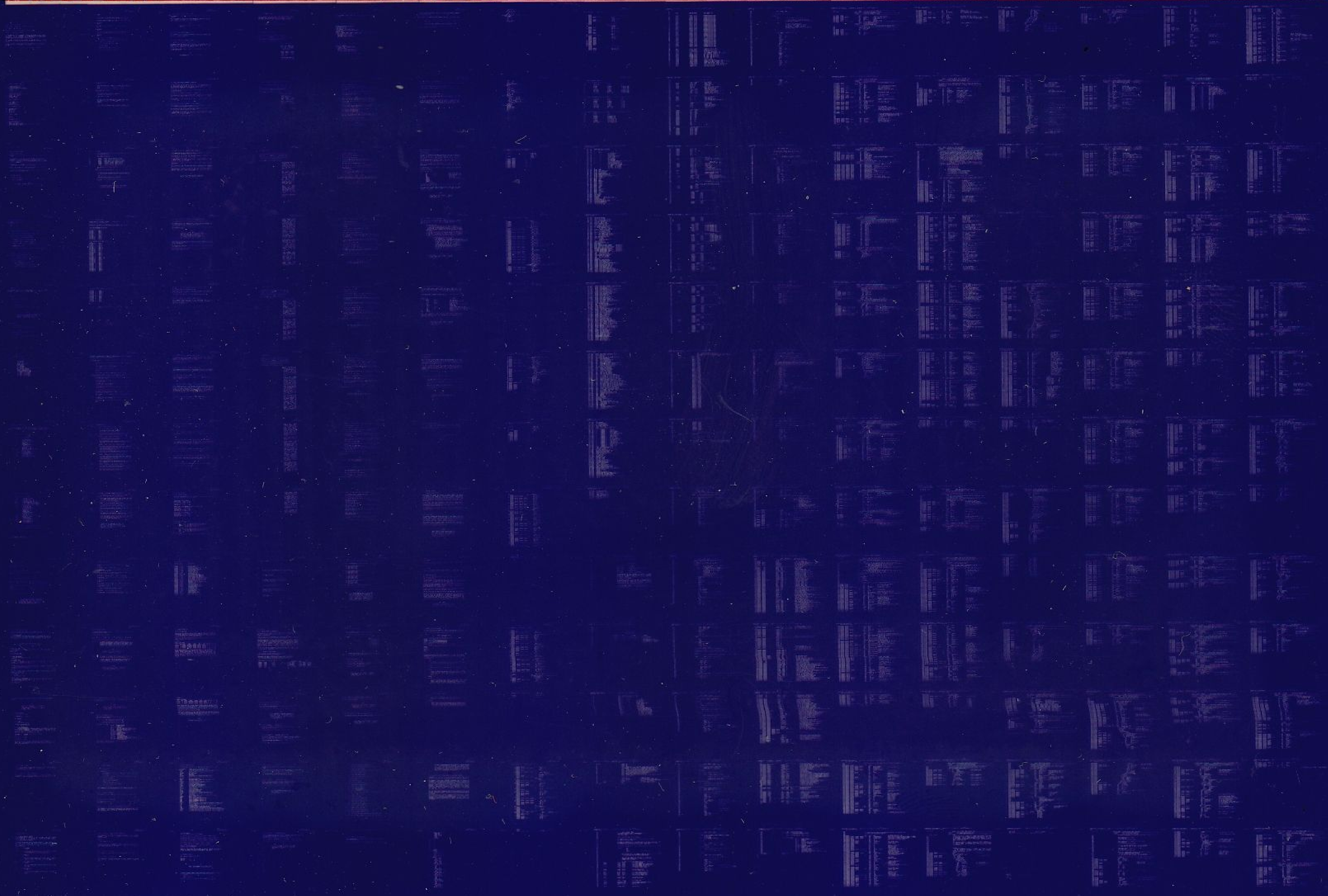
MS11-L*
MS11-M*

MS11-L/M MEMORY
CZMSDB0

AH-F295B-MC
FICHE 1 OF 2

FEB 1981

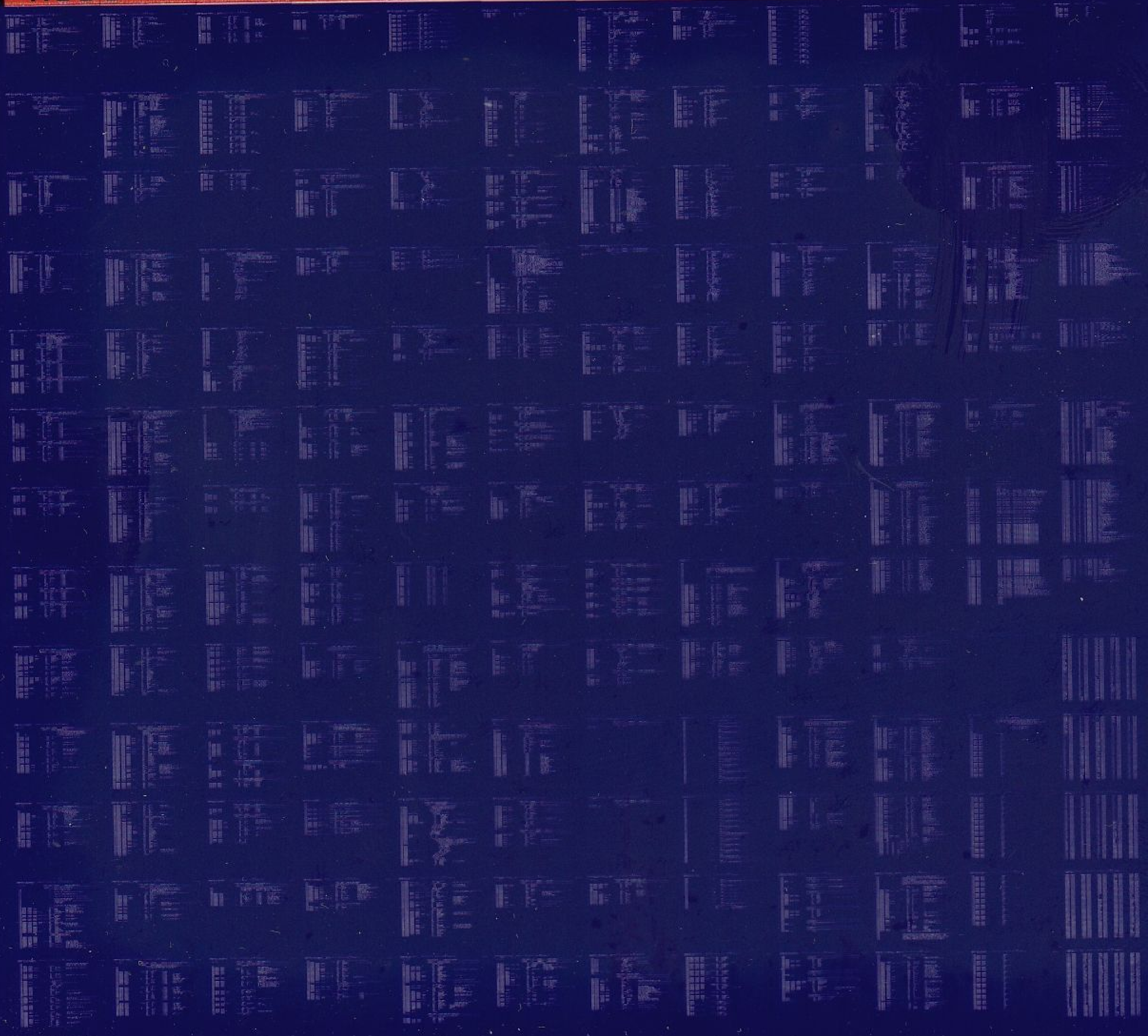


MS11-L*
MS11-M*

MS11-L/M MEMORY
CZMSDB0

AH-F295B-MC
FICHE 2 OF 2

FEB 1981
COPYRIGHT© 79-80
MADE IN USA



IDENTIFICATION

PRODUCT CODE: AC-F294B-MC

PRODUCT NAME: CZMSDB0 MS11-L/M MEMORY DIAGNOSTIC

DATE CREATED: OCTOBER 1980

MAINTAINER: BASE SYSTEM DIAGNOSTIC ENGINEERING

AUTHOR: MICHAEL D BIBEAULT

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.

DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT THAT IS NOT SUPPLIED BY DIGITAL.

TABLE OF CONTENTS

1.0 GENERAL PROGRAM INFORMATION

1.1 Program Purpose (Abstract)

a. Intended for use on all PDP-11's which meet the conditions in 1.2.1.

b. This program will be used by system managers and operators to determine the correct operation of main memory and also it will be primarily used by field service and manufacturing to isolate failures to the memory and to isolate failures within the memory to the correct card.

c. The object of this software is to functionally test and verify all main memory functions as fast as possible.

d. There is the capability of testing mixed configurations (MS11-L, MS11-M and what ever else in on the system).

e. It has special a maintenance mode (Field Service Mode) to provide specific functional capabilities.

1.2 System Requirements

1.2.1 Hardware Requirements -

PDP-11 CPU and at least 64K (16 Bit Words) of Memory and Memory Management.

NOTE

Like memory types must be on 16K word boundaries starting at physical address 0.

1.2.2 Software Requirements -

This program is designed to run stand alone or under any of the following monitors:

XXDP
ACT
APT

1.3 Related Documents And Standards

1. PDP-11/04/34/45/55/60 Processor Handbook (EB9340)
2. PDP-11/44 User's Guide (EK-11044-UG)
3. MS11-M User's Guide (EK-MS11M-UG-001)
4. Programming Practices (175-003-009-02)
5. System Macro Manual (MAINDEC-11-DXQAC-C-D)
6. SUPER-MAC Reference Guide (130-380-007-00)
7. Standard APT System to PDP-11 Diagnostic Interface (APT/11-317-07-09)
8. ACT11/XXDP Programming Specification (AUTOCAT-11-QZAUB-B-D)

1.4 Diagnostic Hierarchy Prerequisites

If the program in any way misbehaves, then:

1. Try it again with Cache off (reference Section 2.4.3.1)
2. Inhibit relocation (reference section 2.4.1)
3. Try CPU Diagnostics
4. Try Memory Management Diagnostics
5. Try Cache Diagnostics (where applicable)
6. Try UNIBUS Map Diagnostics (where applicable)

1.5 Assumptions

This program assumes the correct operation of the CPU, Memory Management, Cache, and the UNIBUS Map. This program occupies (initially) Bank 0 (0-16K). The XXDP loaders are in bank 1.

2.0 OPERATING INSTRUCTIONS

2.1 Loading & Starting Procedures

2.1.1 Quick Starting -

1. Load address 200
2. Set switch register for options (normally 0)
3. Start

NOTE

> If on an 11/24 using MS11-L Memory BE SURE that the peripheral page jumper is in place; failure to do so sends the diagnostic to Never-Never Land.

2.1.2 Stopping -

1. Set SW8, and/or
2. Type control "C" (Reference section 2.4.4.1).

2.1.3 Restarting (Preserve Configuration Table) -

1. Load address 202
2. Set switch register for options (Normally 0)
3. Start

2.1.4 Switch Register Options -

SWITCH	USE
15	HALT ON ERROR
14	LOOP ON TEST
13	INHIBIT ERROR TYPEOUTS
12	INHIBIT RELOCATION
11	QUICK VERIFY
10	BELL ON ERROR
9	LOOP ON ERROR
8	HALT PROGRAM (UNRELOCATE & RESTORE LOADERS)
7	DETAILED ERROR REPORTS
6	INHIBIT CONFIGURATION MAP
5	LIMIT MAX ERRORS PER BANK
4	FAT TERMINAL (132 COLUMNS OR BETTER)
3	TEST MODE - SEE DOCUMENT
2	TEST MODE - SEE DOCUMENT
1	TEST MODE - SEE DOCUMENT
0	DETECT SINGLE BIT ERRORS

2.2 Default Test Sequence

The following two lists give the test protocol for parity and ECC Memory. Tests marked with a "*" are not normally run except under ACT or APT, or through a Field Service Command (Reference Section 2.4.4.8).

2.2.1 Test Protocol For Parity Memory -

Pattern	Pattern Name	Time (sec/16K)
34	Soft Error Test	<1
6	Initial Data Test	<1
17	Holding 1's and 0's Test	<1
7	Address Bit Test	<1
1	Address Test	<1
2	Complement Address Test	<1
3	3 XOR 9 Test	1
4	Rotating 0's Test	1
5	Rotating 1's Test	1
21	Marching 1's and 0's Test	1
35	Worst Case Noise Parity Test	n/a
* 22	Refresh Test	10
* 23	Shifting Diagonal Test	10
26	Random Data Test	<1
* 24	Fast Galloping Pattern Test	20
* 31	Sob-a-long Test	3
* 32	Write Recovery Test	<1
* 33	Branch Gobble Test	35
34	Soft Error Test	<1

2.2.2 Test Protocol For ECC Memory -

Pattern	Pattern Name	Time (sec/16K)
5	Rotating 1's Test	1
@ 25	Interrupt Enable Test	<1
+@ 11	Single Bit Error Test	<2
+@ 12	Write Byte Clears SBE Test	<1
+@ 13	Create Double Bit Error Test	1
%+@ 14	Write Inhibit DATIP w/DBE Test	1
+@ 15	Write Inhibit of Byte w/DBE Test	1
+@ 16	Write Inhibit of Word w/DBE Test	<1
34	Soft Error Test	<1
6	Initial Data Test	<1
10	Byte Address Test	<1
17	Holding 1's and 0's Test	<1
7	Address Bit Test	<1
1	Address Test	<1
2	Complement Address Test	<1
4	Rotating 0's Test	1
5	Rotating 1's Test	1
21	Marching 0's and 1's Test	1
@ 20	Marchin 0's and 1's in CB's Test	<1
* 22	Refresh Test	10
26	Random Data Test	<1
* 24	Fast Galloping Pattern Test	20
* 31	Sob-a-long Test	3
* 32	Write Recovery Test	<1
* 33	Branch Gobble Test	35
34	Soft Error Test	<1

@ - Run only on the first Pass when under ACT or APT

+ - Run twice for each 16K Bank if Interleaved

% - Run only for MF11S-K

At the end of each Pass the program will run cleanup Patterns #30, and #27 for all banks.

2.3 Special Environments

2.3.1 XXDP -

The first pass will be a quick verify pass if and only if it is in chain mode.

2.3.2 ACT & APT Automatic Mode -

The program will not create double bit errors (DBE's) after the 1st pass.

2.3.2.1 APT Execution Times -

Here are some measured execution times for an 11/44 with cache under APT

	1st QV Pass	2nd Pass & onward
128K MS11-M (non-interleaved)	10 min 15 sec	7 min 40 sec
128K MS11-L	9 min 50 sec	7 min 30 sec
256K MS11-M (interleaved)	19 min 50 sec	14 min 45 sec

The tirst pass will be a quick verify pass

NOTE

Even though the first pass is a QV pass it takes longer than the subsequent non-QV passes due to the fact that it is running more patterns, some of which (patterns #24 and #33 for example) can be extrememly time consuming.

2.3.2.2 APT Environment Table -

The following table gives some of the standard settings for the APT E-Table. They may be modified as noted as the user sees fit.

FIRST PASS RUN TIME:
This parameter should be set according to the amount and type of memory to be tested. The above table (APT Execution Times) gives some measured times. For any patterns deleted (through use of the Device Descriptor Words) reference section 2.2 for individual pattern times.

NOTE

The times given in section 2.2 are for 16K chunks of memory, not 128K boards!

LONGEST TEST TIME:
This parameter should be set to the execution time of the longest pattern being run. for the default case this is 35 seconds for Pattern #33.

ADDITIONAL RUN TIME:
Not Used By Program.

SOFTWARE ENVIRONMENT:
For APT auto mode this parameter should be set to a "1". For dump mode set this to a "0".

ENVIRONMENT MODE:
When this parameter is set to a "0" the program does it's own sizing. If the users sets bit #7 however, he must specify the types and amounts of memory to be tested.

SWITCH 1:
The default setting of this switch is "101". APT uses this as the switch register for the program. Reference section 2.4.1 for more information on switch settings.

SWITCH 2:
This switch, if set to any non-zero number, is used to limit the amount of passes APT will make. The program will hang after this count has been reached.

CPU OPTIONS:
Not Used By Program.

MEMORY TYPE n (n=1 to 4)
If bit #7 of ENVIRONMENT MODE is set these four words are used to log the different types of memory to be tested. If bit #7 is not set these location are not used.

MAXIMUM ADDRESS n (n=1 to 4)

These four words are used in conjuntion with the corresponding MEMORY TYPE words to indicate the highest address that memory type occupies.

NOTE

> The above two parameters do not actually have to represent an accurate configuration of memory. All the program looks for is an accurate tally of memory amount!

INTERRUPT VECTOR n (n=1 to 2)
Not Used By Program.

BUS PRIORITY n (n=1 to 2)
Not Used By Program.

BASE ADDRESS:
Not Used By Program.

DEVICE MAP:
Not Used By Program.

CONTROLLER DESCRIPTOR CODE n (n=1 to 2)
Not Used By Program.

DEVICE DESCRIPTOR CODES:
The Device Descriptor codes are used by the program to determine which patterns it will run. The default values of these words are all ''1''s, indicating that all of the patterns shown in section 2.2 are executed (save for exceptions as noted there). Each set of words controls a table in the program as follows:

DD WORDS	PROGRAM TABLE (Symbolic location)
Words 0-1	MKCSRT
Words 2-3	MKPAT
Words 4-5	MJPAT

Bit #0 set in the first word indicates that the first pattern in the table will be executed, bit #1 the second, bit #2 the third,... bit #0 of the second word indicates that the 17th entry in the table will be executed, and so on.

2.3.3 No SBE Free Banks -

If the program cannot find any SBE (Single Bit Error) free locations (in non-protected ECC memory) it will print out an error message and continue testing by-passing the ECC logic tests.

2.3.4 Mixed Parity & ECC Configurations -

The program will function normally in mixed environments. The sequence of testing may seem strange due to the recursive test mode algorithm (reference sections 2.4.1.1, 2.4.1.2, & 2.4.1.3).

2.4 Program Options

2.4.1 Switch Register Details -

If a hardware switch register is not available then the software switch register is in location 176. IF under APT if BIT7 is set in the E-TABLE symbolic location "$ENVM" the APT software switch register will be used (location $SWREG).

To change the software switch register contents: Type "control G". This will cause display the current value of the SWR and prompt for the octal input of the new SWR value from the terminal. This routine will ignore you (not respond to control "G") if you have a hardware switch register.

```
SW15      = HALT ON ERROR
                (100000)

                Continuing from this halt will first check for a change
                in the software switch register ("Control G" in the TTY
                input buffer) then it will continue testing.

SW14      = LOOP ON TEST
                (40000)

                This will cause looping on the present test or  pattern
                (back  to  last  scope trap).  If in a pattern then the
                looping will be for an entire bank of 16K addresses.

SW13      = INHIBIT ERROR TYPEOUTS
                (20000)

                This will cause returns from the error routine  without
                the  typed  messages.  Other on error functions are not
                affected.

SW12      = INHIBIT RELOCATION
                (10000)

                This prevents the program from moving and  consequently
                prevents  the  program  from  testing  at  least 32K of
                memory.
```

```
SW11      = QUICK VERIFY
                (4000)

                If this switch is selected approximately  one  64th  of
                the possible combinations of SBE's & DBE's are tested.

                Each pass complete typeout will indicate this  mode  by
                preceding the pass number with "QV".

SW10      = BELL ON ERROR
                (2000)

                This causes a bell (or beep or  click)  on  each  error
                trap

SW9       = LOOP ON ERROR
                (1000)

                This will cause looping from failure point back to  the
                last correctly initialized area of the current test.

SW8       = HALT PROGRAM
                (400)

                This initiates the following sequence:

                1.  If program is relocated it moves back to bank zero.

                2.  Flush out all possible DBE's.

                3.  Turns off Memory Management.

                4.  Restore loaders.

                5.  Unmap the Unibus Map (if there is one).

                6.  Halt if under APT or ACT branch sel.
```

```
SW7        = DETAILED ERROR REPORTS
                   (200)

                   After any normal error  report  is  typed  this  option
                   causes  the  contents  of the following registers to be
                   typed:
                   R0, R1, R2, R3, R4, R5, SP, "CONTROL", "CPUERR"

SW6        = INHIBIT CONFIGURATION MAP
                   (100)

                   This inhibits the printing of a map showing the  memory
                   configuration - reference section 7.3

SW5        = LIMIT MAX ERRORS PER BANK
                   (40)

                   This will limit the number of error typeouts per  bank.
                   The  default  is  10.  DECIMAL,  however  this  can be
                   changed by changing location "ERRMAX" manually.
SW4        = FAT TERMINAL
                   (20)

                   This informs the program that the console terminal  has
                   a width of at least 132 columns (LA36 with wide paper).
```

SW3-1 = TEST MODE

Test modes determine the recursion algorithm to be used during pattern tests.

	MODE	NAME	DESCRIPTION
(0)	0	BAFPAF	Banks forward, patterns forward
(2)	1	BAFPAR	Banks forward, patterns reverse
(4)	2	BAWPAF	Banks worst first, patterns forward.
(6)	3	BAWPAR	Banks worst first, patterns reverse.
(10)	4	PAFBAF	Patterns forward, banks forward
(12)	5	PAFBAW	Patterns forward, banks worst first
(14)	6	PARBAF	Patterns reverse, banks forward
(16)	7	PARBAW	Patterns reverse, banks worst first.

For more details reference section 2.4.1.1, 2.4.1.2 and 2.4.1.3.

SW0 = DETECT SINGLE BIT ERRORS (SBI's)
(1)

For manufacturing purposes this switch should always be on. For field service purposes this switch should always be off.

This switch will allow all ECC Single Bit errors to be reported by disabling error correction.

Error printouts of SBE's are not distinguishable from DBE's.

NOTE

If Double Bit Errors are found in the memory, this switch should be set to make sure that new data can be written to the DBE locations.

2.4.1.1 Test Mode Example -

Example analysis of mode 5 "PAFBAW". Assume Banks 0 & 1 are MS11-L and Banks 2,3,4,& 5 are MS11-M.

Assume also that Bank 3 is known bad by the program via the sizing routine or previous runs The testing sequence would be as follows:

```
;TEST MS11-M MEMORY TYPES FIRST
;TEST KNOWN BAD MEMORY (BANK 3)

PATTERN 17,     BANK 3
PATTERN  7,     BANK 3
PATTERN  1,     BANK 3
PATTERN  2,     BANK 3
PATTERN  4,     BANK 3
PATTERN  5,     BANK 3
PATTERN 21,     BANK 3
PATTERN 20,     BANK 3
PATTERN 22,     BANK 3
PATTERN 26,     BANK 3

;TEST PRESUMED GOOD MEMORY (BANKS 2,4,5)

PATTERN 17,     BANK 2
PATTERN  7,     BANK 2
PATTERN  1,     BANK 2
PATTERN  2,     BANK 2
PATTERN  4,     BANK 2
PATTERN  5,     BANK 2
PATTERN 21,     BANK 2
PATTERN 20,     BANK 2
PATTERN 22,     BANK 2
PATTERN 26,     BANK 2
PATTERN 17,     BANK 4
PATTERN  7,     BANK 4
PATTERN  1,     BANK 4
PATTERN  2,     BANK 4
PATTERN  4,     BANK 4
PATTERN  5,     BANK 4
PATTERN 21,     BANK 4
PATTERN 20,     BANK 4
PATTERN 22,     BANK 4
PATTERN 26,     BANK 4
PATTERN 17,     BANK 5
PATTERN  7,     BANK 5
PATTERN  1,     BANK 5
PATTERN  2,     BANK 5
PATTERN  4,     BANK 5
PATTERN  5,     BANK 5
PATTERN 21,     BANK 5
PATTERN 20,     BANK 5
PATTERN 22,     BANK 5
PATTERN 26,     BANK 5
```

```
;RELOCATE & TEST PROGRAM SPACE (BANK 0 _& 1)

PATTERN  1,     BANK 0
PATTERN  2,     BANK 0
PATTERN  3,     BANK 0
PATTERN  4,     BANK 0
PATTERN  5,     BANK 0
PATTERN 26,     BANK 0
PATTERN  1,     BANK 1
PATTERN  2,     BANK 1
PATTERN  3,     BANK 1
PATTERN  4,     BANK 1
PATTERN  5,     BANK 1
PATTERN 26,     BANK 1
```

NOTE

This is an example & not an actual sequence.

The pattern sequence was forward (the simple patterns first, complex patterns last) sequence of patterns (MS11-M = 17, 7, 1, 2, 4, 5, 21, 20, 22, 26)(MS11-L = 1, 2, 3, 4, 5, 26).

If the bank selection is forward the banks will be tested in the following order:

1. ECC banks that are not protected or program space (from 0 to 200).

2. Parity banks that are not program space (from 0 to 200).

3. The program now relocates & tests:

4. ECC banks that were protected or program space (from 0 to 200).

5. Parity banks that were program space (from 0 to 200).

If bank selection is worst first the configuration table will be consulted and banks will be tested in the following order.

1. ECC banks that are known bad and are not protected or program space (from 0 to 200).

2. Parity banks that are known bad and are not program space (from 0 to 200).

3. ECC banks that are presumed good and are not protected or program space (from 0 to 200).

4. Parity banks that are presumed good and are not program space (from 0 to 200).

5. The program now relocates & tests:

6. ECC banks that are known bad and were protected or program space (from 0 to 200).

7. Parity banks that are known bad and were program space (from 0 to 200).

8. ECC banks that are presumed good and were protected or program space (from 0 to 200).

9. Parity banks that are presumed good and were program space (from 0 to 200).

2.4.1.2 Test Mode Details -

MODE 0 = "BAFPAF" banks forward, patterns forward

This is the default and simplest mode.

This mode tests each bank completely from 0 to 200 except those requiring relocation*.

While testing each bank the patterns are run with the simple ones first building to the more complex.

MODE 1 = "BAFPAR" = banks forward, patterns reverse

This mode tests each bank completely from 0 to 200 except those requiring relocation*.

While testing each bank the patterns are run with the most complex ones first, working to the simple ones.

MODE 2 = "BAWPAF" = Banks worst first, patterns forward

This mode first tests each known bad bank completely from 0 to 200 except those requiring relocation*, then presumed good banks are tested from 0 to 200 except those requiring relocation*.

While testing each bank the patterns are run with the simple ones first, building to the more complex.

MODE 3 = "BAWPAR" = Banks worst first, patterns reverse

This mode first tests each known bad bank completely from 0 to 200 except those requiring relocation*, then presumed good banks are tested from 0 to 200 except those requiring relocation*.

While testing each bank the patterns are run with the most complex ones first, working to the simple ones.

MODE 4 = "PAFBAF" = Patterns forward, banks forward

This mode tests each pattern completely with the simple ones first, building to the more complex.

While testing each pattern the banks are run from 0 to 200 except those requiring relocation*.

MODE 5 = "PAFBAW" = Patterns forward, banks worst first

This mode tests each pattern completely with the simple ones first, building to the more complex.

While testing each pattern first each known bad bank from 0 to 200 except those requiring relocation* is run, then presumed good banks are run from 0 to 200 except those requiring relocation*.

MODE 6 = "PARBAF" = Patterns Reverse, Banks Forward

This mode tests each pattern completely with the most complex ones first, working to the simple ones.

While testing each pattern the banks are run from 0 to 200 except those requiring relocation*.

MODE 7 = "PARBAW" = Patterns Reverse, Banks Worst First

This mode tests each pattern completely with the most complex ones first, working to the simple ones.

While testing each pattern first each known bad bank from 0 to 200 except those that require relocation* is run, then presumed good banks are run from 0 to 200 except those requiring relocation*.

NOTE

* Relocation is required to test the bank(s) in program space and also to test any ECC banks protected by diagnostic checkmode with the inhibit mode pointer off (zero)!

2.4.1.3 Test Mode Applications -

1. To verify correct operation of the memory system use Mode 0 "BAFPAF".

   Advantages: Easy to understand.

   Disadvantages: In case of a failing Bank, it may take a long time to find the failure.

2. To get detailed error information on known bad Banks (found by sizing routine) use Mode 2 "BAWPAF".

   Advantages: Seeks Bad Banks. Easy to understand.

   Disadvantages: Failures other than zeros & ones may take a long time to find.

3. To get good error info on any memory problem fast use Mode 4 "PAFBAF".

   Advantages: Covers all banks fast. Easy to understand.

   Disadvantages: Failures from only complex patterns may take a long time to find.

4. To find any problem fast use Mode 7 "PARBAW".

   Advantages: Covers all Banks fast.

   Disadvantages: Difficult to understand failures reported are not necessarily the most basic failure modes.

2.4.2 Display Register -

A software display register exists in location 174 in addition to any hardware display existence.

Display fields are as follows:

```
      15 ! 14  13  12  11  10   9   8 ! 7   6   5 ! 4   3   2   1
Relocated !            Bank  #        ! Not Used  !     Pattern #
          !                           !           !
=================================================================
```

PATTERN # = The number of the pattern presently being run. All patterns are described in section 6.2. Any pattern can be found in the Diagnostic by Looking up the symbolic Tags "MTOONN" and "MTPONN" - where "NN" is the Pattern number. MTOONN refers to the routine that sets up for the test Pattern whereas MTPONN is the actual pattern itself.

NOTE

The pattern # is not necessarily an indication of degree of difficulty.

BANK = The number of the Bank (16K) of memory under test (0-200). these bits directly map to physical address bits (21:15).

RELOCATED = This bit indicates that the program is relocated and no longer in Bank 0. It will be relocated to the first known good non-protected memory bank indicated on the configuration map (reference section 7.3).

NOTE

Another way to obtain this information is to type a CONTROL/T at the console (reference Section 2.4.4.5).

2.4.3 Special Memory Locations -

2.4.3.1 CACHE Constant -

The CACHE constant is located at symbolic location "CACHK" and is used to enable CACHE.

NOTE

> Bit 0 in the CACHE constant has no effect since it is unconditionally set by the program whenever it tries to enable CACHE.

2.4.3.2 Configuration Table

The configuration table is located at symbolic location "CONFIG" and has the following format:

```
CONFIG: First 16K Configuration words (2 each)
        2nd   16K Configuration words (2 each)
         .........................
        200th 16K configuration words (2 each)
```

Configuration Words:

LOW:	BIT 0	ERRORS PRESENT
	BIT 1	MEMORY EXISTS
	BIT 2-4	RESERVED
	BIT 5	SKIP ECC LOGIC TESTS FLAG (1 =SKIP)
	BIT 6	PROTECTED REGION OF AN ECC MEMORY
	BIT 7	PROTECTED (PROGRAM SPACE)
	BIT 8-11	CSR CODE
	BIT 12-15	INTERLEAVED CSR CODE
MED:	BIT 0-7	NUMBER OF ERRORS
	BIT 8-10	MEMORY TYPE
	BIT 11	CSR TESTED OK
	BIT 12	INTERLEAVE ENABLED
	BIT 13	"BACKGROUND PATTERN VALID" FLAG
	BIT 14	BANK SELECTED FOR TEST BY FIELD SERVICE MODE
	BIT 15	LOADERS HOME BANK

This table is used as the source for the configuration Map (reference. section 7.3).

2.4.4 Terminal Commands -

2.4.4.1 Control "C"

This command will:

1. If Switch 8 (Halt Program) in the switch register is set halt the program.

2. If Switch 8 is not set, unrelocate if program was relocated.

3. Flush out any DBE's.

4. Turn off Memory Management.

5. Attempt to Boot RK05 Drive 0.

6. Failing 4, attempt to Boot RK04 Drive 1.

7. Failing 5, go to 4.

This command will only be recognized at the completion of the current test or pattern, or at the end of a line of an error message.

2.4.4.2 Control "D" (Debug)

This command to enter a modified version of ODT has been deleted.

2.4.4.3 Control "E" (procEEd)

This command would allow you to exit ODT. It is has also been deleted.

2.4.4.4 Control "K" (Kill error printout and skip pattern)

This command will allow you to stop an error printout and skip to the next pattern. This is handy, for example, when you have a whole bank full of errors, have gotten enough information, and wish to skip to the next pattern.

2.4.4.5 Control "T" (Tell me what's happening)

This command will print out the information encoded in the display register. This is mainly intended for CPU's without a hardware display register.

Example:

```
RELOCATED BANK= 23 PAT= 26
```

By use of Field Service Command 17 "Trace" can be set so that it will automatically type out the bank and pattern numbers as each pattern is run. (Reference section 2.4.4.8.18).

2.4.4.6 Control "S" (Stop)

This command will stop typeout (soon) and will wait for a Control "Q".

2.4.4.7 Control "Q" (Quintinue)

This command will continue typing that has been stopped by Control "S". If there has been no Control "S" typed then this command is ignored.

2.4.4.8 Control "F" (Field Service mode)

This command will cause you to enter a mode which looks for sub commands.

When the program is looking for a sub command any number that is not a legal command will cause a mini help message to be typed. Therefore when in doubt type 99 (CR) and you will get help.

NOTE

> Typing just carriage return is a default command 0.

2.4.4.8.1 Field Service Command 0 (Exit)

This command will exit Field Services Mode and return to whatever task it was in prior to typing control "F". Note typing just carriage return is a default Command 0.

2.4.4.8.2 Field Service Command 1 (Read CSR)

This command will typeout the contents of the CSR.

If there is more than one CSR on the CPU (or if the program has not determined the CSR status yet), it will Ask you "WHICH CSR(0-F)" to which you must respond with an Hexidecimal number from 0 to F. Note typing just carriage return is a default 0.

If the CSR you select causes a trap to 4 the program will type "THIS CSR DOES NOT EXIST".

NOTE

> CSR references are done in accordance with section 5.0.

2.4.4.8.3 Field Service Command 2 (Load CSR)
This command will enable you to load the CSR.

If there is more than one CSR on the CPU (or if the program has not yet determined the CSR status yet) it will ask you "WHICH CSR(0-F)" to which you must respond with an Hexidecimal number from 0 to F. Note typing just carriage return is a default 0.

If the CSR you select causes a trap to 4 the program will type "THIS CSR DOES NOT EXIST".

The CSR will be read and displayed as in command 1.

The program will then ask you for the "CSR?" to which you must respond with an Octal number. Note typing just carriage return is a default 0.

The program will then load the CSR and Read it again displaying its new contents.

2.4.4.8.4 Field Service Command 3 (Examine Memory)

This command will allow you to examine any physical address and does the necessary memory management mapping for you.

The program will ask you for the "PHYSICAL ADDRESS (0-17757776)" to which you must respond with an Octal number.

If the address access causes a trap to 4 the program will type "TIMEOUT TRAP". If the address access causes a trap to 114 the program will type "PARITY ABORT".

The contents of your physical address will be typed.

2.4.4.8.5 Field Service Command 4 (Modify Memory)

This command allows you to modify any physical address and does the necessary memory management mapping for you.

The program will ask you for the "PHYSICAL ADDRESS (0-17757776)" to which you must respond with an Octal number.

If the address access causes a trap to 4 the program will type "TIMEOUT TRAP". If the address access causes a trap to 114 the program will type "PARITY ABORT".

The program will type "OLD DATA WAS" and the contents of your physical address.

The program will then type "INPUT NEW DATA" to which you must respond with an Octal number. Note typing just carriage return is a default 0.
The program will attempt to write this new data into your physical address after which it will read it again and type "DATA IS NOW" and the new contents of your physical address.

NOTE

If you can't change the data, that would indicate that you have a Double Bit Error in that double word pair.

2.4.4.8.6 Field Service Command 5 (Select Bank & Pattern)

This command allows you to run any bank with any pattern forever.

The program will ask you "BANK(0-200)" to which you must respond with an Octal number. If the bank is not accessible. The program will type "BANK NOT ACCESSIBLE" and ask question over.

The program will then ask "PATTERN (0-37)" to which you must respond with an Octal number.

NOTE

> Any pattern can be run including those that are not part of the APT E-TABLE defaults (reference section 6.2.1). If you select Pattern 0, the program will ask "PATTERN 0 DATA IS?" to which you must respond with an Octal number.

If the Bank you selected requires relocation the program will type "BANK REQUIRES RELOCATION" and exit this command. Note normally this is true for Bank 0.

The program will then arm the console keyboard for interrupts and type "TO ESCAPE TYPE ANY KEY!".

The test pattern will be entered and run until a console key is depressed to escape this loop.

2.4.4.8.7 Field Service Command 6 (Type Configuration Map)

This command types the configuration map.

This is useful after a long run (overnight) to see all the banks that are marked as bad. (Especially if your console is a video terminal).

For a detailed explanation of the map reference section 7.3.

2.4.4.8.8 Field Service Command 7 (SOB-A-LONG TEST)

This command allows execution of the SOB-A-LONG Test on all non-protected Banks reference Section 6.2.2.26. Operation is identical to command 5 except that no Pattern or Bank is entered and each pass causes a Bell.

2.4.4.8.9 Field Service Command 8 (Error Summary)

This command types out the number of passes and the total number of errors. If there were any errors it will type out the Banks and the number of errors per bank up to 255 DECIMAL.

This becomes useful after long runs (all night) on systems with a video consoie terminal.

2.4.4.8.10 Field Service Command 9 (Refresh TEST)

This command allows execution of the Refresh Test on all non-protected Banks reference Section 6.2.2.19. Operation is identical to command 5 except that no Pattern or Bank is entered and each pass causes a Bell.

2.4.4.8.11 Field Service Command 10 (Set Fill Count)

This command allows setting of the terminal fill count (necessary for LA30's, ASR33's, and VT05's). It is normally set to zero for LA36's, VT52's, VT100's, etc.

2.4.4.8.12 Field Service Command 11 (Enter Kamikaze Mode)

This command allows you to run patterns that are normally not executed unless under APT or ACT. They are usually very time consuming and can result in failures that are fatal to the program. In effect you are trying to find a hardware failure regardless of the consequences. Note that most crashes do not wipe out the display information which is telling you what the program was doing just prior to failure. There are two ways to die here - Impatience and Crashes.

2.4.4.8.13 Field Service Command 12 (Exit Kamikaze Mode)

Return to the default mode of testing (undo Command 12).

2.4.4.8.14 Field Service Command 13 (Turn Cache Off)

This changes the Cache constant to bypass cache (reference section 2.4.3.1).

2.4.4.8.15 Field Service Command 14 (Turn Cache On)

This changes the Cache constant to use cache (reference section 2.4.3.1).

2.4.4.8.16 Field Service Command 15 (Test Only Selected Banks)

This command allows you to center the test effort on only those banks that you are troubleshooting. You may also test banks that require relocation and were inaccessable via command 5.

2.4.4.8.17 Field Service Command 16 (Resume Testing All Banks)

Return to the default mode of testing (undo Command 15).

2.4.4.8.18 Field Service Command 17 (Resume Testing All Banks)

Enable "Trace". After exiting field service mode, the program will type out the bank and pattern numbers as each pattern is run.

2.4.4.8.19 Field Service Command 18 (Resume Testing All Banks)

Disable "Trace". (undo Command 16).

2.5 Execution Times

2.5.1 Typical (System) -

Execution time depends on many variables; however here are some measured times on an 11/44 with cache:

128K words of MS11-L Memory

Normal Pass	0 Min 50 Sec
Quick Verify	0 Min 50 Sec
Kamikaze Mode	10 Min 5 Sec
Kamikaze QV	10 Min 5 Sec

128K words of MS11-M Memory (Non-Interleaved)

Normal Pass	2 Min 25 Sec
Quick Verify	1 Min 0 Sec
Kamikaze Mode	11 Min 0 Sec
Kamikaze QV	10 Min 30 Sec

128K words of MS11-M Memory (Interleaved)

Normal Pass	3 Min 55 Sec
Quick Verify	1 Min 50 Sec
Kamikaze Mode	22 Min 0 Sec
Kamikaze QV	20 Min 5 Sec

2.5.2 Calculations (System)

Normal Pass
Add 18 Sec per BANK of Non-Intereaved MS11-M
Add 15 Sec per BANK of Interleaved MS11-M
Add 6 Sec per BANK of MS11-L

Quick Verify Pass
Add 8 Sec per BANK of Non-Interleaved MS11-M
Add 7 Sec per BANK of Interleaved MS11-M
Add 6 Sec per BANK of MS11-L

Kamikaze Mode
Add 10 min. per 128K words for approximate pass times.

2.5.3 Typical (Patterns)

Pattern	Time	Description
MT0000	;<1 SEC	DATA PATTERN TEST
MT0001	;<1 SEC	ADDRESS TEST
MT0002	;<1 SEC	COMPLEMENT ADDRESS TEST
MT0003	; 1 SEC	3 XOR 9 WORST CASE NOISE TEST
MT0004	; 1 SEC	ROTATING ZEROS TEST
MT0005	; 1 SEC	ROTATING ONES TEST
MT0006	;<1 SEC	INITIAL DATA TEST
MT0007	;<1 SEC	ADDRESS BIT TEST
MT0010	;<1 SEC	BYTE ADDRESSING TEST
MT0011	;<2 SEC	CREATE SINGLE BIT ERROR TEST
MT0012	;<1 SEC	WRITE BYTE CLEARS SBE TEST
MT0013	; 1 SEC	CREATE DOUBLE BIT ERROR TEST
MT0014	; 1 SEC	WRITE INHIBIT DURING DATIP WITH DBE
MT0015	; 1 SEC	WRITE INHIBIT OF BYTE WITH DBE
MT0016	;<1 SEC	WRITE INHIBIT OF WORD WITH DBE
MT0017	;<1 SEC	HOLDING 1'S & 0'S TEST
MT0020	;<1 SEC	MARCHING 1'S & 0'S IN CHECK BITS
MT0021	; 1 SEC	MARCHING 0'S & 1'S TEST
MT0022	;10 SEC	REFRESH TEST
MT0023	;10 SEC	SHIFTING DIAGONAL TEST
MT0024	;20 SEC	FAST GALLOPING PATTERN TEST
MT0025	;<1 SEC	INTERRUPT ENABLE TEST
MT0026	;<1 SEC	RANDOM DATA TEST
MT0027	; 1 SEC	UNIQUE BANK TEST
MT0030	; 1 SEC	FLUSH OUT DBE'S TEST
MT0031	; 3 SEC	SOB-A-LONG TEST
MT0032	;<1 SEC	WRITE RECOVERY TEST
MT0033	;35 SEC	BRANCH GOBBLE TEST
MT0034	;<1 SEC	SOFT ERROR TEST
MT0035	;<1 SEC	WORST CASE PARITY TEST

3.0 ERROR INFORMATION

3.1 Error Reporting

Most errors are reported using the EMT trap and handler provided by SYSMAC.SML. Most errors will be of the ''MEMORY DATA ERROR'' type which will be described here. MEMORY DATA ERRORS will also cause the bank to be marked as Bad in the configuration table.

Other errors are best explained by referencing the specific typeout and if necessary the program listing.

Example 1:

```
MEMORY DATA ERROR
  PC    BANK  VADD     PADD     GOOD     BAD      XOR   CSR MTYP  INT PAT
022132   37  060006 03700006  000000  000100  000100  0    M     -   06
022132   37  060006 03700006  000000  000100  000100  0    M     -   06
022132   37  060006 03700006  000000  000100  000100  0    M     -   06
022132   37  060006 03700006  000000  000100  000100  0    M     -   06
```

While testing Bank 37 at virtual address 60006 (virtual addresses are always between 60000 and 157776 for mapping purposes), physical address 3700006 (that's Bank 37 physical 6 within the Bank) with Pattern 6 (Initial Data Test), the good data expected was 0 but the data actually read (BAD) was 100, the exclusive OR at Good & Bad yields 100 which indicates only failing bit(s) (Bit 6). It is an MS11-M (ECC) Memory and it's not interleaved. The CSR is located at 172000.

Example 2:

```
MEMORY DATA ERROR
  PC    BANK  VADD      PADD     GOOD     BAD      XOR  CSR MTYP  INT PAT
022132   35  060000  03500000  000000  000001  000001  0   M     1   06
022132   35  060002  03500002  000000  000100  000100  0   M     1   06
022132   35  060006  03500006  000000  000100  000100  0   M     1   06
```

While testing Bank 35, virtual address 60000, physical address 3700000 with Pattern 6 (Initial Data Test), the good data expected was 0 but the data actually read (BAD) was 1, the exclusive OR at Good & Bad yields 1 which indicates only failing bit(s) (Bit 0). It is an MS11-M (ECC) Memory and it's interleaved; so since Address Bit 1 was not asserted, the CSR is located at 172000.

While also in Bank 35, virtual addresses 60002 and 60006 were expected to have 0, but the data read was 100, the exclusive OR of Good & Bad yields 100 which indicates one failing bit (Bit 6). Since it is interleaved MS11-M memory, and Address Bit 1 is asserted, the CSR is located at 172102 (CSR number 1 under the INT column)

NOTE

Subsequent errors of the same test do not type a new heading.

3.2 Error Abbreviations

The following is a list of all abbreviations used in error reports.

# OF ERRORS	Number of Errors that were detected.
1ST ADD	First Address that failed.
ARRAY	The array number that was locked up in the MS11-M CSR.
APT#	The # of CPU's APT expects on the system.
APTCORE	APT Core size.
APTMOS	APT MOS size.
BAD	Bad data.
BAD-WD1	Bad Word #1 of a double word data value.
BAD-WD2	Bad Word #2 of a double word data value.
BAD-CHK	Bad Check Code Bits.
BANK	The Bank number. Banks are 16K words long.
BD-CC	Bad Check Code Bits.
CHKBITS	The 7 bit value of the Check Code Bits.
CONTRL	The CACHE Control register.
CPUERR	CPU Error register.
CSR	Control and Status Register.
CSRNO	CSR NUMBER (0-F Hexidecimal).
DATARG	The CACHE Data Register.
DBE	Double Bit Error (uncorrectable error).
DEV ADD	Device Address.
ECC	Error Correctable Code.
GD-CC	Good Check Code Bits.
GD-CHK	Good Check Code Bits.
GD-WD1	Good Word #1 of a double word data value.
GD-WD2	Good Word #2 of a double word data value.
GOOD	Good data.
INT	Interleaved (Address Bit 1 asserted) CSR number.
LSIZE	MS11-L Size.
MEMERR	Memory Error register.
MMR0	Memory Management Register #0.
MMR1	Memory Management Register #1.
MMR2	Memory Management Register #2.
MMR3	Memory Management Register #3.
MSIZE	MS11-M Size.
MTYP	Memory Type (MS11-L,MS11-M,MF11S-K,BIPOLAR or UNIBUS Parity).
PADD	Physical Address (asserted by the program after mapping).
PAT	Pattern number.
PC	Program Counter at the time the error occurred.
SBE	Single Bit Error (correctable error).
VADD	Virtual Address (asserted by the program before mapping).
WROTE1	The data that was written into the 1st half of a double word.
WROTE2	The data that was written into the 2nd half of a double word.
XOR	Exclusive OR of the good and bad data. Shows the bad bits.

3.3 Error Halts

There are several Halts in the program.

All unused trap vectors contain a trap catcher (.WORD .+2,HALT).

An undefined TRAP instruction halts at symbolic location "$HALT2".

The APT down load sequence will halt at symbolic location "APTHLT".

Halt on Error option (SW15 Set) at symbolic location "$HALT".

Halt program (SW8 Set) at symbolic location "$EXHALT".

Power Fail will normally halt at the end of the shut down sequence (symbolic location "$DOWN").

Power Fail has a fatal Halt at symbolic location "$ILLUP" which can be caused by power up occurring before power down sequence completed or by power down before a power up sequence is completed.

4.0 PROGRESS REPORTS

Pass complete typeouts as follows:

```
END PASS        #       0
END PASS        #       1
END PASS        #QV     2
```

NOTE

Pass 2 was flagged as a Quick Verify Pass. (Because of a change in SW5)

To obtain progress reports while executing, typing a Control "T" will print out the information encoded in the display register.

Example:

```
BANK= 2 PAT= 34
```

Reference Section 2.4.4.8.18 for more information on Tracing.

5.0 CSR INFORMATION TABLES

The following is a picture view of the current control status registers which can be tested by this program. It shows bit assignments and definitions to provide a handy reference, and shows the similarities and differences between each one:

NOTE

All unused bits in each CSR are equal to zero.

5.1 CORE/MOS PARITY REGISTER

```
-------------------------------------------------
I  I  I  I  I  I  I  I  I  I  I  I  I  I  I  I  I
!PE!  !  !  !      ADDRESS        !  !  !WP!  !AE!
I  I  I  I  I  I  I  I  I  I  I  I  I  I  I  I  I
-------------------------------------------------
 15 14 13 12 11 10 09 08 07 06 05 04 03 02 01 00
```

Bit assignments are defined as follows:

BIT15 PARITY ERROR

BITS 11-5 ERROR ADDRESS High order address bits of address of parity error (Bits 17-11 of address).

BIT02 WRITE WRONG PARITY Normal parity (odd) when clear; other parity (even) when set.

BIT00 ACTION ENABLE No action when clear trap to vector 114 when set.

5.2 MOS BIPOLAR PARITY REGISTER (USED IN THE 11/45-55)

```
-------------------------------------------------
I  I  I  I  I  I  I  I  I  I  I  I  I  I  I  I  I
!PE!  !  !  !  !  !  !  !  !  !  !  !WP!  !  !AE!
I  I  I  I  I  I  I  I  I  I  I  I  I  I  I  I  I
-------------------------------------------------
 15 14 13 12 11 10 09 08 07 06 05 04 03 02 01 00
```

Bit assignments are defined as follows:

```
                                                 BIT15 PARITY ERROR

                                                 BIT02
                                                 WRITE WRONG PARITY
                                                 Normal  parity  (odd)
                                                 when   clear;   other
                                                 parity (even) when set

                                                 BIT00 ACTION ENABLE No
                                                 action when clear trap
                                                 to  vector  114  when
                                                 set.
```

5.3 MF11S-K CSR

```
-------------------------------------------------
I  I  I  I  I  I  I  I  I  I  I  I  I  I  I  I  I
!DE!  !SI!  !        ADDRESS        !SE!IP!DC!EC!EE!
I  I  I  I  I  I  I  I  I  I  I  I  I  I  I  I  I
-------------------------------------------------
 15 14 13 12 11 10 09 08 07 06 05 04 03 02 01 00
```

BIT ASSIGNMENTS ARE DEFINED AS FOLLOWS:

BIT15 DOUBLE ERROR Set whenever DBE occurs. If BIT2=0, the error address will be stored in Bits 11-5. If BIT2 =1, the check bits read will be stored in BITS 11-5.

BIT 13 SET INHIBIT MODE When this bit is set to a 1, it enables the Inh Mode Pointer to inhibit either the first or second 16K from ever going into the Diag. Check or ECC Disable mode. When this bit is set to zero, the entire memory operates in in Diagnostic Check or ECC Disable Mode.

BITS 11-5 ERROR ADDRESS With BIT02 cleared they contain the high order error address (Bits 17-11); with BIT02 set they contain the check bits for ECC.

BIT04 SINGLE ERROR Set whenever single error occurs

BIT03 INHIBIT MODE POINTER The Inhibit Mode Pointer works in conjunction with the Set Inhibit Mode bit. When BIT13 is set to a 1, a 16K portion of memory is inhibitted from operating in the ECC Disable mode or Diagnostic Check mode. the Inhibit Mode Pointer indicates which 16K is being inhibited; e.g.-when BIT 3 =1, the second 16K of memory is inhibitted. When BIT 13 is set to a 0, BIT 3 becomes inoperative.

BIT02 DIAGNOSTIC CHECK MODE When set enables read-write of check bits(see Bits 11-5). If a DBE occurs in this mode (with BIT1 =0), BIT15 in the CSR is set but the check bits from memory are stored in CSR Bits 11-5 and not the DBE address bits.

BIT01 DISABLE ERROR CORRECTION When set no single error correction takes place and the error is not logged in the csr; correct check bits are still written to the memory however.

BIT00 DOUBLE ERROR ENABLE When set enables trap to vector 114 on double error.

5.4 MS11-L CSR

```
------------------------------------------------
I  I  I  I  I  I  I  I  I  I  I  I  I  I  I  I  I
!PE!EU!  !  !        ADDRESS        !  !  !WP!  !AE!
I  I  I  I  I  I  I  I  I  I  I  I  I  I  I  I  I
------------------------------------------------
 15 14 13 12 11 10 09 08 07 06 05 04 03 02 01 00
```

Bit assignments are defined as follows:

BIT15 PARITY ERROR

BIT14 EUB ERROR RETRIEVAL If the memory is on an Extended UNIBUS, when BIT14 is zero, the low order failing addresses are available (Bits 11-17); when BIT14 is one, the high order failing addresses are available (Bits 18-21 of address). If the memory is on a UNIBUS, a jumper disables this bit so that it is read only, and equal to zero.

BITS 11-5 ERROR ADDRESS With BIT14 set, they contain the high order parity error address (Bits 21-18 of address); with BIT14 cleared, they contain the low order parity error address (Bits 17-11 of address).

BIT02 WRITE WRONG PARITY Normal parity (odd) when clear; other parity (even) when set.

BIT00 ACTION ENABLE No action when clear; trap to vector 114 when set.

5.5 MS11-M CSR

```
-------------------------------------------------
I  I  I  I  I  I  I  I  I  I  I  I  I  I  I  I  I
!DE!EU!SI!  !        ADDRESS        !SE!IP!DC!EC!EE!
I  I  I  I  I  I  I  I  I  I  I  I  I  I  I  I  I
-------------------------------------------------
 15 14 13 12 11 10 09 08 07 06 05 04 03 02 01 00
```

Bit assignments are defined as follows:

BIT15 UNCORRECTABLE ERROR This bit is set if a DBE occurs, and the error address is stored in the CSR. This bit is also set in the ECC Disable mode if an SBE or DBE occurs.

BIT14 EUB ERROR RETRIEVAL If the memory is on an Extended UNIbus, when BIT14 is zero and either BIT4 or BIT 15 is a one, the low order failing addresses are available (Bits 11-17); when BIT14 is one, the high order failing addresses are available (Bits 18-21 of address). If the memory is on a UNIBUS, a jumper disables this bit so that it is read only, and equal to zero.

BIT13 SET INHIBIT MODE When this bit is set to a 1, it enables the Inh Mode Pointer to inhibit either the first or second 16K from ever going into the Diag. Check or ECC Disable mode. When this bit is set to a 0, it allows the Diag. Check mode and/or ECC Disable mode to operate over the entire memory on the board.

BITS 11-5 ERROR ADDRESS With BIT02 cleared and BIT14 set, they contain the high order error address (Bits 21-18); when BIT02 and BIT14 are cleared, they contain the low order error address (Bits 17-11); when BIT02 is set they contains check bits for ECC.

BIT04 SINGLE ERROR Set whenever single error occurs.

BIT03 INHIBIT MODE POINTER The Inhibit Mode Pointer works in conjunction with the Set Inhibit Mode bit. when BIT13 is set to a 1, a 16K portion of memory is inhibitted from operating in the ECC Disable mode or Diagnostic check mode. the Inhibit Mode Pointer indicates which 16K is being inhibited; e.g.-if BIT3 =1, the second 16K of memory is inhibitted. when BIT13 is set to a 0, BIT3 becomes inoperative.

BIT02 DIAGNOSTIC CHECK MODE When set enables read-write of check bits(see Bits 11-5). If a DBE occurs in this mode (with BIT1=0), BIT15 is set, but the check bits read are stored in Bits 11-5, not the DBE address bits.

BIT01 DISABLE ERROR CORRECTION When set no single error correction takes place. A single bit error will set BIT04 and BIT15 and assert BUS PBL L if BIT00 is asserted; a double error will set set BIT15 and assert BUS PBL L if BIT00 is asserted. The error address is stored in the CSR, and correct check bits are generated and stored on a write.

BIT00 UNCORRECTABLE ERROR ENABLE When set enables trap to vector 114 on uncorrectable error.

6.0 SUB-TEST SUMMARIES

6.1 Tests

TEST 1

BIT TEST OF ALL CSR'S/MATCH ALL CSR'S WITH MEMORY
(CSR Access may cause wrong Type of Traps)

TEST 2

TEST BANK 0 ACCESSES
Failures are fatal.

TEST 3

TEST BANKS 1-200 (OCTAL) FOR ZEROS AND ONES
Errors are not typed here - only logged in the configuration table

TEST 4

ECC INHIBIT MODE POINTER TEST

TEST 5

DIAGNOSTIC MODE DISPATCH ROUTINE
This test runs all the patterns in the mode selected.

TEST 6

UNIQUE BANK TEST
Pattern 27 is run

6.2 Patterns

6.2.1 General Pattern Information

Actual patterns are identified by symbolic locations "MTPXYY" where X may be any sub program indicator (A,B,C,etc) or 0 and YY will be the number of the pattern.

Setup procedures for each pattern are identified by symbolic locations "MTOOYY" where YY will be the number of the pattern.

Patterns reside in 4 scripts that are scanned for execution. Symbolic location "MKCSRT" is a table of patterns that can run once for each ECC bank (twice for interleaved MS11-M's). Symbolic location "MKPAT" is a table of patterns that can run on each Bank of ECC memory. Symbolic location "MJPAT" is a table of patterns that can run on each Bank of Parity memory. Symbolic 'ocation "FSPAT" is a table of patterns that can be run in Field Serv .e Mode (command 5).

The 1st 3 scripts are completely controlled by the APT E-table (even if not running under APT). Modifications to this table can be made (1) with APT, or (2) manually.

Example E-table Segment:

```
;THE FOLLOWING LOCATIONS SPECIFY WHICH PATTERNS
;ARE TO BE RUN FOR PARTICULAR MEMORIES
;
;REFERENCE THE TABLE LISTED BELOW TO RELATE BITS TO PATTERNS.
;BITO SET WILL RUN THE FIRST ENTRY IN THE TABLE, BITO SET
;IN THE SECOND WORD WILL RUN THE 17TH ENTRY IN THE TABLE...
;
;NOTE**NULL TESTS DO NOT TAKE ANY TIME
;                                                      RECOMMENDED VALUE
$DDW0:    .WORD        177777              ;ECC CSR TESTS         177777 TABLE _= MKCSRT:
$DDW1:    .WORD        177777              ;ECC CSR TESTS         177777 TABLE _= MKCSRT:
$DDW2:    .WORD        177777              ;ECC PATTERNS          103777 TABLE _= MKPAT:
$DDW3:    .WORD        177777              ;ECC PATTERNS          177777 TABLE _= MKPAT:
$DDW4:    .WORD        177777              ;PARITY PATTERNS       003777 TABLE _= MJPAT:
$DDW5:    .WORD        177777              ;PARITY PATTERNS       177774 TABLE _= MJPAT:
```

6.2.2 Specific Patterns

6.2.2.1 Pattern 0 Basic Data Test

Writes & Reads R2 into a 16K Bank.

This is used for Zeros and Ones testing and in Field Service Mode for any console selected pattern.

It can execute out of the USER Instruction PAR's.

NOTE

> It is frequently modified dynamically such that (1) it returns after writing only (the 1st NOP is replaced with a RETURN) or (2) it only counts Errors (the code PERR02 and NOP are replaced with INC @#PATERR).

6.2.2.2 Pattern 1 Address Test

Writes & Reads an incrementing pattern equivalent to physical addressed into a 16K Bank.

It can execute out of the USER Instruction PAR's.

6.2.2.3 Pattern 2 Complement Address Test

Writes the complement of the physical address from high addresses to low (write down) and reads from low addresses to high (read up).

This provides the complement of the coverage of Pattern 1 in both data pattern and addressing sequence.

It can execute out of the USER Instruction PAR's.

6.2.2.4 Pattern 3 3 XOR 9

Writes & Reads a pattern that complements as address bits 3 and 9 change.

This pattern is run 4 times (1) with Zeros & Ones, (2) with Ones & Zeros, (3) with 401 & Ones, and (4) with Ones & 401. The pattern of the 401 is to force a the parity bits to become involved.

It can execute out of the USER Data PDR's, the User Instruction PAR's, the Kernel Data PAR's and the Supervisor Data PAR's.

6.2.2.5 Pattern 4 Rotating Zeros Test

Writes a background pattern of Ones. Rotates a Zero Carry Bit left thru each par of bytes (18 times) and then checks that the carry is Zero and the word (2 bytes) is still all Ones.

It can execute out of the User Data PAR's and the Kernel Data PAR's.

NOTE

> It is not uncommon to observe the good data equal to the bad data. This indicates that the carry was not clear after 18 ROLB's.

6.2.2.6 Pattern 5 Rotating Ones Test

Writes a background pattern of Zeros. Rotates a One carry bit left thru each pair of bytes (18 times) and then checks that the Carry is a One and the Word (2 Bytes) is still all Zeros.

This provides the complement of the coverage of Pattern 4 in data.

It can execute out of the User Data PAR's and the Kernel Data PAR's.

NOTE

It is not uncommon to observe the good data equal the bad data. This indicates that the Carry was not set after 18 ROLB's.

6.2.2.7 Pattern 6 Initial Data Test

Writes & Reads a double word first with all bits 0 except 1 (for every bit position), Second with all bits 1 except 1 (for every bit position).

This is a very quick check of the data paths.

6.2.2.8 Pattern 7 Address Bit Test

```
Writes a background of all Zeros.
Read Address 1 for a 0 Byte.
Complement Address 1.
Read Address 1 for a non 0 Byte.
For each Address Bit position from Bit 1:
Virtual (2, 4, 10, 20, 40, 100, 200, 400,  1000,  2000,  4000,  10000,
60000, 20000)
Physical (60002, 60004, 60010,  60020,  60040,  60100,  60200,  60400,
61000, 62000, 64000, 70000, 140000, 100000)
  Read Address for a 0 word.
  Complement Address contents.
  Read Address for a non-zero word.
```

This is a very quick check of the address bit uniqueness.

6.2.2.9 Pattern 10 Byte Addressing Test

```
With ECC Disabled.
Writes all ones to a double word.
For each of the 4 Bytes in the Double Word.
  Clears one byte.
  Reads all 4 bytes from double word.
  Checks for only proper byte clear.
  All other bytes set to all ones.
```

This is only done on one double word address.

NOTE

This is run for ECC memory only

6.2.2.10 Pattern 11 Single Bit Error Test

1. Create a Single Bit Error.

2. Read data Uncorrected (with ECC Disable).

3. Check that SBE and DBE flags are set, and the error address is latched.

4. Read First Word of data corrected (with ECC Enabled)

5. Check that the CSR Single Bit Error Flag was set, and the error address was latched.

6. Clear SBE Flag.

7. Read Second word of data corrected (with ECC Enabled).
8. Check that the CSR Single Bit Error Flag was set.

9. Do (1-7) for a Single Bit Error in each of 32 positions of a double word.
   i.e. (32 TIMES)

10. If not in Quick Verify Mode then Do (1-8) for data consisting of 1 bit set in each of 32 positions of a double word.
    i.e. (32 X 32 = 1024 Times)

11. Do (1-9) for complemented data (1 Bit clear in each of 32 positions of a double word).
    i.e. (1024 X 2 = 2048 Times)
    or (32 X 2 = 64 Times (Quick Verify))

12. Do (1-7) for a double word equal to (000000,000000), and all possible Single Bit Error combinations forced into the Check Bits (CSR bits 5-11).

13. Clear any errors out of test locations.

This insures that all Single Bit Errors can be corrected and detected.

NOTE

This test is run for ECC memory only

6.2.2.11 Pattern 12 Write Byte Clears SBE Test

1. Create a Single Bit Error.

2. Write a Byte of Double Word to Ones.

3. Read a Byte of Double Word.

4. If this is MS11-M, the SBE flag should be SET.
   If this is MF11S-K the SBE flag should be SET if this is the byte with the error.

5. The Byte should have been equal to Ones.

6. Do (1-5) for each of the 4 Bytes of the Double Word

7. Do (1-6) for a Single Bit Error in each of 32 positions of a Double Word
   i.e. (32 Times)

8. If not in Quick Verify Mode then do (1-7) for data consisting of 1 Bit set in each of 32 positions of a double word.
   i.e. (32 X 32 = 1024 Times)

9. Clear any errors out of test locations.

This insures that single bit errors in the data portion (not in checkbits) can be cleared by writing the corresponding byte and that writing any other byte does not change the existing single bit error.

NOTE

This test is run for ECC memory only.

6.2.2.12 Pattern 13 Create Double Bit Error Test

1. Create a Double Bit Error.

2. Access the Data (TST instruction).

3. Check that the CSR DBE Flag is set, and the error address is latched.

4. Initialize CSR to allow parity traps on DBE's.

5. Access the Data (TST Instruction).

6. Check that a parity trap occurred.

7. Do (1-6) for the 2nd Bit of each Double Bit Error in each of 32 positions of a double word less the one position of the 1st Bad Bit.
   i.e. (31 Times)

8. If not in Quick Verify Mode then Do (1-7) for the 1st Bit of each of Double Bit Error in each of 32 positions of a double word.
   i.e. (31 X 32 = 992 Times)

9. Do (1-8) for complemented data (Ones versus Zeros in Double Word)
   i.e. (992 X 2 = 1984 Times)
   or (31 X 2 = 62 Times (Quick Verify))

10. Do (1-6) for a double word equal to (000000,000000), and all possible Double Bit Error combinations forced into each of the check bits (CSR bits 5-11).

11. Clear any errors out of test locations.

This insures that all Double Bit Errors can be created and detected and cause traps.

NOTE

> This test is only run during the first (QV) PASS when under ACT or APT, and is run for ECC memory only.

6.2.2.13 Pattern 14 Write Inhibit During DATIP With DBE Test

1. Create a Double Bit Error.

2. Do ASRB on Test Location.

3. Check that Double Word is STILL Bad (Unchanged-with DBE).

4. Do (2-3) on all 4 Bytes of Double Word.

5. Do (1-4) for the 2nd bit of each Double Bit Error in each of 32 positions of a Double Word less the one position of the 1st Bad Bit.
   i.e. (31 Times)

6. If not in Quick Verify Mode then Do (1-5) for the 1st Bit of each Double Bit Error in each of 32 positions of a double word.
   i.e. (32 X 32 = 922 Times)

7. Do (1-6) for complemented data (Ones versus Zeros in Double Word).
   i.e. (922 X 2 = 1984 Times)
   or (31 X 2 = 62 Times (Quick Verify))

8. Do (1-4) for a double word equal to (000000,000000), and all possible Double Bit Error combinations forced into the Check Bits(CSR bits 5-11).

9. Clear any errors out of test locations.

This insures that the Double Bit Error can be cleared by a DATIP to any affected Byte.

NOTE

This test is only run during the first (QV) pass when under ACT or APT, and is run for MF11S-K only.

6.2.2.14 Pattern 15 Write Inhibit Of Byte With DBE

1. Create a Double Bit Error.

2. Do a MOVB immediate to test byte..

3. Check that Double Word is still Bad (unchanged-with DBE).

4. Do (2-3) on all 4 Bytes of Double Word.

5. Do (1-4) for the 2nd Bit of each Double Bit Error in each of 32 positions of a double word less the one position of the 1st Bad Bit.
   i.e. (31 Times)

6. If not in Quick Verify Mode then Do (1-5) for the 1st Bit of each Double Bit Error in each of 32 positions of a Double Word.
   i.e. (31 X 32 = 922 Times)

7. Do (1-6) for Complemented Data (Ones versus Zeros in Double Word).
   i.e. (992 X 2 = 1984 Times)
   or (31 X 2 = 62 Times (Quick Verify))

8. Do (1-4) for a double word equal to (000000,000000), and all possible Double Bit Error combinations forced into the Check Bits (CSR bits 5-11).

9. Clear any errors out of test locations.

This insures that no Double Bit Error can be cleared by a MOVB to any affected Byte.

NOTE

This test is only run during the first (QV) pass when under ACT or APT, and is run for ECC memry only.

6.2.2.15 Pattern 16 Write Inhibit Of Word With DBE Test

1. Create a Double Bit Error.

2. Do MOV IMMEDIATE on test location.

3. Check that Double Word is STILL Bad (unchanged-with DBE).

4. Do (2-3) on both Double Words.

5. Do (1-4) for the 2nd Bit of each Double Bit Error in each of 32 positions of a Double Word less the one position of the 1st Bad Bit.
   i.e. (31 Times)

6. If not in Quick Verify Mode then Do (1-5) for the 1st Bit of each Double Bit Error in each of 32 positions of a Double Word.
   i.e. (32 X 32 = 992 Times)

7. Do (1-6) for Complemented Data (Ones versus Zeros in Double Word).
   i.e. (992 X 2 = 1984 Times)
   or (31 X 2 = 62 Times (Quick Verify))
8. Do (1-4) for a double word equal to (000000,000000), and all possible Double Bit Error combinations forced into the Check Bits (CSR bits 5-11).

9. Clear any errors out of test locations.

This insures that no Double Bit Error can be cleared by a MOV to any affected word.

NOTE

> This test is only run during the first (QV) pass when under ACT or APT, and is run for ECC memory only.

6.2.2.16 Pattern 17 Holding 1's & 0's Test

1. Write a 16K Bank with alternating Bytes of Zeros & Ones writing a Byte at a time.

2. Read each Word for correct pattern.

3. Do (1-2) again for a complement pattern.

This checks the memory for the capability of holding 0's & 1's.

6.2.2.17 Pattern 20 Marching 0's & 1's In Check Bits Test

1. Write Double Words of 000000,,000000 which causes check bits to equal 077 while addressing increments.
   (Write Up/077 --> check bits)

2. If in Quick Verify Mode then Go to Step (5).

3. Read Double Words & check while writing 000000,,100000 and addressing decrements.
   (Down/077 --> 100)
   This flips all the checkbits.

4. Read Double Words & check while writing Zeros while addressing increments.
   (Up/100 --> 077)

5. Read Double Words & check while writing 000000,,100000 & addressing increments.
   (Up/077 --> 100)

6. Read Double Words & check while writing Zeros while addressing decrements.
   (Down/100 --> 077)

7. Read Double Words & check while Addressing increments.
   (Up/077)

This checks the integrity of the MOS chips that store the checkbits.

6.2.2.18 Pattern 21 Marching 0's & 1's Test

1. Write a Background of alternating Bytes of Zeros & Ones

2. For the 16K Bank addressing Down
   (a) Read check a word
   (b) Byte Swap a word
   (c) Read check a word

3. For the 16K Bank addressing Up
   (a) Read check a word
   (b) Byte Swap a word
   (c) Read check a word

4. For the 16K Bank addressing Up
   (a) Read check a word
   (b) Byte Swap a word
   (c) Read check a word

5. For the 16K Bank addressing Down
   (a) Read check a word
   (b) Byte Swap a word
   (c) Read check a word

This checks the integrity of the 32 Bit Double Words.

It can execute out of the User Data PAR's.

NOTE

> It is not uncommon to see a misleading error typeout because the second test in each case is based upon a byteswap of the first test which may or may not have failed. If the error report indicates errors in pairs with the bad bit in the second report being the same bit position relative to a byte then you should ignore the second error report.

6.2.2.19 Pattern 22 Refresh Test

1. Write a diagonal pattern of ones on every KDIAG(TH) stripe & write zeros elsewhere.

   This pattern is on addresses not bit positions.

   Example:

```
Address                                 MSB's
                                        ------------------
              LSB's                     ! 0 0 0 1 0 0 0 1
                                        ! 0 0 1 0 0 0 1 0
                                        ! 0 1 0 0 0 1 0 0
                                        ! 1 0 0 0 1 0 0 0
                                        ! 0 0 0 1 0 0 0 1
                                        ! 0 0 1 0 0 0 1 0
                                        ! 0 1 0 0 0 1 0 0
                                        ! 1 0 0 0 1 0 0 0
```

   NOTE

   Example uses KDIAG of value 4 more typical is a value of 8. Consult the symbolic definition of "KDIAG" in the program listing to be sure.

2. Disturb each row for > 3.2ms

3. Read check diagonal pattern

4. Do (1-3) KDIAG times moving the placement of the diagonal stripe to cover all address positions.

5. Do (1-4) for a complement pattern (zeros in a background of ones)

   NOTE

   This test is not normally executed except under APT or ACT. It may be invoked VIA Field Service Command 13 (Kamikaze Mode).

6.2.2.20 Pattern 23 Shifting Diagonal Pattern Test

Similar in overall operation to pattern 22 except it does not delay for refresh and disturb rows.

NOTE

> This test is not normally executed except under APT or ACT. It may be invoked VIA Field Service Command 13 (Kamikaze Mode).

6.2.2.21 Pattern 24 Fast Galloping Pattern Test

This does a classical galloping pattern except that addressing is incremented by 400 Octal (every 64th double word)

NOTE

> This test is not normally executed except under APT or ACT. It may be invoked VIA Field Service Command 13 (Kamikaze Mode).

6.2.2.22 Pattern 25 Interrupt Enable Test

1. Set CSR to Allow Uncorrectable Error Traps.
2. Access Test Double Words.
3. Check that no Uncorrectable Error Trap occurred.
4. Enable CSR for SBE Traps.
5. Access Test Double Words.
6. Check that no SBE Trap occurred.
7. Write a SBE in 1 Byte.
8. Disable CSR Traps.
9. Access Test Double Words.
10. Check that no Traps occurred.
11. Enable CSR for SBE Traps.
12. Access Test Double Words.
13. Check to Insure Trap Occurred.
14. Do (7-13) for the 3 other Bytes in the Double Word.
15. Create a DBE in 1 Byte.
16. Disable CSR Traps.
17. Access the Test Double Word.
18. Check that no Traps occurred.
19. Enable CSR for DBE Traps.
20. Access the Test Double Word.
21. Check to Insure Trap Occurred.
22. Enable CSR for SBE Traps.
23. Access the Test Double Word.
24. Check to Insure Trap Occurred.
25. Do (15-24) for the 3 other Bytes in the Double Word.

This insures that SBE's & DBE's give the correct type of traps.

NOTE

This test is run for ECC memory only.

6.2.2.23 Pattern 26 Random Data Test

Write Random Data in a 16K Bank while incrementing the Addresses.
Read check Random Data.

This routine regenerates the same random numbers by using the same seed as the write sequence. After the read check the seed is updated so that the next use of this pattern will not invoke the same sequence of random numbers.

If you wish to change the random sequence so that it is different than any other run in the same configuration then there are 2 ways of doing so.

1. Modify symbolic locations "SEEDHI" and "SEEDLO" to any number you like.

2. Enter Field Service Mode and execute this pattern (command 5) on some (any good) bank for a short time (30 sec or so).

This can execute out of the User Data PAR's, the Kernel Data PAR's, and the Supervisor Data PAR's.

6.2.2.24 Pattern 27 Unique Bank Test

This pattern uses Pattern 0 to write & read the Bank number in each bank.

It does not test Banks that require relocation to test.

It does not run as part of any script but rather is always run after normal pattern tests are complete.

6.2.2.25 Pattern 30 Flush Out DBE's Test

This Reads each location then moves the old value back in. This is done with ECC Disabled and therefore corrects any DBE's or SBE's (if possible).

It does not run as part of any script but rather is always run just prior to the End of Pass Code, as part of a Control "C" (Boot) command, as part of End of Pass shutdown for ACT or XXDP Chain Mode, as part of hanging sequence after an error if under ACT or APT, and as part of a shutdown sequence directed by Switch 8 (Halt Program).

6.2.2.26 Pattern 31 SOB-A-LONG Test

Rationalization

In order to concentrate the memory cycles of a test into a particular address, we must cut the overhead cycles to a minimum. Frequently, the instruction itself may provide adequate data or set up a background in which any complemented bit may find it hard to survive.

The SOB instruction is the only PDP-11 instruction that is (1) a single operand, (2) can be repeatedly executed at the same PC and, (3) can escape this repetitious loop.

Hence, it can be possible to SOB a MOS cell to death (or at least brain wash him), and to SOB a core into over-heating (or at least warm discomfort).

The SOB Routine will be loaded and called with R0 set equal to the SOB constant "SOBK", R1 set equal to the complement of a "SOB R0,." Instruction "100776".

Simplified SOB Example:

```
1$:     SOB         R0,1$                   ;SOB till R0 underflows
        MOV         R1,1$                   ;Write complement of SOB
        CMP         R1,1$                   ;Read & check not SOB
        BEQ            2$                   ;Skip if OK
        SOBFAIL                             ;Trap & report error
2$:     SOBMOV1                             ;Code to get self moved
        SOBMOV2                             ;Forward 1 word and run again
        SOBMOV3
        SOBMOV4
        SOBMOV...
```

The value of the SOB constant can be found at symbolic location "SOBK" (typical 25 decimal).

This test is not in the normal script of execution but may be added via the APT E-TABLE, reference symbolic locations "MKPAT", "MJPAT", "$DDW2-5". Field Service Mode command 8 is the normal method of running this pattern.

NOTE

> This test is not normally executed except under APT or ACT. It may be invoked VIA Field Service Command 13 (Kamikaze Mode).

6.2.2.27 Pattern 32 Write Recovery Test

This test causes a WRITE, READ, WRITE, READ, ... to occur in memory and if the 1st, 3rd, 5th, ...READ is bad the program may bomb or if the 2nd, 4th, 6th, ... READ is bad the program will gracefully type out the error.

Write Recovery Test
This test differs from other tests in that it consists of a small test program actually running in the bank under test.

The program is self modifying and may be difficult to debug. To aid in the debug, remember that the bank and margin are being displayed. This will allow the user to at least see which memory bank failed.

The test consists of 1/2 of the bank stored with "MOV R2,-(PC)" and the other 1/2 containing "177667". "177667" is the complement of "JMP (R0)" instruction. R2 contains "COM -(R1)" instruction on entry to the bank and R1 contains the highest test address in that bank.

If you understand this so far the rest is easy.

The test execution is as follows:

1. The "MOV R2,-(PC)" instruction executes storing the contents of R2 in the address it vacated (due to -(PC).
2. Since R2 contains a "COM -(R1)" instruction it complements the highest address under test. this address contained "177667" so after the COM -(R1) it equals 110 cleverly this is the "JMP (R0)" instruction.
3. This sequence continues until the "MOV R2,-(PC)" instructions reach the middle of the test bank. then the "JMP (R0)" instruction is met and executed. R0 contained the return address back to test 13.
4. These steps are repeated for each bank under test.

NOTE

This test is not normally executed except under APT or ACT. It may be invoked VIA Field Service Command 13 (Kamikaze Mode).

6.2.2.28 Pattern 33 Branch Gobble Test

This test loads a small routine into the memory under test. The routine moves itself along in memory one word after each pas so that when it reaches the end every instruction has executed from every location with the exception of the beginning and end of each test area.

The Branch Gobble's general format after you eliminate setup code and code to move the program along is as follows.

```
BGTEST:  0                                      ;TEST WORD
BRGOBB:  SEC
         ADCB         BGTEST                    ;INC LOW BYTE
         BMI          1$                        ;END LOOP AFTER 128 TIMES
         INCB         BGTEST+1                  ;INC HIGH BYTE
         BR           BRGOBB                    ;LOOP 128 TIMES
1$:      BVS          2$                        ;BRANCH IF V-BIT SET (SHOULD BE)
         ERROR                                  ;ERROR TRAP
2$:      CLV                                    ;CLEAR V-BIT
         INCB         BGTEST                    ;INC HIGH BYTE ONE LAST TIME
         BCS          3$                        ;BRANCH IF C-BIT SET (SHOULD NOT BE)
         BVC          3$                        ;BRANCH IF V-BIT CLEAR (SHOULD NOT BE)
         BMI          4$                        ;BRANCH IF N-BIT SET (SHOULD BE)
3$:      ERROR                                  ;ERROR TRAP
4$:      RETURN
```

This code origionally came from the PDP-11 Family Instruction Exerciser DZQKA-A. The first MOS memorys fell succeptable to this section of that diagnostic and it has been an important memory exerciser ever since.

NOTE

> This test is not normally executed except under APT or ACT. It may be invoked VIA Field Service Command 13 (Kamikaze Mode).

6.2.2.29 Pattern 34 Soft Error Test

Rationalization

MOS chips have a failure mode in which they can randomly pick or drop bits. This is caused by Alpha particles bombarding the cell. If the cell is very small (and they are) then the electrons displaced by the Alpha particle are sufficient to cause the cell to change from a one to a zero or from a zero to a one.

This test is controlled by the main program so that it is used to create a pattern of 125252 and 52525 on alternate passes of the program. The configuration table is used to flag banks that have the pattern invalidated because another pattern was written over this background.

This pattern is nothing more than a clever use of pattern 0.

6.2.2.30 Pattern 35 Worst Case Parity Test

1. Force Write Wrong Parity in each 1K word block of the Memory Under Test.

2. Read with Parity Trapping enabled, making sure that a trap occurrs.

3. Make sure error address bits are set correctly.

4. Write good parity without trapping, and make sure no trap occurrs when read.

NOTE

This test is run for parity memory which is not controlled by the same CSR as the program.

6.2.2.31 Pattern 999 Null Test

This is an instant return added to preserve the software structure.

This pattern replaces any real patterns when the APT E-Table does not specify a pattern to be run.

7.0 PROGRAM FEATURES

7.1 Fast Data Access Rates

One of the main areas of concern in testing memory in systems environments is speed. One of the prime reasons that system programs like RSTS,IAS and MUMPS can crash due to memory failures not detectable by memory diagnostics (0-124K,0-2 MEG,etc.) is because of multiple NPR devices contending for the bus. After some delay a NPR device becomes bus master and does several memory transfers at memory data rates.

On the other hand most diagnostics when writing reading and/or checking patterns spend most of their time fetching instructions and operands out of their program space and proportionally little time accessing the memory under test.

This diagnostic's error detecting abilities have been optimized around the primary design criteria of speed. To this end the following steps have been taken.

7.1.1 Fast City

Utilization of Memory Management Registers as Non Memory Bus, Non UNIBUS, Bipolar Memory. Since User Mode is only used for relocation and Data Space is never used, then subroutines can be executed from the UIPAR's, UDPAR's, KDPAR's, SDPAR's and with some Bit Pattern restrictions the UIPDR's, UDPDR's, KDPDR's, and SDPDR's.

The program runs in Kernel mode and Patterns are executed in Supervisor mode for mapping purposes. All core patterns and some MOS Patterns are subroutines that are moved to this Bipolar region referred to in the program as Fast City.

NOTE

> 18-Bit PDP-11's cannot execute from the PAR's because their PAR's are only 12 bits wide; they also have no Supervisor Mode. Therefore, all patterns are executed in memory, using User Mode (reference Section 7.5).

7.1.2 SOB's

Utilization of the full PDP-11 Instruction Set to speed pattern algorithms (principally the SOB).

7.1.3 CACHE

CACHE is used between pattern tests to decrease program pass times. CACHE can be defeated by the operator (reference section 2.4.3.1).

7.2 Bank Zero Testing

Bank Zero has been traditionally neglected by memory diagnostics for the following reason.

The vector space exists there and ALL traps must not access test pattern data. If the area is tested the diagnostic must not use any traps, and it is against the rules for power to fail.

Systems with Memory Management can overcome this because all traps are to Kernel Virtual space even if the power should fail (caution must be observed because power up goes to physical address 24 (because the Memory Management Unit comes up off)).

However, Catch 22 is that the diagnostic is not APT compatible in this mode because APT Accesses Physical Memory Locations.

The PDP-11/44 can over come this because the UNIBUS Map can fool APT.

Because of the previous arguments this program does not relocate in the true since of the word (i.e. no position independent code was written (at least not on purpose)), but rather this program moves and remaps (hereafter referred to as relocates). This enables the complete testing of Bank Zero or any other program space or privileged space exactly as all other banks are tested. (The conditional test to see if a bank is protected is complemented when relocated).

NOTE

> The program will relocate only in the first pass under APT; after this, the program will remain fixed in Banks 0 and 1.

7.3 Memory Configuration Map

This map is printed out immediately after sizing the memory unless SW6 is set (reference section 2.4.1). It can also be printed at any later time in Field Service Mode (reference section 2.4.4.8.7)

Example:

```
                       MEMORY CONFIGURATION MAP
                              16K BANKS
                1       2       3       4       5       6       7
        012345670123456701234567012345670123456701234567012345670123
ERRORS          XX
CPU MAP 11111111111111111111111111111111
INTRLV  ----------------3333333333333333
MEMTYPE LLLLLLLLMMMMMMMMMMMMMMMMMMMMMMMM
CSR     00000000111111112222222222222222
PROTECT PP      I       I       P
            0       1       2       3       4       5       6
        456701234567012345670123456701234567012345670123456701234567
ERRORS
CPU MAP
INTRLV
MEMTYPE
CSR
PROTECT
```

Displayed are Banks 0-73 Octal (2 meg words). If the Fat Terminal Switch was set (reference section 2.4.1) then all Banks (0-167) would be shown. If this was an 18-Bit PDP-11 (eg - 11/34), only Banks 0-7 would be printed.
The fields:

ERRORS:

The sizing routine could not write zeros and ones in Banks 10 & 11, hence they are marked as bad with X's.

CPU MAP:

The CPU was able to access banks 0-37 (512 K words).

INTRLV:

There is interleaving on Banks 20-37, with CSR 2 (172104) controlling the Address Bit 1 Non-Asserted addresses, and CSR 3 (172106) controlling the Address Bit 1 Asserted addresses.

ERRORS:

MEMTYPE:

Banks 0-7 are Memory Type L (MS-11L), and Banks 10-37 are Memory Type M (MS11-M); while Banks 40-167 do not exist. Memory Type K would indicate MF11S-K, Memory Type P would indicate UNIBUS Parity, and Memory Type B would indicate

11/45-type Bipolar memory.

CSR:

Banks 0-7 are assigned to CSR 172100, 10-17 to CSR 172102, and 20-37 to interleaved CSR's 172104 and 172106.

PROTECT:

Banks 0 and 1 are protected because they are program space. Bank 0 and 1 can also be protected because they are in the bottom 16K of an MS11-M CSR. The protection is hierarchical and program space overshadows MS11-M protection. Banks 0 and 1 will not be tested until the program relocates. If any bank is protected by MS11-M (or MF11S-K) and not because it is in program space it will have an "I" typed in this row. This is to point out where the protected banks start for each ECC CSR. Note the "P" at Bank 30; This points out the "Shadow" protection which occurrs when two MS11-M memories are interleaved. Therefore, Bank 30 will not be tested until the program has relocated.

7.4 Everything You've Always Wanted To Know About SUPERMAC ...

SUPER-MAC is a set of structured programming macros that allows programs to be written in a high level, easily understood language.

As a general rule, most SUPER-MAC statements can be single-line statements or multiple-line (nested) block statements. A single-line statement must be completed on one source line; no continuation lines are allowed. Single-line statements should be as short and simple as possible. Comments may also be included on a source line. All the general ruies, conditions, etc., that govern MACRO-11 also govern SUPER-MAC. Spacing on a source lne is very important. The elements should be separated by a comma or a space. Tabs should never be used for spacing. For example: The expression A+B is interpreted different than A + B.

All the conditional statements can be written as multiple-line nested biocks. Each level of nesting within a block must be terminated with an associated END statement. Each level of nesting should be indented two spaces.
User written macros or assembly language instructions may be included in a program if desired. As a debugging aid, if the symbol LST$$ is defined, it will cause generated code and labels to be listed. All programs must begin with the macro call SMACIT. This call initializes SUPER-MAC. All legal PDP-11 source and destination operands are legai in SUPER-MAC.

7.4.1 Sample Source File -

```
        .ENABL ABS
        .ENABL AMA
        .MCALL .SUPER
        .SUPER
        ;LST$$_=0
        BIT5_=40
A:      0
B:      0
C:      0
D:      0
E:      0
F:      0
G:      0
H:      0
I:      0
J:      0
        .PAGE
;LET EXAMPLES
        LET R0 :_= A
        LET B :_= C + D
        LET E :_= F +  1
        LET G :_= H +  2
        LET J :_= J +  01
        LET A :B_= B
;IF EXAMPLES
        IF A IS TRUE
          MOV     23,D
        END ;OF IF A
        IF B IS FALSE
          MOV     34,E
        END ;OF IF B
        IF A EQ B THEN LET C :_= D
        IF A LT B
          MOV    C,D
        ELSE
          MOV    E,D
        END ;OF IF A
        IF A EQ B AND C NE D
          MOV    F,G
        END ;OF IF A
        IF A EQ B OR C NE D
          MOV    F,G
        END ;OF IF A
        IFB A EQ B AND C EQ  1
          MOV    H,J
        ELSE
          MOV    E,J
        END ;OF IFB A
        IFB A EQ B ANDB C EQ  1
          MOV    H,J
        ELSE
          MOV    E,J
        END ;OF IFB A
        IF RESULT IS EQ
          MOV    A,B
        END ;OF IF RESULT
        IF  BIT5 SET.IN A
          MOV   B,C
        END ;OF IF  BIT5
        IF  BIT5 OFF.IN A
          MOV   C,D
        END ;OF IF  BIT5
;ON.ERROR IS LIKE AN IF STATEMENT ON THE C-BIT
;ON.ᐧRROR EXAMPLES
        ON.ERROR
          MOV   A,B
        ELSE
          MOV   C,B
        END ;OF ON.ERROR
        ON.NOERROR
```

```
          MOV    C,B
        ELSE
          MOV    A,B
        END ;OF ON.NOERROR
        ON.ERROR THEN LET A :B_= B
;FOR EXAMPLES
        FOR I :_=  -5 TO  23
          INC    A
        END ;OF FOR I
        FOR RO :_=  0 TO  140 BY  4
          DEC    A(RO)
        END ;OF FOR RO
        FOR I :_=  133 DOWNTO  3 BY  2
          ADD    A,B
        END ;OF FOR I
;BEGIN EXAMPLES
        BEGIN ALPHA
          FOR RO :_=  0 TO  167
            MOVB           A(RO),B
            IF B LT  0 THEN LEAVE ALPHA
          END ;OF FOR RO
          FOR RO :_=  400 TO  567
            IF B GE  0 THEN LEAVE ALPHA
          END ;OF FOR RO
        END ALPHA
;$RETURN EXAMPLES
        $RETURN
        $RETURN ERROR
        $RETURN NOERROR
;CASE EXAMPLES
        MOV      A,RO
        CASE RO
          A
          B
          C
          D
          E
          F
        END ;OF CASE RO

        .END
```

```
7.4.2  Sample Listing File (with No Expanded Macros) - -
.MAIN.  MACRO M1111  01-APR-79 16:41  PAGE 2

      1 000000                                  .ENABL ABS
      2                                         .ENABL AMA
      3                                         .MCALL .SUPER
      4 000000                                  .SUPER
      5                                         ;LST$$ =0
      6         000040                          BIT5_=40
      7 000000  000000                  A:      0
      8 000002  000000                  B:      0
      9 000004  000000                  C:      0
     10 000006  000000                  D:      0
     11 000010  000000                  E:      0
     12 000012  000000                  F:      0
     13 000014  000000                  G:      0
     14 000016  000000                  H:      0
     15 000020  000000                  I:      0
     16 000022  000000                  J:      0
```

```
                                        ;LET EXAMPLES
000024                                          LET R0 :_= A
000030                                          LET B :_= C + D
000044                                          LET E :_= F +  1
000056                                          LET G :_= H +  2
000072                                          LET J :_= J +  01
000100                                          LET A :B_= B
                                        ;IF EXAMPLES
000106                                          IF A IS TRUE
000114  012737   000023   000006                  MOV     23,D
000122                                          END ;OF IF A
000122                                          IF B IS FALSE
000130  012737   000034   000010                  MOV     34,E
000136                                          END ;OF IF B
000136                                          IF A EQ B THEN LET C :_= D
000154                                          IF A LT B
000164  013737   000004   000006                  MOV    C,D
000172                                          ELSE
000174  013737   000010   000006                  MOV    E,D
000202                                          END ;OF IF A
000202                                          IF A EQ B AND C NE D
000222  013737   000012   000014                  MOV    F,G
000230                                          END ;OF IF A
000230                                          IF A EQ B OR C NE D
000250  013737   000012   000014                  MOV    F,G
000256                                          END ;OF IF A
000256                                          IFB A EQ B AND C EQ  1
000276  013737   000016   000022                  MOV    H,J
000304                                          ELSE
000306  013737   000010   000022                  MOV    E,J
000314                                          END ;OF IFB A
000314                                          IFB A EQ B ANDB C EQ  1
000334  013737   000016   000022                  MOV    H,J
000342                                          ELSE
000344  013737   000010   000022                  MOV    E,J
000352                                          END ;OF IFB A
000352                                          IF RESULT IS EQ
000354  013737   000000   000002                  MOV    A,B
000362                                          END ;OF IF RESULT
000362                                          IF  BIT5 SET.IN A
000372  013737   000002   000004                  MOV    B,C
000400                                          END ;OF IF  BIT5
000400                                          IF  BIT5 OFF.IN A
000410  013737   000004   000006                  MOV    C,D
000416                                          END ;OF IF  BIT5
                                        ;ON.ERROR IS LIKE AN IF STATEMENT ON THE C-BIT
                                        ;ON.ERROR EXAMPLES
000416                                          ON.ERROR
000420  013737   000000   000002                  MOV    A,B
000426                                          ELSE
000430  013737   000004   000002                  MOV    C,B
000436                                          END ;OF ON.ERROR
000436                                          ON.NOERROR
000440  013737   000004   000002                  MOV    C,B
```

```
72 000446                                   ELSE
73 000450  013737  000000  000002             MOV   A,B
74 000456                                   END ;OF ON.NOERROR
```

```
 75 000456                                        ON.ERROR THEN LET A :B_= B
 76                                       ;FOR EXAMPLES
 77 000466                                        FOR I :_=  -5 TO  23
 78 000474  005237  000000                          INC    A
 79 000500                                        END ;OF FOR I
 80 000514                                        FOR R0 :_=  0 TO  140 BY  4
 81 000516  005360  000000                          DEC    A(R0)
 82 000522                                        END ;OF FOR R0
 83 000534                                        FOR I :_=  133 DOWNTO  3 BY  2
 84 000542  063737  000000  000002                  ADD    A,B
 85 000550                                        END ;OF FOR I
 86                                       ;BEGIN EXAMPLES
 87 000566                                        BEGIN ALPHA
 88 000566                                          FOR R0 :_=  0 TO  167
 89 000570  116037  000000  000002                    MOVB        A(R0),B
 90 000576                                            IF B LT  0 THEN LEAVE ALPHA
 91 000604                                          END ;OF FOR R0
 92 000614                                          FOR R0 :_=  400 TO  567
 93 000620                                            IF B GE  0 THEN LEAVE ALPHA
 94 000626                                          END ;OF FOR R0
 95 000636                                        END ALPHA
 96                                       ;$RETURN EXAMPLES
 97 000636                                        $RETURN
 98 000640                                        $RETURN ERROR
 99 000644                                        $RETURN NOERROR
100                                       ;CASE EXAMPLES
101 000650  013700  000000                        MOV      A,R0
102 000654                                        CASE R0
103 000664  000000                                  A
104 000666  000002                                  B
105 000670  000004                                  C
106 000672  000006                                  D
107 000674  000010                                  E
108 000676  000012                                  F
109 000700                                        END ;OF CASE R0
110
111         000001                                .END
```

7.4.3 Sample Listing File (with Expanded Macros) - -

```
.MAIN.  MACRO M1111  01-APR-79 16:10  PAGE 2

     1 000000                                 .ENABL ABS
     2                                        .ENABL AMA
     3                                        .MCALL .SUPER
     4 000000                                 .SUPER
     5          000000                        LST$$_=0
     6          000040                        BIT5_=40
     7 000000   000000               A:       0
     8 000002   000000               B:       0
     9 000004   000000               C:       0
    10 000006   000000               D:       0
    11 000010   000000               E:       0
    12 000012   000000               F:       0
    13 000014   000000               G:       0
    14 000016   000000               H:       0
    15 000020   000000               I:       0
    16 000022   000000               J:       0
```

```
.MAIN.  MACRO M1111  01-APR-79 16:10  PAGE 3

     18                                         ;LET EXAMPLES
     19 000024                                          LET R0 :_= A
        000024  013700  000000                          MOV A,R0
     20 000030                                          LET B :_= C + D
        000030  013737  000004  000002                  MOV C,B
        000036  063737  000006  000002                  ADD D,B
     21 000044                                          LET E :_= F +  1
        000044  013737  000012  000010                  MOV F,E
        000052  005237  000010                          INC E
     22 000056                                          LET G :_= H +  2
        000056  013737  000016  000014                  MOV H,G
        000064  062737  000002  000014                  ADD  2,G
     23 000072                                          LET J :_= J +  01
        000072  062737  000001  000022                  ADD  01,J
     24 000100                                          LET A :B_= B
        000100  113737  000002  000000                  MOVB B,A
     25                                         ;IF EXAMPLES
     26 000106                                          IF A IS TRUE
        000106  005737  000000                          TST A
        000112  001403                                  BEQ L0
     27 000114  012737  000023  000006                    MOV     23,D
     28 000122                                          END ;OF IF A
        000122                                  L0:
     29 000122                                          IF B IS FALSE
        000122  005737  000002                          TST B
        000126  001003                                  BNE L1
     30 000130  012737  000034  000010                    MOV     34,E
     31 000136                                          END ;OF IF B
        000136                                  L1:
     32 000136                                          IF A EQ B THEN LET C :_= D
        000136  023737  000000  000002                  CMP A,B
        000144  001003                                  BNE L2
        000146  013737  000006  000004                  MOV D,C
        000154                                  L2:
     33 000154                                          IF A LT B
        000154  023737  000000  000002                  CMP A,B
        000162  002004                                  BGE L3
     34 000164  013737  000004  000006                    MOV   C,D
     35 000172                                          ELSE
        000172  000403                                  BR L4
        000174                                  L3:
     36 000174  013737  000010  000006                    MOV   E,D
     37 000202                                          END ;OF IF A
        000202                                  L4:
     38 000202                                          IF A EQ B AND C NE D
        000202  023737  000000  000002                  CMP A,B
        000210  001007                                  BNE L5
        000212  023737  000004  000006                  CMP C,D
        000220  001403                                  BEQ L5
     39 000222  013737  000012  000014                    MOV   F,G
     40 000230                                          END ;OF IF A
        000230                                  L5:
     41 000230                                          IF A EQ B OR C NE D
        000230  023737  000000  000002                  CMP A,B
```

```
000236  001404                          BEQ L6
000240  023737  000004  000006          CMP C,D
000246  001403                          BEQ L7
```

```
       000250                              L6:
    42 000250  013737  000012  000014                MOV   F,G
    43 000256                                      END ;OF IF A
       000256                              L7:
    44 000256                                      IFB A EQ B AND C EQ  1
       000256  123737  000000  000002              CMPB A,B
       000264  001010                              BNE L10
       000266  023727  000004  000001              CMP C, 1
       000274  001004                              BNE L10
    45 000276  013737  000016  000022                MOV   H,J
    46 000304                                      ELSE
       000304  000403                              BR L11
       000306                              L10:
    47 000306  013737  000010  000022                MOV   E,J
    48 000314                                      END ;OF IFB A
       000314                              L11:
    49 000314                                      IFB A EQ B ANDB C EQ  1
       000314  123737  000000  000002              CMPB A,B
       000322  001010                              BNE L12
       000324  123727  000004  000001              CMPB C, 1
       000332  001004                              BNE L12
    50 000334  013737  000016  000022                MOV   H,J
    51 000342                                      ELSE
       000342  000403                              BR L13
       000344                              L12:
    52 000344  013737  000010  000022                MOV   E,J
    53 000352                                      END ;OF IFB A
       000352                              L13:
    54 000352                                      IF RESULT IS EQ
       000352  001003                              BNE L14
    55 000354  013737  000000  000002                MOV   A,B
    56 000362                                      END ;OF IF RESULT
       000362                              L14:
    57 000362                                      IF  BIT5 SET.IN A
       000362  032737  000040  000000              BIT  BIT5,A
       000370  001403                              BEQ L15
    58 000372  013737  000002  000004                MOV   B,C
    59 000400                                      END ;OF IF  BIT5
       000400                              L15:
    60 000400                                      IF  BIT5 OFF.IN A
       000400  032737  000040  000000              BIT  BIT5,A
       000406  001003                              BNE L16
    61 000410  013737  000004  000006                MOV   C,D

    62 000416                                      END ;OF IF  BIT5
       000416                              L16:
    63                                     ;ON.ERROR IS LIKE AN IF STATEMENT ON THE C-BIT
    64                                     ;ON.ERROR EXAMPLES
    65 000416                                      ON.ERROR
       000416  103004                              BCC L17
    66 000420  013737  000000  000002                MOV   A,B
    67 000426                                      ELSE
       000426  000403                              BR L20
       000430                              L17:
```

```
68 000430  013737  000004  000002            MOV    C,B

69 000436                                   END ;OF ON.ERROR
   000436                         L20:
70 000436                                   ON.NOERROR
```

```
     000436  103404                              BCS L21
  71 000440  013737  000004  000002                 MOV   C,B
  72 000446                                      ELSE
     000446  000403                              BR L22
     000450                            L21:
  73 000450  013737  000000  000002                 MOV   A,B
  74 000456                                      END ;OF ON.NOERROR
     000456                            L22:
  75 000456                                      ON.ERROR THEN LET A :B_= B
     000456  103003                              BCC L23
     000460  113737  000002  000000              MOVB B,A
     000466                            L23:
  76                                   ;FOR EXAMPLES
  77 000466                                      FOR I :_=  -5 TO  23
     000466  012737  177773  000020              MOV  -5,I
     000474                            B0:
  78 000474  005237  000000                         INC   A
  79 000500                                      END ;OF FOR I
     000500  005237  000020                      INC I
     000504  023727  000020  000023              CMP I, 23
     000512  003770                              BLE B0
     000514                            F0:
  80 000514                                      FOR R0 :_=  0 TO  140 BY  4
     000514  005000                              CLR R0
     000516                            B1:
  81 000516  005360  000000                         DEC   A(R0)
  82 000522                                      END ;OF FOR R0
     000522  062700  000004                      ADD  4,R0
     000526  020027  000140                      CMP R0, 140
     000532  003771                              BLE B1
     000534                            E1:
  83 000534                                      FOR I :_=  133 DOWNTO  3 BY  2
     000534  012737  000133  000020              MOV  133,I
     000542                            B2:
  84 000542  063737  000000  000002                 ADD   A,B
  85 000550                                      END ;OF FOR I
     000550  162737  000002  000020              SUB  2,I
     000556  023727  000020  000003              CMP I, 3
     000564  002366                              BGE B2
     000566                            E2:
  86                                   ;BEGIN EXAMPLES
  87 000566                                      BEGIN ALPHA
     000566                            B3:
  88 000566                                         FOR R0 :_=  0 TO  167
     000566  005000                              CLR R0
     000570                            B4:
  89 000570  116037  000000  000002                    MOVB        A(R0),B
  90 000576                                            IF B LT  0 THEN LEAVE ALPHA
     000576  005737  000002                      TST B
     000602  002415                              BLT E3
  91 000604                                         END ;OF FOR R0
     000604  005200                              INC R0
     000606  020027  000167                      CMP R0, 167
     000612  003766                              BLE B4
```

```
   000614                          E4:
92 000614                                   FOR R0 : =  400 TO  567
   000614  012700  000400                  MOV  400,R0
```

```
        000620                          B5:
     93 000620                                          IF B GE  0 THEN LEAVE ALPHA
        000620  005737  000002                  TST B
        000624  002004                          BGE E3
     94 000626                                    END ;OF FOR R0
        000626  005200                          INC R0
        000630  020027  000567                  CMP R0, 567
        000634  003771                          BLE B5
        000636                          E5:
     95 000636                                  END ALPHA
        000636                          E3:
     96                                 ;$RETURN EXAMPLES
     97 000636                                  $RETURN
        000636  000207                          RTS PC
     98 000640                                  $RETURN ERROR
        000640  000261                          SEC
        000642  000207                          RTS PC
     99 000644                                  $RETURN NOERROR
        000644  000241                          CLC
        000646  000207                          RTS PC
    100                                 ;CASE EXAMPLES
    101 000650  013700  000000                  MOV     A,R0
    102 000654                                  CASE R0
        000654  010046                          MOV R0,-(SP)
        000656  006316                          ASL @SP
        000660  004737  000700                  JSR PC,L24
    103 000664  000000                            A
    104 000666  000002                            B
    105 000670  000004                            C
    106 000672  000006                            D
    107 000674  000010                            E
    108 000676  000012                            F
    109 000700                                  END ;OF CASE R0
        000700                          L24:
        000700  062616                          ADD (SP)+,@SP
        000702  013646                          MOV @(SP)+,-(SP)
        000704  004736                          JSR PC,@(SP)+
    110
    111         000001                          .END
```

7.5 Memory Management Mapping

7.5.1 Memory Management Mapping For The 11/44 -

PAR	SUPERVISOR	KERNEL	USER
0	Program	Program	Dst Bk/Fst Mem
1	Program	Program	Src Bk/Fst Mem
2	Program	Program	Src Bk/Fst Mem
3	Test Area	Program	Src Bk/Fst Mem
4	Test Area	Program	Dst Bk/Fst Mem
5	Test Area	Program	Dst Bk/Fst Mem
6	Test Area	Map to CSR's	Dst Bk/Fst Mem
7	Perif Page	Perif Page	Dst Bk/Fst Mem

7.5.2 Memory Management Mapping For UNIBUS-11's With Supervisor Mode (eg 11/45) -

PAR	SUPERVISOR	KERNEL	USER
0	Program	Program	Dst Bk
1	Program	Program	Src Bk
2	Program	Program	Src Bk
3	Test Area	Program	Src Bk
4	Test Area	Program	Dst Bk
5	Test Area	Program	Dst Bk
6	Test Area	Map to CSR's	Dst Bk
7	Perif Page	Perif Page	Dst Bk

7.5.3 Memory Management Mapping For UNIBUS-11's W/o Supervisor Mode (eg 11/34) -

PAR	KERNEL	USER
0	Program	Program/Dst Bk
1	Program	Program/Src Bk
2	Program	Program/Src Bk
3	Program	Test Area/Src Bk
4	Program	Test Area/Dst Bk
5	Program	Test Area/Dst Bk
6	Map to CSR's	Test Area/Dst Bk
7	Perif Page	Perif Page/Dst Bk

```
.TITLE  CZMSDBO MS11-L/M DIAGNOSTIC
.REM    &


                              IDENTIFICATION
                              --------------

  PRODUCT CODE:           AC-F294B-MC

  PRODUCT NAME:           CZMSDBO MS11-L/M MEMORY DIAGNOSTIC

  PRODUCT DATE:           DECEMBER 1979

  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 MANUAL.

  THE SOFTWARE DESCRIBED IN THIS DOCUMENT IS FURNISHED TO  THE
  PURCHASER  UNDER  A  LICENSE  FOR  USE  ON A SINGLE COMPUTER
  SYSTEM  AND  CAN  BE  COPIED  (WITH  INCLUSION  OF DIGITAL'S
  COPYRIGHT NOTICE) ONLY FOR USE IN SUCH SYSTEM, EXCEPT AS MAY
  OTHERWISE BE PROVIDED IN WRITING BY DIGITAL.

  DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY  FOR
  THE  USE OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT THAT IS
  NOT SUPPLIED BY DIGITAL.

  COPYRIGHT (C)  1979 DIGITAL EQUIPMENT CORPORATION
&
```

```
;                                  REVISION HISTORY
;                                  ================

;  REVISION      DATE              AUTHOR                     CHANGES
;  ========    =========      ================               =======
;   CZMSDA     01-DEC-79     MICHAEL D BIBEAULT      NONE=NEW PROGRAM
;
;   CZMSDB     01-OCT-80     MICHAEL D BIBEAULT      1) COMPATIBLE WITH 11/24
;                                                    2) SIZNG ROUTINE WILL ACCEPT ALL
;                                                       LEGAL MEMORY CONFIGURATIONS
;                                                    3) ALL FIELD SERVICE COMMANDS
;                                                       OPERATIVE
;
;
;
```

```
        .SBTTL  OPERATIONAL SWITCH SETTINGS
        .SBTTL  ;SWITCH REGISTER DEFINITIONS
        .SBTTL  ;*
        .SBTTL  ;*        SWITCH              USE
        .SBTTL  ;*        ------              --------------------
        .SBTTL  ;*          15                HALT ON ERROR
        .SBTTL  ;*          14                LOOP ON TEST
        .SBTTL  ;*          13                INHIBIT ERROR TYPEOUTS
        .SBTTL  ;*          12                INHIBIT RELOCATION
        .SBTTL  ;*          11                QUICK VERIFY
        .SBTTL  ;*          10                BELL ON ERROR
        .SBTTL  ;*           9                LOOP ON ERROR
        .SBTTL  ;*           8                HALT PROGRAM (UNRELOCATED & RESTORE LOADERS)
        .SBTTL  ;*           7                DETAILED ERROR REPORTS
        .SBTTL  ;*           6                INHIBIT CONFIGURATION MAP
        .SBTTL  ;*           5                LIMIT MAX ERRORS PER BANK
        .SBTTL  ;*           4                FAT TERMINAL (132 COLUMNS OR BETTER)
        .SBTTL  ;*           3                TEST MODE - SEE DOCUMENT
        .SBTTL  ;*           2                TEST MODE - SEE DOCUMENT
        .SBTTL  ;*           1                TEST MODE - SEE DOCUMENT
        .SBTTL  ;*           0                DETECT SINGLE BIT ERRORS
```

```
000000                          .ENABL  ABS
                                .ENABL  AMA
                                .DSABL  GBL
                                ;NOTE:  CZMSDB.SML IS THE SUPER.MAC SOURCE AND IS RELEASED WITH
                                ;THIS PROGRAM.  ALL THESE .MCALL STATEMENTS REFERENCE THAT FILE.
                                .MCALL  SMACIT,..PUSH,..POP,..TAG,..BRAN,.EMIT,.EMITN,.EMITL,.EMITR
                                .MCALL  .IFOPR,.IS,.GENBR,.OPADD,.OPSUB,CLEAR,SET,CLEARB,SETB
                                .MCALL  RNE,REQ,RLT,RGE,RGT,RLE,RPL,RMI,RHI,RLOS,RHIS,RLO,RCS,RCC
                                .MCALL  IF,.OR,.IFARI,.LEAVE,.GOTO,OR,AND,THEN,ELSE,WHILE,CASE
                                .MCALL  FOR,TO,DOWNTO,REPEAT,UNTIL,THRU,END,BEGIN
                                .MCALL  $$END,LEAVE,JUMPTO,GOTO,PUSH,POP,LET
                                .MCALL  .SIMPLE,.ARITH,ORB,ANDB,IFB,UNTILB,WHILEB,ON.ERROR,ON.NOERROR
                                .MCALL  $CALL,$RETURN

                                .NLIST  TTM                     ;I WANT-FAT PAPER!
                                .LIST   MC,SYM                  ;LIST MACRO CALLS, SYMBOL TABLE
                                .NLIST  MD,CND,ME               ;DON'T LIST MACRO DEFS & CONDITIONALS & EXPANSIONS
                                ;LST$$= 0                       ;DEFINED TO LIST SUPERMAC EXPANSIONS
           163000               $SWR=   163000                  ;USE THESE SYSMAC SWITCHES
           000001               $TN=    1                       ;FIRST TEST NUMBER TO ONE(1)
000000                          SMACIT
```

```
                                         .SBTTL  DEFINE  TRAPS
                               ;ALL ENTRIES HERE MUST HAVE A CORRESPONDING ENTRY IN THE
                               ;TRAP TABLE "$TRPAD" (NEAR END OF PROGRAM).
                               ;*TRAP DEFINITIONS
                               ;
                               ;HERE IS HOW TRAPS WORK IN THIS PROGRAM
                               ;
                               ;ALL TRAPS EXECUTE A "TRAP" INSTRUCTION WHICH TAKES THE PROGRAM
                               ;TO SYMBOLIC LOCATION "$TRAP"
                               ;
                               ;AT $TRAP THE PROGRAM PICKS UP THE RIGHT BYTE OF THE TRAP INSTRUCTION
                               ;AND INDEXES INTO A TABLE AT LOCATION "$TRPAD" WHICH SENDS THE PROGRAM TO
                               ;THE SPECIFIC ROUTINE TO HANDLE THAT SPECIFIC TRAPS TASK.
                               ;
                               ;THE ULTIMATE DESTINATION OF A TRAP INSTRUCTION CAN BE GUESSED AT AS FOLLOWS
                               ;
                               ;EXAMPLE:        NOP
                               ;                NOP
                               ;                NOP
                               ;                KERNEL                  ;ENTER KERNEL MODE
                               ;                NOP
                               ;
                               ;        ADD A DOLLAR SIGN TO THE SYMBOLIC NAME AND CHECK THE CRF FOR SOMETHING CLOSE
                               ;        IN THIS CASE THE CRF HAS $KERNE LISTED AS 032546
                               ;        AT LOCATION 32546 YOU FIND THE ROUTINE $KERNEL
                               ;
                               ;        NOTE THAT CRF SYMBOLS ARE TRUCNATED TO 6 CHARACTERS
                               ;        SYMBOLIC NAMES GREATER THAT 6 CHARACTERS ARE USED SO I CAN
                               ;        REMEMBER WHAT THEY MEAN!
                               ;
                               ;        ALL GOOD TRAP ROUTINES RETURN VIA AN "RTI" INSTRUCTION
          104401               TYPEIT= 104401          ;;TTY TYPEOUT ROUTINE
          104402               TYPOC=  104402          ;;TYPE OCTAL NUMBER (WITH LEADING ZEROS)
          104403               TYPOS=  104403          ;;TYPE OCTAL NUMBER (NO LEADING ZEROS)
                               ;TYPON= 104404          ;;TYPE OCTAL NUMBER (AS PER LAST CALL)
          104405               TYPDS=  104405          ;;TYPE DECIMAL NUMBER (WITH SIGN)
                               ;TYPBN= 104406          ;;TYPE BINARY (ASCII) NUMBER

          104407               GTSWR=  104407          ;;GET SOFT-SWR SETTING
          104410               CKSWR=  104410          ;;TEST FOR CHANGE IN SOFT-SWR

          104411               RDCHR=  104411          ;;TTY TYPEIN CHARACTER ROUTINE
          104412               RDLIN=  104412          ;;TTY TYPEIN STRING ROUTINE
          104413               RDOCT=  104413          ;;READ AN OCTAL NUMBER FROM TTY
          104414               RDDEC=  104414          ;;READ A DECIMAL NUMBER FROM TTY

          104415               SAVREG= 104415          ;;SAVE R0-R5 ROUTINE
          104416               RESREG= 104416          ;;RESTORE R0-R5 ROUTINE

          104417               KERNEL= 104417          ;ENTER KERNEL MODE

          104420               ENERGIZE=104420         ;TURN ON MEMORY MANAGEMENT & TRAPS
          104421               DEENERGIZE=104421       ;TURN OFF MEMORY MANAGEMENT & TRAPS
          104422               KMAP=   104422          ;MAP KERNEL 1 TO 1

          104423               CACHON= 104423          ;TURN ON CACHE
          104424               CACHOFF=104424          ;TURN OFF CACHE
```

```

        104425            LOADCSR=104425          ;LOAD CORRECT CSR
        104426            READCSR=104426          ;READ CORRECT CSR

        104427            PERR01= 104427          ;PROGRAM DETECTED ERROR
        104430            PERR02= 104430          ;PROGRAM DETECTED ERROR
        104431            PERR03= 104431          ;PROGRAM DETECTED ERROR
        104432            PERR04= 104432          ;PROGRAM DETECTED ERROR
        104433            PERR07= 104433          ;PROGRAM DETECTED ERROR
        104434            PERR10= 104434          ;PROGRAM DETECTED ERROR
        104435            PERR11= 104435          ;PROGRAM DETECTED ERROR
        104436            PERR12= 104436          ;PROGRAM DETECTED ERROR
        104437            PERR13= 104437          ;PROGRAM DETECTED ERROR
        104440            PERR14= 104440          ;PROGRAM DETECTED ERROR
        104441            PERR15= 104441          ;PROGRAM DETECTED ERROR
        104442            PERR16= 104442          ;PROGRAM DETECTED ERROR
        104443            PERR17= 104443          ;PROGRAM DETECTED ERROR
        104444            PERR20= 104444          ;PROGRAM DETECTED ERROR
        104445            PERR21= 104445          ;PROGRAM DETECTED ERROR
        104446            PERR22= 104446          ;PROGRAM DETECTED ERROR
        104447            PERR23= 104447          ;PROGRAM DETECTED ERROR
        104450            PERR24= 104450          ;PROGRAM DETECTED ERROR
        104451            PERR25= 104451          ;PROGRAM DETECTED ERROR
        104452            PERR26= 104452          ;PROGRAM DETECTED ERROR
        104453            PERR27= 104453          ;PROGRAM DETECTED ERROR
        104454            PERR30= 104454          ;PROGPAM DETECTED ERROR
        104455            PERR31= 104455          ;PROGRAM DETECTED ERROR
        104456            PERR32= 104456          ;PROGRAM DETECTED ERROR
        104457            PERR33= 104457          ;PROGRAM DETECTED ERROR
        104460            PERR34= 104460          ;PROGRAM DETECTED ERROR
        104461            PERR35= 104461          ;PROGRAM DETECTED ERROR
        104462            PERR36= 104462          ;PROGRAM DETECTED ERROR
        104463            PERR37= 104463          ;PROGRAM DETECTED ERROR
        104464            PERR40= 104464          ;PROGRAM DETECTED ERROR
        104465            PERR41= 104465          ;PROGRAM DETECTED ERROR
        104466            PERR42= 104466          ;PROGRAM DETECTED ERROR
        104467            PERR43= 104467          ;PROGRAM DETECTED ERROR

        104470            ECCDIS= 104470          ;DISABLE ECC ON ALL CSR'S
        104471            ECC1DIS=104471          ;DISABLE ECC ON 1 SELECTED CSR
        104472            ECCINIT=104472          ;INITIALIZE ALL ECC CSR'S
        104473            ECC1INIT=104473         ;INITIALIZE 1 SELECTED ECC CSR
        104474            CBCSR=  104474          ;WRITE GENERATED CHECKBITS IN ALL CSR'S
        104475            CB1CSR= 104475          ;WRITE GENERATED CHECKBITS IN 1 SELECTED CSR
        104476            WASSBE= 104476          ;WAS THERE A SBE ON ANY CSR?
        104477            WAS1SBE=104477          ;WAS THERE A SBE ON 1 SELECTED CSR?
        104500            WASDBE= 104500          ;WAS THERE A DBE ON ANY CSR?
        104501            WAS1DBE=104501          ;WAS THERE A DBE ON 1 SELECTED CSR?
        104502            CLRCSR= 104502          ;CLEAR ALL CSR'S
        104503            CLR1CSR=104503          ;CLEAR 1 SELECTED CSR
        104504            CHKDIS= 104504          ;DISABLE ECC & WRITE CHECKBITS FROM ALL CSR'S
        104505            CHK1DIS=104505          ;DISABLE ECC & WRITE CHECKBITS FROM 1 SELECTED CSR
        104506            ENASBE= 104506          ;ENABLE TRAPS ON SBE'S FROM ALL CSR'S
        104507            ENA1SBE=104507          ;ENABLE TRAPS ON SBE'S FROM 1 SELECTED CSR
        104510            TSTREAD=104510          ;TEST LOC (R1) & TST FOR SBE (WITHOUT FETCHES)
        104511            INVALID=104511          ;INVALIDATE BACKGROUND PATTERN ON "BANK"
        104512            ERRGEN =104512          ;CHECK ERROR ADDRESS
```

```
                                  .SBTTL  DEFINE  BASIC PDP11 STUFF

                          ;*INITIAL ADDRESS OF THE STACK POINTER
        002000            STACK=  2000            ;;FIRST ADDRESS OF THE STACK
        002000            KERSTK= STACK           ;;KERNEL STACK
        000740            SUPSTK= 740             ;;SUPERVISOR STACK
        000700            USESTK= 700             ;;USER STACK
        104000            ERROR=EMT               ;;BASIC DEFINITION OF ERROR CALL
        000004            SCOPE=IOT               ;;BASIC DEFINITION OF SCOPE CALL
        177776            PSW=    177776          ;;PROCESSOR STATUS WORD
                          ;STKLMT=177774          ;;STACK LIMIT REGISTER
                          ;PIRQ=  177772          ;;PROGRAM INTERRUPT REQUEST REGISTER
        177570            DSWR=   177570          ;;HARDWARE SWITCH REGISTER
        177570            DDISP=  177570          ;;HARDWARE DISPLAY REGISTER
        177546            LKS=    177546          ;;LINE CLOCK (KW11-L) STATUS REGISTER

                          ;*MISCELLANEOUS DEFINITIONS
        000011            HT=     11              ;;CODE FOR HORIZONTAL TAB
        000012            LF=     12              ;;CODE LINE FEED
        000015            CR=     15              ;;CODE CARRIAGE RETURN
        000200            CRLF=   200             ;;CODE FOR CARRIAGE RETURN-LINE FEED
        000007            MFPT=   7               ;;CODE FOR PROCESSOR TYPE INSTRUCTION

                          ;*GENERAL PURPOSE REGISTER DEFINITIONS
                          ;SP=R6                  ;;STACK POINTER
                          ;KSP=SP                 ;;KERNEL STACK POINTER
        000006            SSP=SP                  ;;SUPERVISOR STACK POINTER
        000006            USP=SP                  ;;USER STACK POINTER
                          ;PC=R7                  ;;PROGRAM COUNTER

                          ;*"SWITCH REGISTER" SWITCH DEFINITIONS
        100000            SW15=   100000
        040000            SW14=   40000
        020000            SW13=   20000
        010000            SW12=   10000
        004000            SW11=   4000
        002000            SW10=   2000
        001000            SW9=    1000
        000400            SW8=    400
        000200            SW7=    200
        000100            SW6=    100
        000040            SW5=    40
        000020            SW4=    20
        000010            SW3=    10
        000004            SW2=    4
        000002            SW1=    2
        000001            SW0=    1

                          ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
        100000            BIT15=  100000
        040000            BIT14=  40000
        020000            BIT13=  20000
        010000            BIT12=  10000
        004000            BIT11=  4000
        002000            BIT10=  2000
        001000            BIT9=   1000
        000400            BIT8=   400
```

```
000200                  BIT7=   200
000100                  BIT6=   100
000040                  BIT5=   40
000020                  BIT4=   20
000010                  BIT3=   10
000004                  BIT2=   4
000002                  BIT1=   2
000001                  BIT0=   1

                        ;*BASIC "CPU" TRAP VECTOR ADDRESSES
000004                  ERRVEC= 4                       ;;TIME OUT AND OTHER ERRORS
000010                  RESVEC= 10                      ;;RESERVED AND ILLEGAL INSTRUCTIONS
                        ;TBITVEC=14                     ;;"T" BIT
                        ;TRTVEC=         14                     ;;TRACE TRAP
                        ;BPTVEC=         14                     ;;BREAKPOINT TRAP (BPT)
000020                  IOTVEC= 20                      ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
000024                  PWRVEC= 24                      ;;POWER FAIL
000030                  EMTVEC= 30                      ;;EMULATOR TRAP (EMT) **ERROR**
000034                  TRAPVEC=34                      ;;"TRAP" TRAP
000060                  TKVEC=  60                      ;;TTY KEYBOARD VECTOR
                        ;TPVEC= 64                      ;;TTY PRINTER VECTOR
                        ;LKVEC= 100                     ;;LINE CLOCK (KW11-L) VECTER
000114                  CACHVEC=114                     ;;CACHE ERROR INTERRUPT VECTOR
000114                  PARVEC=CACHVEC
                        ;PIRQVEC=240                    ;;PROGRAM INTERRUPT REQUEST VECTOR
000250                  MMVEC=  250                     ;;MEMORY MANAGEMENT VECTOR
                                .SBTTL  DEFINE  CACHE REGISTERS
                        ;MEMERR = 177744                        ;;CACHE ERROR REGISTER
177746                  CONTRL = 177746                         ;;MEMORY CONTROL REGISTER
177750                  MAINT  = 177750                         ;;MEMORY MAINTENENCE REGISTER
                        ;HITMIS = 177752                        ;;HIT MISS REGISTER "1" IMPLIES HIT IN CACHE
177754                  DATARG  = 177754                        ;;DATA REGISTER

                                .SBTTL  DEFINE  CPU REGISTERS
177766                  CPUERR = 177766                 ;;CPU ERROR REGISTER HOLDS CONDITION THAT CAUSED

                                .SBTTL  DEFINE  MEMORY MANAGEMENT REGISTERS
                                ;*MEMORY MANAGEMENT STATUS REGISTER ADDRESSES
177572                  MMR0=   177572
177574                  MMR1=   177574
177576                  MMR2=   177576
172516                  MMR3=   172516

                                ;*USER "I" PAGE DESCRIPTOR REGISTERS
177600                  UIPDR0= 177600
                        ;UIPDR1=         177602
                        ;UIPDR2=         177604
                        ;UIPDR3=         177606
                        ;UIPDR4=         177610
                        ;UIPDR5=         177612
                        ;UIPDR6=         177614
                        ;UIPDR7=         177616

                                ;*USER "D" PAGE DESCRIPTOR REGISTORS
                        ;UDPDR0=         177620
                        ;UDPDR1=         177622
                        ;UDPDR2=         177624
```

```
                    ;UDPDR3=        177626
                    ;UDPDR4=        177630
                    ;UDPDR5=        177632
                    ;UDPDR6=        177634
                    ;UDPDR7=        177636

                            ;*USER "I" PAGE ADDRESS REGISTERS
177640              FASTCITY=UIPAR0
177640              UIPAR0= 177640          ;PATTERN PROGRAM SPACE
177642              UIPAR1= 177642          ;PATTERN PROGRAM SPACE
177644              UIPAR2= 177644          ;PATTERN PROGRAM SPACE
177646              UIPAR3= 177646          ;PATTERN PROGRAM SPACE
177650              UIPAR4= 177650          ;PATTERN PROGRAM SPACE
177652              UIPAR5= 177652          ;PATTERN PROGRAM SPACE
177654              UIPAR6= 177654          ;PATTERN PROGRAM SPACE
                    ;UIPAR7=        177656          ;PATTERN PROGRAM SPACE

                            ;*USER "D" PAGE ADDRESS REGISTERS
177660              UDPAR0= 177660          ;PATTERN PROGRAM SPACE
                    ;UDPAR1=        177662          ;PATTERN PROGRAM SPACE
                    ;UDPAR2=        177664          ;PATTERN PROGRAM SPACE
                    ;UDPAR3=        177666          ;PATTERN PROGRAM SPACE
                    ;UDPAR4=        177670          ;PATTERN PROGRAM SPACE
                    ;UDPAR5=        177672          ;PATTERN PROGRAM SPACE
                    ;UDPAR6=        177674          ;PATTERN PROGRAM SPACE
177676              UDPAR7= 177676          ;PATTERN PROGRAM SPACE

                            ;*SUPERVISOR "I" PAGE DESCRIPTOR REGISTERS
172200              SIPDR0= 172200
                    ;SIPDR1=        172202
                    ;SIPDR2=        172204
                    ;SIPDR3=        172206
                    ;SIPDR4=        172210
                    ;SIPDR5=        172212
                    ;SIPDR6=        172214
                    ;SIPDR7=        172216

                            ;*SUPERVISOR "D" PAGE DESCRIPTOR REGISTERS
                    ;SDPDR0=        172220
                    ;SDPDR1=        172222
                    ;SDPDR2=        172224
                    ;SDPDR3=        172226
                    ;SDPDR4=        172230
                    ;SDPDR5=        172232
                    ;SDPDR6=        172234
                    ;SDPDR7=        172236

                            ;*SUPERVISOR "I" PAGE ADDRESS REGISTERS
172240              SIPAR0= 172240
                    ;SIPAR1=        172242
                    ;SIPAR2=        172244
172246              SIPAR3= 172246          ;TEST AREA
                    ;SIPAR4=        172250          ;TEST AREA
172252              SIPAR5= 172252                  ;TEST AREA
172254              SIPAR6= 172254                  ;TEST AREA
                    ;SIPAR7=        172256

```

```

                                   ;*SUPERVISOR "D" PAGE ADDRESS REGISTERS
        172260          SDPAR0= 172260
                        ;SDPAR1=          172262
                        ;SDPAR2=          172264
                        ;SDPAR3=          172266
                        ;SDPAR4=          172270
        172272          SDPAR5= 172272
        172274          SDPAR6= 172274
        172276          SDPAR7= 172276

                                   ;*KERNEL "I" PAGE DESCRIPTOR REGISTERS
        172300          KIPDR0= 172300
                        ;KIPDR1=          172302
                        ;KIPDR2=          172304
                        ;KIPDR3=          172306
                        ;KIPDR4=          172310
                        ;KIPDR5=          172312
                        ;KIPDR6=          172314
                        ;KIPDR7=          172316

                                   ;*KERNEL "D" PAGE DESCRIPTOR REGISTERS
                        ;KDPDR0=          172320
                        ;KDPDR1=          172322
                        ;KDPDR2=          172324
                        ;KDPDR3=          172326
                        ;KDPDR4=          172330
                        ;KDPDR5=          172332
                        ;KDPDR6=          172334
                        ;KDPDR7=          172336

                                   ;*KERNEL "I" PAGE ADDRESS REGISTERS
        172340          KIPAR0= 172340
                        ;KIPAR1=          172342
                        ;KIPAR2=          172344
                        ;KIPAR3=          172346
        172350          KIPAR4= 172350
        172352          KIPAR5= 172352
        172354          KIPAR6= 172354
                        ;KIPAR7=          172356

                                   ;*KERNEL "D" PAGE ADDRESS REGISTERS
        172360          KDPAR0= 172360
                        ;KDPAR1=          172362
                        ;KDPAR2=          172364
                        ;KDPAR3=          172366
                        ;KDPAR4=          172370
                        ;KDPAR5=          172372
        172374          KDPAR6= 172374
        172376          KDPAR7= 172376

```

```
                                             .SBTTL  DEFINE  UNIBUS MAP REGISTERS
                                             ;*THE LOWER 16 BITS OF THE MAP REGISTERS ARE LABELED 'MAPLXX'
                                             ;*THE UPPER 6 BITS OF THE MAP REGISTERS ARE LABELED 'MAPHXX'
        170200                      MAPL0 = 170200
        170202                      MAPH0 = 170202
        170204                      MAPL1 = 170204
                                    ;MAPH1 = 170206
                                    ;MAPL2 = 170210
                                    ;MAPH2 = 170212
                                    ;MAPL3 = 170214
                                    ;MAPH3 = 170216
                                    ;MAPL4 = 170220
                                    ;MAPH4 = 170222
                                    ;MAPL5 = 170224
                                    ;MAPH5 = 170226
                                    ;MAPL6 = 170230
                                    ;MAPH6 = 170232
                                    ;MAPL7 = 170234
                                    ;MAPH7 = 170236
                                    ;MAPL10 = 170240
                                    ;MAPH10 = 170242
                                    ;MAPL11 = 170244
                                    ;MAPH11 = 170246
                                    ;MAPL12 = 170250
                                    ;MAPH12 = 170252
                                    ;MAPL13 = 170254
                                    ;MAPH13 = 170256
                                    ;MAPL14 = 170260
                                    ;MAPH14 = 170262
                                    ;MAPL15 = 170264
                                    ;MAPH15 = 170266
                                    ;MAPL16 = 170270
                                    ;MAPH16 = 170272
                                    ;MAPL17 = 170274
                                    ;MAPH17 = 170276
                                    ;MAPL20 = 170300
                                    ;MAPH20 = 170302
                                    ;MAPL21 = 170304
                                    ;MAPH21 = 170306
                                    ;MAPL22 = 170310
                                    ;MAPH22 = 170312
                                    ;MAPL23 = 170314
                                    ;MAPH23 = 170316
                                    ;MAPL24 = 170320
                                    ;MAPH24 = 170320
                                    ;MAPL25 = 170324
                                    ;MAPH25 = 170326
                                    ;MAPL26 = 170330
                                    ;MAPH26 = 170332
                                    ;MAPL27 = 170334
                                    ;MAPH27 = 170336
                                    ;MAPL30 = 170340
                                    ;MAPH30 = 170342
                                    ;MAPL31 = 170344
                                    ;MAPH31 = 170346
                                    ;MAPL32 = 170350
                                    ;MAPH32 = 170352
```

```
                                ;MAPL33 = 170354
                                ;MAPH33 = 170356
                                ;MAPL34 = 170360
                                ;MAPH34 = 170362
                                ;MAPL35 = 170364
                                ;MAPH35 = 170366
                                ;MAPL36 = 170370
                                ;MAPH36 = 170372
                                ;MAPL37 = 170374
                                ;MAPH37 = 170376

                                        .SBTTL  DEFINE  SOFTWARE SWITCH & DISPLAY REGISTERS
        000174                  DISPREG=174
        000176                  SWREG=  176

                                        .SBTTL  DEFINE  CONTROL STATUS REGISTERS
        172100                  CSRADD=172100

                                        .SBTTL  DEFINE  PARAMETERS
        060000                  FIRST=60000                     ;START OF THE 16K TEST PATTERN AREA
        157776                  LAST=157776                     ;END OF THE 16K TEST PATTERN AREA
        040000                  SIZE=40000                      ;SIZE OF THE 16K TEST PATTERN AREA (FOR SOB INSTRUCTIONS)
```

```
                .LIST   MD                              ;BE NICE TO SEE MY DEFINITIONS
                .SBTTL  MACRO   FATAL
;*********************** FATAL ***************************
;
;FATAL IS USED TO REPORT FATAL ERRORS (ERRORS THAT PREVENT
;       THE PROGRAM FROM CONTINUING).
;
;*********************************************************
                .MACRO  FATAL   ARG                     ;***MACRO***MACRO***MACRO***
                .NLIST
                .DSABL  CRF
                .IIF DF LST$$ .LIST  ME
                .ENABL  CRF
                .LIST
                INC     FATAL$                          ;SET FATAL INDICATOR
                ERROR   +ARG
                .DSABL  CRF
                .IIF DF LST$$ .NLIST  ME
                .ENABL  CRF
                .ENDM   FATAL

                .SBTTL  MACRO   TYPE
                .MACRO  TYPE    ARG
                .NLIST
                .DSABL  CRF
                .IIF DF LST$$ .LIST  ME
                .ENABL  CRF
                .LIST
                .IF B ARG
                TYPEIT
                .IFF
                TYPEIT  ,ARG
                .ENDC
                .DSABL  CRF
                .IIF DF LST$$ .NLIST  ME
                .ENABL  CRF
                .ENDM   TYPE
                .SBTTL  MACRO   NEWTST
;*********************** NEWTST **********************
;NEWTST IS USED AS THE FIRST INSTRUCTION OF A TEST.
;IT WILL:
;1)  GENERATE A TEST NUMBER FOR THE LABEL OF THIS TEST
;2)  PUT STARS BEFORE AND AFTER A MESSAGE
;ARGUMENTS
;1)  ASCII  --  THIS IS THE MESSAGE THAT WILL APPEAR
;               ON THE LISTING
;2)  ICOUNT --  IF NON-BLANK AND BIT 11 OF $SWR = 1 IT WILL BE
;               THE NUMBER OF ITERATIONS TO MAKE ON THIS TEST
;3)  RETURN --  IF NON-BLANK WILL BE THE ADDRESS TO
;               WHICH THE NEXT SCOPE STATEMENT WILL
;               LOOP BACK TO.
;4)  COMAND --  IF NON-BLANK WILL BE THE FIRST
;               INSTRUCTION OF THE TEST
;               IF BLANK SCOPE WILL BE THE
;               FIRST INSTRUCTION
;******************************************************
                .MACRO  NEWTST  ASCII,ICOUNT,RETURN,COMAND
                $STN=1
                $NWTST=0
                .NLIST  MC
                .IF B <COMAND>
                $$NEWTEST       \$TN,<ASCII>,SCOPE
                .IFF
                $$NEWTEST       \$TN,<ASCII>,<COMAND>
                .ENDC
                .NLIST
                .LIST   ME
                .LIST
                .IF NE 4000&$SWR
                .IF NB ICOUNT
                .IF LE <ICOUNT-1>
                MOV     #1,$TIMES       ;;DO 1 ITERATION
                .IFF
                MOV     #ICOUNT,$TIMES  ;;DO ICOUNT ITERATIONS
                .ENDC
                .ENDC
                .IF NB RETURN
                MOV     #RETURN,$LPADR  ;;SET SCOPE LOOP ADDRESS
                .ENDC
                .ENDC
                .NLIST
                .LIST   MC
                .LIST
                .NLIST  ME
                .ENDM   NEWTST
```

```
        .SBTTL  MACRO   $$NEWTEST
        .MACRO  $$NEWTEST       A,ASC,COMND
        .IRP    ASCI,<ASC>
        .IF EQ $NWTST
        $NWTST=1
        .SBTTL  T'A'    ASCI
        .NLIST
        .LIST   ME
        .LIST
;*******************************************************************************
;*TEST A        ASCI
        .IFF
        ASCI
        .ENDC
        .ENDM
;*******************************************************************************
TST'A:  COMND
        .NLIST  ME
        $TN=$TN+1
        .ENDM   $$NEWTEST

        .SBTTL  MACRO   SUBTST
;************** SUBTST *******************************
;
;THIS MACRO WILL FORMAT A SUBTEST HEADING WITH STARS
;A .SBTTL WILL BE FORCED & .NLISTED FOR THE TABLE OF CONTENTS.
;
;ARGUMENT:
;1) TXT --      THIS IS THE MESSAGE THAT WILL APPEAR IN THE TABLE OF CONTENTS & LISTING.
;
;EXAMPLE:       SUBTST  <<THIS IS A FUN SUBTST>>
;
;*******************************************************

        .MACRO  SUBTST  ASCII
        .NLIST  MC
        $SUBTST <ASCII>
        .LIST   MC
        .ENDM   SUBTST

        .SBTTL  MACRO   $SUBTST
        .MACRO  $SUBTST ASC
        .IRP    ASCI,<ASC>
        .SBTTL  ASCI
        .NLIST
        .LIST   ME
        .LIST
;*******************************************************************************
;*SUBTEST       ASCI
        .ENDM
;*******************************************************************************
        .NLIST  ME
        .ENDM   $SUBTST
        .SBTTL  MACRO   TYPOCT
;********************* TYPOCT ********************
;
;TYPOCT IS USED TO CHANGE A BINARY NUMBER
;       TO A 6 DIGIT OCTAL NUMBER AND TYPE IT
;
;ARGUMENTS:
;
;1)     NUM     THE NUMBER TO BE TYPED
;
;2)     REMARK  ALLOWS A COMMENT TO BE MADE
;
;ROUTINES REQUIRED
;
;1)     CONVERT BINARY TO OCTAL AND TYPE (.$TYPOCT)
;
;2)     TYPE AN ASCIZ STRING (.$TYPE)
;
;EXAMPLES:
;
;1)     TYPOCT  HILMT,<TYPES THE CONTENTS OF HILMT>
;2)     TYPOCT  #5,<TYPES ' 000005'>
;
;*******************************************************
 
        .MACRO  TYPOCT  NUM,REMARK
        .NLIST
        .DSABL  CRF
        .IIF DF LST$$ .LIST  ME
        .ENABL  CRF
        .LIST
        MOV     NUM,-(SP)               ;;SAVE NUM FOR TYPEOUT
        .IIF NB <REMARK>,                                     ;;REMARK
        TYPOC                           ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
        .DSABL  CRF
        .IIF DF LST$$ .NLIST  ME
        .ENABL  CRF
        .ENDM   TYPOCT
        .SBTTL  MACRO   TYPOCS
;********************* TYPOCS ********************
;
;TYPOCS IS USED TO CHANGE A BINARY NUMBER TO AN OCTAL
        NUMBER AND TYPE 1 TO 6 DIGITS
;       WITH OR WITHOUT LEADING ZEROS.
;
;ARGUMENTS:
;
;1)     NUM     NUMBER TO BE TYPED
;
;2)     REMARK  ALLOWS A COMMENT TO BE MADE
;
;3)     N       NUMBER OF DIGITS (1 TO 6) TO BE TYPED
;
;4)     Z       BLANK=SUPPRESS LEADING ZEROS (TYPES SPACES)
;               NON-BLANK=TYPE LEADING ZEROS
;
;ROUTINES REQUIRED
;
;1)     CONVERT BINARY TO OCTAL AND TYPE (.$TYPOCT)
;
;2)     TYPE AN ASCIZ STRING (.$TYPE)
;
;EXAMPLES:
;1)     TYPOCS  #12345,<TYPES ''5''>,1
;2)     TYPOCS  #004,<TYPES ''04''>,2,X
;3)     TYPOCS  #004,<TYPES '' 4''>,2
;
;*****************************************************************

        .MACRO  TYPOCS  NUM,REMARK,N,Z
        .NLIST
        .DSABL  CRF
        .IIF DF LST$$ .LIST  ME
        .ENABL  CRF
        .LIST
        MOV     NUM,-(SP)       ;;SAVE NUM FOR TYPEOUT
        .IIF NB <REMARK>,                               ;;REMARK
        TYPOS                   ;;GO TYPE--OCTAL ASCII
        .IF NB N
        .BYTE   N               ;;TYPE N DIGIT(S)
        .IFF
        .BYTE   6               ;;TYPE 6 DIGITS
        .ENDC
        .IF NB Z
        .BYTE   1               ;;TYPE LEADING ZEROS
        .IFF
        .BYTE   0               ;;SUPPRESS LEADING ZEROS
        .ENDC
        .DSABL  CRF
        .IIF DF LST$$ .NLIST  ME
        .ENABL  CRF
        .ENDM   TYPOCS
        .SBTTL  MACRO   TYPDEC
;******************** TYPDEC ********************
;
;TYPDEC IS USE TO CHANGE A BINARY NUMBER TO A SIGNED
;       DECIMAL NUMBER AND TYPE IT REPLACING LEADING ZERO
;       WITH SPACES.
;NOTE:  IF THE NUMBER IS NEGATIVE A
;       MINUS SIGN WILL BE TYPED.
;
;ARGUMENTS:
;
;1)     NUM     NUMBER TO BE TYPED
;
;2)     REMARK ALLOWS A COMMENT TO BE MADE
;
;ROUTINES REQUIRED
;
;1)     CONVERT BINARY TO DECIMAL AND TYPE (.$TYPDEC)
;
;2)     TYPE AN ASCIZ STRING (.$TYPE)
;
;EXAMPLES
;
;1)     TYPDEC  SIZE,<TYPE THE CONTENTS OF SIZE>
;2)     TYPDEC  #-10.,<TYPE A MINUS TEN>
;
;****************************************************************

        .MACRO  TYPDEC  NUM,REMARK
        .NLIST
        .DSABL  CRF
        .IIF DF LST$$ .LIST  ME
        .ENABL  CRF
        .LIST
        MOV     NUM,-(SP)       ;;SAVE NUM FOR TYPEOUT
        .IIF NB <REMARK>,                               ;;REMARK
        TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
        .DSABL  CRF
        .IIF DF LST$$ .NLIST  ME
        .ENABL  CRF
        .ENDM   TYPDEC
        .SBTTL  MACRO   BMOV
;***************** BMOV ****************************
;
; THIS MACRO MOVES A BLOCK OF DATA.
;
;ARGUEMENTS:
;
;1)     FROMHERE        THE FIRST ADDRESS OF THE SOURCE BLOCK.
;
;2)     TOHERE          THE FIRST ADDRESS OF THE DESTINATION BLOCK.
;                       IF BLANK THE 1ST ADDRESS OF THE USER INSTRUCTION
;                       PAR'S IS USED (FASTCITY).
;
;3)     SIZE            THE SIZE OF THE SOURCE BLOCK.
;                       IF BLANK A 16 WORD TRANSFER IS ASSUMED.
;                       ''WHY DEFAULT TO 16 WORDS?'' YOU ASK!
;                       ''BECAUSE THAT'S HOW MANY WORDS TO THE USER PAR
;                       REGISTERS & THAT'S WHERE I INTEND TO MOVE LOTS
;                       OF STUFF.'' I REPLY!
;
;***************************************************

        .MACRO  BMOV    FROMHERE,TOHERE,SIZE
          .IF B TOHERE
            .NLIST
            .DSABL      CRF
            .IIF DF LST$$ .LIST  ME
            .ENABL      CRF
            .LIST
            JSR R5,BLOCK1
            FROMHERE
            .DSABL      CRF
            .IIF DF LST$$ .NLIST  ME
            .ENABL      CRF
            .MEXIT
          .ENDC
          .IF B SIZE
            .NLIST
            .DSABL      CRF
            .IIF DF LST$$ .LIST  ME
            .ENABL      CRF
            .LIST
            JSR R5,BLOCK2
            TOHERE
            FROMHERE
            .DSABL      CRF
            .IIF DF LST$$ .NLIST  ME
            .ENABL      CRF
            .MEXIT
          .IFF
            .NLIST
            .DSABL      CRF
            .IIF DF LST$$ .LIST  ME
            .ENABL      CRF
            .LIST
            JSR R5,BLOCK3
            SIZE
```

```
    TOHERE
    FROMHERE
    .DSABL      CRF
    .IIF DF LST$$ .NLIST  ME
    .ENABL      CRF
  .ENDC
.ENDM   BMOV
```

```
        .SBTTL  MACRO   MAP
;***************** MAP ******************************
;
; THIS MACRO MAPS A MEMORY BANK (16K) INTO THE
; TEST PATTERN AREA (SUPERVISOR VIRTUAL (60000-157777)).
;
;ARGUEMENTS:
;
;1)     BANK    THE BANK OF 16K WORDS TO BE MAPPED.
;               THERE ARE 120 BANKS OF 16K WORDS
;
;EXAMPLES
;
;       MAP     LOC             ;LOCATION "LOC" CONTAINS THE # OF THE BANK TO MAP
;
;       MAP     #28.            ;BANK 34 (OCTAL) WILL BE MAPPED
;
;***************************************************

        .MACRO  MAP     BANK
        PUSH    R3
        .NLIST
        .DSABL  CRF
        .IIF DF LST$$ .LIST  ME
        .ENABL  CRF
        .LIST
        .IF B BANK
        MOV     #120.,R3
        .IFF
        MOV     BANK,R3
        .ENDC
        CALL    MAPPER
        .DSABL  CRF
        .IIF DF LST$$ .NLIST  ME
        .ENABL  CRF
        POP     R3
        .ENDM   MAP
        .SBTTL  MACRO   SUPERVISOR
;****************** SUPERVISOR *********************
;
; THIS MACRO SWITCHES TO SUPERVISOR MODE.
;
;ARGUEMENTS: NONE.
;
;***************************************************

        .MACRO  SUPERVISOR
        .NLIST
        .DSABL  CRF
        .IIF DF LST$$ .LIST  ME
        .ENABL  CRF
        .LIST
        BIS     #BIT14,PSW                              ;GO TO SUPERVISOR MODE
        .DSABL  CRF
        .IIF DF LST$$ .NLIST  ME
        .ENABL  CRF
        .ENDM   SUPERVISOR

        .SBTTL  MACRO   USER
;***************** USER ****************************
;
; THIS MACRO SWITCHES TO USER MODE.
;
;ARGUEMENTS: NONE.
;
;***************************************************

        .MACRO  USER
        .NLIST
        .DSABL  CRF
        .IIF DF LST$$ .LIST  ME
        .ENABL  CRF
        .LIST
        BIS     #BIT15!BIT14,PSW                        ;GO TO USER MODE
        .DSABL  CRF
        .IIF DF LST$$ .NLIST  ME
        .ENABL  CRF
        .ENDM   USER
        .SBTTL  MACRO   TESTAREA
;***************** TESTAREA ************************
;
; THIS MACRO SWITCHES TO THE SPECIFIED TEST MODE.
;
;ARGUEMENTS: NONE.
;
;***************************************************

        .MACRO  TESTAREA
        .NLIST
        .DSABL  CRF
        .IIF DF LST$$ .LIST  ME
        .ENABL  CRF
        .LIST
        BIS     TESTMODE,PSW                    ;GO TO SYSTEM TEST MODE
        .DSABL  CRF
        .IIF DF LST$$ .NLIST  ME
        .ENABL  CRF
        .ENDM   TESTAREA
        .SBTTL  MACRO   SET4 & RES4
;***************** SET4 & RES4 *********************
;
; THESE MACROS SET & RESTORE VECTOR 4(TIMEOUT TRAP)
;
; IN IT'S RESTORED MODE TRAPS ARE REPORTED AS SUCH.
;
;ARGUEMENTS:    LOC     ;THE LOCATION TO VECTOR TO (ONLY USED IN "SET4" NOT "RES4")
;
;I USE THE SET4 AND RES4 MACROS AROUND CODE THAT I EXPECT TO TRAP TO 4
;LIKE LOOKING FOR ALL POSSIBLE CSR'S AND ETC.  WHENEVER CODE IS NOT
;SURROUNDED BY SET4 AND RES4 THEN ANY TRAPS TO 4 WILL CAUSE AN ERROR
;PRINTOUT THAT SAYS "UNEXPECTED TRAP TO 4" AND ALL THE ASSOCIATED REGISTER JUNK
;****************************************************

        .MACRO  SET4    ARG
        .NLIST
        .DSABL  CRF
        .IIF DF LST$$ .LIST  ME
        .ENABL  CRF
        .LIST
        MOV     ARG,4
        .DSABL  CRF
        .IIF DF LST$$ .NLIST  ME
        .ENABL  CRF
        .ENDM   SET4

        .MACRO  RES4
        .NLIST
        .DSABL  CRF
        .IIF DF LST$$ .LIST  ME
        .ENABL  CRF
        .LIST
        MOV     #TIMEOUT,4
        CMP     #1,PROTYP               ;IS THIS AN 11/44?
        BNE     101$                    ;BRANCH IF NOT
        CLR     CPUERR                  ;CLEAR OUT THE CPU ERROR REGISTER BITS
101$:
                                        ;THAT A EXPECTED TRAP COULD HAVE SET
        .DSABL  CRF
        .IIF DF LST$$ .NLIST  ME
        .ENABL  CRF
        .ENDM   RES4
```

```
                        .SBTTL  MACRO   DLEFT
                ;****************** DLEFT ****************************
                ;
                ; THIS MACRO DOES A DOUBLE WORD LEFT SHIFT
                ;
                ;ARGUEMENTS:     LOC      ;THE LOCATION TO BE SHIFTED LEFT (CARRY TO LOC+2)
                ;
                ;*****************************************************

                        .MACRO  DLEFT    ARG
                        .NLIST
                        .DSABL  CRF
                        .IIF DF LST$$ .LIST  ME
                        .ENABL  CRF
                        .LIST
                        ROL     ARG
                        ROL     ARG+2
                        .DSABL  CRF
                        .IIF DF LST$$ .NLIST  ME
                        .ENABL  CRF
                        .ENDM   DLEFT
                        .NLIST  MD              ;DON'T NEED TO SEE THEM ANY MORE
```

```
                                        .SBTTL  TRAP CATCHER
        000000                          .=0
000000  000000  000000                  .WORD   0,0
        000177                          .REPT   177             ;.WORD  .+2,HALT

                                        .SBTTL  ACT11 HOOKS
                                ;*THE HOOKS REQUIRED BY ACT11 ARE DEFINED AND SETUP BELOW:
                                ;*      DEFINITIONS:
                                ;*                      1)LOC.46        "END-OF-PASS" HOOK
                                ;*                              =ADDRESS OF END OF PASS ROUTINE
                                ;*                               MODIFIED BY ACT11.
                                ;*                      2)LOC.52        PROGRAM NEEDS HOOK
                                ;*                              BIT 15=1 PROGRAM SHOULD BE POWER
                                ;*                                      FAILED WHILE RUNNING
                                ;*                                    =0 NO POWER FAIL
                                ;*                              BIT 14=1 PROGRAM MEMORY SIZE DEPENDENT
                                ;*                                    =0 NOT MEMORY SIZE DEPENDENT
                                ;*                              BIT 13=1 PROGRAM REQUIRES MANUAL INTERVENTION
                                ;*                                    =0  MANUAL INTERVENTION NOT REQUIRED
                                ;*                              BITS 12-0  MUST BE ZERO'S
        000046                          .=46
000046  014430                          $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
        000052                          .=52
000052  000020                          .WORD   BIT4            ;;2)SET LOC.52 TO INDICATE MEMORY SIZE DEPENDANT
                                        .SBTTL  APT11 HOOKS
        000024                          .=24    ;;SET POWER FAIL TO POINT TO START OF PROGRAM
000024  000200                          200     ;;FOR APT START UP
        000042                          .=42
000042  002000                          STACK                   ;SO RT11 CAN START WITH RUN COMMAND
        000044                          .=44    ;;POINT TO APT INDIRECT ADDRESS PNTR.
000044  062566                          $APTHDR ;;POINT TO APT HEADER BLOCK
        000200                          .=200
000200  000437                  START3: BR      START1          ;"NORMAL" START
000202  000442                          BR      START2          ;RESTART (SAVE ERROR ACCOUNTING)
        000300                          .=300
000300  005037  002566          START1: CLR     RESTART
000304  000137  003630                  JMP     START
000310                          START2: SET     RESTART
000316  000137  003630                  JMP     START
        002000                          .=STACK
```

```
                                   .SBTTL  VARIABLES       INITIALIZED TO ZERO
                            ;*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
                            ;*USED IN THE PROGRAM.
002000                      $CMTAG:                        ;;START OF COMMON TAGS
002000  000000              SELONLY:0                      ;SELECT ONLY BANKS MARKED BY FIELD SERVICE MODE FLAG
002002  000000              DIAGFLAG:0                     ;SET FOR SHIFTING DIAGONAL TEST
002004  000000              KAMIKAZE:0                     ;SET FOR KAMIKAZE MODE TESTING
002006  000000              SKIPKAMI:0                     ;USED TO SKIP RESTORING KAMIKAZE MODE WHEN MODIFIED
                            ;NEXT TWO BYTES ARE DISPLAYED IN THE DISPLAY REGISTER
002010     000              $PATMAR:.BYTE   0              ;PATTERN NUMBER
002011     000              $BANK:  .BYTE   0              ;BANK & SIGN
002012     000              $ERFLG: .BYTE   0              ;;CONTAINS ERROR FLAG
002013     000              $ITEMB: .BYTE   0              ;;CONTAINS ITEM CONTROL BYTE
002014  000000              LASTERROR:.WORD 0              ;NUMBER OF ERRORS ON LAST PASS
002016  000000              ERRPC:  .WORD   0              ;CONTAINS PC OF ERROR FOR TYPEOUT
002020  000000              BADPC:  .WORD   0              ;CONTAINS PC OF ERROR
002022  000000              ERRSP:  .WORD   0              ;CONTAINS SP OF ERROR FOR TYPEOUT
002024  000000              BADSP:  .WORD   0              ;CONTAINS SP OF ERROR
002026  000000              ERRPSW: .WORD   0              ;CONTAINS PSW OF ERROR FOR TYPEOUT
002030  000000              BADPSW: .WORD   0              ;CONTAINS PSW OF ERROR
002032  000000              ADDRESS:.WORD   0              ;;CONTAINS ADDRESS OF 'BAD' DATA
002034  000000              PADDRESS:.WORD  0              ;ADDRESS OF PARITY ERROR
002036  000000  000000      PHYADD: .WORD   0,0            ;22 BIT PHYSICAL ADDRESS
002042  000000              GOOD:   .WORD   0              ;;CONTAINS 'GOOD' DATA
002044  000000              GOOD2:  .WORD   0              ;;CONTAINS 'GOOD2' DATA
002046  000000              GOOD3:  .WORD   0              ;;CONTAINS 'GOOD3' DATA
002050  000000              BAD:    .WORD   0              ;;CONTAINS 'BAD' DATA
002052  000000              BAD2:   .WORD   0              ;;CONTAINS 'BAD2' DATA
002054  000000              BAD3:   .WORD   0              ;;CONTAINS 'BAD3' DATA
002056  000000              BADXOR: .WORD   0              ;XOR OF GOOD & BAD = BAD BITS!
002060  000000              $AUTO:  .WORD   0              ;;AUTOMATIC MODE INDICATOR FOR APT,ACT, & XXDP
002062  000000              FATAL$: .WORD   0              ;FATAL ERROR INDICATOR
002064  000000              SKPERR: .WORD   0              ;USED TO SKIP ERROR MESSAGE IN "$ERRGEN"
002066  000000              NEMCNT: 0                      ;NON-EXISTANT MEMORY COUNTER (HOLES)
002070  000000              PARCNT: 0                      ;PARITY ERROR COUNTER
002072  000000              PATERR: 0                      ;PATTERN ERROR COUNTER
002074  000000              NOPAR:  0                      ;NO PARITY ERROR MODE INDICATOR
002076  000000              NONEM:  0                      ;NO NON-EXISTANT MEMORY (HOLES) MODE INDICATOR
002100  000000              BANK:   0                      ;MEMORY BANK UNDER TEST
002102  000000              BANKINDEX:0                    ;USED TO INDEX INTO CONFIG TABLE
002104  000000              CPUBIT: 0                      ;CONTAINS 1 BIT TO IDENTIFY CPU TO CONFIGURATION TABLE
002106  000000              MUT:    0                      ;MEMORY UNDER TEST FLAG
002110  000000              PATTERN:0                      ;PATTERN NUMBER UNDER TEST
002112  000000              KPFLAG: .WORD   0              ;BANK IS PROTECTED REGION OF ECC
002114  000000              ACFLAG: .WORD   0              ;BANK CAN BE ACCESSED BY THIS CPU
002116  000000              MKFLAG: .WORD   0              ;IF SET INDICATES MS11-M OR MF11S-K UNDER TEST
002120  000000              PFLAG:  .WORD   0              ;BANK IS IN PROGRAM SPACE
002122  000000              RRFLAG: .WORD   0              ;BANK IS WHERE PROGRAM RELOCATION IS REQUIRED TO TEST
002124  000000              RLFLAG: .WORD   0              ;PROGRAM IS RELOCATED FLAG
002126  000000              BMFLAG: .WORD   0              ;"BANK IS IDENTIFIED AS BAD MEMORY" FLAG
002130  000000              EUFLAG: .WORD   0              ;"BANK HAS EUB MEMORY" FLAG
002132  000000              TMFLAG: .WORD   0              ;"TYPE OF MEMORY TO TEST" FLAG; 0 = PARITY, 1 = ECC
002134  000000              INTFLAG:.WORD   0              ;"BANK IS INTERLEAVED" FLAG
002136  000000              INT64K: .WORD   0              ;"BANK IS 64K INTERLEAVED" FLAG
002140  000000              ABORTFLAG:.WORD 0              ;"ABORT OCCURED" FLAG
002142  000000              CTLKVEC:.WORD   0              ;HOLDS OLD KERNAL STACK POINTER IN CASE OF CNTL/K
002144  000000              CSR:    .WORD   0              ;DATA TO OR FROM CSR
```

```
   1155 002146  000000                 CSRNO:  0                         ;CSR ADDRESS NUMBER (4 LSB'S)
   1156 002150  000000                 OLDCSR: .WORD   0                 ;OLD CSR NUMBER(USED IN INH PTR TEST)
   1157                                        ;THESE LOCATIONS STORE GPR'S DURING SUPERVISOR TESTS
   1158 002152  000000                 SUPDR0: 0
   1159 002154  000000                 SUPDR1: 0
   1160 002156  000000                 SUPDR2: 0
   1161 002160  000000                 SUPDR3: 0
   1162 002162  000000                 SUPDR4: 0
   1163 002164  000000                 SUPDR5: 0
   1164 002166  000000                 SUPDR6: 0
   1165 002170  000000                 DUMMY:  0                         ;DUMMY LOCATION FOR ADDRESS PASSING
   1166                                ;THESE LOCATIONS STORE GPR'S & PSW DURING DETAILED ERROR PRINTOUTS
   1167 002172  000000                 DETR0:  0
   1168 002174  000000                 DETR1:  0
   1169 002176  000000                 DETR2:  0
   1170 002200  000000                 DETR3:  0
   1171 002202  000000                 DETR4:  0
   1172 002204  000000                 DETR5:  0
   1173 002206  000000                 DETSP:  0
   1174 002210  000000                 DETPSW: 0
   1175 002212  000000                 DETFLAG:0                         ;DETAILED REPORT FLAG
   1176 002214  000000                 CONTFLAG:0                        ;CSR'S HAVE BEEN TESTED FLAG
   1177 002216  000000                 TOTCSRS:.WORD   0                 ;1 BIT PER EXISTING CSR, EG-
   1178                                                                  ;CSR 0 REPRESENTED BY BIT 15, ETC.
   1179 002220  000000                 CSRFIRST:.WORD  0                 ;FIRST ADDRESS UNDER CONTROL OF THIS CSR
   1180 002222  000000                 CSRLAST: .WORD  0                 ;
   1181 002224  000000                 CSRFBANK:.WORD  0                 ;
   1182 002226  000000                 CSRLBANK:.WORD  0                 ;
   1183 002230  000000                 CSRINT:   .WORD 0                 ;
   1184 002232  000000                 SPLTCSR:  .WORD 0                 ;
   1185 002234  000000  000000         DATBUF: .WORD   0,0               ;TWO WORD DATA BUFFER
   1186 002240  000000  000000         TSTDAT: .WORD   0,0               ;TWO WORD TEST DATA
   1187 002244  000000  000000         SBEMSK: .WORD   0,0               ;TWO WORD SINGLE BIT ERROR MASK
   1188 002250  000000  000000         DBEMSK: .WORD   0,0               ;TWO WORD DOUBLE BIT ERROR MASK
   1189 002254  000000                 SUPDOADD:.WORD  0                 ;ADDRESS OF SUBROUTINE TO EXECUTE IN SUPERVISOR MODE
   1190 002256     000                 PASFLG: .BYTE   0                 ;LOCAL LOOP PASS CONTROL
   1191 002257     000                 UPPFLG: .BYTE   0                 ;LOCAL LOOP PASS CONTROL
   1192 002260  000000                 REALPAT:.WORD   0                 ;REAL PATTERN UNDER TEST
   1193 002262  000000                 OLDCACHE:.WORD  0                 ;BACKED UP VALUE OF CACHE CONTROL REGISTER
   1194 002264  000000                 PARTHERE:.WORD  0                 ;PARITY TRAPS SOMETIMES GO TO ADDRESS STORED HERE
   1195 002266  000000                 FSSTACK:.WORD   0                 ;STACK SAVED HERE IF IN FIELD SERVICE MODE
   1196 002270  000000                 NEWBANK:.WORD   0                 ;USED FOR RELOCATION TO A NEW BANK
   1197 002272  000000                 SOURCE: .WORD   0                 ;SOURCE OF DATA WORDS FOR CHECKBIT GENERATION SUBROUTINE
   1198 002274  000000                 CHECK:  .WORD   0                 ;CHECKBITS TO BE LOADED INTO CSR
   1199 002276  000000                 PCBUMP: .WORD   0                 ;VALUE TO BUMP THE PC BY TO RECOVER AFTER A PARITY TRAP
   1200 002300  000000                 CSRINC: .WORD   0                 ;VALUE TO INCREMENT ADDRESS BY TO REMAIN IN THE SAME CSR
   1201 002302  000000                 CSRLOOP:.WORD   0                 ;LOOP CONTROL FOR CSR TESTING
   1202 002304  000000                 SUCCESS:.WORD   0                 ;FLAG SET BY SUCCESSFULL TASK OR SUBROUTINE
   1203 002306  000000                 ZEROS:  .WORD   0                 ;FOR AID IN "MOV" INSTRUCTIONS
   1204 002310  000000                 TIME:   .WORD   0                 ;SECONDS THAT BATTERIES SHOULD LAST
   1205 002312  000000                 SKIPMK: .WORD   0                 ;FLAG TO SKIP MKCONTROL SUBROUTINE
   1206 002314  000000                 NULLFLAG:.WORD  0                 ;SET WHEN RUNNING NULL PATTERNS
   1207 002316  000000                 QVFLAG: 0                         ;FLAGS QUICK VERIFY PASS UNDER APT, ACT, OR XXDP CHAIN MODE
   1208 002320  000000                 ACTFLAG:0                         ;FLAGS ACT AUTOMATIC MODE PROGRAMMING RULES
   1209 002322  000000                 APTFLAG:0                         ;FLAGS APT AUTOMATIC MODE PROGRAMMING RULES
   1210 002324  000000                 XXDPCHAIN:0                       ;FLAGS XXDP CHAIN MODE PROGRAMMING RULES
   1211                                ;NOTE: THESE TWO BYTES MUST STAY TOGETHER
```

```
 1212 002326     000                   $NULL:  .BYTE   0            ;;CONTAINS NULL CHARACTER FOR FILLS
 1213 002327     000                   $FILLS: .BYTE   0            ;CONTAINS # OF FILL CHARACTERS
 1214 002330     000                   $TPFLG: .BYTE   0            ;;"TERMINAL NOT AVAILABLE" FLAG
 1215                                          .EVEN
 1216 002332  000000                   $ESCAPE:0                    ;;ESCAPE ON ERROR ADDRESS
 1217 002334  000000                   EVEN:   0                    ;USED FOR ALTERNATE DATA PATTERNS
 1218 002336  000000                   STRIPES:0                    ;COUNTS DIAGONAL STRIPES
 1219 002340  000000                   COUNT:  0                    ;BACKED UP COPY OF STRIPES
 1220 002342  000000                   NOTAB:  0                    ;NO TABLE BEING PRINTED - NOW
 1221 002344  000000                   BSIZE:  0                    ;SIZE OF 11/45 MOS MEMORY IN K WORDS
 1222 002346  000000                   KSIZE:  0                    ;SIZE OF MF11S-K MEMORY IN K WORDS
 1223 002350  000000                   LSIZE:  0                    ;SIZE OF MS11-L MEMORY IN K WORDS
 1224 002352  000000                   MSIZE:  0                    ;SIZE OF MS11-M MEMORY IN K WORDS
 1225 002354  000000                   PSIZE:  0                    ;SIZE OF UNIBUS PARITY MEMORY IN K WORDS
 1226 002356  000000                   TOOMANY:0                    ;FLAGS WHEN TOO MANY ERRORS HAVE BEEN PRINTED FOR A BANK
 1227 002360  000000                   READONLY:0                   ;FLAG TO PATTERNS TO READ ONLY
 1228 002362  000000  000000           TESTADD:0,0                  ;THE ADDRESS TO RUN CSR TESTS ON
 1229 002366  000000                   UNITOP: 0                    ;HIGHEST ACCESSABLE BANK OF MEMORY THRU UNIBUS MAP
 1230 002370  000000                   STOPOK: 0                    ;FLAG TO ALLOW STOPPING WITH SWITCH REGISTER
 1231 002372  000000                   APTPAR: .WORD   0            ;AMOUNT OF PARITY MEMORY ACCORDING TO APT
 1232 002374  000000                   APTECC: .WORD   0            ;AMOUNT OF ECC MEMORY ACCORDING TO APT
 1233 002376  000000                   NOFSMODE:0                   ;FLAG TO DISABLE FIELD SERVICE MODE
 1234 002400  000000                   NOERROR:0                    ;"THIS IS NOT AN ERROR" FLAG
 1235 002402  000000                   LOADBANK:0                   ;BANK LOADERS ARE RELOCATED TO
 1236 002404  000000                   TEMP:   0                    ;USED FOR JUNK
 1237 002406  000000                   QUICK:  0                    ;QUICK STOP FLAG FOR APT POWER FAIL
 1238 002410  000000                   NOSCOPE:0                    ;"NO SCOPE LOOP ALLOWED" FLAG
 1239 002412  000000                   FSINFLAG:0                   ;"FIELD SERVICE - NO INTERNAL INTERLEAVE" FLAG
 1240 002414  000000                   APTSIZE:0                    ;APT SIZING INFO FLAG
 1241 002416  000000                   FS7FLAG:0                    ;TRUE WHEN IN FIELD SERVICE COMMAND 7
 1246 002420  000000                   CONFGERROR:0                 ;CONFIGURATION ERROR FLAG
 1247 002422  000000                   I:      0                    ;USED FOR GENERAL PURPOSE INDEXING
 1248 002424  000000                   NO22BIT:0                    ;NO 22-BIT MODE FLAG
 1249 002426  000000                   NOSUPER:0                    ;NO SUPERVISOR MODE FLAG
 1250 002430  000000                   ERRADD: .WORD   0            ;HOLDS THE CSR'S ERROR ADDRESS
 1251 002432  000000  000000  000000   CSRINFO:0,0,0,0,0,0,0,0      ;USED TO STORE INFORMATION ABOUT THE 16
      002440  000000  000000  000000
      002446  000000  000000
 1252 002452  000000  000000  000000           0,0,0,0,0,0,0,0      ;POSSIBLE CSR'S
      002460  000000  000000  000000
      002466  000000  000000
 1253 002472  000000                   LINK1:  0                    ;USED TO HOLD LINKS TO PATTERNS WHICH
 1254 002474  000000                   LINK2:  0                    ;CAN EXECUTE IN THE PAR/PDR'S OR NOT
 1255 002476  000000                   CSRHOLD:0                    ;USED TO STORE CSR VALUES FOR CSR TESTS
 1256 002500  000000                   KFLAG:  0                    ;USED TO FLAG MF11S-K MEMORY TO TESTS
 1257 002502  000000  000000           PGMCSR: .WORD   0,0          ;POINTS TO PROGRAM CSR
 1258 002506  000000                   INHECC: .WORD   0            ;FLAGS INHIBIT ECC TESTS ON RELOCATION
 1259 002510  000000                   INHBANK:.WORD   0            ;
 1260 002512  000000                   FULLREL:.WORD   0            ;
 1298 002514                           $CMTGE: ;*END OF COMMON TAGS
```

```
                                                .SBTTL  VARIABLES         INITIALIZED TO NON ZERO
     002514  000001  000000             CACHKN: 1,0                       ;CACHE CONSTANT (MOVED TO CONTRL TO TURN ON CACHE)
     002520  001415                     CACHKF: 1415                      ;CACHE CONSTANT (MOVED TO CONTRL TO TURN OFF CACHE)
     002522  040000                     TESTMODE:40000                    ;USED TO SELECT THE PROPER TEST MODE FOR A PATTERN RUN
     002524  000012                     ERRMAX: 10.                       ;MAX # OF ERRORS PER BANK WITH SW11
     002526  000167                     LASTBANK:167                      ;HIGHEST BANK OF MEMORY
     002530  170000                     LASTBLOCK:170000                  ;HIGHEST BANK OF MEMORY+1 (IN PAR FORMAT)
     002532  000031                     SOBK:   25.                       ;SOB CONSTANT
     002534  002000                     KSTACK: STACK                     ;STACK BEGINNING
     002536  000001                     LOADHOME:1                        ;HOME BANK OF LOADERS
     002540  177777                     WORST:  177777                    ;SET IF TESTING BANKS IN WORST FIRST MODE(1ST PASS)
     002542  176543                     SEEDHI: 176543                    ;WORKING SEED HI (USED FOR RANDOM NUMBER GENERATOR)
     002544  123456                     SEEDLO: 123456                    ;WORKING SEED LO (USED FOR RANDOM NUMBER GENERATOR)
     002546  176543                     MSEEDH: 176543                    ;MASTER SEED HI (USED FOR RANDOM NUMBER GENERATOR)
     002550  123456                     MSEEDL: 123456                    ;MASTER SEED LO (USED FOR RANDOM NUMBER GENERATOR)
     002552  177777                     HEADER: 177777                    ;USED TO PRINT HEADINGS ONLY ONCE
     002554  177777                     ONES:   177777                    ;FOR AID IN ''MOV'' INSTRUCTIONS
     002556  000003                     FLIPLOC:3                         ;COUNTER FOR FLIPING DATA ON WORST CASE NOISE TEST
     002560  052525                     SOFTPAT:52525                     ;PATTERN FOR SOFT ERROR BACKGROUND TESTS
     002562  000000                     $LPADR: .WORD   0                 ;;CONTAINS SCOPE LOOP ADDRESS
     002564  000000                     $LPERR: .WORD   0                 ;;CONTAINS SCOPE RETURN FOR ERRORS
     002566  000000                     RESTART:0                         ;RESTART (START ADD 202) FLAG
     002570  000000                     $ERTTL: .WORD   0                 ;;CONTAINS TOTAL ERRORS


                                        ;************************ NOTE THESE TWO LOCATIONS MUST STAY TOGETHER ************
     002572  000377                     BAKPAT: .WORD   377               ;BACKGROUND PATTERN                              *
     002574  177400                     SWAPAT: .WORD   177400            ;SWAPPED BAKPAT                                  *
                                        ;*********************************************************************************


     002576  177570                     SWR:      .WORD   DSWR            ;;ADDRESS OF SWITCH REGISTER
     002600  177570                     DISPLAY:  .WORD   DDISP           ;;ADDRESS OF DISPLAY REGISTER
     002602  177560                     $TKS:     177560                  ;;TTY KBD STATUS
     002604  177562                     $TKB:     177562                  ;;TTY KBD BUFFER
     002606  177564                     $TPS:     177564                  ;;TTY PRINTER STATUS REG. ADDRESS
     002610  177566                     $TPB:     177566                  ;;TTY PRINTER BUFFER REG. ADDRESS
     002612     012                     $FILLC:   .BYTE   12              ;;INSERT FILL CHARS. AFTER A ''LINE FEED''
     002613     207     377     377     $BELL:    .ASCIZ  <207><377><377> ;;CODE FOR BELL
     002616     000
     002617     077                     $QUES:    .ASCII  /?/             ;;QUESTION MARK
     002620     015                     $CRLF:    .ASCII  <15>            ;;CARRIAGE RETURN
     002621     012     000             $LF:      .ASCIZ  <12>            ;;LINE FEED
                                                  .EVEN
```

```
                                .SBTTL  CONFIGURATION TABLE
                        ;CONFIG:FIRST 16K CONFIGURATION WORDS (2 EACH)
                        ;       2ND     16K CONFIGURATION WORDS (2 EACH)
                        ;       ........................
                        ;       200TH   16K CONFIGURATION WORDS (2 EACH)
                        ;
                        ;CONFIGURATION WORDS:
                        ;               LOW:    BIT 0     ERRORS PRESENT
                        ;                       BIT 1     MEMORY SUCESSFULLY ACCESSED
                        ;                       BIT 2-4   RESERVED
                        ;                       BIT 5     SKIP ECC LOGIC TESTS FLAG (1=SKIP)
                        ;                       BIT 6     PROTECTED REGION OF ECC MEMORY
                        ;                       BIT 7     PROTECTED (PROGRAM SPACE)
                        ;                       BIT 8-11  CSR CODE
                        ;                       BIT 12-15 INTERLEAVED CSR CODE
                        ;               HIGH:   BIT 0-7   NUMBER OF ERRORS
                        ;                       BIT 8-10  MEMORY TYPE
                        ;                       BIT 11    INTERLEAVED BOARD TYPE (0=128K, 1=64K)
                        ;                       BIT 12    INTERLEAVE ENABLED
                        ;                       BIT 13    "BACKGROUND PATTERN VALID" FLAG
                        ;                       BIT 14    BANK SELECTED FOR TEST BY FIELD SERVICE MODE
                        ;                       BIT 15    LOADERS HOME BANK
002624  000201          CONFIG: .REPT   201
003630                  CONFIEND:
```

```
                                                        .SBTTL  *********************** MAIN ***************************
003630                                          START:  SUBTST  <<INITIALIZE VARIABLES TO ZERO>>
                                                ;*******************************************************************************
                                                ;*SUBTEST        INITIALIZE VARIABLES TO ZERO
                                                ;*******************************************************************************
003630  000005                                          RESET
003632  013706  002534                                  MOV     KSTACK,SP       ;;SETUP THE STACK POINTER
003636  012700  002000                                  MOV     #$CMTAG,R0      ;;FIRST LOCATION TO BE CLEARED
003642  005020                                  1$:     CLR     (R0)+           ;;CLEAR MEMORY LOCATION
003644  022700  002514                                  CMP     #$CMTGE,R0      ;;DONE?
003650  001374                                          BNE     1$              ;LOOP BACK IF NO
003652  012737  000167  002526                          MOV     #167,LASTBANK   ;RESTORE LASTBANK (THIS MUST BE DONE PRIOR TO SYSTEM SIZING)
003660                                                  SUBTST  <<CLEAR NON-PROGRAM SPACE>>
                                                ;*******************************************************************************
                                                ;*SUBTEST        CLEAR NON-PROGRAM SPACE
                                                ;*******************************************************************************
                                                        ;THIS ATTEMPS TO GET RID OF ANY PARITY ERRORS BY WRITING INTO
                                                        ;EVERY LOCATION THAT IS NOT LOADED INTO BY THE PROGRAM OR ALLOCATED
                                                        ;TO THE XXDP LOADERS
003660  012737  000001  002074                          MOV     #1,NOPAR                        ;PARITY ACTION = COUNT & IGNORE
003666  005000                                          CLR     R0
003670  000241                                  2$:     CLC
003672  005520                                          ADC     (R0)+
003674  020027  160000                                  CMP     R0,#160000
003700  103773                                          BLO     2$
003702  005037  002074                                  CLR     NOPAR                           ;RESTORE DEFAULT PARITY ACTION
```

```
   1411 003706                                      SUBTST  <<TYPE OF SYSTEM SIZER>>
                                         ;**********************************************************************
                                         ;*SUBTEST        TYPE OF SYSTEM SIZER
                                         ;**********************************************************************
   1412 003706  000401                              BR      SYSSIZ                  ;SKIP OVER VARIABLE LOCATION
   1413 003710  000000                   PROTYP:  .WORD    0
   1414 003712                           SYSSIZ:  SET4     #4$
   1415 003720  005737  177746                      TST     CONTRL                  ;SEE IF CACHE REGISTER RESPONDS
   1416 003724                                      SET4    #9$                     ;YES - DO WE HAVE 11/44 TYPE CACHE
   1417 003732  005737  177750                      TST     MAINT                   ;OR 11/60 TYPE CACHE?
   1418 003736  000411                              BR      5$                      ;BRANCH IF 11/44 TYPE CACHE
   1419 003740  012737  000014  002520  9$:         MOV     #14,CACHKF              ;TURN OFF CONSTANT FOR 11/60 CACHE
   1420 003746  000405                              BR      5$
   1421 003750  005037  002514          4$:         CLR     CACHKN                  ;NO CACHE ON SYSTEM
   1422 003754  012737  002306  064002              MOV     #ZEROS,DT14             ;DO NOT PRINT CONTRL ERROR MESSAGES
   1423 003762                          5$:         SET4    #6$
   1424 003770  005737  172516                      TST     MMR3                    ;DO WE HAVE AN MMR3?
   1425 003774  005037  172516                      CLR     MMR3                    ;YES WE DO
   1426 004000  052737  000020  172516              BIS     #BIT4,MMR3              ;SEE IF THERE IS 22-BIT MODE
   1427 004006  032737  000020  172516              BIT     #BIT4,MMR3
   1428 004014  001024                              BNE     10$                     ;BRANCH IF 22-BIT RELOCATION
   1429 004016  000411                              BR      7$                      ;BRANCH IF MMR3 BUT NO 22-BIT RELOC.
   1430                                  ;* 11/34 TYPE MACHINES ENTER HERE
   1431 004020  012737  140000  002522  6$:         MOV     #140000,TESTMODE        ;MAKE TEST MODE USER
   1432 004026  005237  002426                      INC     NOSUPER                 ;NO SUPERVISOR MODE
   1433 004032  005037  063652                      CLR     DT5+10
   1434 004036  005037  064012                      CLR     DT14+10
   1435                                  ;* 11/45 TYPE MACHINES ENTER HERE
   1436 004042  005237  002424          7$:         INC     NO22BIT                 ;NO 22 BIT MODE
   1437 004046  012737  000007  002526              MOV     #7,LASTBANK             ;124K MEMORY MAX. MEMORY SIZE
   1438 004054  005037  063654                      CLR     DT5+12                  ;DO NOT TRY TO PRINT ERROR REGISTER
   1439 004060  005037  064014                      CLR     DT14+12                 ;ERROR MESSAGES, BECAUSE THERE IS
   1440                                                                              ;IS NO ERROR REGISTER!
   1441 004064  000417                              BR      8$
   1442 004066                          10$:        SET4    #8$
   1443 004074  000007                              MFPT                            ;TYPE OF PROCESSOR TEST: THIS INSTRUCTION
   1444                                                                              ; (AVAILABLE ON NEWER PROCESSORS ONLY) PLACES
   1445                                                                              ; A CODE IN THE LOWER BYTE OF R0 THAT
   1446                                                                              ; INDICATES THE  PROCESSOR TYPE. 1=11/44
   1447                                                                              ; 3=11/24
   1448 004076  110037  003710                      MOVB    R0,PROTYP               ;MOV THE CODE TO PROTYP
   1449 004102  022737  000003  003710              CMP     #3,PROTYP               ;IS THIS AN 11/24?
   1450 004110  001005                              BNE     8$                      ;BRANCH IF NOT - WE HAVE AN 11/44
   1451 004112  005237  002426                      INC     NOSUPER                 ;NO SUPERVISOR MODE
   1452 004116  012737  140000  002522              MOV     #140000,TESTMODE        ;MAKE TEST MODE USER
   1453
   1454 004124                          8$:         RES4
   1457 004146                                      SUBTST  <<INITIALIZE VARIABLES TO NON ZERO>>
                                          ;*********************************************************************************
                                          ;*SUBTEST       INITIALIZE VARIABLES TO NON ZERO
                                          ;*********************************************************************************
   1458 004146                                      SET     WORST
   1459 004154  012737  000003  002556              MOV     #3,FLIPLOC
   1460 004162                                      SET     HEADER
   1461 004170  012737  176543  002546              MOV     #176543,MSEEDH
   1462 004176  012737  123456  002550              MOV     #123456,MSEEDL
   1463 004204  013737  002546  002542              MOV     MSEEDH,SEEDHI   ;PRIME THE RANDOM NUMBER GENERATOR
   1464 004212  013737  002550  002544              MOV     MSEEDL,SEEDLO   ;BOTH HIGH AND LOW WORDS
   1465 004220  012737  000377  002572              MOV     #377,BAKPAT
   1466 004226  012737  177400  002574              MOV     #177400,SWAPAT
   1471 004234                                      SUBTST  <<INITIALIZE VECTORS>>
                                          ;*********************************************************************************
                                          ;*SUBTEST       INITIALIZE VECTORS
                                          ;*********************************************************************************
   1472 004234  012737  055334  000020              MOV     #$SCOPE,IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
   1473 004242  012737  000340  000022              MOV     #340,IOTVEC+2   ;;LEVEL 7
   1474 004250  012737  055642  000030              MOV     #$ERROR,EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
   1475 004256  012737  000340  000032              MOV     #340,EMTVEC+2   ;;LEVEL 7
   1476 004264  012737  062602  000034              MOV     #$TRAP,TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
   1477 004272  012737  000340  000036              MOV     #340,TRAPVEC+2;LEVEL 7
   1478 004300  012737  051524  000024              MOV     #$PWRDN,PWRVEC ;;POWER FAILURE VECTOR
   1479 004306  012737  000340  000026              MOV     #340,PWRVEC+2   ;;LEVEL 7
   1480 004314  012737  037720  000114              MOV     #PARITY,PARVEC;GET READY FOR PARITY ERRORS
   1481 004322  012737  000340  000116              MOV     #340,PARVEC+2
   1482 004330  012737  040114  000010              MOV     #PDP1105,RESVEC;RESERVED INSTRUCTION TRAP
   1483 004336  012737  000340  000012              MOV     #340,RESVEC+2
   1484 004344  012737  040070  000004              MOV     #TIMEOUT,ERRVEC;SETUP TIMEOUT ERRORS
   1485 004352  012737  000340  000006              MOV     #340,ERRVEC+2   ;SET PRIORITY OF ERROR TRAPS
   1486 004360  012737  040102  000250              MOV     #MMTRAP,MMVEC   ;VECTOR FOR MEMORY MANAGEMENT
   1487 004366  012737  000340  000252              MOV     #340,MMVEC+2
   1492 004374  104423                              CACHON                  ;TURN CACHE ON
```

```
004376                                  SUBTST  <<INITIALIZE PATTERNS>>
                        ;***********************************************************************************************
                        ;*SUBTEST         INITIALIZE PATTERNS
                        ;***********************************************************************************************
                                        ;THE APT E-TABLE DETERMINES WHICH PATTERNS ARE GOING TO BE RUN.
                                        ;EACH BIT SET REPRESENTS A PATTERN TABLE ENTRY THAT IS TO BE LEFT
                                        ;ALONE (TO BE RUN).  EACH BIT CLEARED REPRESENTS A PATTERN TABLE ENTRY
                                        ;THAT IS TO BE OVERLAYED WITH THE ADDRESS OF A NULL PATTERN.
004376  012700  062552                  MOV     #$DDW0,R0
004402  012001                          MOV     (R0)+,R1
004404  012703  017366                  MOV     #MKCSRT,R3
004410  012702  000020                  MOV     #16.,R2
004414  004737  004514                  CALL    PATPLUG
004420  012001                          MOV     (R0)+,R1
004422  012702  000010                  MOV     #8.,R2
004426  004737  004514                  CALL    PATPLUG
004432  012001                          MOV     (R0)+,R1
004434  012703  017616                  MOV     #MKPAT,R3
004440  012702  000020                  MOV     #16.,R2
004444  004737  004514                  CALL    PATPLUG
004450  012001                          MOV     (R0)+,R1
004452  012702  000010                  MOV     #8.,R2
004456  004737  004514                  CALL    PATPLUG
004462  012001                          MOV     (R0)+,R1
004464  012703  020002                  MOV     #MJPAT,R3
004470  012702  000020                  MOV     #16.,R2
004474  004737  004514                  CALL    PATPLUG
004500  012001                          MOV     (R0)+,R1
004502  012702  000010                  MOV     #8.,R2
004506  004737  004514                  CALL    PATPLUG
004512  000417                          BR      SUBAAA

004514                  PATPLUG:SUBTST  <<SUBR  PLUG IN NULL PATTERNS>>
                        ;***********************************************************************************************
                        ;*SUBTEST         SUBR    PLUG IN NULL PATTERNS
                        ;***********************************************************************************************
004514                                  FOR I := #1 TO R2
004522  006001                            ROR   R1
004524                                    ON.NOERROR                ;IF CARRY CLEAR
004526  012713  026354                      MOV #MT0999,(R3)
004532                                    END ;OF ON.ERROR
004532  062703  000002                    ADD #2,R3
004536                                  END ;OF FOR
004550  000207                          RETURN
```

```
1535 004552                           SUBAAA: SUBTST  <<CLEAR THE CONFIGURATION TABLE>>
                                      ;********************************************************************************
                                      ;*SUBTEST       CLEAR THE CONFIGURATION TABLE
                                      ;********************************************************************************
1536                                          ;THIS ZEROS (UNLESS WE STARTED AT ADDRESS 202) THE CONFIG TABLE
1537                                          ;WHICH IS FULLY DISCRIBED AT LOCATION "CONFIG".
1538                                          .ENABLE LSB
1539 004552                                   IF RESTART IS FALSE
1540 004560  012700  002624                     MOV   #CONFIG,R0
1541 004564  005020                   1$:       CLR   (R0)+
1542 004566  022700  003630                     CMP   #CONFIEND,R0
1543 004572  001374                             BNE   1$
1544 004574                                   END ;OF IF RESTART
1545                                          .DSABL  LSB
1546 004574  012737  000002  002104           MOV     #BIT1,CPUBIT            ;SET ID BIT
1547 004602                                   SUBTST  <<SIZE FOR A HARDWARE SWITCH REGISTER>>
                                      ;********************************************************************************
                                      ;*SUBTEST       SIZE FOR A HARDWARE SWITCH REGISTER
                                      ;********************************************************************************
1548                                          ;;IF NOT FOUND OR IT IS
1549                                          ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
1550                                          .ENABL  LSB
1551 004602                                   SET4    #3$             ;TRAPS TO 4 GOTO 3$
1552 004610  012737  177570  002576           MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWITCH REGISTER
1553 004616  012737  177570  002600           MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
1554 004624                                   IF #-1 EQ @SWR          ;IF NO TRAP FROM REFERENCE TO @SWR AND @SWR = #-1
1555 004634  000403                             BR    2$              ;;BRANCH IF NO TIMEOUT
1556 004636  012716  004644           3$:       MOV   #2$,(SP)        ;;SET UP FOR TRAP RETURN
1557 004642  000002                             RTI
1558 004644                           2$:       RES4                  ;RESET TRAPS TO 4 TO DEFAULT
1559 004666  012737  000176  002576             MOV   #SWREG,SWR      ;;POINT TO SOFTWARE SWR
1560 004674  012737  000174  002600             MOV   #DISPREG,DISPLAY
1561 004702                                   END ;OF IF #-1
1562                                          .DSABL  LSB
```

```
004702                                  SUBAAB: SUBTST  <<SETUP ACT, APT, & XXDP>>
                                        ;*****************************************************************************
                                        ;*SUBTEST        SETUP ACT, APT, & XXDP
                                        ;*****************************************************************************
                                                ;THIS SETS UP A BUNCH OF FLAGS TO TELL THE PROGRAM EVERYTHING
                                                ;IT CARES TO KNOW ABOUT APT, ACT, & XXDP.
004702  005037  062474                          CLR     $PASS           ;CLEAR PASS COUNT
004706                                          IFB #BIT5 SET.IN $ENVM
004716                                            SET   $TPFLG          ;INDICATE NO TERMINAL
004724                                          END ;OF IFB #BIT5
004724                                          IFB #BIT7 SET.IN $ENVM
004734                                            SET   APTSIZE
004742                                          END ;OF IFB #BIT7
004742                                          IFB $ENV EQ #1
004752                                            SET   APTFLAG,QVFLAG,$AUTO,QUICK
005002  012737  045210  000024                    MOV   #APTDOWN,PWRVEC
005010  012737  062510  002576                    MOV   #$SWREG,SWR     ;USE APT SWR
005016                                          ELSE
005020                                            IF 42 NE #STACK AND 42 NE #0
005036                                              SET QVFLAG,$AUTO
005052                                              IF 42 EQ #$ENDAD
005062                                                SET       ACTFLAG
005070                                              ELSE
005072                                                SET       XXDPCHAIN
005100                                              END ;OF IF 42
005100                                            END ;OF IF 42
005100                                          END ;OF IFB $ENV
```

```
005100                                  SUBTST  <<PROTECT PROGRAM & LOADERS>>
                                ;************************************************************************************
                                ;*SUBTEST       PROTECT PROGRAM & LOADERS
                                ;************************************************************************************
005100  052737  000200  002624          BIS     #BIT7,CONFIG            ;PROTECT PROGRAM SPACE (BANK 0)
005106  052737  000200  002630          BIS     #BIT7,CONFIG+4          ;PROTECT LOADER SPACE (BANK 1)
005114                                  IF #$ENDAD NE 42                ;NOT ACT-11?
005124                                    TYPE  MSG000                  ;TYPE PROGRAM TITLE
005130                                  END ;OF IF #$ENDAD

005130                                  SUBTST  <<CHECK SYSTEM FOR CACHE>>
                                ;************************************************************************************
                                ;*SUBTEST       CHECK SYSTEM FOR CACHE
                                ;************************************************************************************
                                        ;* THIS FIGURES OUT IF THERE IS A CACHE ON THE SYSTEM,
                                        ;* WHAT TYPE OF SYSTEM IT IS, AND WHETHER IT IS ENABLED
                                        ;* OR DISABLED.
005130                                  SET4    #3$
005136  005737  177746                  TST     CONTRL                  ;IS THERE A CONTROL REGISTER?
005142                                  SET4    #2$
005150  005737  177750                  TST     MAINT                   ;IS THERE A MAINTENANCE REGISTER?
005154                                  SET4    #1$
005162  005737  177754                  TST     DATARG                  ;IS THERE A DATA REGISTER?
005166                                  TYPE    MSG117                  ; 11/44
005172  000405                          BR      4$
005174                          1$:     TYPE    MSG116                  ; 11/34
005200  000402                          BR      4$
005202                          2$:     TYPE    MSG118                  ; 11/60
005206  052737  000014  177746  4$:     BIS     #BIT2!BIT3,CONTRL       ;SET CACHE DISABLE BITS
005214  042737  000014  177746          BIC     #BIT2!BIT3,CONTRL       ;CLEAR CACHE DISABLE BITS
005222  032737  000004  177746          BIT     #BIT2,CONTRL            ;IS THE BIT SET?
005230  001004                          BNE     7$                      ;BRANCH IF THE BIT IS SET
005232  032737  000010  177746          BIT     #BIT3,CONTRL            ;IS THE BIT SET?
005240  001413                          BEQ     6$                      ;BRANCH IF THE BIT IS SET
005242                          7$:     TYPE    MSG121                  ; CACHE BYPASSED
005246  104424                          CACHOFF
005250  013737  002514  002516          MOV     CACHKN,CACHKN+2         ;SAVE INFO ABOUT CACHE
005256  005037  002514                  CLR     CACHKN                  ;CACHE CANNOT BE USED - IT'S BYPASSED
005262  000404                          BR      8$
005264                          3$:     TYPE    MSG119                  ; NO
005270                          6$:     TYPE    MSG120                  ; CACHE AVAILABLE
```

```
  1672 005274                                           SUBTST  <<SETUP USER & SUPERVISOR STACK>>
                                                ;*************************************************************************
                                                ;*SUBTEST        SETUP USER & SUPERVISOR STACK
                                                ;*************************************************************************
  1673 005274  104421                           8$:     DEENERGIZE                      ;TURN OFF MEMORY MANAGEMENT
  1674 005276  005737  002426                           TST     NOSUPER                 ;IS THERE A SUPERVISOR MODE?
  1675 005302  001011                                   BNE     5$                      ;NO-SKIP SUPERVISOR SETUP.
  1676
  1677                                                  ;SET PREVIOUS MODE TO SUPERVISOR
  1678 005304  042737  030000  177776                   BIC     #BIT13!BIT12,PSW
  1679 005312  052737  010000  177776                   BIS     #BIT12,PSW
  1680
  1681 005320                                           PUSH    #SUPSTK
  1682 005324  006606                                   MTPI    SSP
  1683
  1684                                                  ;SET PREVIOUS MODE TO USER
  1685 005326  052737  030000  177776   5$:             BIS     #BIT13!BIT12,PSW
  1686
  1687 005334                                           PUSH    #USESTK
  1688 005340  006606                                   MTPI    USP
  1689
  1690 005342                                           SUBTST  <<GET SOFTWARE SWITCH REGISTER IF NECESSARY>>
                                                ;*************************************************************************
                                                ;*SUBTEST        GET SOFTWARE SWITCH REGISTER IF NECESSARY
                                                ;*************************************************************************
  1694 005342                                           IF $AUTO IS FALSE                       ;IF NOT(APT OR ACT)
  1695 005350                                             IF SWR EQ #SWREG              ;IF SOFTWARE SWITCH REG SELECTED
  1696 005360  104407                                       GTSWR                               ;;GET SOFT-SWR SETTINGS
  1697 005362                                             END ;OF IF SWR
  1698 005362                                           END ;OF IF $AUTO
  1702
  1703 005362                                           SUBTST  <<GET MEMORY MANAGEMENT READY>>
                                                ;*************************************************************************
                                                ;*SUBTEST        GET MEMORY MANAGEMENT READY
                                                ;*************************************************************************
  1707 005362  104422                                   KMAP                            ;MAP KERNEL SPACE 1 TO 1
  1711 005364                                           MAP                             ;MAP SUPERVISOR SPACE (TEST AREA) 1 TO 1
  1712 005400  104420                                   ENERGIZE                        ;TURN ON MEMORY MANAGEMENT
```

```
005402                                  NEWTST  <<BIT TEST OF ALL CSR'S>>
                                ;*************************************************************************
                                ;*TEST 1        BIT TEST OF ALL CSR'S
                                ;*************************************************************************
005402  000004                  TST1:   SCOPE
                                ;* THE FIRST PART OF THE CONFIGURATION ANALYSIS DOES THE FOLLOWING:
                                ;*      1) FINDS WHICH CSR'S RESPOND, AND PUTS THEM INTO THE CSR INFORMATION
                                ;*              TABLE, AND STORES ANOTHER BIT FOR ''TOTCSRS''.
                                ;*      2) TESTS THE CSR BITS COMMON TO ALL CSR'S.
                                ;*      3) FIGURES OUT IF THERE IS AN EUB BIT, AN ECC BIT, AND THE EXISTANCE
                                ;*              OF THE ERROR ADDRESS BITS, AND MARKS THIS IN THE CSR
                                ;*              INFORMATION TABLE.
                                ;*      4) TESTS THE BITS PARTICULAR TO THAT TYPE OF CSR.
                                ;*      5) IF ANY BITS TEST BAD IN THE CSR UNDER TEST, THE CSR OK BIT IN THE
                                ;*              CSR INFORMATION TABLE IS CLEARED.
                                ;* THE INFORMATION BITS ONE THROUGH THREE FORM A CODE WHICH GIVES THE TYPE
                                ;* OF CSR:
                                ;*                          ERR. ADDR.   PARITY  NOT EUB
                                ;*         TYPE               BIT2        BIT1    BIT0      CODE TOTALS
                                ;* UNIBUS PARITY               1           1       1            7
                                ;*      MS11-L                 1           1       0            6
                                ;*      MF11S-K                1           0       1            5
                                ;*      MS11-M                 1           0       0            4
                                ;* 11/45 BIPOLAR               0           1       1            3
                                ;*
                                ;* THIS MEMORY CODE WILL BE USED IN THE SECOND PART OF THIS ANALYSIS
                                ;*
005404  005005                          CLR     R5              ;R5 IS THE TOTAL CSR NUMBER
005406  005000                          CLR     R0              ;R0 IS A TABLE INDEX
005410  012703  172100                  MOV     #CSRADD,R3      ;R3 HAS THE CSR ADDRESS
005414  012737  000001  002074          MOV     #1,NOPAR        ;IGNORE PARITY ERRORS
005422                                  SET4    #2$
005430  005713                  1$:     TST     (R3)    ;DOES THE CSR RESPOND?
005432  052705  000001                  BIS     #1,R5
005436  005004                          CLR     R4              ;CLEAR THE LAST CSR INDICATOR
005440  052760  000007  002432          BIS     #7,CSRINFO(R0)  ;SET ALL THE MEMORY INFO BITS
005446  052760  000030  002432          BIS     #BIT4!BIT3,CSRINFO(R0)  ;YES-MARK IT IN CSR INFORMATION TABLE
005454  004537  006010                  JSR     R5,TEST         ;TEST BIT 0 AND 15
005460  100001                          .WORD   BIT15!BIT0
005462  012713  040000                  MOV     #BIT14,(R3)     ;IS THERE A BIT 14 RESPONDING
005466  032713  040000                  BIT     #BIT14,(R3)     ;(IT'S THE EUB BIT)
005472  001403                          BEQ     3$              ;BRANCH IF NO EUB BIT
005474  042760  000001  002432          BIC     #BIT0,CSRINFO(R0) ;CLEAR EUB INFO IN THE CSR TABLE
005502  005013                  3$:     CLR     (R3)            ;CLEAR THE CSR UNDER TEST
005504  012713  020000                  MOV     #BIT13,(R3)     ;DOES BIT 13 RESPOND
005510  032713  020000                  BIT     #BIT13,(R3)     ;(TO TEST FOR ECC CSR)
005514  001417                          BEQ     4$              ;BRANCH IF NOT ECC CSR
005516  042760  000002  002432          BIC     #BIT1,CSRINFO(R0) ;CLEAR PARITY INFO IN THE CSR TABLE
005524  012713  020000                  MOV     #BIT13,(R3)     ;SET THE INHIBIT MODE POINTER TO 1ST 16K
005530  004537  006010                  JSR     R5,TEST         ;TEST BIT 3
005534  000010                          .WORD   BIT3
005536  012713  020000                  MOV     #BIT13,(R3)
005542  004537  006010                  JSR     R5,TEST         ;TEST BIT 1 AND 4
005546  000022                          .WORD   BIT4!BIT1
005550  012713  020000                  MOV     #BIT13,(R3)
```

```
005554  004537  006010          4$:     JSR     R5,TEST           ;TEST BIT 2
005560  000004                          .WORD   BIT2
005562  005013                          CLR     (R3)
005564  052713  007740                  BIS     #7740,(R3)        ;ARE THERE ERROR ADDRESS BITS?
005570  032713  007740                  BIT     #7740,(R3)
005574  001404                          BEQ     5$                ;BRANCH IF NO ERROR ADDR. BITS.
005576  004537  006010                  JSR     R5,TEST           ;TEST BITS 5->11
005602  007740                          .WORD   7740
005604  000403                          BR      6$                ;SKIP OVER THE INFORMATION REPORTING
005606  042760  000004  002432  5$:     BIC     #BIT2,CSRINFO(R0) ;REPORT THAT THERE ARE NO ERROR ADDRESS BITS.
005614  032760  000002  002432  6$:     BIT     #BIT1,CSRINFO(R0) ;IS THIS CSR AN ECC CSR?
005622  001014                          BNE     7$                ;BRANCH IF NOT
005624  032760  000001  002432          BIT     #BIT0,CSRINFO(R0) ;IS THE EUB BIT SET?
005632  001410                          BEQ     7$                ;BRANCH IF IT IS
                                        ;WE MUST NOW TEST FOR MS11-M ON THE UNIBUS
005634  012713  007760                  MOV     #7760,(R3)        ;PUT PATTERN & SBE BIT INTO BITS 4->11
005640  022713  007760                  CMP     #7760,(R3)        ;ARE THEY STILL THERE?
005644  001403                          BEQ     7$                ;YES - BRANCH FOR MF11S-K CSR
005646  042760  000001  002432          BIC     #BIT0,CSRINFO(R0) ;NO - SET EUB BIT FOR MS11-M
005654  005013                  7$:     CLR     (R3)              ;CLEAR CSR
005656  022760  000040  002432          CMP     #40,CSRINFO(R0)   ;IS THIS A LEGAL CONFIGURATION?
005664  100004                          BPL     10$               ;BRANCH IF IT'S LEGAL
005666  016037  002432  002050          MOV     CSRINFO(R0),BAD
005674  104021                          ERROR   +21               ;ILLEGAL TYPE ERROR
005676  032760  000004  002432  10$:    BIT     #BIT2,CSRINFO(R0) ;DOES THIS CSR HAVE ERROR BITS
005704  001016                          BNE     2$                ;BRANCH IF TRUE
005706  032760  000002  002432          BIT     #BIT1,CSRINFO(R0) ;ARE THE OTHER 2 BITS SET?
005714  001404                          BEQ     11$               ;BRANCH IF NOT
005716  032760  000001  002432          BIT     #BIT0,CSRINFO(R0)
005724  001006                          BNE     2$
005726  016037  002432  002050  11$:    MOV     CSRINFO(R0),BAD
005734  104021                          ERROR   +21               ;ILLEGAL TYPE ERROR
005736  005060  002432                  CLR     CSRINFO(R0)       ;CLEAR THE CSR INFO-IT WILL NOT EXIST IN THE PROGRAM
005742  062700  000002          2$:     ADD     #2,R0             ;INCREMENT TO NEXT CSR TABLE
005746  062703  000002                  ADD     #2,R3             ;INCREMENT TO NEXT CSR
005752  006305                          ASL     R5
005754  103001                          BCC     8$                ;IS THERE A CSR 0?
005756  005204                          INC     R4                ;YES - SET CSR PRESENT FLAG
005760  022700  000040          8$:     CMP     #40,R0            ;ARE WE DONE?
005764  001221                          BNE     1$                ;BRANCH IF MORE TO DO
005766  000241                          CLC
005770  006005                          ROR     R5                ;RESYNC R5
005772  005704                          TST     R4                ;WAS THERE A CSR 0?
005774  001402                          BEQ     9$                ;BRANCH IF NOT
005776  052705  100000                  BIS     #BIT15,R5         ;YES - SET IN THE CSR TABLE
006002  010537  002216          9$:     MOV     R5,TOTCSRS        ;STORE R5 IN TOTCSRS
006006  000437                          BR      CTEST             ;JUMP OVER SUBROUTINE
                                        ;THIS SUBROUTINE TESTS THE CSR BITS
006010  012501                  TEST:   MOV     (R5)+,R1          ;GET THE BIT TO TEST
006012  050113                          BIS     R1,(R3)           ;SET THAT IN THE CSR UNDER TEST
006014  030113                          BIT     R1,(R3)           ;IS THE BIT STILL THERE?
006016  001013                          BNE     1$                ;BRANCH IF STILL THERE
006020  011337  002144                  MOV     (R3),CSR          ;READ CSR
006024  010137  002042                  MOV     R1,GOOD
006030  032713  100020                  BIT     #BIT15!BIT4,(R3);IS IT BECAUSE OF A SBE OR DBE?
006034  001004                          BNE     1$                ;BRANCH IF IT IS
006036  104035                          ERROR   +35               ;BIT SET ERROR
```

```
006040  042760  000010  002432          BIC     #BIT3,CSRINFO(R0) ;CLEAR CSR OK BIT
006046  040113                  1$:     BIC     R1,(R3)           ;CLEAR THE SELECTED BIT
006050  030113                          BIT     R1,(R3)           ;IS IT CLEARED?
006052  001413                          BEQ     2$                ;BRANCH IF IT IS CLEARED
006054  011337  002144                  MOV     (R3),CSR          ;READ CSR
006060  010137  002042                  MOV     R1,GOOD
006064  032713  100020                  BIT     #BIT15!BIT4,(R3);IS IT BECAUSE OF A SBE OR DBE?
006070  001004                          BNE     2$                ;BRANCH IF TRUE
006072  104010                          ERROR   +10               ;BIT CLEAR ERROR
006074  042760  000010  002432          BIC     #BIT3,CSRINFO(R0) ;CLEAR CSR OK BIT
006102  000205                  2$:     RTS     R5
006104  000000                  TRACE:  .WORD   0
```

```
                                           ;THE FOLLOWING ROUTINE DETERMINES WHICH CSR CONTROLS PROGRAM SPACE
                                           ;
006106  104424                             CTEST:  CACHOFF
006110  012737  177777  002502                     MOV     #177777,PGMCSR
006116  012737  002000  172350                     MOV     #2000,KIPAR4            ;SET UP MAP REGISTER
006124  012701  002362                             MOV     #TESTADD,R1
006130  012737  100000  002362                     MOV     #100000,TESTADD
006136  012737  100002  002364                     MOV     #100002,TESTADD+2
006144  005000                                     CLR     R0                      ;CLEAR CSR COUNTER
006146  005037  002146                             CLR     CSRNO
006152  013703  002216                             MOV     TOTCSRS,R3              ;OBTAIN CSR MAP
006156  000240                                     NOP                             ;DEBUG AID
006160  006303                             4$:     ASL     R3                      ;PUT HIGH ORDER BIT INTO C BIT
006162  103407                                     BCS     2$                      ;BRANCH IF CSR EXISTS
006164  062700  000002                     1$:     ADD     #2,R0                   ;UPDATE CSR COUNTER
006170  010037  002146                             MOV     R0,CSRNO
006174  005703                                     TST     R3                      ;IS MAP EMPTY?
006176  001464                                     BEQ     3$                      ;BRANCH IF SO
006200  000767                                     BR      4$
006202  000240                             2$:     NOP                             ;DEBUG AID
006204  000241                                     CLC                             ;CLEAR CARRY
006206  032760  000002  002432                     BIT     #BIT1,CSRINFO(R0)       ;IS THIS PARITY MEMORY?
006214  001414                                     BEQ     5$                      ;BRACH IF NOT
006216  052760  000004  172100                     BIS     #BIT2,CSRADD(R0)        ;SET WRITE WRONG PARITY
006224  012771  123456  000000                     MOV     #123456,@(R1)           ;WRITE DATA
006232  012771  123456  000002                     MOV     #123456,@2(R1)
006240  005060  172100                             CLR     CSRADD(R0)              ;RESTORE CSR
006244  000414                                     BR      6$
006246  012760  000000  172100             5$:     MOV     #0,CSRADD(R0)           ;CLEAR THE CSR UNDER TEST
006254  012771  123456  000000                     MOV     #123456,@(R1)           ;WRITE DATA
006262  012771  123456  000002                     MOV     #123456,@2(R1)
006270  012760  020006  172100                     MOV     #20006,CSRADD(R0)       ;SET DIAG CHECK MODE
006276  005771  000000                     6$:     TST     @(R1)                   ;WRITE CHECKBITS TO CSR
006302  016004  172100                             MOV     CSRADD(R0),R4           ;WRITE CSR TO R4
006306  032760  000002  002432                     BIT     #BIT1,CSRINFO(R0)       ;PARITY MEMORY?
006314  001403                                     BEQ     7$                      ;BRACH IF NOT
006316  005704                                     TST     R4                      ;PARITY ERROR?
006320  100411                                     BMI     8$                      ;BRACH IF SO
006322  000720                                     BR      1$                      ;TRY NEXT CSR
006324  000240                             7$:     NOP                             ;DEBUG AID
006326  072427  177773                             ASH     #-5,R4
006332  042704  177600                             BIC     #^C177,R4
006336  022704  000157                             CMP     #157,R4                 ;CORRECT CHECKBITS?
006342  001310                                     BNE     1$                      ;BRANCH IF NOT
006344  010037  002502                     8$:     MOV     R0,PGMCSR
006350  000240                             3$:     NOP                             ;DEBUG AID
006352  104502                                     CLRCSR                          ;CLEAR ALL CSR'S
006354  012771  000000  000000                     MOV     #0,@(R1)                ;RESTORE TEST LOCATIONS
006362  012771  000000  000002                     MOV     #0,@2(R1)
006370  023727  002502  177777                     CMP     PGMCSR,#177777
006376  001402                                     BEQ     FINT                    ;IF PROGRAM CSR NOT FOUND GO TO FINT
006400  000137  007004                             JMP     CLRMEM                  ;GO TO SIZING ROUTINE IF FOUND
```

```
                                        ;
                                        ; IF PGMCSR WAS NOT FOUND BY THE PRECEEDING ROUTINE, THIS ROUTINE TRIES
                                        ;TO FIND IT FOR INTERLEAVED MEMORIES
                                        ;
006404                                  FINT:   SET4    #2$                     ;NE MEMORY TRAPS GO TO 2$
006412  012771  123456  000000          1$:     MOV     #123456,@(R1)           ;WRITE DATA AT FIRST LOCATION OF BANK 2 IN BOARD
006420  012771  123456  000002                  MOV     #123456,@2(R1)          ;WRITE DATA AT SECOND LOCATION OF BANK 2 IN BOARD
006426  062737  010000  172350                  ADD     #10000,KIPAR4           ;UPDATE PAR4 TO POINT TO UPPER BOARDS
006434  000766                                  BR      1$                      ;KEEP GOING TILL NO MORE MEMORY
006436  012700  177776                  2$:     MOV     #-2,R0
006442  013703  002216                          MOV     TOTCSRS,R3              ;PUT CSR MAP IN R3
006446  062700  000002                  3$:     ADD     #2,R0                   ;UPDATE CSR COUNTER
006452  010037  002146                          MOV     R0,CSRNO                ;UPDATE CSRNO
006456  006303                                  ASL     R3
006460  103403                                  BCS     4$                      ;BRANCH IF CSR EXISTS
006462  005703                                  TST     R3                      ;ANY CSR'S LEFT?
006464  001405                                  BEQ     5$                      ;BRANCH IF NOT
006466  000767                                  BR      3$                      ;LOOK FOR NEXT CSR
006470  012760  020006  172100          4$:     MOV     #20006,CSRADD(R0)       ;SET DIAGNOSTIC CHECK MODE IN CSR
006476  000763                                  BR      3$                      ;LOOK FOR NEXT CSR
006500                                  5$:     SET4    #6$                     ;NE MEMORY TRAPS NOW GO TO 6$
006506  012700  177776                          MOV     #-2,R0                  ;RESET CSR POINTER
006512  012737  002000  172350                  MOV     #2000,KIPAR4            ;REMAP PAR4 TO POINT TO BANK 2
006520  005771  000000                          TST     @(R1)                   ;TEST NONASSERTED LOCATIONS
006524  062700  000002                  6$:     ADD     #2,R0                   ;UPDTAE CSR POINTER
006530  010037  002146                          MOV     R0,CSRNO
006534  022700  000040                          CMP     #40,R0                  ;NOT FOUND?
006540  001515                                  BEQ     10$                     ;BRANCH IF NOT
006542  016004  172100                          MOV     CSRADD(R0),R4           ;GET CSR CONTENTS
006546  072427  177773                          ASH     #-5,R4
006552  042704  177600                          BIC     #^C177,R4               ;CLEAR ALL BUT CHECKBITS
006556  022704  000157                          CMP     #157,R4                 ;PROPER CHECKBITS?
006562  001401                                  BEQ     7$                      ;BRANCH IF SO
006564  000757                                  BR      6$                      ;TRY NEXT CSR IF NOT
006566  110037  002502                  7$:     MOVB    R0,PGMCSR               ;WRITE NON-ASSERTED CSR # IN PGMCSR
006572                                          SET4    #8$                     ;NE TRAPS GO TO 8$
006600  012700  177776                          MOV     #-2,R0
006604  013703  002216                          MOV     TOTCSRS,R3              ;PUT CSR MAP IN R3
006610  062700  000002                  23$:    ADD     #2,R0                   ;UPDATE CSR COUNTER
006614  010037  002146                          MOV     R0,CSRNO                ;UPDATE CSRNO
006620  006303                                  ASL     R3
006622  103403                                  BCS     24$                     ;BRANCH IF CSR EXISTS
006624  005703                                  TST     R3                      ;ANY CSR'S LEFT?
006626  001405                                  BEQ     25$                     ;BRANCH IF NOT
006630  000767                                  BR      23$                     ;LOOK FOR NEXT CSR
006632  012760  020006  172100          24$:    MOV     #20006,CSRADD(R0)       ;SET DIAGNOSTIC CHECK MODE IN CSR
006640  000763                                  BR      23$                     ;LOOK FOR NEXT CSR
006642  012700  177776                  25$:    MOV     #-2,R0
006646  005771  000002                          TST     @2(R1)                  ;TEST ASSERTED LOCATIONS
006652  062700  000002                  8$:     ADD     #2,R0
006656  010037  002146                          MOV     R0,CSRNO
006662  022700  000040                          CMP     #40,R0
006666  001442                                  BEQ     10$
006670  016004  172100                          MOV     CSRADD(R0),R4
006674  072427  177773                          ASH     #-5,R4
006700  042704  177600                          BIC     #^C177,R4
006704  022704  000157                          CMP     #157,R4                 ;PROPER CHECKBITS?
```

```
  1947 006710  001401                             BEQ     9$                        ;BRANCH IF SO
  1948 006712  000757                             BR      8$                        ;TRY NEXT CSR IF NOT
  1949 006714  110037  002503             9$:     MOVB    R0,PGMCSR+1               ;WRITE ASSERTED CSR # IN PGMCSR
  1950 006720  052737  100000  002502             BIS     #BIT15,PGMCSR             ;SET INTERLEAVED INDICATOR IN PGMCSR
  1951 006726  104502                             CLRCSR
  1952 006730  012737  002000  172350             MOV     #2000,KIPAR4
  1953 006736                                     SET4    #12$                      ;NE MEMORY TRAPS GO TO 12$
  1954 006744  012771  000000  000000     11$:    MOV     #0,@(R1)                  ;WRITE DATA AT FIRST LOCATION OF BANK 2 IN BOARD
  1955 006752  012771  000000  000002             MOV     #0,@2(R1)                 ;WRITE DATA AT SECOND LOCATION OF BANK 2 IN BOARD
  1956 006760  062737  010000  172350             ADD     #10000,KIPAR4             ;UPDATE PAR4 TO POINT TO UPPER BOARDS
  1957 006766  000766                             BR      11$
  1958 006770  104423                     12$:    CACHON
  1959 006772  000404                             BR      CLRMEM
  1960 006774                             10$:    TYPE    MSG126                    ;ERROR - PROGRAM CSR NOT FOUND!
  1961 007000  005037  002502                     CLR     PGMCSR                    ;SET TO DEFAULT OF 0
```

```
007004                                 SUBTST  <<CLEAR ALL MEMORY SPACE FROM BANK 2 ON>>
                               ;******************************************************************************
                               ;*SUBTEST        CLEAR ALL MEMORY SPACE FROM BANK 2 ON
                               ;******************************************************************************

                               ;
                               ;THIS ROUTINE CLEARS ALL MEMORY SPACE BEGINNING AT ADDRESS 200,000 AND
                               ;CONTINUES UNTIL THERE IS NO MEMORY LEFT. IT SHOULD CLEAR ANY PARITY ERRORS
                               ;CREATED BY THE LAST ROUTINE, AND CLEAN UP ANY JUNK LEFT HANGING AROUND IN
                               ;HIGHER MEMORY.
                               ;
007004                         CLRMEM: SET4    #CLREX          ;NONEM TRAPS GO TO CLREX
007012  005037  006104                 CLR     TRACE
007016  012737  000001  002074         MOV     #1,NOPAR        ;IGNORE PARITY ERRORS
007024  012737  002000  172350         MOV     #2000,KIPAR4    ;SET UP MAP TO START AT BANK 2
007032  012701  100000                 MOV     #100000,R1      ;R1 MAPS TO KIPAR4
007036  020127  177776         1$:     CMP     R1,#177776      ;WHOLE 16K BANK DONE?
007042  001003                         BNE     2$              ;KEEP GOING IF NOT
007044  012737  177777  006104         MOV     #-1,TRACE       ;USE TRACE FLAG TO FLAG END OF BANK
007052  005021                 2$:     CLR     (R1)+           ;CLEAR CONTENTS & INCREMENT
007054  005737  006104                 TST     TRACE           ;EOB FLAG SET?
007060  001001                         BNE     3$              ;GO TO NEXT BANK IF SO
007062  000765                         BR      1$
007064  062737  001000  172350 3$:     ADD     #1000,KIPAR4    ;SET MAP FOR NEXT BANK
007072  005037  006104                 CLR     TRACE           ;RESET FLAG
007076  012701  100000                 MOV     #100000,R1      ;RESET R1
007102  000755                         BR      1$              ;CLEAR NEXT BANK
007104  000240                 CLREX:  NOP
007106  005037  006104                 CLR     TRACE
007112                                 RES4
```

```
007134                                  ANA2:   SUBTST  <<MATCH ALL CSR'S WITH MEMORY>>
                                        ;*******************************************************************************
                                        ;*SUBTEST        MATCH ALL CSR'S WITH MEMORY
                                        ;*******************************************************************************
                                        ;* THE SECOND PART OF THE ANALYSIS MATCHES UP THE CSR'S WITH THE MEMORY, AND
                                        ;* INSTALLS ALL THE INFORMATION FOUND IN THE CONFIGURATION TABLE.  FOR ECC,
                                        ;* THIS IS DONE BY TAKING EACH CSR FOUND IN THE PREVIOUS SECTION SEQUENTIALLY
                                        ;* AND CHECKING THROUGH ALL OF MEMORY, ONE BANK AT AT TIME, TO SEE WHICH BANKS
                                        ;* RESPOND TO THE CSR IN QUESTION. THE FIRST DOUBLE WORD PAIR IN EACH BANK ARE
                                        ;* WRITTEN WITH DATA AND DIAGNOSTIC CHECK MODE SET IN THE CSR IN ORDER TO AC-
                                        ;* COMPLISH THIS. ALL POSSIBLE CONFIGURATIONS OF DOUBLE WORD PAIRS (NON-INTER-
                                        ;* LEAVED, 64K INTERLEAVED, OR 128K INTERLEAVED) ARE CHECKED FOR EACH BANK
                                        ;* THROUGH USE OF TESTADD AND KERNEL INSTRUCTION PAGE ADDRESS REGISTERS 4 AND
                                        ;* 5. IF WE GET THE PROPER CHECKBITS BACK, WE HAVE A MATCH. IF NOT, THE ROUT-
                                        ;* INE CHECKS FOR SINGLE OR DOUBLE BIT ERRORS.
                                        ;*                IF ONE OR THE OTHER IS FOUND, THE ERROR ADDRESS IS CHECKED
                                        ;* TO SEE IF IT IS THAT BANK.  IF IT IS, WE HAVE A MATCH.  AT THE END OF EACH
                                        ;* BANK PASS, FOR EACH CSR PASS, THE PROGRAM COMES UP WITH A NUMBER, STORED IN
                                        ;* "I", WHICH DENOTES THE FOLLOWING:
                                        ;*
                                        ;*        I        MEMORY DESCRIPTION
                                        ;*        -        ------------------
                                        ;*        0        NON-EXISTANT MEMORY
                                        ;*        1        NON-INTERLEAVED MEMORY
                                        ;*        2        64K INTERLEAVED, A1 NOT ASSERTED MEMORY
                                        ;*        3        128K INTERLEAVED, A1 NOT ASSERTED MEMORY
                                        ;*        4        64K INTERLEAVED, A1 ASSERTED MEMORY
                                        ;*        5        128K INTERLEAVED, A1 ASSERTED MEMORY
                                        ;*
                                        ;*  NOTE - I=2 THROUGH I=5 CAN ONLY OCCUR WITH MS11-M MEMORY.
                                        ;*
                                        ;* NOTE THAT PARITY MEMORY WRITES WRONG PARITY TO THE DOUBLE WORDS, THEN LOOKS
                                        ;* FOR THE PARITY ERROR BIT TO BE SET.  IF THE BIT IS SET, WE HAVE A MATCH.
                                        ;*
007134                                          SET4    #6$                     ;NE MEMORY TRAPS GO TO 4
007142  005037  002274                          CLR     CHECK                   ;CLEAR CHECK
007146  012701  002362                          MOV     #TESTADD,R1             ;SET UP THE VIRTUAL ADDR. POINTER
007152  013703  002216                          MOV     TOTCSRS,R3              ;MOVE CSR MAP INTO R3
007156  005000                                  CLR     R0                      ;CLEAR THE CSR POINTER
007160  005005                                  CLR     R5                      ;CLEAR THE PROGRAM CSR STATUS POINTER
007162  005737  002424                          TST     NO22BIT                 ;IS THIS AN 11/44 OR 11/24?
007166  001403                                  BEQ     7$                      ;BRANCH IF IT IS
007170  005037  002530                          CLR     LASTBLOCK               ;ADJUST LASTBLOCK INDICATOR FOR 124K MACHINE
007174  000413                                  BR      1$                      ;BRANCH OVER NEXT PIECE OF CODE
007176  022737  000167  002526  7$:     CMP     #167,LASTBANK           ;IS THERE UNIBUS MEMORY ABOVE 17000000?
007204  001407                                  BEQ     1$                      ;BRANCH IF NOT
007206  013702  002526                          MOV     LASTBANK,R2             ;SET UP A NEW LAST BLOCK INDICATOR
007212  005202                                  INC     R2
007214  072227  000011                          ASH     #9.,R2
007220  010237  002530                          MOV     R2,LASTBLOCK
007224  012702  000004          1$:     MOV     #4,R2                   ;R2 IS INDEX FOR CONFIG TABLE
007230  012737  001000  172350          MOV     #1000,KIPAR4            ;SET KIPAR4 FOR BANK 1
007236  012737  001000  172352          MOV     #1000,KIPAR5            ;SET KIPAR5 FOR BANK 1
007244  006303                  2$:     ASL     R3                      ;DOES THIS CSR EXIST?
007246  103420                                  BCS     3$                      ;BRANCH IF IT DOES EXIST
007250  062700  000002                          ADD     #2,R0                   ;INCREMENT THE CSR POINTER
007254  010037  002146                          MOV     R0,CSRNO                ;STORE IT IN CSRNO ALSO
```

```
007260  005703                          TST     R3                      ;ARE THERE ANY MORE CSR'S TO DO?
007262  001370                          BNE     2$                      ;BRANCH IF ALL CSRS NOT DONE
007264  012737  001000  172350          MOV     #1000,KIPAR4            ;RESTORE KIPAR4
007272  012737  001200  172352          MOV     #1200,KIPAR5            ;RESTORE KIPAR5
007300  013706  002534                  MOV     KSTACK,SP               ;RESTORE STACK
007304  000137  010474                  JMP     SUBAAS                  ;JUMP TO SUBAAS IF ALL CSR'S ARE DONE
007310  010037  002146          3$:     MOV     R0,CSRNO                ;MAKE SURE CSRNO IS UPDATED
007314  104424                  13$:    CACHOFF                         ;TURN THE CACHE OFF
007316  000240                          NOP
007320  012737  100000  002362  45$:    MOV     #100000,TESTADD         ;SET UP VIRTUAL ADDRESS TO KIPAR4
007326  012737  120002  002364          MOV     #120002,TESTADD+2       ;SET UP VIRTUAL ADDRESS TO KIPAR5
007334  032762  000040  002624          BIT     #BIT5,CONFIG(R2)        ;IS THIS A BANK TO SKIP?
007342  001402                          BEQ     43$                     ;NO - BRANCH AROUND NEXT INSTRUCTION
007344  000137  010420                  JMP     6$                      ;YES - GO TO END OF BANK
007350  005037  002422          43$:    CLR     I                       ;CLEAR THE MEMORY CONFIGURATION COUNTER
007354  005771  000000          4$:     TST     @(R1)                   ;TEST TO SEE THAT THERE IS MEMORY PRESENT
007360  005237  002422                  INC     I
007364                                  PUSH    @(R1),@2(R1)            ;SAVE THE LOCATIONS UNDER TEST
007374  032760  000002  002432          BIT     #BIT1,CSRINFO(R0)       ;IS THIS PARITY MEMORY?
007402  001414                          BEQ     34$                     ;NO - BRANCH
007404  052760  000004  172100          BIS     #BIT2,CSRADD(R0)        ;SET WRITE WRONG PARITY
007412  012771  123456  000000          MOV     #123456,@(R1)           ;SET THE FIRST LOCATION UNDER TEST
007420  012771  123456  000002          MOV     #123456,@2(R1)          ;SET THE SECOND LUT
007426  005060  172100                  CLR     CSRADD(R0)              ;CLEAR THE CSR
007432  000411                          BR      41$                     ;TEST LOCATIONS
007434  012771  123456  000000  34$:    MOV     #123456,@(R1)           ;SET THE FIRST LOCATION UNDER TEST
007442  012771  123456  000002          MOV     #123456,@2(R1)          ;SET THE SECOND LUT
007450  104503                          CLR1CSR                         ;RESET CSR
007452  104475                          CB1CSR                          ;SET DIAG. CHECK MODE IN CSR UNDER TEST
007454  000240                          NOP                             ;DEBUG AID
007456  005771  000000          41$:    TST     @(R1)                   ;READ THE FIRST LUT TO WRITE CKBITS. INTO CSR
007462  104426                          READCSR                         ;READ THE CSR UNDER TEST
007464  000240                          NOP                             ;DEBUG AID
007466  013704  002144                  MOV     CSR,R4                  ;GET THE CHECKBITS FROM THE CSR
007472  000240                          NOP                             ;DEBUG AID
007474  010437  002404                  MOV     R4,TEMP                 ;SAVE IN TEMP FOR LATER
007500  104503                          CLR1CSR                         ;RESET CSR
007502                                  POP     @2(R1),@(R1)            ;RESTORE LOCATIONS UNDER TEST
007512  032760  000002  002432          BIT     #BIT1,CSRINFO(R0)       ;IS THIS PARITY MEMORY?
007520  001404                          BEQ     42$                     ;NO - BRANCH
007522  005704                          TST     R4                      ;DID WE GET A PARITY ERROR?
007524  100414                          BMI     25$                     ;YES - FILL IN CONFIG TABLE
007526  000137  010420                  JMP     6$                      ;NO - JUMP TO END OF BANK
007532  072427  177773          42$:    ASH     #-5,R4                  ;MANIPULATE THE CSR BITS
007536  042704  177600                  BIC     #^C177,R4               ;INTO A USABLE FORM.
007542  000240                          NOP                             ;DEBUG AID
007544  022704  000157                  CMP     #157,R4                 ;DO THE CHECKBITS COMPARE TO WHAT WAS WRITTEN?
007550  001402                          BEQ     25$                     ;BRANCH IF THERE IS A MATCH
007552  000137  010100                  JMP     22$                     ;ELSE BRANCH IF NOT THE SAME
                                ;*
                                ;* WE COME HERE IF THERE IS A MATCH
                                ;*
007556  010004                  25$:    MOV     R0,R4                   ;GET THE CSR NUMBER
007560  006204                          ASR     R4                      ;SET IT UP FOR USE IN THE
007562  000304                          SWAB    R4                      ;CONFIGURATION TABLE.
007564  042704  170377                  BIC     #170377,R4              ;CLEAR OFF EXTRANEOUS BITS
007570  032737  000004  002422          BIT     #BIT2,I                 ;INTERLEAVED A1 ASSERTED MEMORY FOUND?
```

```
007576  001402                             BEQ     15$                       ;BRANCH IF NOT
007600  072427  000004                     ASH     #4,R4                     ;PUT CSR NUMBER IN INTERLEAVED CSR SLOT
007604  050462  002624           15$:      BIS     R4,CONFIG(R2)             ;PUT CSR NUMBER IN CONFIG. TABLE
007610  016004  002432                     MOV     CSRINFO(R0),R4            ;GET MEMORY TYPE
007614  042704  177770                     BIC     #^C7,R4                   ;CLEAR OFF THE EXTRANEOUS BITS
007620  000304                             SWAB    R4                        ;MOVE INTO PROPER POSITION
007622  050462  002626                     BIS     R4,CONFIG+2(R2)           ;SET IT INTO THE CONFIG TABLE
007626  022737  000001  002422             CMP     #1,I                      ;WAS THIS NON-INTERLEAVED MEMORY?
007634  001431                             BEQ     24$                       ;BRANCH IF IT WAS
007636  052762  010000  002626             BIS     #BIT12,CONFIG+2(R2)       ;SET THE INTERLEAVED BIT
007644  010204                             MOV     R2,R4                     ;SAVE THE CURRENT BANK INDEX
007646  032737  000001  002422             BIT     #BIT0,I                   ;WAS THIS 128K INTERLEAVED?
007654  001006                             BNE     5$                        ;BRANCH IF TRUE
007656  052762  004000  002626             BIS     #BIT11,CONFIG+2(R2)       ;SET 64K INTERLEAVED FLAG IN CONFIG
007664  062704  000020                     ADD     #20,R4                    ;SET NEW BANK POINTER TO 4 BANKS AHEAD
007670  000402                             BR      16$                       ;JUMP OVER NEXT INSTRUCTION
007672  062704  000040           5$:       ADD     #40,R4                    ;SET NEW BANK POINTER 8 BANKS AHEAD
007676  052764  000040  002624   16$:      BIS     #BIT5,CONFIG(R4)          ;SET SKIP ECC LOGIC TESTS FLAG
007704  056264  002624  002624             BIS     CONFIG(R2),CONFIG(R4)     ;SET OTHER INFO INTO THAT BANK
007712  056264  002626  002626             BIS     CONFIG+2(R2),CONFIG+2(R4)
                                           ;*
                                           ;* THIS SECTION IS EXECUTED ONLY WHEN THE BANK=1
                                           ;*
007720  022737  001000  172350   24$:      CMP     #1000,KIPAR4              ;IS THIS BANK 1 ?
007726  001402                             BEQ     30$                       ;BRANCH IF TRUE
007730  000137  010260                     JMP     33$                       ;ELSE JUMP TO END OF THIS BANK
007734  032737  100020  002404   30$:      BIT     #BIT15!BIT4,TEMP          ;WAS THERE A SBE OR DBE?
007742  001417                             BEQ     10$                       ;BRANCH IF NOT
007744  013704  002404                     MOV     TEMP,R4                   ;GET CSR CONTENTS
007750  072427  177767                     ASH     #-9.,R4                   ;MAKE ERROR ADDRESS INTO BANK #
007754  022704  000001                     CMP     #1,R4                     ;ERROR IN BANKS 0 OR 1?
007760  003010                             BGT     10$                       ;BRANCH IF NOT
007762  052762  000001  002624             BIS     #BIT0,CONFIG(R2)          ;SET ERROR FLAG IN CONFIG TABLE
007770  105262  002626                     INCB    CONFIG+2(R2)              ;ADD ONE TO BANK ERROR COUNT
007774                                     SET     CONFGERROR                ;PRINT CONFIG TABLE
010002  053737  002630  002624   10$:      BIS     CONFIG+4,CONFIG           ;SET UP INFORMATION IN BANK ZERO
010010  053737  002632  002626             BIS     CONFIG+6,CONFIG+2
010016  000240                             NOP                               ;DEBUG AID
010020  022737  000001  002422             CMP     #1,I                      ;WAS THIS NON-INTERLEAVED MEMORY
010026  001002                             BNE     46$                       ;NO - BRANCH OVER NEXT STMT.
010030  000137  010420                     JMP     6$                        ;YES - JUMP TO END OF THIS BANK
010034  012704  000020           46$:      MOV     #20,R4                    ;SET UP COUNTER FOR 64K INTERLEAVED
010040  032737  000001  002422             BIT     #BIT0,I                   ;WAS IT 128K INTERLEAVED?
010046  001402                             BEQ     26$                       ;BRANCH IF NOT
010050  062704  000020                     ADD     #20,R4                    ;SET UP COUNTER FOR 128K INTERLEAVED
010054  053764  002624  002624   26$:      BIS     CONFIG,CONFIG(R4)         ;SET OTHER BANK WITH SAME INFORMATION
010062  053764  002626  002626             BIS     CONFIG+2,CONFIG+2(R4)     ;AS IN BANK 0
010070  052764  000040  002624             BIS     #BIT5,CONFIG(R4)          ;SET SKIP ECC LOGIC TESTS FLAG
010076  000470                             BR      33$                       ;BRANCH
                                           ;*
                                           ;* IF CHECKBITS DID NOT MATCH, WE COME HERE
                                           ;*
010100  032737  100020  002144   22$:      BIT     #BIT15!BIT4,CSR           ;SBE OR DBE FLAGS SET?
010106  001001                             BNE     8$                        ;BRANCH IF TRUE
010110  000463                             BR      33$                       ;CHECK TO SEE IF IT IS MS11-M
010112  013704  002146           8$:       MOV     CSRNO,R4                  ;GET CSRNO
010116  042764  000006  172100             BIC     #6,CSRADD(R4)             ;TURN OFF DIAG CHECK & ECC DISABLE
```

```
010124                                  PUSH    R0,R1                   ;SAVE R0 & R1
010130  016401  172100                  MOV     CSRADD(R4),R1           ;GET CSR INFORMATION
010134  072127  177773                  ASH     #-5,R1                  ;SET UP ERROR ADDRESS
010140  042701  177600                  BIC     #^C177,R1
010144  005737  002424                  TST     NO22BIT                 ;IS THIS AN 11/44 OR 11/24?
010150  001015                          BNE     27$                     ;BRANCH IF NOT
010152  052764  040000  172100          BIS     #BIT14,CSRADD(R4)       ;GET EXTENDED ERROR ADDRESS BITS
010160  016400  172100                  MOV     CSRADD(R4),R0           ;READ FROM CSR
010164  042764  040000  172100          BIC     #BIT14,CSRADD(R4)       ;TURN OFF EUB BIT
010172  042700  177037                  BIC     #^C740,R0               ;SET UP EXTENDED BITS
010176  006300                          ASL     R0
010200  006300                          ASL     R0
010202  060001                          ADD     R0,R1                   ;SET UP TOTAL ERROR ADDRESS
010204  010104                  27$:    MOV     R1,R4                   ;SAVE IN R4
010206                                  POP     R1,R0                   ;RESTORE R0 & R1
010212  072427  000005                  ASH     #5,R4                   ;SET ERROR ADDRESS UP IN PAR NOTATION
010216  020437  172350                  CMP     R4,KIPAR4               ;DOES IT EQUAL KIPAR4?
010222  001001                          BNE     28$                     ;BRANCH IF FALSE
010224  000403                          BR      35$                     ;YES - MARK INFO IN CONFIG TABLE
010226  020437  172352          28$:    CMP     R4,KIPAR5               ;DOES IT EQUAL KIPAR5?
010232  001012                          BNE     33$                     ;BRANCH IF FALSE
010234  052762  000001  002624  35$:    BIS     #BIT0,CONFIG(R2)        ;SET BANK ERROR FLAG
010242  105262  002626                  INCB    CONFIG+2(R2)            ;INCREMENT BANK ERROR COUNTER
010246                                  SET     CONFGERROR              ;PRINT CONFIG TABLE
010254  000137  007556                  JMP     25$                     ;YES - MARK INFO IN CONFIG TABLE
                                        ;*
                                        ;* THIS SECTION SETS UP ALL THE POSSIBLE CONFIGURATIONS OF
                                        ;* MS11-M MEMORY.
                                        ;*
010260  032760  000003  002432  33$:    BIT     #BIT0!BIT1,CSRINFO(R0)  ;IS THIS MS11-M MEMORY?
010266  001054                          BNE     6$                      ;NO - GO TO END OF BANK
010270  032760  000004  002432          BIT     #BIT2,CSRINFO(R0)
010276  001450                          BEQ     6$
010300  022737  000001  002422          CMP     #1,I                    ;IS THIS 1ST TIME THROUGH?
010306  103410                          BLO     18$                     ;BRANCH IF NOT
010310  162737  000002  002364          SUB     #2,TESTADD+2            ;TRY AS 64K INTERLEAVED
010316  062737  004000  172352          ADD     #4000,KIPAR5            ;A1 NON-ASSERTED MEMORY
010324  000137  007354                  JMP     4$                      ;TRY TO MATCH AGAIN
010330  022737  000004  002422  18$:    CMP     #4,I                    ;4TH TIME THROUGH?
010336  001404                          BEQ     20$                     ;YES - BRANCH
010340  022737  000002  002422          CMP     #2,I                    ;2ND TIME THROUGH
010346  103405                          BLO     12$                     ;NO - BRANCH
010350  062737  004000  172352  20$:    ADD     #4000,KIPAR5            ;TRY AS 128K INTERLEAVED
010356  000137  007354                  JMP     4$                      ;TRY TO MATCH AGAIN
010362  022737  000003  002422  12$:    CMP     #3,I                    ;THIRD TIME THROUGH?
010370  103413                          BLO     6$                      ;NO - BRANCH
010372  062737  000002  002362          ADD     #2,TESTADD              ;TRY TESTING THE BANK
010400  062737  000002  002364          ADD     #2,TESTADD+2            ;AS A1 ASSERTED
010406  162737  004000  172352          SUB     #4000,KIPAR5            ;64K INTERLEAVED MEMORY
010414  000137  007354                  JMP     4$                      ;TRY TO MATCH AGAIN
                                        ;*
                                        ;*END OF BANK ROUTINE
                                        ;*
010420  104503                  6$:     CLR1CSR                         ;CLEAR THE CSR UNDER TEST
010422  062702  000004                  ADD     #4,R2                   ;UPDATE CONFIGURATION POINTER
010426  062737  001000  172350          ADD     #1000,KIPAR4            ;UPDATE KIPAR4 TO NEXT BANK
010434  013737  172350  172352          MOV     KIPAR4,KIPAR5           ;AND UPDATE KIPAR5
```

```
010442  000240                             NOP                          ;DEBUG AID
010444  023737  002530  172350             CMP     LASTBLOCK,KIPAR4     ;HAVE WE DONE THE WHOLE MEMORY SPACE?
010452  002002                             BGE     19$                  ;BRANCH IF DONE
010454  000137  007320                     JMP     45$                  ;JUMP IF NOT DONE
010460  062700  000002          19$:       ADD     #2,R0                ;INCREMENT CSR POINTER
010464  000240                             NOP                          ;DEBUG AID
010466  104423                             CACHON                       ;TURN ON THE CACHE
010470  000137  007224                     JMP     1$                   ;JUMP TO TRY NEXT CSR
```

```
010474  104423                      SUBAAS: CACHON          ;MAKE SURE THE CACHE IS ON
010476  104472                              ECCINIT         ;TRAP ON DOUBLE BIT ERRORS (NORMAL)
010500                                      NEWTST  <<TEST BANK 0 ACCESSES>>
                                    ;*******************************************************************
                                    ;*TEST 2        TEST BANK 0 ACCESSES
                                    ;*******************************************************************
010500  000004                      TST2:   SCOPE
                                            ;THIS DOES A "TST" INSTRUCTION ON EVERY LOCATION IN BANK #0 TO SEE
                                            ;IF IT GETS ANY PARITY TRAPS.
                                            ;SINCE EVERY LOCATION IS EITHER LOADED OR WRITTEN INTO BY THE PROGRAM
                                            ;PRIOR TO THIS POINT - THEN A PARITY ERROR IMPLIES THAT THERE IS A
                                            ;HARDWARE FAILURE IN THE MEMORY.
                                            ;THESE ERRORS ARE COUNTED AND A FATAL ACTION IS TAKEN
010502  005037  002070                      CLR     PARCNT                  ;CLEAR PARITY ERROR COUNTER
010506  012737  000001  002074              MOV     #1,NOPAR                ;SET THE NO PARITY ERROR FLAG
010514  005037  002066                      CLR     NEMCNT                  ;CLEAR NON-EXISTANT MEMORY ERROR COUNTER
010520  012737  000001  002076              MOV     #1,NONEM                ;SET THE NON-EXISTANT MEMORY ERROR MODE TO COUNT
010526                                      SET4    #NONEXIST               ;TRAPS TO 4 GOTO NONEXIST
010534  005000                              CLR     R0
010536  012701  040000                      MOV     #SIZE,R1
010542  104424                              CACHOFF                         ;TURN CACHE OFF
010544  005720                      1$:     TST     (R0)+                   ;SEE IF I CAN DO A READ ACCESS WITHOUT A PARITY TRAP
010546  077102                              SOB     R1,1$
010550  104423                              CACHON                          ;TURN CACHE ON
                                            ;SEE IF ANY FAILURES
010552  005737  002070                      TST     PARCNT                  ;ANY PARITY ERRORS?
010556  001403                              BEQ     2$                      ;NO - SKIP
010560                                      FATAL   3
010566  005737  002066              2$:     TST     NEMCNT                  ;ANY NON-EXISTANT MEMORY (HOLES)?
010572  001406                              BEQ     3$                      ;SKIP IF EQUAL
010574  162737  000002  002032              SUB     #2,ADDRESS              ;UPDATE 1ST ADDRESS FAILURE FROM AUTO INCREMENT #
010602                                      FATAL   4
010610  053737  002104  002624      3$:     BIS     CPUBIT,CONFIG           ;SET CORRECT ACCESSED BIT ON BANK 0
010616                                      RES4                            ;RESET TRAPS TO 4 TO DEFAULT

010640                                      SUBTST  <<ENABLE ECC FOR CORRFCT TRAPS>>
                                    ;*******************************************************************
                                    ;*SUBTEST       ENABLE ECC FOR CORRECT TRAPS
                                    ;*******************************************************************
010640                                      IF #SW0 SET.IN @SWR OR ACTFLAG IS TRUE
010656  104506                                ENASBE                        ;TRAP ON SINGLE BIT ERRORS
010660                                      ELSE
010662  104472                                ECCINIT                       ;TRAP ON DOUBLE BIT ERRORS (NORMAL)
010664                                      END ;OF IF #SW0
```

```
 2264 010664                                     NEWTST  <<TEST BANKS 1-200 (OCTAL) FOR ZEROS & ONES>>
                                        ;*********************************************************************************
                                        ;*TEST 3        TEST BANKS 1-200 (OCTAL) FOR ZEROS & ONES
                                        ;*********************************************************************************
      010664  000004                    TST3:   SCOPE
 2265                                           ;EACH BANK IS TESTED FOR EXISTANCE AND IF IT EXISTS
 2266                                           ;THEN IS IS TESTED FOR ZEROS & ONES.
 2267                                           ;EXCEPT -
 2268                                           ;       PROTECTED BANKS (WHERE THE PROGRAM IS) ARE ONLY TESTED BY
 2269                                           ;       "TST" INSTRUCTIONS LIKE BANK #0
 2270                                           ;ANY BAD BANKS ARE LOGGED IN THE CONFIGURATION TABLE.
 2271                                           ;THIS ROUTINE IS ONLY DOING A SMART SIZE - NOT ACTUAL TESTING!
 2272 010666  005037  002100                    CLR     BANK
 2273 010672  012737  000001  002074            MOV     #1,NOPAR                ;SET NO PARITY ERROR FLAG
 2274 010700  012737  000002  002076            MOV     #2,NONEM                ;SET NON-EXISTANT MEMORY MODE TO EXIT TEST LOOP
 2275 010706                                    SET4    #NONEXIST               ;TRAPS TO 4 GOTO NONEXIST
 2276 010714  022737  000001  003710            CMP     #1,PROTYP               ;IS THIS AN 11/44?
 2277 010722  001407                            BEQ     1$                      ;BRANCH IF TRUE
 2278 010724  012737  011522  002472            MOV     #MTST3+4,LINK1          ;SET UP LINKS
 2279 010732  012737  011524  002474            MOV     #MTST3+6,LINK2
 2280 010740  000411                            BR      TAG9$
 2281 010742                            1$:     BMOV    MTST3                   ;PUT IN FAST MEMORY
 2282 010750  012737  177644  002472            MOV     #UIPAR2,LINK1           ;SET UP LINKS
 2283 010756  012737  177646  002474            MOV     #UIPAR3,LINK2
 2284 010764  005237  002100            TAG9$:  INC     BANK
 2285 010770  023737  002526  002100            CMP     LASTBANK,BANK           ;DONE?
 2286 010776  103457                            BLO     TAG2$                   ;YES - SKIP TO NEXT TEST
 2287 011000  013701  002100                    MOV     BANK,R1
 2288 011004  006301                            ASL     R1
 2289 011006  006301                            ASL     R1                      ;BANK * 4
 2290 011010  010137  002102                    MOV     R1,BANKINDEX
 2291 011014  005037  002072                    CLR     PATERR                  ;CLEAR PATTERN ERROR COUNTER
 2292 011020  005037  002070                    CLR     PARCNT                  ;CLEAR PARITY ERROR COUNTER
 2293 011024  005037  002066                    CLR     NEMCNT                  ;CLEAR NON-EXISTANT MEMORY COUNTER (HOLES)
 2294 011030                                    MAP     BANK                    ;MAP SUPERVISOR SPACE (TEST AREA) TO BANK
 2295 011044  105761  002624                    TSTB    CONFIG(R1)              ;IS THIS BANK PROTECTED?
 2296 011050  100555                            BMI     TSTBANK                 ;YES - GO TEST BANK SPECIAL
 2297 011052  012777  000207  171412  WARN1:  MOV     #207,@LINK1             ;PUT "RETURN" INSTRUCTION AFTER WRITE ROUTINE
 2298 011060  012700  060000                    MOV     #FIRST,R0
 2299 011064  010004                            MOV     R0,R4
 2300 011066  012701  040000                    MOV     #SIZE,R1
 2301 011072  010103                            MOV     R1,R3
 2302 011074  005002                            CLR     R2                      ;DATA IS ZEROS
 2303 011076  104424                            CACHOFF                         ;TURN CACHE OFF
 2304 011100                                    TESTAREA                        ;ENTER SUPERVISOR MODE
 2305 011106  022737  000001  003710            CMP     #1,PROTYP               ;IS THIS AN 11/44?
 2306 011114  001403                            BEQ     1$                      ;BRANCH IF TRUE
 2307 011116  004737  011516                    CALL    MTST3
 2308 011122  000402                            BR      2$
 2309 011124  004737  177640            1$:     CALL    FASTCITY                ;CALL TO THE USER INSTRUCTION PAR'S
 2310 011130  104417                    2$:     KERNEL                          ;ENTER KERNEL MODE
 2311 011132  104423                            CACHON                          ;TURN CACHE ON
 2312 011134  000416                            BR      TAG3$                   ;SKIP NEXT INSTRUCTION
 2313 011136  005037  002100            TAG2$:  CLR     BANK
 2314 011142                                    RES4                            ;RESET TRAPS TO 4 TO DEFAULT
 2315 011164  005037  002074                    CLR     NOPAR                   ;INDICATE DEFAULT PARITY ACTION
 2316 011170  000564                            BR      SUBAAI
```

```
011172 005737 002066          TAG3$:   TST     NEMCNT                  ;ANY TRAPS?
011176 001401                          BEQ     1$                      ;NO - SKIP
011200 000671                          BR      TAG9$                   ;NOW - TRY NEXT BANK
011202 104424                 1$:      CACHOFF                         ;TURN CACHE OFF
011204                                 TESTAREA                        ;ENTER SUPERVISOR MODE
011212 004777 171256                   CALL    @LINK2                  ;FINISH PATTERN
011216 104417                          KERNEL                          ;ENTER KERNEL MODE
011220 104423                          CACHON                          ;TURN CACHE ON
011222 005737 002072                   TST     PATERR                  ;ANY PATTERN ERRORS
011226 001040                          BNE     2$                      ;YES - SKIP
011230 005737 002070                   TST     PARCNT                  ;ANY PARITY ERRORS
011234 001035                          BNE     2$                      ;YES - SKIP
011236 005737 002066                   TST     NEMCNT                  ;ANY NON EXISTANT MEMORY
011242 001032                          BNE     2$                      ;YES - SKIP
011244 012700 060000                   MOV     #FIRST,R0
011250 010004                          MOV     R0,R4
011252 012701 040000                   MOV     #SIZE,R1
011256 010103                          MOV     R1,R3
011260 013702 002554                   MOV     ONES,R2                 ;DATA IS ONES
011264 012777 000240 171200            MOV     #000240,@LINK1          ;PUT ''NOP'' INSTRUCTION BACK IN SUBROUTINE
011272 104424                          CACHOFF                         ;TURN CACHE OFF
011274                                 TESTAREA                        ;ENTER TEST MODE
011302 022737 000001 003710            CMP     #1,PROTYP               ;IS THIS AN 11/44?
011310 001403                          BEQ     5$                      ;BRANCH IF IT IS
011312 004737 011516                   CALL    MTST3                   ;DO IN MEMORY IF NOT
011316 000402                          BR      6$                      ;JUMP OVER NEXT INSTRUCTION
011320 004737 177640          5$:      CALL    FASTCITY                ;CALL TO THE USER INSTRUCTION PAR'S
011324 104417                 6$:      KERNEL                          ;ENTER KERNEL MODE
011326 104423                          CACHON                          ;TURN CACHE ON
011330 013700 002102          2$:      MOV     BANKINDEX,R0
011334 005737 002072                   TST     PATERR                  ;ANY PATTERN ERRORS?
011340 001006                          BNE     3$                      ;YES - SKIP
011342 005737 002070                   TST     PARCNT                  ;ANY PARITY ERRORS?
011346 001003                          BNE     3$                      ;YES - SKIP
011350 005737 002066                   TST     NEMCNT                  ;ANY HOLES?
011354 001406                          BEQ     4$                      ;NONE - SKIP
011356 052760 000001 002624   3$:      BIS     #BIT0,CONFIG(R0)        ;SET ERROR BIT IN THIS BANK
011364                                 SET     CONFGERROR              ;FORCE PRINTING OF CONFIGURATION TABLE
011372 053760 002104 002624   4$:      BIS     CPUBIT,CONFIG(R0)       ;SET ACCESSED BIT
011400 000137 010764                   JMP     TAG9$


                                       ;TEST A PROTECTED BANK
011404                        TSTBANK:PUSH     R1
011406 012737 000001 002076            MOV     #1,NONEM                ;SET NON-EXISTANT MEMORY TO COUNT
011414 012700 060000                   MOV     #FIRST,R0
011420 012701 020000                   MOV     #20000,R1
011424 104424                          CACHOFF                         ;TURN CACHE OFF
011426                                 TESTAREA                        ;ENTER TEST MODE
011434 005720                 4$:      TST     (R0)+
011436 077102                          SOB     R1,4$
011440 104417                          KERNEL                          ;ENTER KERNEL MODE
011442 104423                          CACHON                          ;TURN CACHE ON
011444 012737 000002 002076            MOV     #2,NONEM                ;RESET NON-EXISTANT MEMORY TO EXIT TEST LOOP
011452                                 POP     R1
011454                                 IF PARCNT NE #0
011462 052761 000001 002624              BIS   #BIT0,CONFIG(R1)        ;ERROR BANK
011470                                   SET   CONFGERROR
```

```
 2374 011476                                        END ;OF IF PARCNT
 2375 011476                                        IF NEMCNT EQ #0
 2376 011504  053761  002104  002624                  BIS   CPUBIT,CONFIG(R1)          ;ACCESSED BANK
 2377 011512                                        END ;OF IF NEMCNT
 2378 011512  000137  010764                        JMP     TAG9$
 2379 011516  010220                       MTST3:   MOV     R2,(R0)+                   ;V177640
 2380 011520  077102                                SOB     R1,MTST3                   ;V177642
 2381 011522  000240                                NOP                                ;V177644
 2382 011524  012401                       2$:      MOV     (R4)+,R1                   ;V177646
 2383 011526  020102                                CMP     R1,R2                      ;V177650
 2384 011530  001402                                BEQ     3$                         ;V177652
 2385 011532  005237  002072                        INC     PATERR                     ;V177654
 2386 011536  077306                       3$:      SOB     R3,2$                      ;V177660
 2387 011540  000207                                RETURN                             ;V177662
```

```
  2389 011542                                  SUBAAI: SUBTST  <<FIND SHADOW INHIBIT MODE POINTERS>>
                                               ;*******************************************************************************
                                               ;*SUBTEST        FIND SHADOW INHIBIT MODE POINTERS
                                               ;*******************************************************************************
  2390                                         ;* THIS SECTION LOOKS FOR INTERLEAVED MS11-M MEMORIES AND FIGURES OUT
  2391                                         ;* WHERE THE SHADOW INHIBIT MODE POINTERS ARE LOCATED.  THESE AREAS
  2392                                         ;* ARE THEN MARKED AS PROGRAM SPACE.
  2393 011542  005037  002100                          CLR     BANK                        ;RESET BANK TO ZERO
  2394 011546  004737  044240                  SHADL1: CALL    EXBANK                      ;SET BANK PARAMETERS
  2395 011552  013700  002102                          MOV     BANKINDEX,R0
  2396 011556                                          IF ACFLAG IS TRUE AND INTFLAG IS TRUE
  2397 011572                                            IF INT64K IS TRUE
  2398 011600  062700  000020                              ADD #20,R0                      ;POINT TO BANKINDEX + 4
  2399 011604  062737  000010  002100                      ADD #10,BANK                    ;POINT TO BANK + 8
  2400 011612                                            ELSE
  2401 011614  062702  000040                              ADD #40,R2                      ;POINT TO BANKINDEX + 8
  2402 011620  062737  000020  002100                      ADD #20,BANK                    ;POINT TO BANK + 16
  2403 011626                                            END; OF IF INT64K
  2404 011626  052760  000200  002624                    BIS     #BIT7,CONFIG(R0)          ;MAKE NEW BANK PROGRAM SPACE
  2405 011634                                          ELSE
  2406 011636  005237  002100                            INC     BANK                      ;GO TO NEXT BANK
  2407 011642                                          END; OF IF ACFLAG
  2408 011642  023737  002526  002100                  CMP     LASTBANK,BANK               ;HAVE WE DONE ALL THE BANKS?
  2409 011650  002336                                  BGE     SHADL1                      ;BRANCH IF NOT
```

```
011652                                          NEWTST  <<ECC INHIBIT MODE POINTER TEST>>
                                        ;*****************************************************************************
                                        ;*TEST 4         ECC INHIBIT MODE POINTER TEST
                                        ;*****************************************************************************
011652  000004                          TST4:   SCOPE
                                                ;THE MS11-M OR MF11S-K INHIBIT ECC DISABLE AND DIAGNOSTIC CHECK MODE
                                                ;ON THE BOTTOM  FIRST OR SECOND 16K WORDS CONTROLLED BY A CSR.  THIS
                                                ;IS CONSIDERED  TO BE A  PROTECTED BANK  BY THE  PROGRAM.  IT MAY BE
                                                ;QUITE  COMPLEX TO DETERMINE ON A  GIVEN SYSTEM  CONFIGURATION WHICH
                                                ;BANKS CAN BE PROTECTED;
                                                ;SO
                                                ;THIS ROUTINE ATTEMPS TO  CREATE A DOUBLE BIT ERROR IN ADDRESS 0 & 2
                                                ;OF  EVERY ECC BANK.  ECC HARDWARE WILL PREVENT THIS  FROM HAPPENING
                                                ;IN  PROTECTED BANKS  WHICH SHOULD ALWAYS  INCLUDE BANK ZERO - WHERE
                                                ;THE PROGRAM IS.
                                                ;
                                                ;
                                                ;WARNING:!!!!!!!!!!
                                                ;  IN CASE OF HARDWARE FAILURE IT IS COMMON  THAT A DOUBLE BIT ERROR
                                                ;  WILL BE  CREATED ON  THE KERNEL  STACK & "CRASH"  THE  DIAGNOSTIC
                                                ;DURING  THIS ROUTINE.  YOUR ONLY CLUE IS THAT YOU CAN GET AS FAR AS
                                                ;THIS ROUTINE BUT NOT PAST IT!
011654  104424                                  CACHOFF                                         ;TURN CACHE OFF
011656  012737  177777  002150                  MOV     #-1,OLDCSR
011664                                          FOR BANK := #0 TO LASTBANK
011670  012701  060000                            MOV   #FIRST,R1                               ;SET UP VIRT ADDR POINTER
011674  004737  044240                            CALL  EXBANK
011700  013700  002102                            MOV   BANKINDEX,R0
011704                                            IF ACFLAG IS TRUE
011712                                              IF MKFLAG IS TRUE
011720                                              IF SKIPMK IS FALSE
011726                                                IF INTFLAG IS TRUE
011734  012703  040000                                  MOV     #40000,R3                       ;SET INDEX COUNTER
011740  012737  000001  002232                          MOV     #1,SPLTCSR                      ;MAP AS INTERLEAVED BANK
011746                                                ELSE
011750  012703  000002                                  MOV     #2,R3                           ;SET INDEX COUNTER
011754                                                END; OF IF INTFLAG
011754  116002  002625                                MOVB      CONFIG+1(R0),R2
011760  006302                                        ASL       R2
011762  042702  177741                                BIC       #^C36,R2
011766  010237  002146                                MOV       R2,CSRNO
011772                                                IF CSRNO NE OLDCSR
012002  013737  002146  002150                          MOV     CSRNO,OLDCSR
012010                                                  IF PFLAG IS FALSE
012016  052760  000100  002624                            BIS  #BIT6,CONFIG(R0)
012024                                                  END; OF IF PFLAG
012024  004737  012160                                  CALL    IMPTEST
```

```
012030                                      IF INTFLAG IS TRUE
012036  116002  002625                        MOVB  CONFIG+1(R0),R2
012042  072227  177775                        ASH   #-3,R2
012046  042702  177741                        BIC   #^C36,R2
012052  010237  002146                        MOV   R2,CSRNO
012056  062701  000002                        ADD   #2,R1                 ;FIX POINTER FOR A1 ASSERTED HALF
012062  004737  012160                        CALL  IMPTEST
012066  005037  002232                        CLR   SPLTCSR
012072                                      END; OF IF INTFLAG
012072                                    END; OF IF CSRNO
012072                                  END; OF IF SKIPMK
012072                                  END; OF IF MKFLAG
012072                                END; OF IF ACFLAG
012072                              END; OF FOR BANK
012106                              MAP                                   ;MAP TEST SPACE TO BANK 0
012122  005037  002100              CLR     BANK
012126                              IF #SW0 SET.IN @SWR OR ACTFLAG IS TRUE
012144  104506                        ENASBE                              ;TRAP ON SINGLE BIT ERRORS
012146                              ELSE
012150  104472                        ECCINIT                             ;TRAP ON DOUBLE BIT ERRORS (NORMAL)
012152                              END; OF IF #SW0
012152  104423                      CACHON                                ;TURN THE CACHE BACK ON
012154  000137  012422              JMP     SUBAAR                        ;JUMP OVER THE SUBROUTINE
```

```
012160  005004                         IMPTEST:CLR      R4
012162                                         MAP BANK                                   ;MAP SUPERVISOR SPACE (TEST AREA) TO BANK
012176  005005                                 CLR      R5
012200  012737  020000  002144                 MOV      #BIT13,CSR
012206                                         TESTAREA                                   ;ENTER TEST MODE
012214                                         PUSH     (R1)                              ;SAVE TEST LOCATION
012216  060301                                 ADD      R3,R1                  ;INDEX TO NEXT LOCATION
012220                                         PUSH     (R1)                   ;SAVE TEST LOCATION
012222  104505                                 CHK1DIS                         ;DISABLE ECC & WRITE CHECKBITS FOR 1 CSR
012224  010411                                 MOV      R4,(R1)                ;WRITE CHECKBITS (ALL ZEROS)
012226  160301                                 SUB      R3,R1
012230  010411                                 MOV      R4,(R1)
012232  104503                                 CLR1CSR                                    ;CLEAR CSR
012234  005711                                 TST      (R1)                              ;READ CHECKBITS INTO REAL CSR
012236  104501                                 WAS1DBE                                    ;WAS THERE A DOUBLE BIT ERROR

                                               ;THIS MAKES SURE THAT SBE'S DON'T LOOK LIKE PROTECTED AREAS

012240                                         ON.NOERROR ;1
012242  012737  020000  002144                   MOV    #BIT13,CSR
012250  104505                                   CHK1DIS                                  ;DISABLE ECC & WRITE CHECKBITS FOR 1 CSR
012252  013711  002554                           MOV    ONES,(R1)
012256  060301                                   ADD    R3,R1
012260  013711  002554                           MOV    ONES,(R1)
012264  160301                                   SUB    R3,R1
012266  104503                                   CLR1CSR                                  ;CLEAR CSR
012270  005711                                   TST    (R1)
012272  104501                                   WAS1DBE                                  ;WAS THERE A DOUBLE BIT ERROR
012274                                           ON.NOERROR ;2
012276  012737  027400  002144                     MOV #27400,CSR
012304  104505                                     CHK1DIS                                ;DISABLE ECC & WRITE CHECKBITS FOR 1 CSR
012306  010411                                     MOV R4,(R1)
012310  060301                                     ADD R3,R1                 ;ADD INDEX TO GET TO SECOND WORD
012312  010411                                     MOV R4,(R1)
012314  160301                                     SUB R3,R1                 ;SUBTRACT INDEX TO FIRST WORD
012316  104503                                     CLR1CSR                                ;CLEAR CSR
012320  005711                                     TST (R1)
012322  104501                                     WAS1DBE                                ;WAS THERE A DOUBLE BIT ERROR
012324                                             ON.NOERROR ;3
012326  012737  074000  002144                       MOV        #74000,CSR
012334  104505                                       CHK1DIS                              ;DISABLE ECC & WRITE CHECKBITS FOR 1 CSR
012336  010411                                       MOV        R4,(R1)
012340  060301                                       ADD        R3,R1        ;INDEX TO SECOND WORD
012342  010411                                       MOV        R4,(R1)
012344  104503                                       CLR1CSR                              ;CLEAR CSR
012346  160301                                       SUB        R3,R1        ;GO BACK TO FIRST WORD
012350  005711                                       TST        (R1)
012352  104501                                       WAS1DBE                              ;WAS THERE A DOUBLE BIT ERROR
012354                                             END ;OF ON.NOERROR ;3
012354                                           END ;OF ON.NOERROR ;2
012354                                         END ;OF ON.NOERROR ;1
012354                                         ON.ERROR
012356  005205                                   INC    R5                                ;IDENTIFY AS BAD BANK
012360                                         END ;OF ON.ERROR
012360  104471                                 ECC1DIS                                    ;DISABLE ERROR CORRECTION
012362  010411                                 MOV      R4,(R1)                ;CLEAR OUT DOUBLE BIT ERROR!
012364  060301                                 ADD      R3,R1                  ;INDEX TO SECOND WORD
```

```
012366  010411                          MOV     R4,(R1)                 ;CLEAR OUT DOUBLE BIT ERROR!
012370  104503                          CLR1CSR
012372  005705                          TST     R5
012374  001405                          BEQ     1$
012376  050560  002624                  BIS     R5,CONFIG(R0)
012402  105260  002626                  INCB    CONFIG+2(R0)
012406  104036                          ERROR   +36
012410                          1$:     POP     (R1)                    ;RESTORE TEST LOCATION (2ND WORD)
012412  160301                          SUB     R3,R1                   ;GO BACK TO FIRST WORD
012414                                  POP     (R1)                    ;RESTORE TEST LOCATION (1ST WORD)
012416  104417                          KERNEL
012420  000207                          RETURN
```

```
   2892
   2893 012422                        SUBAAR: SET      STOPOK          ;PROGRAM CAN NOW BE HALTED
```

```
2896 012430                                     SUBTST  <<LEGAL CONFIGURATION CHECK>>
                                      ;*************************************************************************
                                      ;*SUBTEST        LEGAL CONFIGURATION CHECK
                                      ;*************************************************************************
2897 012430  012700  000020                     MOV     #16.,R0
2898 012434  012701  002432                     MOV     #CSRINFO,R1
2899 012440  005021                     1$:     CLR     (R1)+
2900 012442  077002                             SOB     R0,1$
2904 012444                                       FOR BANK := #0 TO LASTBANK
2905 012450  004737  044240                         CALL        EXBANK
2906 012454  013700  002102                         MOV BANKINDEX,R0
2928
2929 012460                                         IF ACFLAG IS TRUE
2930 012466  116003  002625                           MOVB        CONFIG+1(R0),R3
2931 012472  042703  177760                           BIC         #^C17,R3
2932 012476  006303                                   ASL         R3
2933 012500  005263  002432                           INC         CSRINFO(R3)
2934 012504                                           IF MKFLAG IS TRUE
2935                                                  ;MAKE SURE THAT EACH BANK HAS NO MORE THAN 2 CSRS
2936 012512                                             BEGIN   LEGALCSR
2937 012512                                               IF INTFLAG IS TRUE
2938 012520  116003  002625                                 MOVB            CONFIG+1(R0),R3
2939 012524  010304                                         MOV R3,R4
2940 012526  042703  177760                                 BIC #^C17,R3
2941 012532  072427  177774                                 ASH #-4,R4
2942 012536  042704  177760                                 BIC #^C17,R4
2943 012542                                                 IF R3 EQ R4
2944 012546  042760  014000  002626                           BIC         #BIT11!BIT12,CONFIG+2(R0)
2945 012554  042760  170000  002624                           BIC         #170000,CONFIG(R0)
2946 012562                                                   LEAVE LEGALCSR
2947 012564                                                 END; OF IF R3
2948 012564                                                 IF KFLAG IS FALSE
2949 012572                                                   LEAVE     LEGALCSR
2950 012574                                                 END; OF IF KFLAG
2951 012574  006303                                         ASL R3
2952 012576  006304                                         ASL R4
2953 012600  005263  002432                                 INC CSRINFO(R3)
2954 012604  005264  002432                                 INC CSRINFO(R4)
2955 012610                                               ELSE
2956 012612                                                 LEAVE       LEGALCSR
2957 012614                                               END; OF IF INTFLAG
2958 012614                                             TYPE    MSG124                        ;# OF CSR'S IS WRONG
2959 012620                                             TYPOCS  BANK,<TYPE BANK #>,3
2960 012630                                             SET     CONFGERROR
2961 012636                                             END     LEGALCSR
2962 012636        -                                  END ;OF IF MKFLAG
2963 012636                                         END ;OF IF ACFLAG
2964 012636                                       END; OF FOR BANK
2965 012652                                     PUSH    R5,R0                                 ;SAVE CONTENTS OF R5, R0
2966 012656  005000                             CLR     R0                                    ;CLEAR REGISTERS
2967 012660  005001                             CLR     R1
2968 012662  005005                             CLR     R5
2969 012664  005037  013074                     CLR     MBERR                                 ;CLEAR ERROR INDICATOR
2970 012670  022761  000010  002432  2$:        CMP     #10,CSRINFO(R1)                       ;IS CURRENT CSR <= 10
2971 012676  002043                             BGE     5$                                    ;BRANCH IF SO
2972 012700  022761  000020  002432             CMP     #20,CSRINFO(R1)                       ;IS CURRENT CSR < 20
2973 012706  002003                             BGE     3$                                    ;BRANCH IF SO
```

```
2974 012710  004737  013216                     CALL   ILLCSR                          ;CALL ERROR ROUTINE
2975 012714  000434                             BR     5$                              ;TRY NEXT CSR
2976 012716  016005  002624             3$:     MOV    CONFIG(R0),R5                   ;MOVE LOW WORD TO R5
2977 012722  032705  000002                     BIT    #BIT1,R5                        ;DOES MEMORY EXIST HERE?
2978 012726  001415                             BEQ    4$                              ;BRANCH IF NOT
2979 012730  042705  170377                     BIC    #^C7400,R5                      ;ISOLATE CSR NUMBER IN
2980 012734  072527  177771                     ASH    #-7,R5                          ;REGISTER 5
2981 012740  020501                             CMP    R5,R1                           ;IS IT THE CURRENT CSR?
2982 012742  001007                             BNE    4$                              ;TRY NEXT WORD OF CONFIG IF NOT
2983 012744  032760  010000  002626             BIT    #BIT12,CONFIG+2(R0)             ;IS IT INTERLEAVED?
2984 012752  001003                             BNE    4$                              ;BRANCH IF SO
2985 012754  012737  000001  013074             MOV    #1,MBERR                        ;SET ERROR INDICATOR
2986 012762  062700  000004             4$:     ADD    #4,R0                           ;UPDATE CONFIG COUNTER
2987 012766  022700  000340                     CMP    #340,R0                         ;CONFIG TABLE ALL DONE?
2988 012772  001351                             BNE    3$                              ;BRANCH IF NOT
2989 012774  005737  013074                     TST    MBERR                           ;ERRORS FOUND?
2990 013000  001402                             BEQ    5$                              ;TRY NEXT CSR IF NOT
2991 013002  004737  013216                     CALL   ILLCSR                          ;CALL ERROR ROUTINE
2992 013006  005000                     5$:     CLR    R0                              ;REINITIALIZE CONFIG COUNTER
2993 013010  005037  013074                     CLR    MBERR                           ;CLEAR ERROR INDICATOR
2994 013014  062701  000002                     ADD    #2,R1                           ;UPDATE CSR COUNTER
2995 013020  022701  000040                     CMP    #40,R1                          ;ALL CSR'S DONE?
2996 013024  001321                             BNE    2$                              ;BRANCH IF NOT
2997 013026                                     POP    R0,R5                           ;RESTORE REGISTERS
2998 013032  005037  013074                     CLR    MBERR                           ;RESET ERROR INDICATOR
3002 013036  005001                             CLR    R1                      ;CLEAR
3003 013040  005000                             CLR    R0                      ;COUNTERS
3004 013042  032760  000002  002624     6$:     BIT    #BIT1,CONFIG(R0)        ;MEMORY PRESENT?
3005 013050  001404                             BEQ    7$                      ;BRANCH IF NOT
3006 013052  005201                             INC    R1                      ;ADD ONE BANK TO COUNT
3007 013054  062700  000004                     ADD    #4,R0                   ;UPDATE COUNTER
3008 013060  000770                             BR     6$                      ;TRY NEXT BANK
3009 013062  062701  177777             7$:     ADD    #-1,R1                  ;ADJUST COUNT
3010 013066  010137  002526                     MOV    R1,LASTBANK             ;STORE COUNT
3011 013072  000402                             BR     SKUJ
3012 013074  000000                     MBERR:  .WORD 0                        ;SAVE SPACE FOR ERROR INDICATOR
3013 013076  000000                     PHEBE:  .WORD 0                        ;SAVE SPACE FOR ODD BOUNDARY INTERLEAVED INDICATOR
3014 013100  005000                     SKUJ:   CLR    R0                      ;CLEAR CONFIG COUNTER
3015 013102  005037  013076                     CLR    PHEBE                   ;CLEAR COUNTER
3016 013106  032760  000002  002624     1$:     BIT    #BIT1,CONFIG(R0)        ;IS THERE MEMORY PRESENT?
3017 013114  001431                             BEQ    3$                      ;BRANCH IF NOT
3018 013116  032760  010000  002626             BIT    #BIT12,CONFIG+2(R0)     ;IS IT INTERLEAVED?
3019 013124  001005                             BNE    2$                      ;BRANCH IF SO
3020 013126  005237  013076                     INC    PHEBE                   ;INCREMENT COUNTER
3021 013132  062700  000004                     ADD    #4,R0                   ;INCREMENT CONFIG COUNTER
3022 013136  000763                             BR     1$                      ;TRY NEXT BANK
3023 013140  023727  013076  000010     2$:     CMP    PHEBE,#10               ;IS THE COUNTER EQUAL TO...
3024 013146  001417                             BEQ    4$                      ;ONE OF THE SPECIAL VALUES.
3025 013150  023727  013076  000030             CMP    PHEBE,#30               ;IF IT IS...
3026 013156  001413                             BEQ    4$                      ;BRANCH TO 4$
3027 013160  023727  013076  000050             CMP    PHEBE,#50
3028 013166  001407                             BEQ    4$
3029 013170  023727  013076  000070             CMP    PHEBE,#70
3030 013176  001403                             BEQ    4$
3031 013200  005037  013076             3$:     CLR    PHEBE                   ;CLEAR INDICATOR
3032 013204  000403                             BR     5$
3033 013206  012737  000001  013076     4$:     MOV    #1,PHEBE                ;SET INDICATOR
```

```
3034 013214  000421                    5$:     BR      SUBAAP          ;BRANCH TO NEXT SUBTEST
3035 013216  010102                    ILLCSR: MOV     R1,R2           ;R2 HAS CSR NUMBER
3036 013220  006202                            ASR     R2              ;MAKE ACCEPTABLE FOR PRINTING
3037 013222  022702  000012                    CMP     #10.,R2
3038 013226  100002                            BPL     1$
3039 013230  062702  000007                    ADD     #7,R2
3040 013234  062702  000060            1$:     ADD     #60,R2
3041 013240  110237  074767                    MOVB    R2,MSGA122      ;PUT NUMBER INTO ERROR MESSAGE
3042 013244                                    TYPE    MSG122
3043 013250                                    SET     CONFGERROR
3044 013256  000207                            RETURN
```

```
013260                                  SUBAAP: SUBTST  <<PRINT CONFIGURATION DETAILS>>
                                        ;******************************************************************************
                                        ;*SUBTEST        PRINT CONFIGURATION DETAILS
                                        ;******************************************************************************
013260                                          CLEAR   BSIZE,KSIZE,LSIZE,MSIZE,PSIZE
013304  013702  002526                          MOV     LASTBANK,R2
013310  006302                                  ASL     R2
013312  006302                                  ASL     R2
013314                                          FOR R1 := #0 TO R2 BY #4
013316                                           IF CPUBIT SET.IN CONFIG(R1)
013326                                             IF #BIT10 SET.IN CONFIG+2(R1)
013336                                               IF #BIT8 SET.IN CONFIG+2(R1)
013346                                                 IF #BIT9 SET.IN CONFIG+2(R1)
013356                                                   LET PSIZE := PSIZE + #1
013362                                                 ELSE
013364                                                   LET KSIZE := KSIZE + #1
013370                                                 END;IF BIT9
013370                                               ELSE
013372                                                 IF #BIT9 SET.IN CONFIG+2(R1)
013402                                                   LET LSIZE := LSIZE + #1
013406                                                 ELSE
013410                                                   LET MSIZE := MSIZE + #1
013414                                                 END; IF BIT9
013414                                               END;IF BIT8
013414                                             ELSE
013416                                               IF #BIT9 SET.IN CONFIG+2(R1)
013426                                                 IF #BIT8 SET.IN CONFIG+2(R1)
013436                                                   LET BSIZE := BSIZE + #1
013442                                                 END; OF IF #BIT8
013442                                               END; OF IF #BIT9
013442                                             END;IF BIT10
013442                                           END; OF IF CPUBIT
013442                                          END ;OF FOR ALL BANKS IN TABLE

013452  005037  002422                          CLR     I
013456                                          FOR R1 := #0 TO #10 BY #2
013460  006361  002344                            ASL   BSIZE(R1)
013464  006361  002344                            ASL   BSIZE(R1)
013470  006361  002344                            ASL   BSIZE(R1)
013474  006361  002344                            ASL   BSIZE(R1)                      ;BSIZE(R1) := BSIZE(R1) * 16.
013500  066137  002344  002422                    ADD   BSIZE(R1),I                    ;I <- I + BSIZE(R1)
013506                                          END; FOR R1
013520                                          FOR R1 := #0 TO #200 BY #4
013522                                              IF CPUBIT SET.IN CONFIG(R1)
013532                                                LET UNITOP := UNITOP + #1
013536                                              END; OF IF CPUBIT
013536                                          END; OF FOR R1
013550  006337  002366                          ASL     UNITOP
013554  006337  002366                          ASL     UNITOP
013560  006337  002366                          ASL     UNITOP
013564  006337  002366                          ASL     UNITOP          ;UNITOP := UNITOP * 16.
013570                                          IF I LT UNITOP THEN LET I := UNITOP
013606                                          TYPE    $CRLF
013612  005737  002344                          TST     BSIZE
013616  001405                                  BEQ     1$
013620                                          TYPDEC  BSIZE
013626                                          TYPE    MSG071
```

```
013632  005737  002346          1$:     TST     KSIZE
013636  001405                          BEQ     2$
013640                                  TYPDEC  KSIZE
013646                                  TYPE    MSG072
013652  005737  002350          2$:     TST     LSIZE
013656  001405                          BEQ     3$
013660                                  TYPDEC  LSIZE
013666                                  TYPE    MSG112
013672  005737  002352          3$:     TST     MSIZE
013676  001405                          BEQ     4$
013700                                  TYPDEC  MSIZE
013706                                  TYPE    MSG113
013712  005737  002354          4$:     TST     PSIZE
013716  001405                          BEQ     5$
013720                                  TYPDEC  PSIZE
013726                                  TYPE    MSG114
013732                          5$:     TYPDEC  I
013740                                  TYPE    MSG070
013744                                  IF #SW6 OFF.IN @SWR
013754  004737  036570                    CALL  PCONFIG
013760                                  END; OF IF #SW6
```

```
  3124 013760                                 SUBTST  <<CHECK APT SIZING>>
                                      ;****************************************************************************
                                      ;*SUBTEST        CHECK APT SIZING
                                      ;****************************************************************************
  3125 013760                                 IF APTFLAG IS TRUE AND APTSIZE IS TRUE
  3126 013774  005037  002404                   CLR    TEMP
  3127 014000  012700  062516                   MOV    #$MAMS1,R0
  3128 014004                                   FOR R2 := #0 TO #4
  3129 014006                                     IFB 1(R0) NE #0
  3130 014014  111001                                MOVB        (R0),R1
  3131 014016  042701  177400                        BIC         #177400,R1
  3132 014022                                        IF 2(R0) LT #0
  3133 014030  000261                                  SEC
  3134 014032                                        ELSE
  3135 014034  000241                                  CLC
  3136 014036                                        END ;OF IF 2(R0)
  3137 014036  006101                                ROL         R1
  3138 014040  005201                                INC         R1                   ;TO COMPENSATE FOR 4 BANKS BEING (0-3)
  3139 014042  006301                                ASL         R1
  3140 014044  006301                                ASL         R1
  3141 014046  006301                                ASL         R1
  3142 014050  006301                                ASL         R1
  3143 014052  163701  002404                        SUB         TEMP,R1
  3144 014056  010137  002404                        MOV         R1,TEMP
  3145 014062                                        IFB 1(R0) EQ #3
  3146 014072  060137  002372                          ADD       R1,APTPAR
  3147 014076                                        END ;OF IFB 1(R0)
  3148 014076                                        IFB 1(R0) EQ #4
  3149 014106  060137  002374                          ADD       R1,APTECC
  3150 014112                                        END ;OF IFB 1(R0)
  3151 014112  062700  000004                        ADD         #4,R0
  3152 014116                                     END ;OF IFB 1(R0)
  3153 014116                                   END ;OF FOR R2
  3154 014126                                   IF APTPAR NE LSIZE OR APTECC NE MSIZE
  3155 014146  104046                              ERROR        +46
  3156 014150                                   END ;OF IF APTPAR
  3168 014150                                 END ;OF IF APTFLAG
```

```
 3170 014150                              LOOP:   NEWTST  <<DIAGNOSTIC MODE DISPATCH ROUTINE>>
                                          ;*****************************************************************************
                                          ;*TEST 5        DIAGNOSTIC MODE DISPATCH ROUTINE
                                          ;*****************************************************************************
      014150  000004                      TST5:   SCOPE
 3171 014152  005037  002214                      CLR     CONTFLAG
 3172 014156  017700  166414                      MOV     @SWR,R0           ;GET SWITCHES
 3173 014162  042700  177761                      BIC     #^C16,R0          ;MASK TO ONLY MODE BITS
 3174 014166  004770  014176                      CALL    @DISPTBL(R0)      ;DISPATCH TO ROUTINE THROUGH NEXT TABLE
 3175 014172  000137  014216                      JMP     MEMDONE           ;GO TO NEXT TEST
 3176 014176  014650                      DISPTBL:BAFPAF  ;MODE 0;BANKS FORWARD, PATTERNS FORWARD
 3177 014200  014756                              BAFPAR  ;MODE 1;BANKS FORWARD, PATTERNS REVERSE
 3178 014202  015064                              BAWPAF  ;MODE 2;BANKS WORST FIRST, PATTERNS FORWARD
 3179 014204  015214                              BAWPAR  ;MODE 3;BANKS WORST FIRST, PATTERNS REVERSE
 3180 014206  015344                              PAFBAF  ;MODE 4;PATTERNS FORWARD, BANKS FORWARD
 3181 014210  015474                              PAFBAW  ;MODE 5;PATTERNS FORWARD, BANKS WORST FIRST
 3182 014212  015646                              PARBAF  ;MODE 6;PATTERNS REVERSE, BANKS FORWARD
 3183 014214  015776                              PARBAW  ;MODE 7;PATTERNS REVERSE, BANKS WORST FIRST
 3184
 3185 014216  004737  014550              MEMDONE:CALL    DOBACK                            ;CHECK BACKGROUND PATTERN
 3186
 3187 014222                                      NEWTST<<UNIQUE BANK TEST>>
                                          ;*****************************************************************************
                                          ;*TEST 6        UNIQUE BANK TEST
                                          ;*****************************************************************************
      014222  000004                      TST6:   SCOPE
 3188                                             ;MAKE SURE THAT EACH BANK CAN HAVE UNIQUE DATA
 3189                                             ;WRITE AND READ THE BANK NUMBER IN EACH BANK (EXCEPT WHERE THE PROGRAM IS)
 3190 014224                                      IF SELONLY IS FALSE
 3191 014232                                        SET    HEADER,MUT
 3192 014246  004737  024172                        CALL   MT0027
 3193 014252                                        SET    HEADER
 3194 014260  005037  002106                        CLR    MUT
 3195 014264                                      END ;OF IF SELONLY
 3196 014264  004737  014550                      CALL    DOBACK            ;RESTORE BACKROUND PATTERN
 3200
 3201 014270                              FLUSH:  SUBTST  <<FLUSH OUT DBE'S>>
                                          ;*****************************************************************************
                                          ;*SUBTEST       FLUSH OUT DBE'S
                                          ;*****************************************************************************
 3202 014270  004737  024656                      CALL    MT0030
```

```
                                          .SBTTL  END OF PASS ROUTINE
                                        ;*****************************************************************************
                                        ;*INCREMENT THE PASS NUMBER ($PASS)
                                        ;*INDICATE END-OF-PROGRAM AFTER EACH PASSES THRU THE PROGRAM
                                        ;*TYPE "END PASS #XXXXX" (WHERE XXXXX IS A DECIMAL NUMBER)
                                        ;*IF THERES A MONITOR GO TO IT
                                        ;*IF THERE ISN'T JUMP TO LOOP
014274  005037  002412                  $EOP:   CLR     FSINFLAG
014300  012700  002626                          MOV     #CONFIG+2,R0     ;MOVE 2ND WORD OF CONFIG TO R0
014304  042710  020000                  1$:     BIC     #BIT13,(R0)      ;CLEAR BACKGROUND VALID BIT
014310  062700  000004                          ADD     #4,R0            ;INCREMENT TO NEXT BANK
014314  020027  003620                          CMP     R0,#3620         ;DONE?
014320  003771                                  BLE     1$               ;NO - BRANCH
014322  013737  002570  002014                  MOV     $ERTTL,LASTERROR
014330  005237  062474                          INC     $PASS            ;;INCREMENT THE PASS NUMBER
014334  042737  100000  062474                  BIC     #100000,$PASS    ;;DON'T ALLOW A NEG. NUMBER
014342                                          TYPE    MSG077           ;;TYPE "END PASS #"
014346                                          IF #SW11 SET.IN @SWR OR QVFLAG IS TRUE
014364                                            TYPE  MSG035           ;QV
014370  005037  002316                            CLR   QVFLAG
014374                                          END ;OF IF SW11
014374                                          TYPDEC  $PASS
014402  013700  000042                          MOV     42,R0            ;;GET MONITOR ADDRESS
014406  001456                                  BEQ     $DOAGAIN         ;;BRANCH IF NO MONITOR
014410  022700  002000                  $ZAP42: CMP     #STACK,R0        ;ARE WE UNDER RT11
014414  001453                                  BEQ     $DOAGAIN         ;YES - BRANCH
                                                ;WE ARE UNDER (HEAVEN HELP US) XXDP!
014416                                          PUSH    R0
014420  004737  045126                          CALL    SHUTUP
014424                                          POP     R0
014426  000005                                  RESET                    ;;CLEAR THE WORLD
014430  004710                          $ENDAD: CALL    (R0)             ;;GO TO MONITOR
014432  000240                                  NOP                      ;;SAVE ROOM
014434  000240                                  NOP                      ;;FOR
014436  000240                                  NOP                      ;;ACT11
014440                                  $DOAGN: ;UNDO SHUTUP STUFF
                                                ;       RESTORE STACK
                                                ;       ENERGIZE UNIBUS MAP & 22 BIT ADDRESSING
                                                ;       ENERGIZE MEMORY MANAGEMENT
                                                ;       PUT LOADERS BACK HOME
014440  013706  002534                          MOV     KSTACK,SP
014444  005737  002424                          TST     NO22BIT                         ;IS THIS AN 11/44 OR 11/24?
014450  001003                                  BNE     1$
014452  052737  000060  172516                  BIS     #BIT5!BIT4,MMR3
014460  104420                          1$:     ENERGIZE                                ;TURN ON MEMORY MANAGEMENT
014462  013700  002536                          MOV     LOADHOME,R0                     ;DESTINATION BANK
014466  012701  000001                          MOV     #1,R1                           ;SOURCE BANK
014472  004737  043710                          CALL    BANKMOV
014476                                          IF APTFLAG IS TRUE
014504                                            IF $USWR EQ $PASS
014514  012701  000050                  APTHANG:     MOV #50,R1
014520  077001                          2$:          SOB R0,2$
014522  062737  000001  062476                       ADD #1,$DEVCT
014530  005537  062500                               ADC $UNIT
014534  077107                                       SOB R1,2$
014536  005237  062474                               INC $PASS
014542  000764                                       BR  APTHANG
```

```
   3262 014544                                  END ;OF IF $USWR
   3263 014544                                END ;OF IF APTFLAG
   3264 014544  000137  014150          $DOAGAIN: JMP   LOOP            ;RETURN
   3267 014550                          DOBACK: SUBTST  <<WRITE BACKGROUND PATTERNS>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       WRITE BACKGROUND PATTERNS
                                        ;*****************************************************************************
   3268 014550  005037  002110                  CLR     PATTERN
   3269 014554                                  FOR BANK := #0 TO LASTBANK
   3270 014560  004737  044240                    CALL  EXBANK
   3271 014564                                    IF ACFLAG IS TRUE AND RRFLAG IS FALSE
   3272 014600                                      SET         HEADER,MUT
   3273 014614  004737  017456                      CALL        MKTEST          ;CALL   MJTEST WOULD ALSO WORK
   3274 014620  005037  002106                      CLR         MUT
   3275 014624                                      SET         HEADER
   3276 014632                                    END ;OF IF ACFLAG
   3277 014632                                  END ;OF FOR BANK
   3278 014646  000207                          RETURN
```

```
                                          .SBTTL  MTEST MODES

     014650                       BAFPAF: SUBTST  <<BANKS FORWARD,PATTERNS FORWARD      **RECURSIVE**>>
                                  ;*********************************************************************************
                                  ;*SUBTEST        BANKS FORWARD,PATTERNS FORWARD      **RECURSIVE**
                                  ;*********************************************************************************
     014650  005037  002100               CLR     BANK              ;SET BANK TO 0
                                          ;START OF BANK LOOP
     014654  004737  044240       1$:     CALL    EXBANK            ;EXAMINE BANK
     014660  005737  002114               TST     ACFLAG            ;CAN WE ACCESS THIS BANK?
     014664  001412                       BEQ     4$                ;NO - GO TO BANK LOOP TERMINATION
     014666  005737  002122               TST     RRFLAG            ;RELOCATION REQUIRED?
     014672  001007                       BNE     4$                ;YES - GO TO BANK LOOP TERMINATION
     014674  005037  002110               CLR     PATTERN           ;SET PATTERN TO 0
                                          ;START OF PATTERN LOOP
     014700  004737  016150       2$:     CALL    MTEST             ;GO TEST CORRECT MEMORY
                                          ;TERMINATION OF PATTERN LOOP
     014704  004737  044726               CALL    INCPAT            ;GO SEE IF THIS IS THE LAST PATTERN
     014710  001373                       BNE     2$                ;NO - LOOP ON THIS PATTERN
                                          ;TERMINATION OF BANK LOOP
     014712  005037  002214       4$:     CLR     CONTFLAG
     014716  004737  044752               CALL    INCBNK            ;NEXT HIGHER BANK
     014722  002354                       BGE     1$                ;IF NOT DONE - LOOP ON THIS BANK
                                          ;END OF LOOPS
     014724  005737  002124               TST     RLFLAG            ;HAVE WE BEEN RELOCATED?
     014730  001401                       BEQ     5$                ;NO - SKIP
     014732  000207                       RETURN                    ;YES - RETURN
     014734  004737  042466       5$:     CALL    RELOCATE          ;MOVE & MAP PROGRAM
     014740                               ON.ERROR THEN $RETURN
                                          ;**NOTE** RECURSIVE CALL
     014744  004737  014650               CALL    BAFPAF            ;CALL SELF
     014750  004737  043356               CALL    UNRELOCATE        ;UNMOVE & UNMAP PROGRAM
     014754  000207                       RETURN
```

```
014756                        BAFPAR: SUBTST  <<BANKS FORWARD,PATTERNS REVERSE       **RECURSIVE**>>
                              ;*******************************************************************************
                              ;*SUBTEST       BANKS FORWARD,PATTERNS REVERSE       **RECURSIVE**
                              ;*******************************************************************************
014756  005037  002100                CLR     BANK                    ;SET BANK TO 0
                                      ;START OF BANK LOOP
014762  004737  044240        1$:     CALL    EXBANK                  ;EXAMINE BANK
014766  005737  002114                TST     ACFLAG                  ;CAN WE ACCESS THIS BANK?
014772  001412                        BEQ     4$                      ;NO - GO TO BANK LOOP TERMINATION
014774  005737  002122                TST     RRFLAG                  ;RELOCATION REQUIRED?
015000  001007                        BNE     4$                      ;YES - GO TO BANK LOOP TERMINATION
015002  004737  044742                CALL    SETPAT                  ;SET HIGH PATTERN FOR CORRECT MEMORY
                                      ;START OF PATTERN LOOP
015006  004737  016150        2$:     CALL    MTEST                   ;GO TEST CORRECT MEMORY
                                      ;TERMINATION OF PATTERN LOOP
015012  005337  002110                DEC     PATTERN                 ;IS THIS THE LAST PATTERN?
015016  100373                        BPL     2$                      ;NO - LOOP ON THIS PATTERN
                                      ;TERMINATION OF BANK LOOP
015020  005037  002214        4$:     CLR     CONTFLAG
015024  004737  044752                CALL    INCBNK                  ;NEXT HIGHER BANK
015030  002354                        BGE     1$                      ;IF NOT DONE - LOOP ON THIS BANK
                                      ;END OF LOOPS
015032  005737  002124                TST     RLFLAG                  ;HAVE WE BEEN RELOCATED?
015036  001401                        BEQ     5$                      ;NO - SKIP
015040  000207                        RETURN                          ;YES - RETURN
015042  004737  042466        5$:     CALL    RELOCATE                ;MOVE & MAP PROGRAM
015046                                ON.ERROR THEN $RETURN
                                      ;**NOTE** RECURSIVE CALL
015052  004737  014756                CALL    BAFPAR                  ;CALL SELF
015056  004737  043356                CALL    UNRELOCATE              ;UNMOVE & UNMAP PROGRAM
015062  000207                        RETURN
```

```
015064                          BAWPAF: SUBTST  <<BANKS WORST FIRST,PATTERNS FORWARD  **RECURSIVE**>>
                                ;*****************************************************************************
                                ;*SUBTEST        BANKS WORST FIRST,PATTERNS FORWARD  **RECURSIVE**
                                ;*****************************************************************************
015064  005037  002100                  CLR     BANK            ;SET BANK TO 0
                                        ;START OF BANK LOOP
015070  004737  044240          1$:     CALL    EXBANK          ;EXAMINE BANK
015074  005737  002114                  TST     ACFLAG          ;CAN WE ACCESS THIS BANK?
015100  001415                          BEQ     4$              ;NO - GO TO BANK LOOP TERMINATION
015102  005737  002126                  TST     BMFLAG          ;IS THIS BAD MEMORY (WORST FIRST)?
015106  001412                          BEQ     4$              ;NO - GO TO BANK LOOP TERMINATION
015110  005737  002122                  TST     RRFLAG          ;RELOCATION REQUIRED?
015114  001007                          BNE     4$              ;YES - GO TO BANK LOOP TERMINATION
015116  005037  002110                  CLR     PATTERN         ;SET PATTERN TO 0
                                        ;START OF PATTERN LOOP
015122  004737  016150          2$:     CALL    MTEST           ;GO TEST CORRECT MEMORY
                                        ;TERMINATION OF PATTERN LOOP
015126  004737  044726                  CALL    INCPAT          ;GO SEE IF THIS IS THE LAST PATTERN
015132  001373                          BNE     2$              ;NO - LOOP ON THIS PATTERN
                                        ;TERMINATION OF BANK LOOP
015134  005037  002214          4$:     CLR     CONTFLAG
015140  004737  044752                  CALL    INCBNK          ;NEXT HIGHER BANK
015144  002351                          BGE     1$              ;IF NOT DONE - LOOP ON THIS BANK
                                        ;END OF LOOPS
015146  005137  002540                  COM     WORST           ;IS THIS AN EVEN NUMBERED PASS?
015152  001003                          BNE     5$              ;YES - SKIP
                                        ;**NOTE** RECURSIVE CALL
015154  004737  015064                  CALL    BAWPAF          ;CALL SELF
015160  000207                          RETURN
015162  005737  002124          5$:     TST     RLFLAG          ;HAVE WE BEEN RELOCATED?
015166  001401                          BEQ     6$              ;NO - SKIP
015170  000207                          RETURN                  ;YES - RETURN
015172  004737  042466          6$:     CALL    RELOCATE        ;MOVE & MAP PROGRAM
015176                                  ON.ERROR THEN $RETURN
                                        ;**NOTE** RECURSIVE CALL
015202  004737  015064                  CALL    BAWPAF          ;CALL SELF
015206  004737  043356                  CALL    UNRELOCATE      ;UNMOVE & UNMAP PROGRAM
015212  000207                          RETURN
```

```
015214                            BAWPAR: SUBTST  <<BANKS WORST FIRST,PATTERNS REVERSE  **RECURSIVE**>>
                                  ;*******************************************************************************
                                  ;*SUBTEST         BANKS WORST FIRST,PATTERNS REVERSE  **RECURSIVE**
                                  ;*******************************************************************************
015214  005037  002100                    CLR     BANK            ;SET BANK TO 0
                                          ;START OF BANK LOOP
015220  004737  044240            1$:     CALL    EXBANK          ;EXAMINE BANK
015224  005737  002114                    TST     ACFLAG          ;CAN WE ACCESS THIS BANK?
015230  001415                            BEQ     4$              ;NO - GO TO BANK LOOP TERMINATION
015232  005737  002126                    TST     BMFLAG          ;IS THIS BAD MEMORY (WORST FIRST)
015236  001412                            BEQ     4$              ;NO - GO TO BANK LOOP TERMINATION
015240  005737  002122                    TST     RRFLAG          ;RELOCATION REQUIRED?
015244  001007                            BNE     4$              ;YES - GO TO BANK LOOP TERMINATION
015246  004737  044742                    CALL    SETPAT          ;SET HIGH PATTERN FOR CORRECT MEMORY
                                          ;START OF PATTERN LOOP
015252  004737  016150            2$:     CALL    MTEST           ;GO TEST CORRECT MEMORY
                                          ;TERMINATION OF PATTERN LOOP
015256  005337  002110                    DEC     PATTERN         ;IS THIS THE LAST PATTERN?
015262  100373                            BPL     2$              ;NO - LOOP ON THIS PATTERN
                                          ;TERMINATION OF BANK LOOP
015264  005037  002214            4$:     CLR     CONTFLAG
015270  004737  044752                    CALL    INCBNK          ;NEXT HIGHER BANK
015274  002351                            BGE     1$              ;IF NOT DONE - LOOP ON THIS BANK
                                          ;END OF LOOPS
015276  005137  002540                    COM     .WORST          ;IS THIS AN EVEN NUMBERED PASS?
015302  001003                            BNE     5$              ;YES - SKIP
                                          ;**NOTE** RECURSIVE CALL
015304  004737  015214                    CALL    BAWPAR          ;CALL SELF
015310  000207                            RETURN
015312  005737  002124            5$:     TST     RLFLAG          ;HAVE WE BEEN RELOCATED?
015316  001401                            BEQ     6$              ;NO - SKIP
015320  000207                            RETURN                  ;YES - RETURN
015322  004737  042466            6$:     CALL    RELOCATE        ;MOVE & MAP PROGRAM
015326                                    ON.ERROR THEN $RETURN
                                          ;**NOTE** RECURSIVE CALL
015332  004737  015214                    CALL    BAWPAR          ;CALL SELF
015336  004737  043356                    CALL    UNRELOCATE      ;UNMOVE & UNMAP PROGRAM
015342  000207                            RETURN
```

```
015344                              PAFBAF: SUBTST  <<PATTERNS FORWARD,BANKS FORWARD       **RECURSIVE**>>
                                    ;*****************************************************************************
                                    ;*SUBTEST        PATTERNS FORWARD,BANKS FORWARD        **RECURSIVE**
                                    ;*****************************************************************************
015344  005037  002110                      CLR     PATTERN             ;SET PATTERN TO 0
                                            ;START OF PATTERN LOOP
015350  005037  002100              1$:     CLR     BANK                ;SET BANK TO 0
                                            ;START OF BANK LOOP
015354  004737  044240              2$:     CALL    EXBANK              ;EXAMINE BANK
015360  004737  044710                      CALL    BANKOK              ;CORRECT MEMORY FOR THIS BANK?
015364  001010                              BNE     4$                  ;NO - GO TO BANK LOOP TERMINATOR
015366  005737  002114                      TST     ACFLAG              ;CAN WE ACCESS THIS BANK?
015372  001405                              BEQ     4$                  ;NO - GO TO BANK LOOP TERMINATION
015374  005737  002122                      TST     RRFLAG              ;RELOCATION REQUIRED?
015400  001002                              BNE     4$                  ;YES - GO TO BANK LOOP TERMINATION
015402  004737  016150                      CALL    MTEST               ;GO TEST CORRECT MEMORY
                                            ;TERMINATION OF BANK LOOP
015406  005037  002214              4$:     CLR     CONTFLAG
015412  004737  044752                      CALL    INCBNK              ;NEXT HIGHER BANK
015416  002356                              BGE     2$                  ;IF NOT DONE - LOOP ON THIS BANK
                                            ;TERMINATION OF PATTERN LOOP
015420  004737  044726                      CALL    INCRPT              ;NEXT HIGHER PATTERN
015424  001351                              BNE     1$                  ;OK - LOOP; ELSE CONTINUE
                                            ;END OF LOOPS
015426  005137  002132                      COM     TMFLAG              ;COMPLEMENT TYPE OF MEMORY
                                                                        ;IS THIS AN EVEN NUMBER PASS?
015432  001403                              BEQ     5$                  ;YES - SKIP
                                            ;**NOTE** RECURSIVE CALL
015434  004737  015344                      CALL    PAFBAF              ;CALL SELF
015440  000207                              RETURN
015442  005737  002124              5$:     TST     RLFLAG              ;HAVE WE BEEN RELOCATED?
015446  001401                              BEQ     6$                  ;NO - SKIP
015450  000207                              RETURN                      ;YES - RETURN
015452  004737  042466              6$:     CALL    RELOCATE            ;MOVE & MAP PROGRAM
015456                                      ON.ERROR THEN $RETURN
                                            ;**NOTE** RECURSIVE CALL
015462  004737  015344                      CALL    PAFBAF              ;CALL SELF
015466  004737  043356                      CALL    UNRELOCATE          ;UNMOVE & UNMAP PROGRAM
015472  000207                              RETURN
```

```
 3455 015474                        PAFBAW: SUBTST  <<PATTERNS FORWARD,BANKS WORST FIRST  **RECURSIVE**>>
                                    ;*******************************************************************************
                                    ;*SUBTEST       PATTERNS FORWARD,BANKS WORST FIRST  **RECURSIVE**
                                    ;*******************************************************************************
 3456 015474  005037  002110                CLR     PATTERN         ;SET PATTERN TO 0
 3457                                       ;START OF PATTERN LOOP
 3458 015500  005037  002100        1$:     CLR     BANK            ;SET BANK TO 0
 3459                                       ;START OF BANK LOOP
 3460 015504  004737  044240        2$:     CALL    EXBANK          ;EXAMINE BANK
 3461 015510  004737  044710                CALL    BANKOK          ;CORRECT MEMORY FOR THIS BANK?
 3462 015514  001013                        BNE     4$              ;NO - GO TO BANK LOOP TERMINATOR
 3463 015516  005737  002114                TST     ACFLAG          ;CAN WE ACCESS THIS BANK?
 3464 015522  001410                        BEQ     4$              ;NO - GO TO BANK LOOP TERMINATION
 3465 015524  005737  002126                TST     BMFLAG          ;IS THIS BAD MEMORY (WORST FIRST)
 3466 015530  001405                        BEQ     4$              ;NO - GO TO BANK LOOP TERMINATION
 3467 015532  005737  002122                TST     RRFLAG          ;RELOCATION REQUIRED?
 3468 015536  001002                        BNE     4$              ;YES - GO TO BANK LOOP TERMINATION
 3469 015540  004737  016150                CALL    MTEST           ;GO TEST CORRECT MEMORY
 3470                                       ;TERMINATION OF BANK LOOP
 3471 015544  005037  002214        4$:     CLR     CONTFLAG
 3472 015550  004737  044752                CALL    INCBNK          ;NEXT HIGHER BANK
 3473 015554  002353                        BGE     2$              ;IF NOT DONE - LOOP ON THIS BANK
 3474                                       ;TERMINATION OF PATTERN LOOP
 3475 015556  004737  044726                CALL    INCRPT          ;NEXT HIGHER PATTERN
 3476 015562  001346                        BNE     1$              ;OK - LOOP; ELSE CONTINUE
 3477                                       ;END OF LOOPS
 3478 015564  005137  002132                COM     TMFLAG          ;COMPLEMENT TYPE OF MEMORY
 3479                                                               ;IS THIS AN EVEN NUMBER PASS?
 3480 015570  001403                        BEQ     5$              ;YES - SKIP
 3481                                       ;**NOTE** RECURSIVE CALL
 3482 015572  004737  015474                CALL    PAFBAW          ;CALL SELF
 3483 015576  000207                        RETURN
 3484 015600  005137  002540        5$:     COM     WORST           ;4TH PASS?
 3485 015604  001003                        BNE     6$              ;YES - SKIP
 3486                                       ;**NOTE** RECURSIVE CALL
 3487 015606  004737  015474                CALL    PAFBAW          ;CALL SELF
 3488 015612  000207                        RETURN
 3489 015614  005737  002124        6$:     TST     RLFLAG          ;HAVE WE BEEN RELOCATED?
 3490 015620  001401                        BEQ     7$              ;NO - SKIP
 3491 015622  000207                        RETURN                  ;YES - RETURN
 3492 015624  004737  042466        7$:     CALL    RELOCATE        ;MOVE & MAP PROGRAM
 3493 015630                                ON.ERROR THEN $RETURN
 3494                                       ;**NOTE** RECURSIVE CALL
 3495 015634  004737  015474                CALL    PAFBAW          ;CALL SELF
 3496 015640  004737  043356                CALL    UNRELOCATE      ;UNMOVE & UNMAP PROGRAM
 3497 015644  000207                        RETURN
```

```
3500 015646                          PARBAF: SUBTST  <<PATTERNS REVERSE,BANKS FORWARD      **RECURSIVE**>>
                                     ;*****************************************************************************
                                     ;*SUBTEST         PATTERNS REVERSE,BANKS FORWARD      **RECURSIVE**
                                     ;*****************************************************************************
3501 015646  004737  044742                  CALL    HIPAT               ;SET HIGHTEST PATTERNS
3502                                         ;START OF PATTERN LOOP
3503 015652  005037  002100          1$:     CLR     BANK                ;SET BANK TO 0
3504                                         ;START OF BANK LOOP
3505 015656  004737  044240          2$:     CALL    EXBANK              ;EXAMINE BANK
3506 015662  004737  044710                  CALL    BANKOK              ;CORRECT MEMORY FOR THIS BANK?
3507 015666  001010                          BNE     4$                  ;NO - GO TO BANK LOOP TERMINATOR
3508 015670  005737  002114                  TST     ACFLAG              ;CAN WE ACCESS THIS BANK?
3509 015674  001405                          BEQ     4$                  ;NO - GO TO BANK LOOP TERMINATION
3510 015676  005737  002122                  TST     RRFLAG              ;RELOCATION REQUIRED?
3511 015702  001002                          BNE     4$                  ;YES - GO TO BANK LOOP TERMINATION
3512 015704  004737  016150                  CALL    MTEST               ;GO TEST CORRECT MEMORY
3513                                         ;TERMINATION OF BANK LOOP
3514 015710  005037  002214          4$:     CLR     CONTFLAG
3515 015714  004737  044752                  CALL    INCBNK              ;NEXT HIGHER BANK
3516 015720  002356                          BGE     2$                  ;IF NOT DONE - LOOP ON THIS BANK
3517                                         ;TERMINATION OF PATTERN LOOP
3518 015722  005337  002110                  DEC     PATTERN             ;NEXT LOWER PATTERN
3519 015726  100351                          BPL     1$                  ;OK - LOOP; ELSE CONTINUE
3520                                         ;END OF LOOPS
3521 015730  005137  002132                  COM     TMFLAG              ;COMPLEMENT TYPE OF MEMORY
3522                                                                     ;IS THIS AN EVEN NUMBER PASS?
3523 015734  001403                          BEQ     5$                  ;YES - SKIP
3524                                         ;**NOTE** RECURSIVE CALL
3525 015736  004737  015646                  CALL    PARBAF              ;CALL SELF
3526 015742  000207                          RETURN
3527 015744  005737  002124          5$:     TST     RLFLAG              ;HAVE WE BEEN RELOCATED?
3528 015750  001401                          BEQ     6$                  ;NO - SKIP
3529 015752  000207                          RETURN                      ;YES - RETURN
3530 015754  004737  042466          6$:     CALL    RELOCATE            ;MOVE & MAP PROGRAM
3531 015760                                  ON.ERROR THEN $RETURN
3532                                         ;**NOTE** RECURSIVE CALL
3533 015764  004737  015646                  CALL    PARBAF              ;CALL SELF
3534 015770  004737  043356                  CALL    UNRELOCATE          ;UNMOVE & UNMAP PROGRAM
3535 015774  000207                          RETURN
```

```
015776                           PARBAW: SUBTST  <<PATTERNS REVERSE,BANKS WORST FIRST  **RECURSIVE**>>
                                 ;*******************************************************************************
                                 ;*SUBTEST        PATTERNS REVERSE,BANKS WORST FIRST  **RECURSIVE**
                                 ;*******************************************************************************
015776  004737  044742                   CALL    HIPAT            ;SET HIGHTEST PATTERN
                                         ;START OF PATTERN LOOP
016002  005037  002100           1$:     CLR     BANK             ;SET BANK TO 0
                                         ;START OF BANK LOOP
016006  004737  044240           2$:     CALL    EXBANK           ;EXAMINE BANK
016012  004737  044710                   CALL    BANKOK           ;CORRECT MEMORY FOR THIS BANK?
016016  001013                           BNE     4$               ;NO - GO TO BANK LOOP TERMINATOR
016020  005737  002114                   TST     ACFLAG           ;CAN WE ACCESS THIS BANK?
016024  001410                           BEQ     4$               ;NO - GO TO BANK LOOP TERMINATION
016026  005737  002126                   TST     BMFLAG           ;IS THIS BAD MEMORY (WORST FIRST)
016032  001405                           BEQ     4$               ;NO - GO TO BANK LOOP TERMINATION
016034  005737  002122                   TST     RRFLAG           ;RELOCATION REQUIRED?
016040  001002                           BNE     4$               ;YES - GO TO BANK LOOP TERMINATION
016042  004737  016150                   CALL    MTEST            ;GO TEST CORRECT MEMORY
                                         ;TERMINATION OF BANK LOOP
016046  005037  002214           4$:     CLR CONTFLAG
016052  004737  044752                   CALL    INCBNK           ;NEXT HIGHER BANK
016056  002353                           BGE     2$               ;IF NOT DONE - LOOP ON THIS BANK
                                         ;TERMINATION OF PATTERN LOOP
016060  005337  002110                   DEC     PATTERN          ;NEXT LOWER PATTERN
016064  100346                           BPL     1$               ;OK - LOOP; ELSE CONTINUE
                                         ;END OF LOOPS
016066  005137  002132                   COM     TMFLAG           ;COMPLEMENT TYPE OF MEMORY
                                                                  ;IS THIS AN EVEN NUMBER PASS?
016072  001403                           BEQ     5$               ;YES - SKIP
                                         ;**NOTE** RECURSIVE CALL
016074  004737  015776                   CALL    PARBAW           ;CALL SELF
016100  000207                           RETURN
016102  005137  002540           5$:     COM     WORST            ;4TH PASS?
016106  001003                           BNE     6$               ;YES - SKIP
                                         ;**NOTE** RECURSIVE CALL
016110  004737  015776                   CALL    PARBAW           ;CALL SELF
016114  000207                           RETURN
016116  005737  002124           6$:     TST     RLFLAG           ;HAVE WE BEEN RELOCATED?
016122  001401                           BEQ     7$               ;NO - SKIP
016124  000207                           RETURN                   ;YES - RETURN
016126  004737  042466           7$:     CALL    RELOCATE         ;MOVE & MAP PROGRAM
016132                                   ON.ERROR THEN $RETURN
                                         ;**NOTE** RECURSIVE CALL
016136  004737  015776                   CALL    PARBAW           ;CALL SELF
016142  004737  043356                   CALL    UNRELOCATE       ;UNMOVE & UNMAP PROGRAM
016146  000207                           RETURN
```

```
016150                        MTEST:  SUBTST  <<SUBR  SETUP MEMORY TEST>>
                              ;*************************************************************************
                              ;*SUBTEST         SUBR    SETUP MEMORY TEST
                              ;*************************************************************************
016150                                SET     HEADER                          ;INITIALIZE HEADER MESSAGE TYPEOUT
016156                                SET     MUT                             ;INDICATE THERE IS A MEMORY UNDER TEST
016164  005037  002256                CLR     PASFLG
016170  005737  002116                TST     MKFLAG                          ;ECC?
016174  001413                        BEQ     MT1                             ;NO - SKIP
016176                                BEGIN   HOLDLOOP
016176                                  IF CONTFLAG IS TRUE THEN LEAVE HOLDLOOP
016204                                  IF SKIPMK IS FALSE
016212  004737  016244                    CALL        MKCONTROL
016216                                  END; OF IF SKIPMK
016216                                END     HOLDLOOP
016216  004737  017456                CALL    MKTEST                          ;YES - DO ECC TESTS
016222  000402                        BR      MT2
016224  004737  017676        MT1:    CALL    MJTEST                          ;DO PARITY TESTS
016230  005037  002106        MT2:    CLR     MUT                             ;NOW - NO MEMORY UNDER TEST
016234                                SET     HEADER                          ;ALLOW HEADERS NORMAL
016242  000207                        RETURN
```

```
  3603 016244                                   MKCONTROL:SUBTST        <<SUBR  TEST ECC CSR LOGIC DISPATCH>>
                                                ;*********************************************************************
                                                ;*SUBTEST          SUBR    TEST ECC CSR LOGIC DISPATCH
                                                ;*********************************************************************
  3604                                                  ;THE NEXT TWO MODULES SOLVE THE PROBLEM OF
  3605                                                  ;HOW TO RUN THE CSR TESTS ON EACH ECC MEMORY
  3606                                                  ;
  3607 016244                                           IF SELONLY IS TRUE THEN $RETURN
  3608 016254                                           IF INHECC IS TRUE THEN $RETURN
  3609 016264                                           PUSH    BANK,R0,R1,R2,R3
  3610 016300  012737  060000  002224                   MOV     #FIRST,CSRFBANK          ;SET FIRST TEST ADDRESS TO FIRST ADDR.
  3611 016306  012737  157776  002226                   MOV     #LAST,CSRLBANK
  3612 016314  005037  002230                           CLR     CSRINT
  3613 016320  005037  002232                           CLR     SPLTCSR
  3614 016324  005037  002302                           CLR     CSRLOOP                  ; AND ZERO THE LOOP COUNTER
  3615 016330  013700  002102                           MOV     BANKINDEX,R0             ;GET THE BANK INDEX
  3616 016334  016001  002624                           MOV     CONFIG(R0),R1            ;GET CSR NUMBER
  3617 016340  000301                                   SWAB    R1
  3618 016342  042701  177760                           BIC     #^C17,R1
  3619 016346  006301                                   ASL     R1
  3620 016350  010137  002476                           MOV     R1,CSRHOLD               ;STORE IN THE LOW BYTE
  3621 016354  005737  002134                           TST     INTFLAG                  ;IS THIS BANK INTERLEAVED?
  3622 016360  001421                                   BEQ     1$                       ;BRANCH IF NOT INTERLEAVED
  3623 016362  005237  002232                           INC     SPLTCSR
  3624 016366  012737  120000  002226                   MOV     #120000,CSRLBANK
  3625 016374  005237  002302                           INC     CSRLOOP                  ;WE MUST LOOP TWICE FOR AN INTERLEAVED BANK
  3626 016400  005237  002230                           INC     CSRINT
  3627 016404  016001  002624                           MOV     CONFIG(R0),R1            ;GET THE INTERLEAVE CSR NUMBER
  3628 016410  072127  177775                           ASH     #-3,R1
  3629 016414  042701  160777                           BIC     #^C17000,R1
  3630 016420  050137  002476                           BIS     R1,CSRHOLD               ;STORE IT IN CSRHOLD'S UPPER BYTE
  3631 016424  005003                           1$:     CLR     R3
  3632 016426  116337  002476  002146  MKLOOP: MOVB    CSRHOLD(R3),CSRNO
  3633 016434  042737  177741  002146          BIC     #^C36,CSRNO              ;CLEAR ANY UNNECESSARY BITS
  3634 016442                                              FOR R2 := #0 TO CSRINT
  3635 016444                                                FOR CSRFIRST := CSRFBANK TO CSRLBANK BY #4000
  3636 016452                                                MAP BANK                 ;MAP TEST SPACE TO BANK
  3637 016466  104511                                        INVALIDATE               ;INVALIDATE BACKROUND PATTERN
  3638 016470                                                  BEGIN CSRSTUFF
  3639 016470  005037  002304                                  CLR       SUCCESS
  3640 016474                                                    IF ACFLAG IS TRUE AND RRFLAG IS FALSE
  3641 016510  013737  002220  002222                              MOV   CSRFIRST,CSRLAST
  3642 016516  062737  004000  002222                              ADD   #4000,CSRLAST
  3643 016524                                                        FOR TESTADD := CSRFIRST TO CSRLAST BY #4
  3644 016532  013737  002362  002364                                  MOV       TESTADD,TESTADD+2
  3645 016540  005737  002232                                          TST       SPLTCSR
  3646 016544  001404                                                  BEQ       1$
  3647 016546  062737  040000  002364                                  ADD       #40000,TESTADD+2
  3648 016554  000403                                                  BR        2$
  3649 016556  062737  000002  002364                          1$:     ADD       #2,TESTADD+2
  3650 016564  004737  017052                                  2$:     CALL      SBETEST
  3651 016570                                                          ON.NOERROR
  3652 016572  104424                                                    CACHOFF                 ;TURN CACHE OFF
  3653 016574  005037  002074                                            CLR     NOPAR           ;INDICATE PARITY ACTION
  3654 016600                                                            FOR I := #0 TO #27
  3655 016604                                                              SET    HEADER
  3656 016612  005037  002256                                              CLR    PASFLG
```

```
016616                                                   LET RO := I
016622                                                   PUSH  R3              ;SAVE LOOP COUNTER
016624   010637  002142                                  MOV   SP,CTLKVEC      ;SAVE VECTOR IN CSR OF ^K
016630   162737  000002 002142                           SUB   #2,CTLKVEC
016636   004737  017356                                  CALL CSRCASE
016642                                                   POP   R3              ;RESTORE LOOP COUNTER
016644                                                 END ;OF FOR I
016660   104423                                        CACHON                  ;TURN CACHE ON
016662                                                   SET   SUCCESS
016670                                                 LEAVE CSRSTUFF
016672                                               END ;OF ON.NOERROR
016672                                             END ;OF FOR TESTADD
016710                                         END ;OF IF
016710                                       END CSRSTUFF
016710                                       IF SUCCESS IS FALSE
016716                                         TYPE    MSGA34
016722                                         TYPOCS  BANK,<TYPES BANK NUMBER>,3
016732                                         TYPE    MSGB34
016736                                       END ;OF IF SUCCESS
016736                                     END; OF FOR CSRFIRST
016754   005237  002232                    INC  SPLTCSR
016760                                   END; OF FOR R2
016770   062737  000002 002224          ADD    #2,CSRFBANK
016776   012737  000001 002232          MOV    #1,SPLTCSR
017004   005203                         INC    R3
017006   020337  002302                 CMP    R3,CSRLOOP
017012   003002                         BGT    1$
017014   000137  016426                 JMP    MKLOOP
017020   104472                 1$:     ECCINIT        ;TRAP ON DOUBLE BIT ERRORS (NORMAL)
017022                                  SET    CONTFLAG
017030   005037  002232                 CLR    SPLTCSR
017034                                  POP    R3,R2,R1,RU,BANK
017050   000207                         RETURN
```

```
017052                        SBETEST:SUBTST  <<CHECK FOR SBE FREE LOCATIONS>>
                              ;*******************************************************************************
                              ;*SUBTEST       CHECK FOR SBE FREE LOCATIONS
                              ;*******************************************************************************
                                        ;IN ORDER TO DETERMINE IF A LOCATION IS SBE FREE I DO THIS
                                        ;
                                        ;WRITE ZEROS WITH ECC DISABLE
                                        ;READ ZEROS BACK
                                        ;IF NOT ZEROS THEN RETURN ERROR
                                        ;
                                        ;WRITE ZEROS WITH ECC ENABLED BUT TRAPS DISABLED
                                        ;READ ZEROS BACK
                                        ;IF NOT ZEROS THEN RETURN ERROR
                                        ;
                                        ;TEST THE LOCATION FROM THE PAR'S (WITH NO PROGRAM FETCHES)
                                        ;IF THERE WERE ANY SBE'S OR DBE'S THEN RETURN ERROR
                                        ;
                                        ;COMPLIMENT ZEROS TO ONES WITH ECC DISABLE
                                        ;READ ONES BACK
                                        ;IF NOT ONES THEN RETURN ERROR
                                        ;
                                        ;WRITE 100,,100000,00000 (CHECKBITS COMPLIMENT OF BEFORE)
                                        ;  WITH ECC ENABLED BUT TRAPS DISABLED
                                        ;TEST THE LOCATION FROM THE PAR'S (WITH NO PROGRAM FETCHES)
                                        ;IF THERE WERE ANY SBE'S OR DBE'S THEN RETURN ERROR
                                        ;
                                        ;IF NONE OF THE ABOVE FORCES A RETURN ERROR THEN RETURN NO.ERROR
                                        .ENABL  LSB
017052                                  PUSH    R0,R1,R4                        ;PUSH R0,R1,R4 ONTO STACK
017060  013701  002362                  MOV     TESTADD,R1
017064  013704  002364                  MOV     TESTADD+2,R4
017070                                  TESTAREA                                ;ENTER TEST MODE
017076  104424                          CACHOFF                                 ;TURN CACHE OFF
017100  104471                          ECC1DIS                                 ;DISABLE ECC ON 1 SELECTED CSR
017102                                  CLEAR   (R1),(R4)
017106  005711                          TST     (R1)
017110  001107                          BNE     SBENT
017112  005714                          TST     (R4)
017114  001105                          BNE     SBENT

017116  104503                          CLR1CSR                                 ;CLEAR 1 SELECTED CSR
017120                                  CLEAR   (R1),(R4)
017124  005711                          TST     (R1)
017126  001100                          BNE     SBENT
017130  005714                          TST     (R4)
017132  001076                          BNE     SBENT

017134  104510                          TSTREAD                 ;TEST LOC (R1) & TST FOR SBE (WITHOUT FETCHES)
017136                                  IF #BIT15!BIT4 SET.IN CSR
017146                                    SET SKPERR                            ;DISABLE ERRGEN'S ERROR PRINTOUT
017154  104512                            ERRGEN
017156  013700  002430                    MOV   ERRADD,R0
017162  072027  177774                    ASH   #-4,R0
017166  042700  177600                    BIC   #^C177,R0
017172                                    IF BANK EQ R0 THEN GOTO SBENT
017200                                  END: OF IF #BIT15
017200  104471                          ECC1DIS                                 ;DISABLE ECC ON 1 SELECTED CSR
```

```
017202  005111                          COM     (R1)
017204  005114                          COM     (R4)
017206  023711  002554                  CMP     ONES,(R1)
017212  001046                          BNE     SBENT
017214  023714  002554                  CMP     ONES,(R4)
017220  001043                          BNE     SBENT

017222  104503                          CLR1CSR                               ;CLEAR 1 SELECTED CSR
017224  005011                          CLR     (R1)
017226  012714  100000                  MOV     #BIT15,(R4)
017232  005711                          TST     (R1)
017234  001035                          BNE     SBENT
017236  022714  100000                  CMP     #BIT15,(R4)
017242  001032                          BNE     SBENT

017244  104510                          TSTREAD                  ;TEST LOC (R1) & TST FOR SBE (WITHOUT FETCHES)
017246                                  IF #BIT15!BIT4 SET.IN CSR
017256                                    SET SKPERR                          ;DISABLE ERRGEN'S ERROR PRINTOUT
017264  104512                            ERRGEN
017266  013700  002430                    MOV   ERRADD,R0
017272  072027  177774                    ASH   #-4,R0
017276  042700  177600                    BIC   #^C177,R0
017302                                    IF BANK EQ R0 THEN GOTO SBENT
017310                                  END; OF IF #BIT15

017310  104417                          KERNEL                                ;ENTER KERNEL MODE
017312  104473                          ECC1INIT                              ;INITIALIZE 1 SELECTED CSR
017314  104423                          CACHON                                ;TURN CACHE ON
017316                                  POP       R4,R1,R0                    ;POP R0,R1 & R4 FROM STACK
017324                                  $RETURN NOERROR

017330  104503                  SBENT:  CLR1CSR                               ;CLEAR 1 SELECTED CSR
017332                                  CLEAR   (R1),(R4)
017336  104417                          KERNEL                                ;ENTER KERNEL MODE
017340  104473                          ECC1INIT                              ;INITIALIZE 1 SELECTED CSR
017342  104423                          CACHON                                ;TURN CACHE ON
017344                                  POP       R4,R1,R0                    ;POP R0,R1 & R4 FROM STACK
017352                                  $RETURN ERROR
                                        .DSABL  LSB
```

```
 3787 017356                     CSRCASE:SUBTST  <<CSR PATTERN CASE STATEMENT>>
                                 ;***************************************************************************
                                 ;*SUBTEST        CSR PATTERN CASE STATEMENT
                                 ;***************************************************************************
 3788 017356                             CASE R0
 3789                                      ;WARNING IF YOU CHANGE THIS TABLE ALSO
 3790                                      ;CHANGE "$DDW0" - "$DDW5" (THE PATTERN BIT MAP)
 3791
 3792                                      ;PAT  TIME              DISCRIPTION
 3793 017366  021132             MKCSRT:   MT0006         ;<1 SEC            INITIAL DATA TEST
 3794 017370  021230                       MT0010         ;<1 SEC            BYTE ADDRESSING TEST
 3795 017372  023622                       MT0025         ;<1 SEC            INTERRUPT ENABLE TEST
 3796 017374  021264                       MT0011         ;<2 SEC            CREATE SINGLE BIT ERROR TEST
 3797 017376  021332                       MT0012         ;<1 SEC            WRITE BYTE CLEARS SBE TEST
 3798 017400  021426                       MT0013         ; 1 SEC            CREATE DOUBLE BIT ERROR TEST
 3799 017402  021502                       MT0014         ; 1 SEC            WRITE INHIBIT DURING DATIP WITH DBE
 3800 017404  021560                       MT0015         ; 1 SEC            WRITE INHIBIT OF BYTE WITH DBE
 3801 017406  021626                       MT0016         ;<1 SEC            WRITE INHIBIT OF WORD WITH DBE
 3802 017410  026354                       MT0999         ; 0 SEC            NULL TEST
 3803 017412  026354                       MT0999         ; 0 SEC            NULL TEST
 3804 017414  026354                       MT0999         ; 0 SEC            NULL TEST
 3805 017416  026354                       MT0999         ; 0 SEC            NULL TEST
 3806 017420  026354                       MT0999         ; 0 SEC            NULL TEST
 3807 017422  026354                       MT0999         ; 0 SEC            NULL TEST
 3808 017424  026354                       MT0999         ; 0 SEC            NULL TEST
 3809 017426  026354                       MT0999         ; 0 SEC            NULL TEST
 3810 017430  026354                       MT0999         ; 0 SEC            NULL TEST
 3811 017432  026354                       MT0999         ; 0 SEC            NULL TEST
 3812 017434  026354                       MT0999         ; 0 SEC            NULL TEST
 3813 017436  026354                       MT0999         ; 0 SEC            NULL TEST
 3814 017440  026354                       MT0999         ; 0 SEC            NULL TEST
 3815 017442  026354                       MT0999         ; U SEC            NULL TEST
 3816 017444  026354                       MT0999         ; 0 SEC            NULL TEST
 3817 017446                             END ;OF CASE R0
 3818 017454  000207                     RETURN
 3821 017456                                   MKTEST: SUBTST  <<SUBR  ECC TEST DISPATCH>>
                                               ;*******************************************************************************
                                               ;*SUBTEST         SUBR    ECC TEST DISPATCH
                                               ;*******************************************************************************
 3825 017456                                           IF #SWO SET.IN @SWR OR ACTFLAG IS TRUE
 3826 017474   104470                                  ECCDIS                                 ;DISABLE ERROR CORRECTION
 3827 017476                                           ELSE
 3828 017500   104502                                  CLRCSR                                 ;CLEAR ALL CSR'S
 3829 017502                                           END ;OF IF
 3830 017502   012737  000002  002074                  MOV     #2,NOPAR                       ;INDICATE PARITY ACTION
 3831 017510   012737  000002  002276                  MOV     #2,PCBUMP                      ;TRAPS ADD 2 TO PC
 3832 017516   013700  002110                          MOV     PATTERN,RO                     ;GET PATTERN NUMBER
 3833 017522   006300                                  ASL     RO                             ;MAKE IT A WORD ADDRESS
 3834 017524                                           IF MKPAT(RO) NE #MT0034 AND MKPAT(RO) NE #MT0999
 3835 017544   104511                                    INVALIDATE           ;INVALIDATE BACKGROUND PATTERN ON "BANK"
 3836 017546                                           END ;OF IF MKPAT(RO)
 3837 017546   010637  002142                          MOV     SP,CTLKVEC                     ;SAVE VECTOR IN CASE OF ^K
 3838 017552   162737  000002  002142                  SUB     #2,CTLKVEC
 3839 017560   004770  017616                          CALL    @MKPAT(RO)                     ;INDEX OFF TABLE
 3840 017564                                           IF #SWO SET.IN @SWR OR ACTFLAG IS TRUE
 3841 017602   104506                                    ENASBE                               ;TRAP ON SINGLE BIT ERRORS
 3842 017604                                           ELSE
 3843 017606   104472                                    ECCINIT                              ;TRAP ON DOUBLE BIT ERRORS (NORMAL)
 3844 017610                                           END ;OF IF #SWO
 3845 017610   005037  002074                          CLR     NOPAR                          ;INDICATE PARITY ACTION
 3846 017614   000207                                  RETURN
 3847
 3848                                                  ;WARNING IF YOU CHANGE THIS TABLE ALSO
 3849                                                  ;CHANGE "$DDWO" - "$DDW5" (THE PATTERN BIT MAP)
 3850                                                  ;PAT     TIME             DISCRIPTION
 3851 017616                                   MKPAT:  ;NOTE MT0034 MUST BE FIRST & LAST
 3852 017616   026070                                  MT0034  ;<1 SEC          ;SOFT ERROR - BACKGROUND PATTERN TEST
 3853 017620   021674                                  MT0017  ;<1 SEC          ;HOLDING 1'S & 0'S TEST
 3854 017622   021166                                  MT0007  ;<1 SEC          ;ADDRESS BIT TEST
 3855 017624   020142                                  MT0001  ;<1 SEC          ;ADDRESS TEST
 3856 017626   020262                                  MT0002  ;<1 SEC          ;COMPLEMENT ADDRESS TEST
 3857 017630   020654                                  MT0004  ; 1 SEC          ;ROTATING ZEROS TEST
 3858 017632   020776                                  MT0005  ; 1 SEC          ;ROTATING ONES TEST
 3859 017634   023006                                  MT0021  ; 1 SEC          ;MARCHING 0'S & 1'S TEST
 3860 017636   021716                                  MT0020  ;<1 SEC          ;MARCHING 1'S & 0'S IN CHECK BITS
 3861 017640   023260                                  MT0022  ;10 SEC          ;REFRESH & SHIFTING DIAGONAL  TEST
 3862 017642   023670                                  MT0026  ;<1 SEC          ;RANDOM DATA TEST
 3863 017644   023356                                  MT0024  ;20 SEC          ;FAST GALLOPING PATTERN TEST
 3864 017646   025160                                  MT0031  ; 3 SEC          ;SOB-A-LONG TEST
 3865 017650   025350                                  MT0032  ;<1 SEC          ;WRITE RECOVERY TEST
 3866 017652   025702                                  MT0033  ;35 SEC          ;BRANCH GOBBLE TEST
 3867 017654   026070                                  MT0034  ;<1 SEC          ;SOFT ERROR - BACKGROUND PATTERN TEST
 3868                                                  ;NOTE MT0034 MUST BE FIRST & LAST
 3869 017656   026354                                  MT0999  ; 0 SEC          ;NULL TEST
 3870 017660   026354                                  MT0999  ; 0 SEC          ;NULL TEST
 3871 017662   026354                                  MT0999  ; 0 SEC          ;NULL TEST
 3872 017664   026354                                  MT0999  ; 0 SEC          ;NULL TEST
 3873 017666   026354                                  MT0999  ; 0 SEC          ;NULL TEST
 3874 017670   026354                                  MT0999  ; 0 SEC          ;NULL TEST
 3875 017672   026354                                  MT0999  ; 0 SEC          ;NULL TEST
 3876 017674   026354                                  MT0999  ; 0 SEC          ;NULL TEST
```

```
017676                                  MJTEST: SUBTST  <<SUBR  PARITY TEST DISPATCH>>
                                        ;**********************************************************************
                                        ;*SUBTEST        SUBR    PARITY TEST DISPATCH
                                        ;**********************************************************************
017676  012737  000002  002074                  MOV     #2,NOPAR                        ;INDICATE PARITY ACTION
017704  012737  000002  002276                  MOV     #2,PCBUMP                       ;TRAPS ADD 2 TO PC
017712  012737  060000  002362                  MOV     #FIRST,TESTADD
017720  012737  060002  002364                  MOV     #FIRST+2,TESTADD+2
017726  013700  002110                          MOV     PATTERN,R0                      ;GET PATTERN NUMBER
017732  006300                                  ASL     R0                              ;MAKE IT A WORD ADDRESS
017734                                          IF MJPAT(R0) NE #MT0034 AND MJPAT(R0) NE #MT0999
017754  104511                                    INVALIDATE            ;INVALIDATE BACKGROUND PATTERN ON "BANK"
017756                                          END ;OF IF MJPAT(R0)
017756  010637  002142                          MOV     SP,CTLKVEC                      ;SAVE VECTOR IN CASE OF ^K
017762  162737  000002  002142                  SUB     #2,CTLKVEC
017770  004770  020002                          CALL    @MJPAT(R0)                      ;INDEX OFF TABLE
017774  005037  002074                          CLR     NOPAR                           ;INDICATE PARITY ACTION
020000  000207                                  RETURN


                                                ;WARNING IF YOU CHANGE THIS TABLE ALSO
                                                ;CHANGE "$DDW0" - "$DDW5" (THE PATTERN BIT MAP)

                                                ;PAT     TIME            DISCRIPTION
020002                                  MJPAT:  ;NOTE MT0034 MUST BE FIRST & LAST
020002  026070                                  MT0034   ;<1 SEC         ;SOFT ERROR - BACKGROUND PATTERN TEST
020004  021132                                  MT0006   ;<1 SEC         ;INITIAL DATA TEST
020006  021674                                  MT0017   ;<1 SEC         ;HOLDING 1'S & 0'S TEST
020010  021166                                  MT0007   ;<1 SEC         ;ADDRESS BIT TEST
020012  020142                                  MT0001   ;<1 SEC         ;ADDRESS TEST
020014  020262                                  MT0002   ;<1 SEC         ;COMPLEMENT ADDRESS TEST
020016  020422                                  MT0003   ; 1 SEC         ;3 XOR 9 WORST CASE NOISE TEST
020020  020654                                  MT0004   ; 1 SEC         ;ROTATING ZEROS TEST
020022  020776                                  MT0005   ; 1 SEC         ;ROTATING ONES TEST
020024  023006                                  MT0021   ; 1 SEC         ;MARCHING 0'S & 1'S TEST
020026  026242                                  MT0035   ;<1 SEC         ;WORSE CASE NOISE PARITY TEST
020030  023260                                  MT0022   ;10 SEC         ;REFRESH TEST
020032  023312                                  MT0023   ;10 SEC         ;SHIFTING DIAGONAL  TEST
020034  023670                                  MT0026   ;<1 SEC         ;RANDOM DATA TEST
020036  023356                                  MT0024   ;20 SEC         ;FAST GALLOPING PATTERN TEST
020040  025160                                  MT0031   ; 3 SEC         ;SOB-A-LONG TEST
020042  025350                                  MT0032   ;<1 SEC         ;WRITE RECOVERY TEST
020044  025702                                  MT0033   ;35 SEC         ;BRANCH GOBBLE TEST
020046  026070                                  MT0034   ;<1 SEC         ;SOFT ERROR - BACKGROUND PATTERN TEST
                                                ;NOTE MT0034 MUST BE FIRST & LAST
020050  026354                                  MT0999   ; 0 SEC         ;NULL TEST
020052  026354                                  MT0999   ; 0 SEC         ;NULL TEST
020054  026354                                  MT0999   ; 0 SEC         ;NULL TEST
020056  026354                                  MT0999   ; 0 SEC         ;NULL TEST
020060  026354                                  MT0999   ; 0 SEC         ;NULL TEST
```

```
                                        .SBTTL  PATTERNS

                                        .SBTTL  MEMORY TEST SETUP ROUTINES
020062                          MT0000: SUBTST  <<MT0000        SETUP DATA PATTERN TEST>>
                                ;*******************************************************************************
                                ;*SUBTEST               MT0000  SETUP DATA PATTERN TEST
                                ;*******************************************************************************
020062  005037  002260                  CLR     REALPAT                 ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
020066  012700  060000                  MOV     #FIRST,R0
020072  012701  040000                  MOV     #SIZE,R1
020076  004737  036330                  CALL    REGCOPY
020102  022737  000001  003710          CMP     #1,PROTYP               ;ARE WE ON AN 11/44?
020110  001406                          BEQ     1$                      ;BRANCH IF YES
020112  012737  026774  002254          MOV     #MTP000,SUPDOADD        ;ELSE DO PATTERN IN MAIN MEMORY
020120  004737  026602                  CALL    SUPD03
020124  000207                          RETURN
020126                          1$:     BMOV    MTP000
020134  004737  026424                  CALL    SUPD01                  ;DO IT IN SUPERVISOR MODE
020140  000207                          RETURN
020142                          MT0001: SUBTST  <<MT0001        SETUP ADDRESS TEST>>
                                ;*******************************************************************************
                                ;*SUBTEST               MT0001  SETUP ADDRESS TEST
                                ;*******************************************************************************
020142  012737  000001  002260          MOV     #1,REALPAT              ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
020150  012700  060000                  MOV     #FIRST,R0
020154  012701  040000                  MOV     #SIZE,R1
020160  005737  002426                  TST     NOSUPER
020164  001005                          BNE     2$
020166  023737  172252  172254          CMP     SIPAR5,SIPAR6
020174  001007                          BNE     4$
020176  000404                          BR      3$
020200  023737  177652  177654  2$:     CMP     UIPAR5,UIPAR6
020206  001002                          BNE     4$
020210  012701  030000          3$:     MOV     #30000,R1
020214  005002                  4$:     CLR     R2
020216  004737  036330                  CALL    REGCOPY
020222  022737  000001  003710          CMP     #1,PROTYP               ;IS THIS AN 11/44?
020230  001406                          BEQ     1$                      ;BRANCH IF IT IS
020232  012737  027020  002254          MOV     #MTP001,SUPDOADD        ;SET UP CALLING ADDRESS
020240  004737  026602                  CALL    SUPD03
020244  000207                          RETURN
020246                          1$:     BMOV    MTP001
020254  004737  026424                  CALL    SUPD01                  ;DO IT IN SUPERVISOR MODE
020260  000207                          RETURN
020262                          MT0002: SUBTST  <<MT0002        SETUP COMPLEMENT ADDRESS TEST>>
                                ;*******************************************************************************
                                ;*SUBTEST               MT0002  SETUP COMPLEMENT ADDRESS TEST
                                ;*******************************************************************************
020262  012737  000002  002260          MOV     #2,REALPAT              ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
020270  012700  160000                  MOV     #LAST+2,R0
020274  012701  040000                  MOV     #SIZE,R1
020300  012704  060000                  MOV     #FIRST,R4
020304  012705  100001                  MOV     #100001,R5
020310  005737  002426                  TST     NOSUPER
020314  001005                          BNE     2$
020316  023737  172252  172254          CMP     SIPAR5,SIPAR6
020324  001013                          BNE     4$
```

```
 3977 020326  000404                            BR      3$
 3978 020330  023737  177652  177654  2$:       CMP     UIPAR5,UIPAR6
 3979 020336  001006                            BNE     4$
 3980 020340  012701  030000          3$:       MOV     #30000,R1
 3981 020344  012700  140000                    MOV     #140000,R0
 3982 020350  012705  120001                    MOV     #120001,R5
 3983 020354  012702  000001          4$:       MOV     #1,R2
 3984 020360  010103                            MOV     R1,R3
 3985 020362  022737  000001  003710            CMP     #1,PROTYP               ;IS THIS AN 11/44?
 3986 020370  001406                            BEQ     1$                      ;BRANCH IF TRUE
 3987 020372  012737  027052  002254            MOV     #MTP002,SUPDOADD        ;SET UP CALLING ADDRESS
 3988 020400  004737  026602                    CALL    SUPDO3
 3989 020404  000207                            RETURN
 3990 020406                          1$:       BMOV    MTP002
 3991 020414  004737  026424                    CALL    SUPDO1
 3992 020420  000207                            RETURN
 3995 020422                                    MT0003: SUBTST  <<MT0003          SETUP 3 XOR 9 WORST CASE NOISE TEST>>
                                                ;*******************************************************************************
                                                ;*SUBTEST        MT0003  SETUP 3 XOR 9 WORST CASE NOISE TEST
                                                ;*******************************************************************************
 3996 020422                                            IF EUFLAG IS TRUE THEN $RETURN
 3997 020432  012737  000003  002260                    MOV     #3,REALPAT                      ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
 3998 020440  005037  002276                            CLR     PCBUMP                          ;TRAPS DO NOT ADD TO PC
 3999 020444  004737  036340                    1$:     CALL    FLIPWARN                        ;SETUP WARNING CONSTANTS & R2
 4000 020450  012701  060000                    2$:     MOV     #FIRST,R1                       ;R1 <-- STARTING ADDRESS
 4001 020454  012703  020000                            MOV     #20000,R3
 4002 020460  072327  177770                            ASH     #-8.,R3                         ;R3 <-- R3 / 256.
 4003 020464  012702  000004                            MOV     #4,R2                           ;SMALL LOOP SIZE
 4004 020470  012705  000100                            MOV     #64.,R5                         ;MEDIUM LOOP SIZE
 4005 020474  022737  000001  003710                    CMP     #1,PROTYP                       ;IS THIS AN 11/44?
 4006 020502  001415                                    BEQ     3$                              ;BRANCH IF IT IS
 4007 020504  104415                                    SAVREG
 4008 020506  012737  027104  002254                    MOV     #MTPA03,SUPDOADD
 4009 020514  004737  026602                            CALL    SUPD03                          ;DO IT IN MAIN MEMORY
 4010 020520  104416                                    RESREG
 4011 020522  012737  027144  002254                    MOV     #MTPB03,SUPDOADD
 4012 020530  004737  026616                            CALL    SUPD04
 4013 020534  000442                                    BR      4$
 4014 020536                                    3$:     BMOV    MTPA03
 4015 020544  104415                                    SAVREG
 4016 020546  004737  026424                            CALL    SUPD01
 4017 020552                                            BMOV    MTPB03
 4018 020560                                            BMOV    MTPC03,KDPAR0,8.
 4019 020572                                            BMOV    MTPD03,SDPAR0,8.
 4020 020604  012737  172360  177642                    MOV     #KDPAR0,UIPAR1                  ;SET UP PAR LINKS
 4021 020612  012737  172260  172374                    MOV     #SDPAR0,KDPAR6
 4022 020620  012737  177644  172276                    MOV     #UIPAR2,SDPAR7
 4023 020626  012737  001032  172272                    MOV     #1032,SDPAR5                    ;CHANGE INST TO BR .+66 (BR TO KDPAR1)
 4024 020634  104416                                    RESREG
 4025 020636  004737  026440                            CALL    SUPD02
 4026 020642  022737  000003  002556            4$:     CMP     #3,FLIPLOC                      ;DONE WITH 4 PATTERNS
 4027                                                                                           ;[(0,177777);(177777,0);(401,177777);(177777,401)]?
 4028 020650  001275                                    BNE     1$                              ;NO - LOOP
 4029 020652  000207                                    RETURN
 4030
 4031 020654                                    MT0004: SUBTST  <<MT0004          SETUP ROTATING ZEROS TEST>>
                                                ;*******************************************************************************
                                                ;*SUBTEST        MT0004  SETUP ROTATING ZEROS TEST
                                                ;*******************************************************************************
 4032 020654  012737  000004  002260                    MOV     #4,REALPAT                      ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
 4033 020662  012737  000004  002276                    MOV     #4,PCBUMP                       ;TRAPS ADD 4 TO PC
 4034 020670  013702  002554                            MOV     ONES,R2
 4035 020674  004737  036470                            CALL    BACKGND                         ;WRITE BACKGROUND OF ONES
 4036 020700  012700  060000                            MOV     #FIRST,R0
 4037 020704  012701  040000                            MOV     #SIZE,R1
 4038 020710  022737  000001  003710                    CMP     #1,PROTYP                       ;IS THIS AN 11/44?
 4039 020716  001406                                    BEQ     1$                              ;BRANCH IF IT IS
 4040 020720  012737  027242  002254                    MOV     #MTPA04,SUPDOADD                ;SET UP LINKS
 4041 020726  004737  026616                            CALL    SUPD04
 4042 020732  000207                                    RETURN
 4043 020734                                    1$:     BMOV    MTPA04
 4044 020742                                            BMOV    MTPB04,KDPAR0,8.
 4045 020754  012737  172360  177652                    MOV     #KDPAR0,UIPAR5
```

```
020762 012737  177654  172376                  MOV     #UIPAR6,KDPAR7
020770 004737  026440                          CALL    SUPD02
020774 000207                                  RETURN
020776                               MT0005: SUBTST  <<MT0005         SETUP ROTATING ONES TEST>>
                                     ;*******************************************************************************
                                     ;*SUBTEST         MT0005  SETUP ROTATING ONES TEST
                                     ;*******************************************************************************
020776 012737  000005  002260                  MOV     #5,REALPAT                      ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
021004 012737  000004  002276                  MOV     #4,PCBUMP                       ;TRAPS ADD 4 TO PC
021012 005002                                  CLR     R2
021014 004737  036470                          CALL    BACKGND                         ;WRITE BACKGROUND OF ZEROS
021020 012700  060000                          MOV     #FIRST,R0
021024 012701  040000                          MOV     #SIZE,R1
021030 022737  000001  003710                  CMP     #1,PROTYP                       ;IS THIS AN 11/44?
021036 001414                                  BEQ     1$                              ;BRANCH IF IT IS
021040 012737  027316  002254                  MOV     #MTP005,SUPDOADD                ;SET UP LINKS
021046 012737  027332  027314                  MOV     #MTP005+14,MTPB04+16
021054 004737  026616                          CALL    SUPD04
021060 012737  027256  027314                  MOV     #MTPA04+14,MTPB04+16            ;RESET TEST'S ORIGINAL VALUE
021066 000207                                  RETURN
021070                               1$:       BMOV    MTP005
021076                                         BMOV    MTPB04,KDPAR0,8.
021110 012737  172360  177652                  MOV     #KDPAR0,UIPAR5
021116 012737  177654  172376                  MOV     #UIPAR6,KDPAR7
021124 004737  026440                          CALL    SUPD02
021130 000207                                  RETURN
```

```
4071 021132                          MT0006: SUBTST  <<MT0006        SETUP INITIAL DATA TEST>>
                                     ;********************************************************************************
                                     ;*SUBTEST       MT0006  SETUP INITIAL DATA TEST
                                     ;********************************************************************************
4072 021132  012737  000006  002260          MOV     #6,REALPAT              ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
4073 021140  012737  000004  002276          MOV     #4,PCBUMP               ;TRAPS ADD 4 TO PC
4074 021146  012701  002362                  MOV     #TESTADD,R1
4075 021152  012737  027352  002254          MOV     #MTP006,SUPDOADD
4076 021160  004737  026602                  CALL    SUPDO3                  ;DO IT IN SUPERVISOR MODE
4077 021164  000207                          RETURN
4078 021166                          MT0007: SUBTST  <<MT0007        SETUP ADDRESS BIT TEST>>
                                     ;********************************************************************************
                                     ;*SUBTEST       MT0007  SETUP ADDRESS BIT TEST
                                     ;********************************************************************************
4079 021166  012737  000007  002260          MOV     #7,REALPAT              ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
4080 021174  005002                          CLR     R2
4081 021176  004737  036470                  CALL    BACKGND                 ;OF ZEROS
4082 021202  012701  060000                  MOV     #FIRST,R1
4083 021206  012702  000001                  MOV     #1,R2
4084 021212  050201                          BIS     R2,R1
4085 021214  012737  027552  002254          MOV     #MTP007,SUPDOADD
4086 021222  004737  026602                  CALL    SUPDO3                  ;DO IT IN SUPERVISOR MODE
4087 021226  000207                          RETURN
4088 021230                          MT0010: SUBTST  <<MT0010        SETUP BYTE ADDRESSING TEST>>
                                     ;********************************************************************************
                                     ;*SUBTEST       MT0010  SETUP BYTE ADDRESSING TEST
                                     ;********************************************************************************
4089 021230  012737  000010  002260          MOV     #10,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
4090 021236  012737  000004  002276          MOV     #4,PCBUMP               ;TRAPS ADD 4 TO PC
4091 021244  013704  002362                  MOV     TESTADD,R4
4092 021250  012737  027652  002254          MOV     #MTP010,SUPDOADD
4093 021256  004737  026602                  CALL    SUPDO3                  ;DO IT IN SUPERVISOR MODE
4094 021262  000207                          RETURN
```

```
 4097 021264                            MT0011: SUBTST  <<MT0011        SETUP CREATE SINGLE BIT ERROR TEST>>
                                        ;*********************************************************************************
                                        ;*SUBTEST        MT0011  SETUP CREATE SINGLE BIT ERROR TEST
                                        ;*********************************************************************************
 4098 021264                                    IF ACTFLAG IS TRUE OR APTFLAG IS TRUE
 4099 021300                                      IF $PASS NE #0 THEN $RETURN
 4100 021310                                    END; OF IF ACTFLAG
 4101 021310  012737  000011  002260            MOV     #11,REALPAT                     ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
 4102 021316  012737  027760  002254            MOV     #MTP011,SUPDOADD
 4103 021324  004737  026602                    CALL    SUPDO3                          ;DO IT IN SUPERVISOR MODE
 4104 021330  000207                            RETURN
 4105 021332                            MT0012: SUBTST  <<MT0012        SETUP WRITE BYTE CLEARS SBE TEST>>
                                        ;*********************************************************************************
                                        ;*SUBTEST        MT0012  SETUP WRITE BYTE CLEARS SBE TEST
                                        ;*********************************************************************************
 4106 021332                                    IF ACTFLAG IS TRUE OR APTFLAG IS TRUE
 4107 021346                                      IF $PASS NE #0 THEN $RETURN
 4108 021356                                    END; OF IF ACTFLAG
 4109 021356  012737  000012  002260            MOV     #12,REALPAT                     ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
 4110 021364  013700  002102                    MOV     BANKINDEX,R0
 4111 021370                                    IF #BIT12 SET.IN CONFIG+2(R0)
 4112 021400  012705  040000                      MOV     #40000,R5
 4113 021404                                    ELSE
 4114 021406  012705  000002                      MOV     #2,R5
 4115 021412                                    END; OF IF #BIT12
 4116 021412  012737  030556  002254            MOV     #MTP012,SUPDOADD
 4117 021420  004737  026602                    CALL    SUPDO3                          ;DO IT IN SUPERVISOR MODE
 4118 021424  000207                            RETURN
 4119 021426                            MT0013: SUBTST  <<MT0013        SETUP CREATE DOUBLE BIT ERROR TEST>>
                                        ;*********************************************************************************
                                        ;*SUBTEST        MT0013  SETUP CREATE DOUBLE BIT ERROR TEST
                                        ;*********************************************************************************
 4120 021426                                    IF ACTFLAG IS TRUE OR APTFLAG IS TRUE
 4121 021442                                      IF $PASS NE #0 THEN $RETURN
 4122 021452                                    END; OF IF ACTFLAG
 4123 021452  012737  000013  002260            MOV     #13,REALPAT                     ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
 4124 021460  012737  031144  002254            MOV     #MTP013,SUPDOADD
 4125 021466  012737  000003  002074            MOV     #3,NOPAR                        ;INDICATE PARITY ACRION
 4126 021474  004737  026602                    CALL    SUPDO3                          ;DO IT IN SUPERVISOR MODE
 4127 021500  000207                            RETURN
 4128 021502                            MT0014: SUBTST  <<MT0014        SETUP WRITE INHIBIT DURING DATIP WITH DBE>>
                                        ;*********************************************************************************
                                        ;*SUBTEST        MT0014  SETUP WRITE INHIBIT DURING DATIP WITH DBE
                                        ;*********************************************************************************
 4129 021502                                    IF ACTFLAG IS TRUE OR APTFLAG IS TRUE
 4130 021516                                      IF $PASS NE #0 THEN $RETURN
 4131 021526                                    END; OF IF ACTFLAG
 4132 021526                                    IF KFLAG IS FALSE THEN $RETURN
 4133 021536  012737  000014  002260            MOV     #14,REALPAT                     ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
 4134 021544  012737  031660  002254            MOV     #MTP014,SUPDOADD
 4135 021552  004737  026602                    CALL    SUPDO3                          ;DO IT IN SUPERVISOR MODE
 4136 021556  000207                            RETURN
 4139 021560                                    MT0015: SUBTST  <<MT0015        SETUP WRITE INHIBIT OF BYTE WITH DBE>>
                                                ;*****************************************************************************
                                                ;*SUBTEST       MT0015  SETUP WRITE INHIBIT OF BYTE WITH DBE
                                                ;*****************************************************************************
 4140 021560                                            IF ACTFLAG IS TRUE OR APTFLAG IS TRUE
 4141 021574                                              IF $PASS NE #0 THEN $RETURN
 4142 021604                                            END ;OF IF ACTFLAG
 4143 021604  012737  000015  002260                    MOV     #15,REALPAT                 ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
 4144 021612  012737  032442  002254                    MOV     #MTP015,SUPDOADD
 4145 021620  004737  026602                            CALL    SUPDO3                      ;DO IT IN SUPERVISOR MODE
 4146 021624  000207                                    RETURN
 4147 021626                                    MT0016: SUBTST  <<MT0016        SETUP WRITE INHIBIT OF WORD WITH DBE>>
                                                ;*****************************************************************************
                                                ;*SUBTEST       MT0016  SETUP WRITE INHIBIT OF WORD WITH DBE
                                                ;*****************************************************************************
 4148 021626                                            IF ACTFLAG IS TRUE OR APTFLAG IS TRUE
 4149 021642                                              IF $PASS NE #0 THEN $RETURN
 4150 021652                                            END ;OF IF ACTFLAG
 4151 021652  012737  000016  002260                    MOV     #16,REALPAT                 ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
 4152 021660  012737  033206  002254                    MOV     #MTP016,SUPDOADD
 4153 021666  004737  026602                            CALL    SUPDO3                      ;DO IT IN SUPERVISOR MODE
 4154 021672  000207                                    RETURN
 4155 021674                                    MT0017: SUBTST  <<MT0017        SETUP HOLDING 1'S & 0'S>>
                                                ;*****************************************************************************
                                                ;*SUBTEST       MT0017  SETUP HOLDING 1'S & 0'S
                                                ;*****************************************************************************
 4156 021674  012737  000017  002260                    MOV     #17,REALPAT                 ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
 4157 021702  012737  033770  002254                    MOV     #MTP017,SUPDOADD
 4158 021710  004737  026602                            CALL    SUPDO3                      ;DO IT IN SUPERVISOR MODE
 4159 021714  000207                                    RETURN
```

```
   4162 021716                                  MT0020: SUBTST  <<MT0020        SETUP MARCHING 0'S & 1'S IN CHECKBITS TEST>>
                                                ;*******************************************************************************
                                                ;*SUBTEST        MT0020  SETUP MARCHING 0'S & 1'S IN CHECKBITS TEST
                                                ;*******************************************************************************
   4163 021716                                          IF ACTFLAG IS TRUE OR ACTFLAG IS TRUE
   4164 021732                                            IF $PASS NE #0 THEN $RETURN
   4165 021742                                          END ;OF IF ACTFLAG
   4166 021742  012737  000020  002260                  MOV     #20,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
   4167 021750  012737  000003  002074                  MOV     #3,NOPAR                ;INDICATE PARITY ACTION
   4168 021756  005001                                  CLR     R1                      ;CLEAR LOOP COUNTER
   4169 021760  005004                                  CLR     R4                      ;CLEAR INTERLEAVE ODD/EVEN FLAG
   4170 021762  012700  060000                          MOV     #FIRST,R0
   4171 021766  013702  002102                          MOV     BANKINDEX,R2            ;SET BANK INDEX
   4172 021772                                  MTL020: IF INTFLAG IS FALSE
   4173 022000                                            BEGIN MTB020
   4174 022000                                            IF NO22BIT IS TRUE
   4175 022006                                              IF BANK EQ #7
   4176 022016  012737  140000  002362                        MOV       #140000,TESTADD ;SET UP 12K NON-INTERLEAVED VIRT ADDR
   4177 022024  012705  140002                                MOV       #140002,R5
   4178 022030                                                LEAVE     MTB020
   4179 022032                                              END; OF IF BANK
   4180 022032                                            END; OF IF NO22BIT
   4181 022032                                            IF NO22BIT IS FALSE
   4182 022040                                              IF BANK EQ #177
   4183 022050  012737  140000  002362                        MOV       #140000,TESTADD
   4184 022056  012705  140000                                MOV       #140000,R5
   4185 022062                                                LEAVE     MTB020
   4186 022064                                              END; OF IF BANK
   4187 022064                                            END; OF IF NO22BIT
   4188 022064  012737  160000  002362                    MOV     #LAST+2,TESTADD
   4189 022072  012705  160002                            MOV     #LAST+4,R5
   4190 022076                                            END     MTB020
   4191 022076  010537  002364                            MOV     R5,TESTADD+2            ;SET UP NON-INTERLEAVED VIRT. ADDR.
   4192 022102                                          ELSE
   4193 022104  005737  002312                            TST     SKIPMK                  ;IS THIS BANK IN SKIP RANGE?
   4194 022110  001401                                    BEQ     1$                      ;BANK IS OUT OF RANGE - DO TEST
   4195 022112  000207                                    RETURN                          ;LEAVE TEST-BANK'S ALREADY TESTED
   4196 022114  012737  120000  002362  1$:               MOV     #120000,TESTADD         ;SET UP 1ST INTERLEAVED VIRT. ADDR.
   4197 022122  012705  160000                            MOV     #LAST+2,R5              ;SET UP END OF BANK FLAG
   4198 022126  010537  002364                            MOV     R5,TESTADD+2            ;SET UP 2ND INT'L. VIRT. ADDR.
   4199 022132  005237  002232                            INC     SPLTCSR                 ;FLAG THE MAPPING ROUTINE FOR INTERLEAVING
   4200 022136  005201                                    INC     R1                      ;SET LOOP COUNTER FOR INTERLEAVING
   4201 022140  005204                                    INC     R4                      ;SET ODD/EVEN FLAG
   4202 022142                                          END; OF IF INTFLAG
   4203 022142  016203  002624                          MOV       CONFIG(R2),R3         ;SET UP CSR NUMBER
   4204 022146                                          IF R4 EQ #2                     ;IF THE SECOND TIME AROUND
   4205 022154  060437  002362                            ADD     R4,TESTADD
   4206 022160  060437  002364                            ADD     R4,TESTADD+2            ;TEST THE A1 ASSERTED ADDRESSES
   4207 022164  060400                                    ADD     R4,R0
   4208 022166  060405                                    ADD     R4,R5
   4209 022170  072327  177775                            ASH     #-3,R3                  ;MOVE INTERLEAVED CSR NUMBER
   4210 022174                                          ELSE
   4211 022176  006303                                    ASL     R3                      ;MOVE CSR NUMBER
   4212 022200                                          END; IF R4
   4213 022200  000303                                  SWAB    R3
   4214 022202  042703  177741                          BIC     #^C36,R3
   4215 022206  010337  002146                          MOV     R3,CSRNO                ;MOVE R3 INTO CSR NUMBER
```

```
022212                                    IF #SWO SET.IN @SWR
022222  104506                              ENASBE                              ;TRAP ON SINGLE BIT ERRORS
022224                                    ELSE
022226  104472                              ECCINIT                             ;TRAP ON UNCORRECTABLE ERRORS
022230                                    END; OF IF #SWO
022230                                    PUSH    R2,R4
022234                                    FOR MTV020 := #0 TO R1
022240                                      PUSH  R1
022242  005002                              CLR   R2                            ;PATTERN TO WRITE INTO BANK
022244  004737  036470                      CALL  BACKGND                       ;SET UP ZEROS IN BANK
022250                                      IF NO22BIT IS TRUE AND MTV020 EQ #1 AND BANK EQ #3
022276  162737  020000  002362                SUB #20000,TESTADD                ;SET UP 12K INTERLEAVED BANK
022304  162705  020000                        SUB #20000,R5
022310  010537  002364                        MOV R5,TESTADD+2
022314                                      END; OF IF NO22BIT
022314  004737  022370                      CALL  MT020Z                        ;START TEST
022320  005237  002232                      INC   SPLTCSR                       ;UPDATE INTERLEAVED MAPPING FLAG
022324                                      POP   R1
022326                                    END; OF FOR MTV020
022340                                    POP     R4,R2
022344  005001                            CLR     R1                            ;RESET LOOP FLAG
022346  005037  002232                    CLR     SPLTCSR                       ;RESET INTERLEAVED MAP FLAG
022352  022704  000001                    CMP     #1,R4                         ;ODD/EVEN FLAG SET?
022356  001605                            BEQ     MTL020                        ;BRANCH IF TRUE
022360  005037  002074                    CLR     NOPAR                         ;INDICATE PARITY ACTION
022364  000207                            RETURN
022366  000000                    MTV020: 0                                     ;VARIABLE FOR PAT 20
```

```
 4244 022370  012702  000004          MT020Z: MOV     #4,R2                    ;SET UP WORD INCR/DECR AMOUNT
 4245 022374  013701  002362                  MOV     TESTADD,R1
 4246 022400  013704  002364                  MOV     TESTADD+2,R4
 4247 022404  012703  100000                  MOV     #BIT15,R3
 4248 022410                                  IF #SW11 SET.IN @SWR OR QVFLAG IS TRUE
 4249 022426                                    GOTO  MT020Y
 4250 022430                                  END ;OF IF #SW11
 4251 022430  022737  000001  003710          CMP     #1,PROTYP                ;IS THIS AN 11/44?
 4252 022436  001411                          BEQ     1$                       ;BRANCH IF IT IS
 4253 022440  012737  034046  002254          MOV     #MTPA20,SUPDOADD
 4254 022446  012737  034062  002264          MOV     #MTPA20+14,PARTHERE      ;VECTOR FOR TRAPS
 4255 022454  004737  026602                  CALL    SUPDO3
 4256 022460  000410                          BR      2$
 4257 022462                          1$:     BMOV    MTPA20
 4258 022470  012737  177654  002264          MOV     #UIPAR6,PARTHERE         ;VECTOR FOR TRAPS
 4259 022476  004737  026424                  CALL    SUPDO1
 4260 022502  022737  000001  003710  2$:     CMP     #1,PROTYP                ;IS THIS AN 11/44?
 4261 022510  001411                          BEQ     4$                       ;BRANCH IF IT IS
 4262 022512  012737  034076  002254          MOV     #MTPB20,SUPDOADD
 4263 022520  012737  034106  002264          MOV     #MTPB20+10,PARTHERE      ;VECTOR FOR TRAPS
 4264 022526  004737  026616                  CALL    SUPDO4
 4265 022532  000410                          BR      MT020Y
 4266 022534                          4$:     BMOV    MTPB20
 4267 022542  012737  177650  002264          MOV     #UIPAR4,PARTHERE         ;VECTOR FOR TRAPS
 4268 022550  004737  026440                  CALL    SUPDO2
 4269 022554  005737  002134          MT020Y: TST     INTFLAG                  ;ARE WE INTERLEAVED?
 4270 022560  001405                          BEQ     7$                       ;BRANCH IF NOT INTERLEAVED
 4271 022562  162701  040000                  SUB     #40000,R1                ;RESET FIRST WORD TO BEGINNING OF BANK
 4272 022566  162704  040000                  SUB     #40000,R4                ;RESET SECOND WORD TO BEGINNING OF BANK
 4273 022572  000404                          BR      8$
 4274 022574  012701  060000          7$:     MOV     #FIRST,R1                ;RESET FIRST WORD TO BEGINNING OF BANK
 4275 022600  012704  060002                  MOV     #FIRST+2,R4              ;RESET SECOND WORD TO BEGINNING OF BANK
 4276 022604  022737  000001  003710  8$:     CMP     #1,PROTYP                ;IS THIS AN 11/44?
 4277 022612  001411                          BEQ     1$                       ;BRANCH IF IT IS
 4278 022614  012737  034126  002254          MOV     #MTPC20,SUPDOADD
 4279 022622  012737  034136  002264          MOV     #MTPC20+10,PARTHERE      ;VECTOR FOR TRAPS
 4280 022630  004737  026602                  CALL    SUPDO3
 4281 022634  000410                          BR      2$
 4282 022636                          1$:     BMOV    MTPC20
 4283 022644  012737  177650  002264          MOV     #UIPAR4,PARTHERE         ;VECTOR FOR TRAPS
 4284 022652  004737  026424                  CALL    SUPDO1
 4285 022656  022737  000001  003710  2$:     CMP     #1,PROTYP                ;IS THIS AN 11/44?
 4286 022664  001411                          BEQ     3$                       ;BRANCH IF IT IS
 4287 022666  012737  034156  002254          MOV     #MTPD20,SUPDOADD
 4288 022674  012737  034172  002264          MOV     #MTPD20+14,PARTHERE      ;VECTOR FOR TRAPS
 4289 022702  004737  026616                  CALL    SUPDO4
 4290 022706  000410                          BR      4$
 4291 022710                          3$:     BMOV    MTPD20
 4292 022716  012737  177654  002264          MOV     #UIPAR6,PARTHERE         ;VECTOR FOR TRAPS
 4293 022724  004737  026440                  CALL    SUPDO2
 4294 022730  022737  000001  003710  4$:     CMP     #1,PROTYP                ;IS THIS AN 11/44?
 4295 022736  001411                          BEQ     5$                       ;BRANCH IF IT IS
 4296 022740  012737  034206  002254          MOV     #MTPE20,SUPDOADD
 4297 022746  012737  034216  002264          MOV     #MTPE20+10,PARTHERE      ;VECTOR FOR TRAPS
 4298 022754  004737  026616                  CALL    SUPDO4
 4299 022760  000410                          BR      6$
 4300 022762                          5$:     BMOV    MTPE20
```

```
   4301 022770  012737  177650  002264          MOV     #UIPAR4,PARTHERE        ;VECTOR FOR TRAPS
   4302 022776  004737  026440                  CALL    SUPD02
   4303 023002  104503                  6$:     CLR1CSR                         ;CLEAR 1 SELECTED CSR
   4304 023004  000207                          RETURN
```

```
023006                                  MT0021: SUBTST  <<MT0021        SETUP MARCHING 0'S & 1'S TEST>>
                                        ;***********************************************************************
                                        ;*SUBTEST       MT0021  SETUP MARCHING 0'S & 1'S TEST
                                        ;***********************************************************************
023006                                          SET NOSCOPE
023014  012737  000021  002260                  MOV     #21,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
023022  013702  002572                          MOV     BAKPAT,R2
023026  004737  036470                          CALL    BACKGND
023032  010203                                  MOV     R2,R3
023034  000303                                  SWAB    R3
023036  012701  160000                          MOV     #LAST+2,R1
023042  010105                                  MOV     R1,R5
023044  012704  060000                          MOV     #FIRST,R4
023050  022737  000001  003710                  CMP     #1,PROTYP       ;IS THIS AN 11/44?
023056  001441                                  BEQ     1$              ;BRANCH IF IT IS
023060  022737  000003  003710                  CMP     #3,PROTYP       ;IS THIS AN 11/24?
023066  001407                                  BEQ     3$              ;BRANCH IF SO
023070  022737  000007  002100                  CMP     #7,BANK
023076  001003                                  BNE     3$
023100  012701  140000                          MOV     #140000,R1
023104  010105                                  MOV     R1,R5
023106  012737  034232  002254  3$:             MOV     #MTPA21,SUPDOADD
023114  004737  026602                          CALL    SUPDO3
023120  012737  034262  002254                  MOV     #MTPB21,SUPDOADD
023126  004737  026616                          CALL    SUPDO4
023132  010401                                  MOV     R4,R1
023134  012737  034316  002254                  MOV     #MTPC21,SUPDOADD
023142  004737  026616                          CALL    SUPDO4
023146  012737  034352  002254                  MOV     #MTPD21,SUPDOADD
023154  004737  026616                          CALL    SUPDO4
023160  000434                                  BR      2$
023162  022737  000177  002100  1$:             CMP     #177,BANK
023170  001003                                  BNE     4$
023172  012701  140000                          MOV     #140000,R1
023176  010105                                  MOV     R1,R5
023200                                  4$:     BMOV    MTPA21
023206  004737  026424                          CALL    SUPDO1

023212                                          BMOV    MTPB21
023220  004737  026440                          CALL    SUPDO2

023224  010401                                  MOV     R4,R1
023226                                          BMOV    MTPC21
023234  004737  026440                          CALL    SUPDO2

023240                                          BMOV    MTPD21
023246  004737  026440                          CALL    SUPDO2
023252  005037  002410          2$:             CLR     NOSCOPE
023256  000207                                  RETURN
```

```
  4354 023260                           MT0022: SUBTST  <<MT0022        SETUP REFRESH & SHIFTING DIAGONAL TEST>>
                                        ;*****************************************************************************
                                        ;*SUBTEST        MT0022  SETUP REFRESH & SHIFTING DIAGONAL TEST
                                        ;*****************************************************************************
  4355 023260  004737  026370                   CALL    KAMITEST                        ;CHECK FOR KAMIKAZE MODE
  4356 023264                                   ON.ERROR THEN $RETURN                   ;IF NOT IN KAMIKAZE MODE RETURN
  4357 023270  012737  000022  002260           MOV     #22,REALPAT                     ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
  4358 023276  012737  034402  002254           MOV     #MTP022,SUPDOADD
  4359 023304  004737  026602                   CALL    SUPDO3                          ;DO IT IN SUPERVISOR MODE
  4360 023310  000207                           RETURN
  4361
  4362 023312                           MT0023: SUBTST  <<MT0023        SHIFTING DIAGONAL TEST>>
                                        ;*****************************************************************************
                                        ;*SUBTEST        MT0023  SHIFTING DIAGONAL TEST
                                        ;*****************************************************************************
  4363 023312  004737  026370                   CALL    KAMITEST                        ;CHECK FOR KAMIKAZE MODE
  4364 023316                                   ON.ERROR THEN $RETURN                   ;IF NOT IN KAMIKAZE MODE RETURN
  4365 023322  012737  000023  002260           MOV     #23,REALPAT                     ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
  4366 023330  012737  034402  002254           MOV     #MTP022,SUPDOADD
  4367 023336                                   SET     DIAGFLAG                        ;IDENTIFY DIAGONAL TEST TO MTP022
  4368 023344  004737  026602                   CALL    SUPDO3                          ;DO IT IN SUPERVISOR MODE
  4369 023350  005037  002002                   CLR     DIAGFLAG
  4370 023354  000207                           RETURN
```

```
023356                              MT0024: SUBTST  <<MT0024        SETUP FAST GALLOPING PATTERN TEST>>
                                    ;*****************************************************************************
                                    ;*SUBTEST       MT0024  SETUP FAST GALLOPING PATTERN TEST
                                    ;*****************************************************************************
023356  004737  026370                      CALL    KAMITEST                ;CHECK FOR KAMIKAZE MODE
023362                                      ON.ERROR THEN $RETURN           ;IF NOT IN KAMIKAZE MODE RETURN
023366                                      SET     NOSCOPE
023374  012737  000024  002260              MOV     #24,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
023402  013702  002572                      MOV     BAKPAT,R2
023406  004737  036470                      CALL    BACKGND
023412  010203                              MOV     R2,R3
023414  010304                              MOV     R3,R4
023416  000304                              SWAB    R4
023420  012701  060000                      MOV     #FIRST,R1
023424  012705  157776                      MOV     #LAST,R5
023430  022737  000001  003710              CMP     #1,PROTYP
023436  001417                              BEQ     1$
023440  022737  000003  003710              CMP     #3,PROTYP
023446  001406                              BEQ     3$
023450  022737  000007  002100              CMP     #7,BANK
023456  001002                              BNE     3$
023460  012705  137776                      MOV     #137776,R5
023464  104415                      3$:     SAVREG
023466  012737  035116  002254              MOV     #MTPB24,SUPDOADD
023474  000440                              BR      2$
023476  022737  000177  002526      1$:     CMP     #177,LASTBANK
023504  001002                              BNE     4$
023506  012705  137776                      MOV     #137776,R5
023512  104415                      4$:     SAVREG
023514                                      BMOV    MTPA24
023522                                      BMOV    MTPB24,SDPAR0,8.
023534                                      BMOV    MTPC24,KDPAR0,8.
023546  012737  172260  002254              MOV     #SDPAR0,SUPDOADD
023554  012737  172260  177676              MOV     #SDPAR0,UDPAR7          ;SET UP PAR LINKS
023562  012737  172360  172272              MOV     #KDPAR0,SDPAR5
023570  012737  177660  172374              MOV     #UDPAR0,KDPAR6
023576  004737  026616              2$:     CALL    SUPD04

                                            ;DO IT AGAIN FOR COMPLEMENT DATA
023602  104416                              RESREG
023604  000302                              SWAB    R2
023606  000303                              SWAB    R3
023610  004737  026616                      CALL    SUPD04
023614  005037  002410                      CLR     NOSCOPE
023620  000207                              RETURN
023622                              MT0025: SUBTST  <<MT0025        SETUP INTERRUPT ENABLE TEST>>
                                    ;*****************************************************************************
                                    ;*SUBTEST       MT0025  SETUP INTERRUPT ENABLE TEST
                                    ;*****************************************************************************
023622                                      IF ACTFLAG IS TRUE OR APTFLAG IS TRUE
023636                                        IF $PASS NE #0 THEN $RETURN
023646                                      END ;OF IF ACTFLAG
023646  012737  000025  002260              MOV     #25,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
023654  012737  035150  002254              MOV     #MTP025,SUPDOADD
023662  004737  026602                      CALL    SUPD03                  ;DO IT IN SUPERVISOR MODE
023666  000207                              RETURN
```

```
023670                             MT0026: SUBTST  <<MT0026        SETUP RANDOM DATA TEST>>
                                   ;*****************************************************************************
                                   ;*SUBTEST        MT0026  SETUP RANDOM DATA TEST
                                   ;*****************************************************************************
023670  012737  000026  002260             MOV     #26,REALPAT
023676  005037  002276                     CLR     PCBUMP                          ;TRAPS DO NOT ADD TO THE PC
023702  013703  002544                     MOV     SEEDLO,R3                       ;INITIALIZE RANDOM NUMBERS
023706  013702  002542                     MOV     SEEDHI,R2
023712  010305                             MOV     R3,R5
023714  010204                             MOV     R2,R4
023716  012701  060000                     MOV     #FIRST,R1
023722  012700  020000                     MOV     #SIZE/2,R0
023726  022737  000001  003710             CMP     #1,PROTYP                       ;DO WE HAVE AN 11/44?
023734  001437                             BEQ     1$                              ;BRANCH IF WE DO
023736  022737  000003  003710             CMP     #3,PROTYP                       ;11/24?
023744  001406                             BEQ     3$                              ;BRANCH IF SO
023746  022737  000007  002100             CMP     #7,BANK
023754  001002                             BNE     3$
023756  012700  014000                     MOV     #14000,R0
023762  104415                     3$:     SAVREG
023764  012737  035622  035722             MOV     #MTPA26+4,MTPD26+14
023772  012737  035616  002254             MOV     #MTPA26,SUPDOADD
024000  004737  026602                     CALL    SUPD03
024004  005037  035646                     CLR     RANODD                          ;FOR ERROR REPORTING
024010  012737  035636  035722             MOV     #MTPB26+4,MTPD26+14             ;SET UP NEXT LINK
024016  012737  035632  002254             MOV     #MTPB26,SUPDOADD
024024  104416                             RESREG
024026  004737  026602                     CALL    SUPD03
024032  000452                             BR      2$
024034  022737  000177  002100     1$:     CMP     #177,BANK
024042  001002                             BNE     4$
024044  012700  014000                     MOV     #14000,R0
024050  104415                     4$:     SAVREG
024052                                     BMOV    MTPA26                          ;WRITE ROUTINE TO FAST MEMORY
024060                                     BMOV    MTPC26,KDPAR0,8.                ;RANDOM SUBPROGRAM TO FAST MEMORY
024072  012737  000730  172376             MOV     #730,KDPAR7                     ;WRITES "BR .-116" IN (BR SDPAR0)
024100                                     BMOV    MTPD26,SDPAR0,8.                ;RANDOM SUBSUBPROGRAM TO FAST MEMORY
024112  012737  172360  177642             MOV     #KDPAR0,UIPAR1
024120  012737  177644  172274             MOV     #UIPAR2,SDPAR6
024126  004737  026424                     CALL    SUPD01                          ;WRITE RANDOM DATA
024132  005037  035646                     CLR     RANODD                          ;FOR ERROR REPORTING
024136                                     BMOV    MTPB26                          ;READ ROUTINE TO FAST MEMORY
024144  012737  172360  177642             MOV     #KDPAR0,UIPAR1                  ;SET UP PAR LINK
024152  104416                             RESREG
024154  004737  026424                     CALL    SUPD01                          ;READ RANDOM DATA
024160  010337  002544             2$:     MOV     R3,SEEDLO                       ;UPDATE FOR NEW RANDOM NUMBERS
024164  010237  002542                     MOV     R2,SEEDHI
024170  000207                             RETURN
```

```
024172                          MT0027: SUBTST  <<MT0027        UNIQUE BANK TEST>>
                                ;************************************************************************
                                ;*SUBTEST        MT0027  UNIQUE BANK TEST
                                ;************************************************************************
                                        ;MAKE SURE THAT EACH BANK CAN HAVE UNIQUE DATA
                                        ;WRITE AND READ THE BANK NUMBER IN EACH BANK (EXCEPT WHERE THE PROGRAM IS)
024172  012737  000027  002260          MOV     #27,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
024200  104502                          CLRCSR                  ;CLEAR CSRS
024202  022737  000001  003710          CMP     #1,PROTYP               ;IS THIS AN 11/44?
024210  001404                          BEQ     1$                      ;BRANCH IF TRUE
024212  012737  026602  002472          MOV     #SUPDO3,LINK1           ;SET UP LINK
024220  000414                          BR      STAR27                  ;BRANCH TO RUN
024222                          1$:     BMOV    MTP034
024230  012737  177646  002254  WARN7:  MOV     #UIPAR3,SUPDOADD
024236  012737  026424  002472          MOV     #SUPDO1,LINK1           ;SET UP LINK
024244                                  SET NOFSMODE
024252                          STAR27: FOR I := #1 TO #2
024260                                    FOR BANK := #0 TO LASTBANK
024264  004737  044240                      CALL EXBANK
024270                                      IF ACFLAG IS TRUE AND RRFLAG IS FALSE
024304  104511                                INVALIDATE                ;INVALIDATE BACKGROUND PATTERN ON "BANK"
024306                                        LET R2 := BANK
024312  012700  060000                        MOV       #FIRST,R0
024316  010004                                MOV       R0,R4
024320  012701  040000                        MOV       #SIZE,R1
024324  010103                                MOV       R1,R3
024326                                        IF I EQ #1
024336  022737  000001  003710                  CMP     #1,PROTYP
024344  001403                                  BEQ     2$
024346  012737  036142  002254                  MOV     #MTP034,SUPDOADD
024354  004777  156112                      2$: CALL    @LINK1
024360                                        END ;OF IF
024360                                        IF I EQ #2
024370  022737  000001  003710                  CMP     #1,PROTYP
024376  001403                                  BEQ     3$
024400  012737  036150  002254                  MOV     #MTP034+6,SUPDOADD
024406  004737  026602                      3$: CALL    SUPDO3
024412                                        END ;OF IF
024412                                      END ;OF IF
024412                                    END ;OF FOR BANK
024426                                  END ;OF FOR I
024442                                  IF FS7FLAG IS TRUE
024450  005037  002376                    CLR   NOFSMODE
024454  000207                            RETURN
024456                                  END ;OF IF FS7FLAG
024456                                  FOR I := #1 TO #2
024464                                    FOR BANK := LASTBANK DOWNTO #0
024472  004737  044240                      CALL EXBANK
024476                                      IF ACFLAG IS TRUE AND RRFLAG IS FALSE
024512                                        LET R2 := BANK
024516  005102                                COM       R2
024520  012700  060000                        MOV       #FIRST,R0
024524  010004                                MOV       R0,R4
024526  012701  040000                        MOV       #SIZE,R1
024532  010103                                MOV       R1,R3
024534                                        IF I EQ #1
024544  022737  000001  003710                  CMP     #1,PROTYP
```

```
 4525 024552  001403                                          BEQ     4$
 4526 024554  012737  036142  002254                          MOV     #MTP034,SUPDOADD
 4527 024562  004777  155704                              4$: CALL    @LINK1
 4528 024566                                                END ;OF IF
 4529 024566                                                IF I EQ #2
 4530 024576  022737  000001  003710                          CMP     #1,PROTYP
 4531 024604  001403                                          BEQ     5$
 4532 024606  012737  036150  002254                          MOV     #MTP034+6,SUPDOADD
 4533 024614  004737  026602                              5$: CALL    SUPDO3
 4534 024620                                                END ;OF IF
 4535 024620                                              END ;OF IF
 4536 024620                                            END ;OF FOR BANK
 4537 024634                                          END ;OF FOR I
 4538 024650  005037  002376                          CLR     NOFSMODE
 4539 024654  000207                                  RETURN
 4542 024656                            MT0030: SUBTST  <<MT0030        SETUP FLUSH OUT DBE'S TEST>>
                                        ;*****************************************************************************
                                        ;*SUBTEST        MT0030  SETUP FLUSH OUT DBE'S TEST
                                        ;*****************************************************************************
 4543 024656  005037  002256                    CLR     PASFLG
 4544 024662                                    SET     FULLREL
 4545 024670  012737  000030  002260    MTA030: MOV     #30,REALPAT                     ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
 4546 024676  012737  000001  002074            MOV     #1,NOPAR                        ;INDICATE COUNT PARITY ERRORS
 4547 024704  022737  000001  003710            CMP     #1,PROTYP
 4548 024712  001007                            BNE     4$
 4549 024714                                    BMOV    MTP030
 4550 024722  012737  026424  002472            MOV     #SUPD01,LINK1
 4551 024730  000406                            BR      1$
 4552 024732  012737  026602  002472    4$:     MOV     #SUPD03,LINK1
 4553 024740  012737  035724  002254            MOV     #MTP030,SUPDOADD
 4554 024746  104470                    1$:     ECCDIS                  ;DISABLE ERROR CORRECTION
 4555 024750                                    SET     NOFSMODE,NOSCOPE
 4556 024764                                    FOR BANK := #0 TO LASTBANK
 4557 024770  004737  044240                      CALL  EXBANK
 4558 024774                                      IF MKFLAG IS TRUE
 4559 025002                                        IF ACFLAG IS TRUE AND RRFLAG IS FALSE
 4560 025016  012701  040000                          MOV       #SIZE,R1
 4561 025022  012700  060000                          MOV       #FIRST,R0
 4562 025026  004777  155440                          CALL      @LINK1
 4563 025032                                        END ;OF IF ACFLAG
 4564 025032                                      END ;OF IF MKFLAG
 4565 025032                                    END ;OF FOR
 4566 025046                                    IF PASFLG IS FALSE
 4567 025054                                      SET PASFLG
 4568 025062  104502                              CLRCSR                                  ;CLEAR CSRS
 4569 025064  004737  042466                      CALL  RELOCATE
 4570 025070                                      ON.ERROR
 4571 025072  104472                                ECCINIT                 ;TRAP ON DOUBLE BIT ERRORS (NORMAL)
 4572 025074                                        CLEAR         NOFSMODE,NOSCOPE,FULLREL
 4573 025110  000207                                RETURN
 4574 025112                                      END ;OF ON.ERROR
 4575 025112  013737  002270  002100              MOV   NEWBANK,BANK
 4576 025120  004737  044240                      CALL  EXBANK
 4577 025124  004737  024670                      CALL  MTA030
 4578 025130  104472                              ECCINIT                   ;TRAP ON DOUBLE BIT ERRORS (NORMAL)
 4579 025132  004737  043356                      CALL  UNRELOCATE
 4580 025136  000207                              RETURN
 4581 025140                                    END ;OF IF PASFLG
 4582 025140  104472                            ECCINIT         ;TRAP ON DOUBLE BIT ERRORS (NORMAL)
 4583 025142                                    CLEAR   NOFSMODE,NOSCOPE,FULLREL
 4584 025156  000207                            RETURN
```

```
025160                              MT0031: SUBTST  <<MT0031        SETUP SOB-A-LONG TEST>>
                                    ;*************************************************************************
                                    ;*SUBTEST        MT0031  SETUP SOB-A-LONG TEST
                                    ;*************************************************************************
025160  004737  026370                      CALL    KAMITEST                        ;CHECK FOR KAMIKAZE MODE
025164                                      ON.ERROR THEN $RETURN                   ;IF NOT IN KAMIKAZE MODE RETURN
025170                                      SET     NOSCOPE
025176  012737  000031  002260              MOV     #31,REALPAT                     ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
025204  005037  002074                      CLR     NOPAR                           ;SETUP PARITY ACTION
025210                                      MAP     BANK                            ;MAP FIRST SO BLOCK MOVE WORKS
025224                                      TESTAREA                                ;ENTER TEST MODE
025232                                      BMOV    MTP031,FIRST,SOBLENGTH/2
025244  104417                              KERNEL                                  ;ENTER KERNEL MODE
025246  013702  002532                      MOV     SOBK,R2
025252  010200                              MOV     R2,R0
025254  012701  100776                      MOV     #100776,R1                      ;COMPLEMENT OF INSTRUCTION "SOB R0,DOT"
025260  012705  060056                      MOV     #FIRST+SOBLENGTH,R5
025264  012737  060002  002254              MOV     #FIRST+2,SUPDOADD
025272  012737  160000  002472              MOV     #LAST+2,LINK1
025300  005737  002426                      TST     NOSUPER
025304  001005                              BNE     1$
025306  023737  172252  172254              CMP     SIPAR5,SIPAR6
025314  001405                              BEQ     2$
025316  000407                              BR      3$
025320  023737  177652  177654  1$:         CMP     UIPAR5,UIPAR6
025326  001003                              BNE     3$
025330  012737  140000  002472  2$:         MOV     #140000,LINK1
025336  004737  026616          3$:         CALL    SUPD04
025342  005037  002410                      CLR     NOSCOPE
025346  000207                              RETURN
```

```
  4616 025350                           MT0032: SUBTST  <<MT0032        SETUP WRITE RECOVERY TEST>>
                                        ;*******************************************************************************
                                        ;*SUBTEST        MT0032  SETUP WRITE RECOVERY TEST
                                        ;*******************************************************************************
  4617 025350  004737  026370                   CALL    KAMITEST                        ;CHECK FOR KAMIKAZE MODE
  4618 025354                                   ON.ERROR THEN $RETURN                   ;IF NOT IN KAMIKAZE MODE RETURN
  4619 025360                                   SET     NOSCOPE
  4620 025366  012737  000032  002260           MOV     #32,REALPAT                     ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
  4621 025374  005037  002074                   CLR     NOPAR                           ;SETUP PARITY ACTION
  4622 025400                                   MAP     BANK                            ;MAP FIRST SO THAT THE BLOCK MOVE WORKS
  4623 025414  012700  010247                   MOV     #10247,R0                       ;OP CODE OF INSTRUCTION "MOV    R2,-(PC)"
  4624 025420  012701  177667                   MOV     #177667,R1                      ;OP CODE OF COMPLEMENT OF INSTRUCTION "JMP (R0)"
  4625 025424  012702  020000                   MOV     #SIZE/2,R2                      ;USED FOR 1/2 BANK LOOP
  4626 025430  010237  002472                   MOV     R2,LINK1
  4627 025434  012703  060000                   MOV     #FIRST,R3
  4628 025440  012704  160000                   MOV     #LAST+2,R4
  4629 025444  005037  002474                   CLR     LINK2
  4630 025450  005737  002426                   TST     NOSUPER
  4631 025454  001005                           BNE     1$
  4632 025456  023737  172252  172254           CMP     SIPAR5,SIPAR6
  4633 025464  001405                           BEQ     2$
  4634 025466  000415                           BR      3$
  4635 025470  023737  177652  177654  1$:      CMP     UIPAR5,UIPAR6
  4636 025476  001011                           BNE     3$
  4637 025500  012704  140000          2$:      MOV     #140000,R4
  4638 025504  012702  014000                   MOV     #14000,R2
  4639 025510  010237  002472                   MOV     R2,LINK1
  4640 025514  012737  000001  002474           MOV     #1,LINK2
  4641
  4642 025522                          3$:      TESTAREA                                ;ENTER TEST MODE
  4643                                          ;MOVE TEST TO MEMORY UNDER TEST
  4644 025530  010023                  4$:      MOV     R0,(R3)+
  4645 025532  010144                           MOV     R1,-(R4)
  4646 025534  077203                           SOB     R2,4$
  4647
  4648 025536  022737  000001  003710           CMP     #1,PROTYP
  4649 025544  001003                           BNE     5$
  4650                                          ;MOVE LAST PART OF TEST TO FASTCITY
  4651 025546                                   BMOV    MTP032
  4652 025554  104417                  5$:      KERNEL                                  ;ENTER KERNEL MODE
  4653
  4654 025556  012702  005141                   MOV     #5141,R2                        ;OP CODE OF INSTRUCTION "COM    -(R1)"
  4655 025562  012700  025700                   MOV     #10$,R0                         ;ADDRESS TO RETURN TO IN R0
  4656 025566  012701  160000                   MOV     #LAST+2,R1                      ;TOP OF BANK
  4657 025572  012737  060000  002254           MOV     #FIRST,SUPDOADD
  4658 025600  005737  002474                   TST     LINK2
  4659 025604  001402                           BEQ     6$
  4660 025606  012701  140000                   MOV     #140000,R1
  4661 025612  004737  026616          6$:      CALL    SUPDO4
  4662 025616  012703  020000                   MOV     #SIZE/2,R3
  4663 025622  012705  000110                   MOV     #110,R5
  4664 025626  012704  060000                   MOV     #FIRST,R4
  4665 025632  005737  002474                   TST     LINK2
  4666 025636  001402                           BEQ     7$
  4667 025640  012703  014000                   MOV     #14000,R3
  4668 025644  022737  000001  003710  7$:      CMP     #1,PROTYP
  4669 025652  001406                           BEQ     8$
```

```
 4670 025654  012737  036012  002254           MOV     #MTP032,SUPDOADD
 4671 025662  004737  026616                   CALL    SUPD04
 4672 025666  000402                           BR      9$
 4673 025670  004737  026440           8$:     CALL    SUPD02
 4674 025674  005037  002410           9$:     CLR     NOSCOPE
 4675 025700  000207                   10$:    RETURN                           ;THIS RETURN ACTS AS A NORMAL RETURN FROM MT0032
 4676                                                                           ;ALSO A RETURN FROM THE ''CALL    SUPD04'' ABOVE
 4677
```

```
  4680 025702                                   MT0033: SUBTST  <<MT0033        SETUP BRANCH GOBBLE TEST>>
                                                ;*****************************************************************************
                                                ;*SUBTEST        MT0033  SETUP BRANCH GOBBLE TEST
                                                ;*****************************************************************************
  4681 025702  004737  026370                           CALL    KAMITEST                        ;CHECK FOR KAMIKAZE MODE
  4682 025706                                           ON.ERROR THEN $RETURN                   ;IF NOT IN KAMIKAZE MODE RETURN
  4683 025712                                           SET     NOSCOPE
  4684 025720  012737  000033  002260                   MOV     #33,REALPAT                     ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
  4685 025726  005037  002074                           CLR     NOPAR                           ;SETUP PARITY ACTION
  4686 025732                                           MAP     BANK                            ;MAP FIRST SO THAT BLOCK MOVE WORKS
  4687
  4688 025746                                           TESTAREA                        ;ENTER TEST MODE
  4689 025754                                           BMOV    MTP033,FIRST,GBLENGTH/2
  4690 025766  104417                                   KERNEL                          ;ENTER KERNEL MODE
  4691
  4692 025770  012705  060076                           MOV     #FIRST+GBLENGTH,R5
  4693 025774  012737  060004  002254                   MOV     #FIRST+4,SUPDOADD
  4694 026002  012701  060002                           MOV     #FIRST+2,R1
  4695 026006  012702  060003                           MOV     #FIRST+3,R2
  4696 026012  012737  160000  002472                   MOV     #LAST+2,LINK1
  4697 026020  005737  002426                           TST     NOSUPER
  4698 026024  001005                                   BNE     1$
  4699 026026  023737  172252  172254                   CMP     SIPAR5,SIPAR6
  4700 026034  001405                                   BEQ     2$
  4701 026036  000407                                   BR      3$
  4702 026040  023737  177652  177654  1$:              CMP     UIPAR5,UIPAR6
  4703 026046  001003                                   BNE     3$
  4704 026050  012737  140000  002472  2$:              MOV     #140000,LINK1
  4705
  4706 026056  004737  026616          3$:              CALL    SUPDO4
  4707 026062  005037  002410                           CLR     NOSCOPE
  4708 026066  000207                                   RETURN
  4709
  4710 026070                                   MT0034: SUBTST  <<MT0034        SOFT ERROR - BACKGROUND PATTERN TEST>>
                                                ;*****************************************************************************
                                                ;*SUBTEST        MT0034  SOFT ERROR - BACKGROUND PATTERN TEST
                                                ;*****************************************************************************
  4711 026070  012737  000034  002260                   MOV     #34,REALPAT
  4712 026076  012700  060000                           MOV     #FIRST,R0
  4713 026102  012701  040000                           MOV     #SIZE,R1
  4714 026106  013702  002560                           MOV     SOFTPAT,R2
  4715 026112  010103                                   MOV     R1,R3
  4716 026114  013705  002102                           MOV     BANKINDEX,R5
  4717 026120  010004                                   MOV     R0,R4
  4718 026122  022737  000001  003710                   CMP     #1,PROTYP                       ;IS THIS AN 11/44?
  4719 026130  001006                                   BNE     1$                              ;BRANCH IF NOT
  4720 026132                                           BMOV    MTP034
  4721 026140  012737  177646  002254                   MOV     #UIPAR3,SUPDOADD
  4722 026146                          1$:              IF #BIT13 SET.IN CONFIG+2(R5)
  4723                                                     ;BACKGROUND PATTERN IS VALID
  4724 026156  022737  000001  003710                      CMP    #1,PROTYP
  4725 026164  001403                                      BEQ    2$
  4726 026166  012737  036150  002254                      MOV    #MTP034+6,SUPDOADD
  4727 026174  004737  026602          2$:                 CALL   SUPDO3                   ;READ IT
  4728 026200                                           ELSE
  4729                                                     ;BACKGROUND PATTERN HAS BEEN INVALIDATED
  4730 026202  022737  000001  003710                      CMP    #1,PROTYP
```

```
026210  001406                                   BEQ    3$
026212  012737  036142  002254                   MOV    #MTP034,SUPDOADD
026220  004737  026602                           CALL   SUPDO3
026224  000402                                   BR     4$
026226  004737  026424                   3$:     CALL   SUPDO1                     ;WRITE IT
026232  052765  020000  002626           4$:     BIS    #BIT13,CONFIG+2(R5)        ;VALIDATE IT
026240                                          END ;OF IF #BIT13
026240  000207                                  RETURN

026242                                  MT0035: SUBTST  <<MT0035        SETUP WORST CASE NOISE PARITY TEST>>
                                        ;*************************************************************************
                                        ;*SUBTEST        MT0035  SETUP WORST CASE NOISE PARITY TEST
                                        ;*************************************************************************
026242  012737  000035  002260                  MOV     #35,REALPAT                ;SET UP TEST NUMBER FOR DISPLAY
026250  013703  002102                          MOV     BANKINDEX,R3
026254  016301  002624                          MOV     CONFIG(R3),R1
026260  000301                                  SWAB    R1
026262  042701  177760                          BIC     #^C17,R1
026266  006301                                  ASL     R1
026270  010137  002146                          MOV     R1,CSRNO
026274  023737  002146  002502                  CMP     CSRNO,PGMCSR
026302  001001                                  BNE     1$
026304  000207                                  RETURN
026306  012702  052524                  1$:     MOV     #52524,R2
026312  004737  036470                          CALL    BACKGND                    ;WRITE BACKROUND OF ALMOST ALT. 1'S AND 0'S
026316  012737  036166  002254                  MOV     #MTP035,SUPDOADD
026324  004737  026602                          CALL    SUPDO3
026330                                          IF QVFLAG IS TRUE THEN $RETURN
026340  005102                                  COM     R2
026342  004737  036470                          CALL    BACKGND                    ;WRITE COMPLEMENT PATTERN INTO MUT
026346  004737  026616                          CALL    SUPDO4
026352  000207                                  RETURN
```

```
   4762 026354                         MT0999: SUBTST  <<MT0999        SETUP NULL TEST>>
                                       ;*******************************************************************************
                                       ;*SUBTEST        MT0999  SETUP NULL TEST
                                       ;*******************************************************************************
   4763 026354  005037  002260                   CLR     REALPAT
   4764 026360                                    SET     NULLFLAG
   4765 026366  000207                            RETURN
   4766
   4767 026370                         KAMITEST:SUBTST <<CHECK FOR KAMIKAZE MODE>>
                                       ;*******************************************************************************
                                       ;*SUBTEST        CHECK FOR KAMIKAZE MODE
                                       ;*******************************************************************************
   4768 026370                                    IF KAMIKAZE IS TRUE OR ACTFLAG IS TRUE OR APTFLAG IS TRUE
   4769 026412                                      $RETURN NOERROR                 ;RUN THE TEST
   4770 026416                                    ELSE
   4771 026420                                      $RETURN ERROR                   ;DON'T RUN THE TEST
   4772 026424                                    END ;OF IF KAMIKAZE
```

```
  4775 026424                               SUPD01: SUBTST  <<SUBR  EXECUTE PATTERN IN SUPERVISOR>>
                                            ;*******************************************************************************
                                            ;*SUBTEST        SUBR    EXECUTE PATTERN IN SUPERVISOR
                                            ;*******************************************************************************
  4776 026424                                       MAP     BANK                                ;MAP SUPERVISOR SPACE (TEST AREA) TO BANK
  4777 026440  004737  055570               SUPD02: CALL    GETDIS
  4778 026444                                       PUSH    $LPERR,$LPADR
  4779 026454  010037  002152                       MOV     R0,SUPDR0
  4780 026460  012700  002154                       MOV     #SUPDR1,R0
  4781 026464  010120                               MOV     R1,(R0)+
  4782 026466  010220                               MOV     R2,(R0)+
  4783 026470  010320                               MOV     R3,(R0)+
  4784 026472  010420                               MOV     R4,(R0)+
  4785 026474  010520                               MOV     R5,(R0)+
  4786 026476  010620                               MOV     SP,(R0)+
  4787 026500  013700  002152                       MOV     SUPDR0,R0
  4788 026504  012737  026520  002562               MOV     #TAG4$,$LPADR
  4789 026512  013737  002562  002564               MOV     $LPADR,$LPERR
  4790 026520  012700  002170               TAG4$:  MOV     #SUPDR6+2,R0
  4791 026524  014006                               MOV     -(R0),SP
  4792 026526  014005                               MOV     -(R0),R5
  4793 026530  014004                               MOV     -(R0),R4
  4794 026532  014003                               MOV     -(R0),R3
  4795 026534  014002                               MOV     -(R0),R2
  4796 026536  014001                               MOV     -(R0),R1
  4797 026540  014000                               MOV     -(R0),R0
  4798 026542                                       SUPERVISOR                  ;ENTER SUPERVISOR MODE
  4799 026550  012706  000740                       MOV     #SUPSTK,SSP
  4800 026554  104424                               CACHOFF                     ;TURN CACHE OFF
  4801 026556  004737  177640                       CALL    FASTCITY                    ;CALL TO THE USER INSTRUCTION PAR'S
  4802 026562  104423                               CACHON                      ;TURN CACHE ON
  4803 026564  104417                               KERNEL                      ;ENTER KERNEL MODE
  4804 026566  000004                               SCOPE
  4805 026570                                       POP     $LPADR,$LPERR
  4806 026600  000207                               RETURN
```

```
026602                           SUPD03: MAP     BANK                    ;MAP SUPERVISOR SPACE (TEST AREA) TO BANK
026616  004737  055570           SUPD04: CALL    GETDIS
026622                                   PUSH    $LPERR,$LPADR
026632  010037  002152                   MOV     R0,SUPDR0
026636  012700  002154                   MOV     #SUPDR1,R0
026642  010120                           MOV     R1,(R0)+
026644  010220                           MOV     R2,(R0)+
026646  010320                           MOV     R3,(R0)+
026650  010420                           MOV     R4,(R0)+
026652  010520                           MOV     R5,(R0)+
026654  010620                           MOV     SP,(R0)+
026656  013700  002152                   MOV     SUPDR0,R0
026662  012737  026676  002562           MOV     #TBG4$,$LPADR
026670  013737  002562  002564           MOV     $LPADR,$LPERR
026676  012700  002170           TBG4$:  MOV     #SUPDR6+2,R0
026702  014006                           MOV     -(R0),SP
026704  014005                           MOV     -(R0),R5
026706  014004                           MOV     -(R0),R4
026710  014003                           MOV     -(R0),R3
026712  014002                           MOV     -(R0),R2
026714  014001                           MOV     -(R0),R1
026716  014000                           MOV     -(R0),R0
026720                                   TESTAREA                        ;ENTER SUPERVISOR MODE
026726  005737  002426                   TST     NOSUPER
026732  001403                           BEQ     1$
026734  012706  000700                   MOV     #USESTK,USP
026740  000402                           BR      2$
026742  012706  000740           1$:     MOV     #SUPSTK,SSP
026746  104424                   2$:     CACHOFF                         ;TURN CACHE OFF
026750  004777  153300                   CALL    @SUPDOADD
026754  104423                           CACHON                          ;TURN CACHE ON
026756  104417                           KERNEL                          ;ENTER KERNEL MODE
026760  000004                           SCOPE
026762                                   POP     $LPADR,$LPERR
026772  000207                           RETURN
```

```
  4846                                           .SBTTL  MEMORY TEST PATTERN ROUTINES
  4847                                   ;*****************************************************************************
  4848                                   ; PATTERN REGISTER CONVENTIONS
  4849                                   ;       R0      FIRST ADDRESS OF PATTERN (FIRST,LAST+2,ETC)
  4850                                   ;       R1      NUMBER OF ADDRESSES IN PATTERN (SIZE)
  4851                                   ;       R2      DATA FOR PATTERN (ONES,52525,ETC)
  4852                                   ;       R3      COPY OF R1 (IF NECESSARY)
  4853                                   ;       R4      COPY OF R0 (IF NECESSARY)
  4854                                   ;       R5      COPY OF R2 (IF NECESSARY)
  4855                                   ;*****************************************************************************
  4856 026774                            MTP000: SUBTST  <<MTP000        BASIC DATA TEST>>
                                         ;*****************************************************************************
                                         ;*SUBTEST        MTP000  BASIC DATA TEST
                                         ;*****************************************************************************
  4857 026774  010220                    1$:     MOV     R2,(R0)+        ;V177640
  4858 026776  077102                            SOB     R1,MTP000       ;V177642
  4859 027000  000240                            NOP                     ;V177644
  4860 027002  012401                    2$:     MOV     (R4)+,R1        ;V177646
  4861 027004  020102                            CMP     R1,R2           ;V177650
  4862 027006  001402                            BEQ     3$              ;V177652
  4863 027010  104430                            PERR02                  ;V177654
  4864 027012  000240                            NOP                     ;V177656
  4865 027014  077306                    3$:     SOB     R3,2$           ;V177660
  4866 027016  000207                            RETURN                  ;V177662
  4867 027020                            MTP001: SUBTST  <<MTP001        ADDRESS TEST>>
                                         ;*****************************************************************************
                                         ;*SUBTEST        MTP001  ADDRESS TEST
                                         ;*****************************************************************************
  4868 027020  010220                    3$:     MOV     R2,(R0)+        ;V177640
  4869 027022  062702  000002                    ADD     #2,R2           ;V177642
  4870 027026  077104                            SOB     R1,3$           ;V177646
  4871 027030  000240                            NOP                     ;V177650
  4872 027032  012400                    1$:     MOV     (R4)+,R0        ;V177652
  4873 027034  020005                            CMP     R0,R5           ;V177654
  4874 027036  001401                            BEQ     2$              ;V177656
  4875 027040  104427                            PERR01                  ;V177660
  4876 027042  062705  000002            2$:     ADD     #2,R5           ;V177662
  4877 027046  077307                            SOB     R3,1$           ;V177666
  4878 027050  000207                            RETURN                  ;V177672
  4879 027052                            MTP002: SUBTST  <<MTP002        COMPLEMENT ADDRESS TEST (WRITE DOWN, READ UP)>>
                                         ;*****************************************************************************
                                         ;*SUBTEST        MTP002  COMPLEMENT ADDRESS TEST (WRITE DOWN, READ UP)
                                         ;*****************************************************************************
  4880 027052  010540                    3$:     MOV     R5,-(R0)        ;V177640
  4881 027054  062705  000002                    ADD     #2,R5           ;V177642
  4882 027060  077104                            SOB     R1,3$           ;V177646
  4883 027062  000240                            NOP                     ;V177650
  4884 027064  162702  000002            1$:     SUB     #2,R2           ;V177652
  4885 027070  012401                            MOV     (R4)+,R1        ;V177656
  4886 027072  020102                            CMP     R1,R2           ;V177660
  4887 027074  001401                            BEQ     2$              ;V177662
  4888 027076  104430                            PERR02                  ;V177664
  4889 027100  077307                    2$:     SOB     R3,1$           ;V177666
  4890 027102  000207                            RETURN                  ;V177670
```

```
027104                        MTPA03: SUBTST  <<MTPA03        3 XOR 9 WORST CASE NOISE TEST (WRITE)>>
                              ;*************************************************************************
                              ;*SUBTEST        MTPA03  3 XOR 9 WORST CASE NOISE TEST (WRITE)
                              ;*************************************************************************
                                      ;R1 = ADDRESS
                                      ;R2 = SMALL LOOP CONSTANT
                                      ;R3 = NUM OF ADD TO TEST (LARGE LOOP)
                                      ;R4 = GOOD DATA
                                      ;R5 = MEDIUM LOOP CONSTANT
                                      .ENABL  LSB
027104  010421                1$:     MOV     R4,(R1)+        ;V177640
027106  010421                        MOV     R4,(R1)+        ;V177642
027110  077203                        SOB     R2,1$           ;V177644
027112  005104                        COM     R4              ;V177646
027114  052704                        BIS     (PC)+,R4        ;V177650
027116  000401                WARN2:  401                     ;V177652 WARNING LOCATION IS MODIFIED BEFORE LOADING
027120  012702  000004                MOV     #4,R2           ;V177654
027124  077511                        SOB     R5,1$           ;V177660
027126  005104                        COM     R4              ;V177662
027130  052704                        BIS     (PC)+,R4        ;V177664
027132  000401                WARN3:  401                     ;V177666 WARNING LOCATION IS MODIFIED BEFORE LOADING
027134  012705  000100                MOV     #64.,R5         ;V177670
027140  077317                        SOB     R3,1$           ;V177674
027142  000207                        RETURN                  ;V177676
                                      .DSABL  LSB

027144                        MTPB03: SUBTST  <<MTPB03        3 XOR 9 WORST CASE NOISE TEST (READ)>>
                              ;*************************************************************************
                              ;*SUBTEST        MTPB03  3 XOR 9 WORST CASE NOISE TEST (READ)
                              ;*************************************************************************
                                      .ENABL  LSB
027144  000137  027204        1$:     JMP     @#MTPC03        ;V177640        GO TO V172360
027150  077203                        SOB     R2,1$           ;V177644
027152  005104                        COM     R4              ;V177646
027154  052704                        BIS     (PC)+,R4        ;V177650
027156  000401                WARN4:  401                     ;V177652 WARNING LOCATION IS MODIFIED BEFORE LOADING
027160  012702  000004                MOV     #4,R2           ;V177654
027164  077511                        SOB     R5,1$           ;V177660
027166  005104                        COM     R4              ;V177662
027170  052704                        BIS     (PC)+,R4        ;V177664
027172  000401                WARN5:  401                     ;V177666 WARNING LOCATION IS MODIFIED BEFORE LOADING
027174  012705  000100                MOV     #64.,R5         ;V177670
027200  077317                        SOB     R3,1$           ;V177674
027202  000207                        RETURN                  ;V177676
                                      .DSABL  LSB
```

```
 4934 027204                          MTPC03: SUBTST  <<MTPC03        TEST DATA SUBPROGRAM>>
                                      ;*****************************************************************************
                                      ;*SUBTEST        MTPC03  TEST DATA SUBPROGRAM
                                      ;*****************************************************************************
 4935 027204  020421                          CMP     R4,(R1)+        ;V172360
 4936 027206  001401                          BEQ     1$              ;V172362
 4937 027210  104431                          PERR03                  ;V172364
 4938 027212  005141                  1$:     COM     -(R1)           ;V172366
 4939 027214  005111                          COM     (R1)            ;V172370
 4940 027216  000137  027222                  JMP     @#MTPD03        ;V172372        GO TO V172260
 4941
 4942 027222                          MTPD03: SUBTST  <<MTPD03        TEST DATA SUBSUBPROGRAM>>
                                      ;*****************************************************************************
                                      ;*SUBTEST        MTPD03  TEST DATA SUBSUBPROGRAM
                                      ;*****************************************************************************
 4943 027222  020421                          CMP     R4,(R1)+        ;V172260
 4944 027224  001401                          BEQ     1$              ;V172262
 4945 027226  104431                          PERR03                  ;V172264
 4946 027230  005127                  1$:     COM     (PC)+           ;V172266
 4947 027232  000000                          0                       ;V172270
 4948 027234  001363                          BNE     MTPC03          ;V172272        GO TO V172360
 4949 027236  000137  027150                  JMP     @#MTPB03+4      ;V172274        GO TO V177644
```

```
027242                          MTPA04: SUBTST  <<MTPA04        ROTATING ZEROS TEST>>
                                ;*****************************************************************************
                                ;*SUBTEST        MTPA04  ROTATING ZEROS TEST
                                ;*****************************************************************************
027242  012705  000010          1$:     MOV     #8.,R5          ;V177640
027246  010504                          MOV     R5,R4           ;V177644
027250  000241                          CLC                     ;V177646
027252  000137  027276                  JMP     @#MTPB04        ;V177650
027256  016004  177776                  MOV     -2(R0),R4       ;V177654
027262  103402                          BCS     2$              ;V177660
027264  020204                          CMP     R2,R4           ;V177662
027266  001401                          BEQ     3$              ;V177664
027270  104432                  2$:     PERR04                  ;V177666
027272  077115                  3$:     SOB     R1,1$           ;V177670
027274  000207                          RETURN                  ;V177672

027276                          MTPB04: SUBTST  <<MTPB04        SUBR    ROTATING BIT>>
                                ;*****************************************************************************
                                ;*SUBTEST        MTPB04  SUBR    ROTATING BIT
                                ;*****************************************************************************
027276  106110                  1$:     ROLB    (R0)            ;V172360
027300  077502                          SOB     R5,1$           ;V172362
027302  106120                          ROLB    (R0)+           ;V172364
027304  106110                  2$:     ROLB    (R0)            ;V172366
027306  077402                          SOB     R4,2$           ;V172370
027310  106120                          ROLB    (R0)+           ;V172372
027312  000137  027256                  JMP     @#MTPA04+14     ;V172374

027316                          MTP005: SUBTST  <<MTP005        ROTATION ONES TEST>>
                                ;*****************************************************************************
                                ;*SUBTEST        MTP005  ROTATION ONES TEST
                                ;*****************************************************************************
027316  012705  000010          1$:     MOV     #8.,R5          ;V177640
027322  010504                          MOV     R5,R4           ;V177644
027324  000261                          SEC                     ;V177646
027326  000137  027276                  JMP     @#MTPB04        ;V177650
027332  016004  177776                  MOV     -2(R0),R4       ;V177654
027336  103002                          BCC     2$              ;V177660 IF THIS HAPPENS THE GOOD & BAD MATCH
027340  020204                          CMP     R2,R4           ;V177662
027342  001401                          BEQ     3$              ;V177664
027344  104432                  2$:     PERR04                  ;V177666
027346  077115                  3$:     SOB     R1,1$           ;V177670
027350  000207                          RETURN                  ;V177672
```

```
027352                                  MTP006: SUBTST  <<MTP006         INITIAL DATA TEST>>
                                        ;*************************************************************************
                                        ;*SUBTEST        MTP006  INITIAL DATA TEST
                                        ;*************************************************************************
                                                        ;             THIS TEST CHECKS THE DI/DO LINES BY
                                                        ;             SHIFTING A 1 THROUGH THE WORD.
027352  012737  000001  002234                  MOV     #1,DATBUF       ;SET THE FIRST TEST BIT
027360  005037  002236                          CLR     DATBUF+2        ;CLEAR 2ND WORD
027364  013771  002234  000000  1$:             MOV     DATBUF,@(R1)    ;WRITE TEST WORD 1
027372  013771  002236  000002                  MOV     DATBUF+2,@2(R1) ;AND TEST WORD 2
027400  017102  000000                          MOV     @(R1),R2
027404  023702  002234                          CMP     DATBUF,R2       ;NOW READ THEM
027410  001401                                  BEQ     2$              ;BR IF FIRST 16 OK
027412  104433                                  PERR07                  ;ERROR TRAP

027414  017102  000002          2$:             MOV     @2(R1),R2
027420  023702  002236                          CMP     DATBUF+2,R2     ;NOW READ SECOND WORD
027424  001401                                  BEQ     3$              ;BR IF OK
027426  104434                                  PERR10                  ;ERROR TRAP

027430  005737  002236          3$:             TST     DATBUF+2        ;HAS LAST BIT BEEN TESTED ?
027434  100405                                  BMI     4$              ;MINUS MEANS BIT 31
027436                                          DLEFT   DATBUF          ;NO, SHIFT TEST BIT LEFT
027446  000746                                  BR      1$              ;GO WRITE NEW TEST DATA
                                                                        ;NOW GOING TO SHIFT A 0 IN DATA DIRECTION
027450  012737  177776  002234  4$:             MOV     #177776,DATBUF  ;PUT A 0 IN BIT 0
027456  012737  177777  002236                  MOV     #-1, DATBUF+2   ;AND 1'S IN ALL OTHERS
027464  013771  002234  000000  5$:             MOV     DATBUF,@(R1)    ;WRITE THE DATA
027472  013771  002236  000002                  MOV     DATBUF+2,@2(R1) ;2 WORDS WORTH
027500  017102  000000                          MOV     @(R1),R2
027504  023702  002234                          CMP     DATBUF,R2       ;NOW READ FIRST WORD
027510  001401                                  BEQ     6$              ;BR IF OK
027512  104433                                  PERR07

027514  017102  000002          6$:             MOV     @2(R1),R2
027520  023702  002236                          CMP     DATBUF+2,R2     ;NOW, READ SECOND WORD
027524  001401                                  BEQ     7$              ;BR IF OK
027526  104434                                  PERR10

027530  005737  002236          7$:             TST     DATBUF+2        ;TESTED BIT 31 YET?
027534  100005                                  BPL     8$              ;BR IF YES, WE'RE DONE
027536                                          DLEFT   DATBUF
027546  000746                                  BR      5$              ;KEEP GOING
027550  000207                  8$:             RETURN
```

```
 5031 027552                            MTP007: SUBTST  <<MTP007        ADDRESS BIT TEST>>
                                        ;*************************************************************************
                                        ;*SUBTEST       MTP007  ADDRESS BIT TEST
                                        ;*************************************************************************
 5032                                                   ;               THIS TEST CHECKS TO SEE THAT EACH ADDRESS
 5033                                                   ;               BIT IN EACH 16K BANK CAN BE ASSERTED UNIQUELY.
 5034                                                   ;               IT CHECKS FOR ADDRESS BITS THAT MAY BE STUCK
 5035                                                   ;               HIGH, STUCK LOW OR STUCK TOGETHER.
 5036 027552  111100                            MOVB    (R1),R0
 5037 027554  105700                            TSTB    R0                      ;READ AND COMPARE FOR ZEROS
 5038 027556  001401                            BEQ     1$                      ;BR IF OK
 5039 027560  104435                            PERR11
 5040
 5041 027562  105111                    1$:     COMB    (R1)                    ;COMPLEMENT THE BYTE
 5042 027564  111100                            MOVB    (R1),R0
 5043 027566  105700                            TSTB    R0                      ;READ FOR NON ZEROS
 5044 027570  001001                            BNE     2$                      ;BR IF OK
 5045 027572  104436                            PERR12
 5046
 5047 027574  040201                    2$:     BIC     R2,R1                   ;MASK OFF THE ASSERTED BIT
 5048 027576  006302                            ASL     R2                      ;SHIFT R2 FOR NEXT BIT
 5049 027600  050201                            BIS     R2,R1                   ;SET THE NEW BIT INTO R1
 5050 027602  011100                            MOV     (R1),R0
 5051 027604  005700                            TST     R0                      ;READ THE NEW ADDRESS
 5052 027606  001401                            BEQ     3$                      ;READ FOR ZEROS
 5053 027610  104437                            PERR13
 5054
 5055 027612  005111                    3$:     COM     (R1)                    ;COMPL THE WORD
 5056 027614  011100                            MOV     (R1),R0
 5057 027616  005700                            TST     R0                      ;READ IT AGAIN
 5058 027620  001001                            BNE     4$
 5059 027622  104440                            PERR14
 5060
 5061 027624  022702  100000            4$:     CMP     #100000,R2
 5062 027630  001407                            BEQ     5$
 5063 027632  022702  010000                    CMP     #10000,R2               ;CHECK FOR MSB IN 4K BANK
 5064 027636  001356                            BNE     2$                      ;NOT LAST BIT, BRANCH
 5065 027640  006302                            ASL     R2
 5066 027642  012701  160000                    MOV     #160000,R1
 5067 027646  000752                            BR      2$
 5068 027650  000207                    5$:     RETURN
```

```
027652                              MTP010: SUBTST  <<MTP010        BYTE ADDRESSING TEST>>
                                    ;*****************************************************************************
                                    ;*SUBTEST        MTP010  BYTE ADDRESSING TEST
                                    ;*****************************************************************************
                                                    ;TEST 3 THIS TEST CHECKS FOR PROPER
                                                    ;          BYTE ADDRESSING WITH ECC DISABLED
027652  010402                              MOV     R4,R2           ;R4 HAS LOWEST ADDRESS
027654  010403                              MOV     R4,R3           ;PUT IT IN R3 ALSO
027656  062702  000004                      ADD     #4,R2           ;POINT R2 TO LAST BYTE +1
027662  012713  177777                      MOV     #-1,(R3)        ;WRITE ALL ONES IN
027666  012763  177777  000002              MOV     #-1,2(R3)       ;THE 4 TEST BYTES
027674  105013                      1$:     CLRB    (R3)            ;CLEAR A BYTE
027676  010401                              MOV     R4,R1           ;INITIALIZE R1 FOR EACH PASS
027700  020201                      2$:     CMP     R2,R1           ;IF EQUAL, JUST READ LAST BYTE
027702  001420                              BEQ     6$              ;BR IF EQUAL
027704  020301                              CMP     R3,R1           ;IS THIS THE BYTE OF ZEROS
027706  001007                              BNE     4$              ;BR IF NOT
027710  111100                              MOVB    (R1),R0
                                            ;WARNING IF YOU OPTOMIZE CHANGE THE PCBUMP FOR THIS ERROR INCASE OF TRAPS
027712  022700  000000                      CMP     #0,R0           ;IT IS, COMPARE FOR ZEROS
027716  001401                              BEQ     3$
027720  104435                              PERR11

027722  005201                      3$:     INC     R1              ;NEXT BYTE
027724  000765                              BR      2$              ;RETURN
027726  111100                      4$:     MOVB    (R1),R0
027730  122700  177777                      CMPB    #-1,R0          ;ITS NOT THE BYTE OF 0'S, READ 1'S
027734  001401                              BEQ     5$
027736  104436                              PERR12

027740  005201                      5$:     INC     R1              ;MOVE TO NEXT BYTE
027742  000756                              BR      2$
027744  112713  177777              6$:     MOVB    #-1,(R3)        ;RESTORE 1'S TO BYTE JUST TESTED
027750  005203                              INC     R3              ;INC TO NEXT BYTE
027752  020302                              CMP     R3,R2           ;WAS THAT JUST THE LAST ONE?
027754  001347                              BNE     1$              ;BR IF NO
027756  000207                              RETURN
```

```
027760                                 MTP011: SUBTST  <<MTP011        SINGLE BIT ERROR TEST>>
                                       ;*******************************************************************************
                                       ;*SUBTEST       MTP011  SINGLE BIT ERROR TEST
                                       ;*******************************************************************************
                                                 ;(1)      CREATE A SINGLE BIT ERROR
                                                 ;
                                                 ;(2)      READ BACK SBE UNCORRECTED (WITH ECC DISABLE)
                                                 ;
                                                 ;(3)      ENABLE ECC & READ CORRECTED DATA
                                                 ;
                                                 ;(4)      CHECK THAT THE SBE FLAG WAS SET FROM THE LAST READ
                                                 ;
                                                 ;(5)      DO (1-4) FOR DATA CONSISTING OF 1 BIT SET IN EACH OF 32
                                                 ;         POSITIONS OF A DOUBLE WORD
                                                 ;         THEN DO IT AGAIN FOR 1 BIT CLEARED IN EACH OF 32 POSITIONS OF
                                                 ;         A DOUBLE WORD
                                                 ;         IE (64 TIMES)
                                                 ;
                                                 ;(6)      DO (1-5) FOR A SBE IN EACH OF 32 BIT POSITIONS
                                                 ;         IE (RUN TEST 64 * 32 = 2048 TIMES)
027760    104503                                 CLR1CSR                       ;CLEAR 1 SELECTED CSR
027762    005737    013076                       TST       PHEBE               ;TEST SPECIAL CASE INDICATOR
027766    001407                                 BEQ       MTLA11              ;BRANCH IF NOT SET
027770    013702    172246                       MOV       SIPAR3,R2           ;SAVE CONTENTS OF SIPAR #3
027774    013737    172252    172246             MOV       SIPAR5,@#SIPAR3     ;COPY CONTENTS OF #5 INTO #3
030002    010237    172252                       MOV       R2,@#SIPAR5         ;COPY CONTENTS OF #3 INTO #5
                                                 ;BIG LOOP
030006    012737    000001    002234   MTLA11:   MOV       #1,DATBUF           ;INITIAL DATA
030014    005037    002236                       CLR       DATBUF+2            ;32 BITS WORTH
                                                 ;MEDIUM LOOP
030020    012737    000001    002244   MTLB11:   MOV       #1,SBEMSK           ;INITIAL ERROR MASK
030026    005037    002246                       CLR       SBEMSK+2            ;32 BITS WORTH
                                                 ;LITTLE LOOP
030032    013737    002234    002240   MTLC11:   MOV       DATBUF,TSTDAT       ;
030040    013737    002236    002242             MOV       DATBUF+2,TSTDAT+2;TO SAVE ORIG DATA
030046    105737    002256                       TSTB      PASFLG  ;COMP DATA ON SECOND PASSONLY
030052    001404                                 BEQ       4$                  ;BR IF FIRST PASS
030054    005137    002240                       COM       TSTDAT              ;SECOND PASS, COMP BOTH WORDS
030060    005137    002242                       COM       TSTDAT+2
030064    013702    002240             4$:       MOV       TSTDAT,R2
030070    013703    002242                       MOV       TSTDAT+2,R3
030074    012737    002240    002272             MOV       #TSTDAT,SOURCE      ;SET UP ADDRESS FOR CHKGEN
030102    004737    041662                       CALL      CHKGEN              ;GEN CHECKBITS ON TSTDAT
                                                 ;**************************************
                                                 ;** CREATE A SINGLE BIT ERROR **
                                                 ;**************************************
030106    013701    002244                       MOV       SBEMSK,R1
030112    074137    002240                       XOR       R1,TSTDAT
030116    013701    002246                       MOV       SBEMSK+2,R1
030122    074137    002242                       XOR       R1,TSTDAT+2
030126    013701    002362             MTLD11:   MOV       TESTADD,R1          ;FIRST TEST ADDRESS
030132    013705    002364                       MOV       TESTADD+2,R5        ;SECOND TEST ADDRESS
030136    104471                                 ECC1DIS                       ;DISABLE ECC ON 1 SELECTED CSR
030140    013711    002240                       MOV       TSTDAT,(R1)         ;WRITE FIRST 16 BITS
030144    104475                                 CB1CSR                        ;WRITE GENERATED CHECKBITS IN 1 SELECTED CSR
030146    013715    002242                       MOV       TSTDAT+2,(R5)       ;WRITE SECOND 16 BITS AND
                                                                               ;CHECK BITS. WE NOW HAVE CHECKBITS
```

```
                                                              ;GENEPATED ON DATBUF AND DATA WITH
                                                              ;ONE BIT IN ERROR (AS PER SBEMSK).
030152  104471                          ECC1DIS               ;DISABLE ECC ON 1 SELECTED CSR
030154  011100                          MOV     (R1),R0
030156  020037  002240                  CMP     R0,TSTDAT     ;READ THE LOW WORD (UNCORRECTED)
030162  001403                          BEQ     6$            ;BR IF OK
030164  010137  002032                  MOV     R1,ADDRESS
030170  104455                          PERR31

030172  011500                  6$:     MOV     (R5),R0
030174  020037  002242                  CMP     R0,TSTDAT+2   ;READ THE HIGH WORD (UNCORRECTED)
030200  001403                          BEQ     7$            ;BR IF OK
030202  010537  002032                  MOV     R5,ADDRESS
030206  104455                          PERR31

030210                          7$:     IF KFLAG IS FALSE
030216  104426                            READCSR
030220                                    IF #BIT4 OFF.IN CSR OR #BIT15 OFF.IN CSR
030240  104045                              ERROR        +45
030242                                    END; OF IF #BIT4
030242                                  END; OF IF KFLAG
030242  0C5737  013076                  TST     PHEBE
030246  001001                          BNE     17$
030250  104512                          ERRGEN
030252  104503                  17$:    CLR1CSR               ;CLEAR 1 SELECTED CSR
030254  011100                          MOV     (R1),R0
030256  020002                          CMP     R0,R2         ;SEE IF ITS BEEN CORRECTED
030260  001401                          BEQ     8$            ;IT SHOULD HAVE BEEN
030262  104456                          PERR32

030264  104510                  8$:     TSTREAD               ;TEST LOC (R1) & TST FOR SBE (WITHOUT FETCHES)
030266  103411                          BCS     9$            ;BR IF IT IS SET
030270                                  SET     HEADER        ;ENABLE PRINTING OF ERROR HEADER INFO
030276  010137  002032                  MOV     R1,ADDRESS
030302  104460                          PERR34
030304                                  SET     HEADER        ;ENABLE PRINTING OF ERROR HEADER INFO

030312  104503                  9$:     CLR1CSR               ;CLEAR 1 SELECTED CSR
030314  011500                          MOV     (R5),R0
030316  020003                          CMP     R0,R3         ;SEE IF ITS BEEN CORRECTED
030320  001401                          BEQ     10$           ;BR IF OK
030322  104456                          PERR32

030324  104510                  10$:    TSTREAD               ;TEST LOC (R1) & TST FOR SBE (WITHOUT FETCHES)
030326  103411                          BCS     11$           ;BR IF YES
030330                                  SET     HEADER        ;ENABLE PRINTING OF ERROR HEADER INFO
030336  010137  002032                  MOV     R1,ADDRESS
030342  104460                          PERR34
030344                                  SET     HEADER        ;ENABLE PRINTING OF ERROR HEADER INFO
030352  104512                  11$:    ERRGEN                ;TEST ERROR ADDRESS
030354  105737  002256                  TSTB    PASFLG
030360  100452                          BMI     15$
030362  005737  002246                  TST     SBEMSK+2      ;TEST FOR LAST MASK BIT
030366  100405                          BMI     12$           ;MINUS MEANS BIT 31
030370                                  DLEFT   SBEMSK
030400  000614                          BR      MTLC11
030402                          12$:    IF #SW11 SET.IN @SWR THEN GOTO 13$
```

```
   5218 030412                                          IF QVFLAG IS TRUE THEN GOTO 13$
   5219 030420  005737  002236                          TST     DATBUF+2          ;LAST DATA BIT ?
   5220 030424  100406                                  BMI     13$               ;WHICH IS BIT 31
   5221 030426                                          DLEFT   DATBUF
   5222 030436  000137  030020                          JMP     MTLB11
   5223 030442  105737  002256                 13$:     TSTB    PASFLG  ;FIRST OR SECOND PASS ?
   5224 030446  001004                                  BNE     14$               ;NON ZERO MEANS WE'RE DONE
   5225 030450  105237  002256                          INCB    PASFLG  ;NOT DONE, GO DO SECOND PASS
   5226 030454  000137  030006                          JMP     MTLA11
   5227 030460  052737  000200  002256         14$:     BIS     #BIT7,PASFLG
   5228 030466  005002                                  CLR     R2
   5229 030470  005003                                  CLR     R3
   5230 030472  005037  002240                          CLR     TSTDAT
   5231 030476  005037  002242                          CLR     TSTDAT+2
   5232 030502  012704  000040                          MOV     #40,R4
   5233 030506  012737  003740  002274         15$:     MOV     #3740,CHECK
   5234 030514  074437  002274                          XOR     R4,CHECK
   5235 030520  006304                                  ASL     R4
   5236 030522  032704  020000                          BIT     #BIT13,R4
   5237 030526  001002                                  BNE     16$
   5238 030530  000137  030126                          JMP     MTLD11
   5239                                                 ;CLEAR OUT ANY DBE'S OR SBE'S
   5240 030534  104471                         16$:     ECC1DIS                   ;DISABLE ECC ON 1 SELECTED CSR
   5241 030536  013701  002362                          MOV     TESTADD,R1
   5242 030542  013705  002364                          MOV     TESTADD+2,R5
   5243 030546                                          CLEAR   (R1),(R5)
   5244 030552  104503                                  CLR1CSR                   ;CLEAR 1 SELECTED CSR
   5245 030554  000207                                  RETURN
```

```
030556                                     MTP012: SUBTST  <<MTP012          WRITE BYTE CLEARS SBE TEST>>
                                           ;*************************************************************************
                                           ;*SUBTEST        MTP012  WRITE BYTE CLEARS SBE TEST
                                           ;*************************************************************************
                                                           ;SINGLE BIT ERROR TEST TO INSURE THAT A WRITE
                                                           ;BYTE CLEARS SINGLE BIT ERRORS.
030556  104503                                     CLR1CSR                 ;CLEAR 1 SELECTED CSR
030560  012737  000001  002234                     MOV     #1,DATBUF       ;INITIAL DATA
030566  005037  002236                             CLR     DATBUF+2        ;32 BITS WORTH
030572  012737  000001  002244     1$:             MOV     #1,SBEMSK       ;INITIAL ERROR MASK
030600  005037  002246                             CLR     SBEMSK+2        ;32 BITS WORTH
030604  013737  002234  002240     2$:             MOV     DATBUF,TSTDAT   ;SAVE ORIGINAL DATA
030612  013737  002236  002242                     MOV     DATBUF+2,TSTDAT+2;BOTH WORDS
030620  012737  002240  002272                     MOV     #TSTDAT,SOURCE  ;NEED ADDRESS FOR CHKGEN
030626  004737  041662                             CALL    CHKGEN          ;GENERATE CHECK BITS
030632  013701  002244                             MOV     SBEMSK,R1
030636  074137  002240                             XOR     R1,TSTDAT
030642  013701  002246                             MOV     SBEMSK+2,R1
030646  074137  002242                             XOR     R1,TSTDAT+2
030652  013704  002362                             MOV     TESTADD,R4      ;FIRST TEST ADDRESS
030656  010401                                     MOV     R4,R1           ;PUT IT IN R1 ALSO
030660  104471                                     ECC1DIS                 ;DISABLE ECC ON 1 SELECTED CSR
030662  013711  002240                             MOV     TSTDAT,(R1)     ;WRITE 16 BITS
030666  104475                                     CB1CSR                  ;WRITE GENERATED CHECKBITS IN 1 SELECTED CSR
030670  060501                                     ADD     R5,R1           ;INDEX UP TO SECOND WORD
030672  013711  002242                             MOV     TSTDAT+2,(R1)   ;WRITE HIGH WORD+CHECKBITS
030676  104503                                     CLR1CSR                 ;CLEAR 1 SELECTED CSR
                                                                           ;IT'S DANGEROUS IF WE DON'T
030700  012702  002244                             MOV     #SBEMSK,R2      ;ADDRESS OF ERROR MASK
030704  160501                                     SUB     R5,R1           ;RETURN TO FIRST WORD
030706  112711  177777             3$:             MOVB    #-1,(R1)        ;WRITE A BYTE OF 1'S
030712  005737  002500                             TST     KFLAG           ;IS THIS MF11S-K
030716  001403                                     BEQ     4$              ;BRANCH IF NOT - IT'S MS11-M
030720  132712  177777                             BITB    #-1,(R2)        ;DID THIS BYTE HAVE THE BAD BIT IN IT?
030724  001420                                     BEQ     6$              ;NO - BRANCH
030726  104510                     4$:             TSTREAD                 ;TEST LOC (R1) & TST FOR SBE (WITHOUT FETCHES)
030730  103011                                     BCC     5$              ;NO - SKIP
030732                                             SET     HEADER          ;ENABLE PRINTING OF ERROR HEADER INFO
030740  010137  002032                             MOV     R1,ADDRESS
030744  104017                                     ERROR   +17
030746                                             SET     HEADER          ;ENABLE PRINTING OF ERROR HEADER INFO

030754  111100                     5$:             MOVB    (R1),R0
030756  122700  177777                             CMPB    #-1,R0          ;CHECK DATA
030762  001414                                     BEQ     7$              ;BR IF OK
030764  104457                                     PERR33

030766  104510                     6$:             TSTREAD                 ;TEST LOC (R1) & TST FOR SBE (WITHOUT FETCHES)
                                                                           ;READ THE BYTE
                                                                           ;SBE ERROR BIT ONLY SET ?
030770  103771                                     BCS     5$              ;SHOULD BE SET, BR IF OK
030772                                             SET     HEADER          ;ENABLE PRINTING OF ERROR HEADER INFO
031000  010137  002032                             MOV     R1,ADDRESS
031004  104460                                     PERR34
031006                                             SET     HEADER          ;ENABLE PRINTING OF ERROR HEADER INFO

031014  132712  177777             7$:             BITB    #-1,(R2)        ;CHECK FOR LAST BYTE
```

```
031020  001012                       BNE     8$               ;
031022  005202                       INC     R2
031024  005201                       INC     R1               ;MOVE TO NEXT BYTE
031026  013704  002362               MOV     TESTADD,R4       ;FIRST TEST ADDRESS
031032  032701  000002               BIT     #2,R1            ;TEST FOR LOWER WORD
031036  001723                       BEQ     3$               ;BR IF IT'S LOW 16 BITS
031040  062704  000002               ADD     #2,R4            ;ADJUST POINTER FOR ERROR REPT.
031044  000720                       BR      3$
031046  005737  002246       8$:     TST     SBEMSK+2         ;LAST ERROR BIT ?
031052  100405                       BMI     9$               ;MINUS MEANS BIT 31
031054                               DLEFT   SBEMSK
031064  000647                       BR      2$
031066                       9$:     IF #SW11 SET.IN @SWR THEN GOTO 10$
031076                               IF QVFLAG IS TRUE THEN GOTO 10$
031104  005737  002236               TST     DATBUF+2         ;LAST DATA BIT?
031110  100405                       BMI     10$              ;MINUS = BIT 31
031112                               DLEFT   DATBUF
031122  000623                       BR      1$
                                     ;CLEAR OUT ANY DBE'S OR SBE'S
031124  104471               10$:    ECC1DIS                  ;DISABLE ECC ON 1 SELECTED CSR
031126  013701  002362               MOV     TESTADD,R1
031132  005011                       CLR     (R1)
031134  060501                       ADD     R5,R1
031136  005011                       CLR     (R1)
031140  104503                       CLR1CSR                  ;CLEAR 1 SELECTED CSR
031142  000207                       RETURN
```

```
 5330 031144                                MTP013: SUBTST  <<MTP013        CREATE DOUBLE BIT ERROR TEST>>
                                            ;*****************************************************************************
                                            ;*SUBTEST        MTP013  CREATE DOUBLE BIT ERROR TEST
                                            ;*****************************************************************************
 5331                                                       ;DOUBLE BIT ERROR FORCE TO CHECK DOUBLE ERROR LOGIC
 5332 031144  104503                                CLR1CSR                 ;CLEAR 1 SELECTED CSR
 5333 031146  012701  002362                        MOV     #TESTADD,R1
 5334 031152  005037  002234                1$:     CLR     DATBUF          ;MAKE INITIAL DATA
 5335 031156  005037  002236                        CLR     DATBUF+2        ;ALL ZEROS
 5336 031162  012737  000001  002244        2$:     MOV     #1,SBEMSK       ;INITIAL SINGLE ERROR MASK
 5337 031170  005037  002246                        CLR     SBEMSK+2        ;SECOND WORD
 5338 031174  012737  000001  002250        3$:     MOV     #1,DBEMSK       ;INITIAL DOUBLE ERROR MASK
 5339 031202  005037  002252                        CLR     DBEMSK+2        ;32 BITS HERE ALSO
 5340 031206  013737  002234  002240        4$:     MOV     DATBUF,TSTDAT   ;
 5341 031214  013737  002236  002242                MOV     DATBUF+2,TSTDAT+2
 5342 031222  105737  002256                        TSTB    PASFLG  ;NO COMPLEMENTING FIRST PASS
 5343 031226  001404                                BEQ     5$
 5344 031230  005137  002240                        COM     TSTDAT          ;COMP FIRST WORD
 5345 031234  005137  002242                        COM     TSTDAT+2        ;SECOND WORD
 5346 031240  104503                        5$:     CLR1CSR                 ;CLEAR 1 SELECTED CSR
 5347 031242  023737  002244  002250                CMP     SBEMSK,DBEMSK   ;CAN'T HAVE THE SAME ERROR BIT SET
 5348 031250  001004                                BNE     6$              ;IN BOTH MASKS
 5349 031252  023737  002246  002252                CMP     SBEMSK+2,DBEMSK+2;COULD BE EQUAL IN SECOND WORD
 5350 031260  001460                                BEQ     13$             ;GO MAKE THEM NOT EQUAL
 5351 031262  012737  002240  002272        6$:     MOV     #TSTDAT,SOURCE  ;SOURCE ADDRESS FOR CHKGEN
 5352 031270  004737  041662                        CALL    CHKGEN          ;GO GENERATE CHECK BITS
 5353 031274  013702  002244                        MOV     SBEMSK,R2
 5354 031300  074237  002240                        XOR     R2,TSTDAT
 5355 031304  013702  002246                        MOV     SBEMSK+2,R2
 5356 031310  074237  002242                        XOR     R2,TSTDAT+2
 5357 031314  013702  002250                        MOV     DBEMSK,R2
 5358 031320  074237  002240                        XOR     R2,TSTDAT
 5359 031324  013702  002252                        MOV     DBEMSK+2,R2
 5360 031330  074237  002242                        XOR     R2,TSTDAT+2
 5361 031334  104471                        16$:    ECC1DIS                 ;DISABLE ECC ON 1 SELECTED CSR
 5362 031336  013731  002240                        MOV     TSTDAT,@(R1)+   ;WRITE 16 BITS
 5363 031342  104475                                CB1CSR                  ;WRITE GENERATED CHECKBITS IN 1 SELECTED CSR
 5364 031344  013771  002242  000000                MOV     TSTDAT+2,@(R1)  ;WRITE HIGH WORD
 5365 031352  104503                                CLR1CSR                 ;CLEAR 1 SELECTED CSR
 5366 031354  162701  000002                        SUB     #2,R1           ;ADJUST TEST ADDRESS
 5367 031360  005771  000000                        TST     @(R1)           ;READ THE LOCATION
 5368 031364  104501                                WAS1DBE                 ;WAS THERE ANY DOUBLE BIT ERRORS ON 1 SELECTED CSR
 5369 031366  103411                                BCS     9$              ;IT SHOULD BE SET
 5370 031370                                        SET     HEADER
 5371 031376  011137  002032                        MOV     (R1),ADDRESS
 5372 031402  104030                                ERROR   +30
 5373 031404                                        SET     HEADER
```

```
5376 031412  104512                        9$:     ERRGEN
5377 031414  105737  002256                        TSTB    PASFLG
5378 031420  100452                                BMI     14$
5379 031422  005737  002252                13$:    TST     DBEMSK+2          ;CHECK MASK FOR LAST BIT
5380 031426  100405                                BMI     10$               ;MINUS = BIT31
5381 031430                                        DLEFT   DBEMSK
5382 031440  000662                                BR      4$
5383 031442                                10$:    IF #SW11 SET.IN @SWR THEN GOTO 11$
5384 031452                                        IF QVFLAG IS TRUE THEN GOTO 11$
5385 031460  005737  002246                        TST     SBEMSK+2          ;CHECK SINGLE ERROR MASK TOO
5386 031464  100405                                BMI     11$               ;BR IF DONE
5387 031466                                        DLEFT   SBEMSK
5388 031476  000636                                BR      3$
5389 031500  105737  002256                11$:    TSTB    PASFLG  ;FIRST PASS
5390 031504  001003                                BNE     12$               ;NON ZERO MEANS WE'RE DONE
5391 031506  105237  002256                        INCB    PASFLG  ;FIRST PASS, NOT DONE
5392                                               ;CLEAR OUT ANY DBE'S OR SBE'S
5393 031512  000617                                BR      1$                ;KEEP GOING
5394 031514  052737  000200  002256        12$:    BIS     #BIT7,PASFLG      ;SET UP FOR CHECK BIT TEST
5395 031522  005037  002240                        CLR     TSTDAT
5396 031526  005037  002242                        CLR     TSTDAT+2
5397 031532  012737  000040  002244                MOV     #40,SBEMSK
5398 031540  012737  000100  002250                MOV     #100,DBEMSK
5399 031546  012737  003740  002274        14$:    MOV     #3740,CHECK
5400 031554  013702  002244                        MOV     SBEMSK,R2
5401 031560  074237  002274                        XOR     R2,CHECK
5402 031564  013702  002250                        MOV     DBEMSK,R2
5403 031570  074237  002274                        XOR     R2,CHECK
5404 031574  006337  002250                        ASL     DBEMSK
5405 031600  032737  020000  002250                BIT     #BIT13,DBEMSK
5406 031606  001652                                BEQ     16$
5407 031610  006337  002244                        ASL     SBEMSK
5408 031614  032737  004000  002244                BIT     #BIT11,SBEMSK
5409 031622  001006                                BNE     15$
5410 031624  013737  002244  002250                MOV     SBEMSK,DBEMSK
5411 031632  006337  002250                        ASL     DBEMSK
5412 031636  000743                                BR      14$
5413 031640  104471                        15$:    ECC1DIS                   ;DISABLE ECC ON 1 SELECTED CSR
5414 031642  012701  002362                        MOV     #TESTADD,R1
5415 031646                                        CLEAR   @(R1)+,@(R1)
5416 031654  104503                                CLR1CSR                   ;CLEAR 1 SELECTED CSR
5417 031656  000207                                RETURN
```

```
031660                                  MTP014: SUBTST  <<MTP014        WRITE INHIBIT DURING DATIP WITH DBE TEST>>
                                        ;*****************************************************************************
                                        ;*SUBTEST        MTP014  WRITE INHIBIT DURING DATIP WITH DBE TEST
                                        ;*****************************************************************************
                                                        ;THIS TEST CHECKS THE WRITE INHIBIT ON DOUBLE
                                                        ;BIT ERRORS DURING A DATIP OPERATION BY USE
                                                        ;OF AN 'ASRB' INSTRUCTION.
031660                                          IF KFLAG IS TRUE THEN $RETURN

                                                        ;NOTE- THIS TEST WILL ONLY BE RUN FOR MF11S-K MEMORY.
031670  005037  002234                  1$:     CLR     DATBUF                  ;INITIAL DATA
031674  005037  002236                          CLR     DATBUF+2                ;2 WORDS WORTH
031700  012737  000001  002244          2$:     MOV     #1,SBEMSK               ;INITIAL ERROR MASK
031706  005037  002246                          CLR     SBEMSK+2                ;
031712  012737  000001  002250          3$:     MOV     #1,DBEMSK               ;DOUBLE ERROR MASK
031720  005037  002252                          CLR     DBEMSK+2                ;2 WORDS
031724  013737  002234  002240          4$:     MOV     DATBUF,TSTDAT           ;PRESERVE ORIG DATA
031732  013737  002236  002242                  MOV     DATBUF+2,TSTDAT+2
031740  105737  002256                          TSTB    PASFLG  ;SECOND PASS YET ?
031744  001404                                  BEQ     5$                      ;BR IF NO
031746  005137  002240                          COM     TSTDAT                  ;COMPL DATA  ON SECOND PASS
031752  005137  002242                          COM     TSTDAT+2
031756  104503                          5$:     CLR1CSR                         ;CLEAR 1 SELECTED CSR
031760  023737  002250  002244                  CMP     DBEMSK,SBEMSK           ;CHECK FOR SAME MASKS
031766  001004                                  BNE     6$                      ;BR IF OK
031770  023737  002252  002246                  CMP     DBEMSK+2,SBEMSK+2
031776  001476                                  BEQ     11$                     ;BR IF THEY'RE EQUAL
032000  012737  002240  002272          6$:     MOV     #TSTDAT,SOURCE          ;SET UP ADDRESS FOR CHKGEN
032006  004737  041662                          CALL    CHKGEN                  ;GENERATE CHECK BITS
032012  013701  002244                          MOV     SBEMSK,R1
032016  074137  002240                          XOR     R1,TSTDAT
032022  013701  002246                          MOV     SBEMSK+2,R1
032026  074137  002242                          XOR     R1,TSTDAT+2
032032  013701  002250                          MOV     DBEMSK,R1
032036  074137  002240                          XOR     R1,TSTDAT
032042  013701  002252                          MOV     DBEMSK+2,R1
032046  074137  002242                          XOR     R1,TSTDAT+2
032052  012701  002362                  7$:     MOV     #TESTADD,R1             ;TEST ADDRESS
032056  104471                                  ECC1DIS                         ;DISABLE ECC ON 1 SELECTED CSR
032060  013731  002240                          MOV     TSTDAT,@(R1)+           ;WRITE FIRST 16 BITS
032064  104475                                  CB1CSR                          ;WRITE GENERATED CHECKBITS IN 1 SELECTED CSR
032066  013771  002242  000000                  MOV     TSTDAT+2,@(R1)          ;SECOND 16 BITS+CHECKBITS
032074  105037  002257                          CLRB    UPPFLG                  ;INDICATE LOWER WORD
032100  013703  002362                          MOV     TESTADD,R3              ;TEST ADDRESS
032104  104503                          8$:     CLR1CSR                         ;CLEAR 1 SELECTED CSR
032106  106223                                  ASRB    (R3)+                   ;SPECIAL DATIP INSTRUCTION
032110  015100                                  MOV     @-(R1),R0
032112  023700  002240                          CMP     TSTDAT,R0               ;CHECK FOR UNCHANGED DATA
032116  001404                                  BEQ     9$                      ;SHOULD BE UNCHANGED
032120  017137  000000  002032                  MOV     @(R1),ADDRESS
032126  104455                                  PERR31

032130  062701  000002                  9$:     ADD     #2,R1                   ;POINT TO UPPER WORD
032134  017100  000000                          MOV     @(R1),R0
032140  023700  002242                          CMP     TSTDAT+2,R0             ;READ IT
032144  001404                                  BEQ     10$                     ;BR IF UNCHANGED
032146  017137  000000  002032                  MOV     @(R1),ADDRESS
```

```
 5479 032154  104455                               PERR31
 5480
 5481 032156  122737  000003  002257  10$:   CMPB    #3,UPPFLG               ;LOWER WORD
 5482 032164  001403                          BEQ     11$             ;BR IF NO
 5483 032166  105237  002257                  INCB    UPPFLG
 5484 032172  000744                          BR      8$
 5485 032174  105737  002256          11$:   TSTB    PASFLG
 5486 032200  100453                          BMI     15$             ;BRANCH IF WE'RE TESTING CHECK BITS
 5487 032202  005737  002252                  TST     DBEMSK+2        ;LAST BIT IN MASK ?
 5488 032206  100405                          BMI     12$             ;BR IF BIT 31
 5489 032210                                  DLEFT   DBEMSK
 5490 032220  000641                          BR      4$
 5491 032222                          12$:   IF #SW11 SET.IN @SWR THEN GOTO 13$
 5492 032232                                  IF QVFLAG IS TRUE THEN GOTO 13$
 5493 032240  005737  002246                  TST     SBEMSK+2        ;LAST BIT IN SINGLE ERROR MASK ?
 5494 032244  100405                          BMI     13$             ;BR IF YES
 5495 032246                                  DLEFT   SBEMSK
 5496 032256  000615                          BR      3$
 5497 032260  105737  002256          13$:   TSTB    PASFLG  ;WHICH PASS
 5498 032264  001004                          BNE     14$             ;BR IF WE'RE DONE
 5499 032266  105237  002256                  INCB    PASFLG  ;INDICATE SECOND PASS COMING
 5500                                         ;CLEAR OUT ANY DBE'S OR SBE'S
 5501 032272  000137  031670                  JMP     1$              ;GO DO IT!
 5502 032276  052737  000200  002256  14$:   BIS     #BIT7,PASFLG
 5503 032304  005037  002240                  CLR     TSTDAT
 5504 032310  005037  002242                  CLR     TSTDAT+2
 5505 032314  012737  000040  002244          MOV     #40,SBEMSK
 5506 032322  012737  000100  002250          MOV     #100,DBEMSK
 5507 032330  012737  003740  002274  15$:   MOV     #3740,CHECK
 5508 032336  013702  002244                  MOV     SBEMSK,R2
 5509 032342  074237  002274                  XOR     R2,CHECK
 5510 032346  013702  002250                  MOV     DBEMSK,R2
 5511 032352  074237  002274                  XOR     R2,CHECK
 5512 032356  006337  002250                  ASL     DBEMSK
 5513 032362  032737  020000  002250          BIT     #BIT13,DBEMSK
 5514 032370  001630                          BEQ     7$
 5515 032372  006337  002244                  ASL     SBEMSK
 5516 032376  032737  004000  002244          BIT     #BIT11,SBEMSK
 5517 032404  001006                          BNE     16$
 5518 032406  013737  002244  002250          MOV     SBEMSK,DBEMSK
 5519 032414  006337  002250                  ASL     DBEMSK
 5520 032420  000743                          BR      15$
 5521 032422  104471                  16$:   ECC1DIS                 ;DISABLE ECC ON 1 SELECTED CSR
 5522 032424  012701  002362                  MOV     #TESTADD,R1
 5523 032430                                  CLEAR   @(R1)+,@(R1)
 5524 032436  104503                          CLR1CSR                 ;CLEAR 1 SELECTED CSR
 5525 032440  000207                          RETURN
```

```
032442                                  MTP015: SUBTST  <<MTP015        WRITE INHIBIT OF BYTE WITH DBE>>
                                        ;*******************************************************************************
                                        ;*SUBTEST       MTP015  WRITE INHIBIT OF BYTE WITH DBE
                                        ;*******************************************************************************
                                                        ;CHECK FOR WRITE INHIBIT DURING A WRITE BYTE.
                                                        ;CHECKS FOR UNCORRECTED DATA.
032442  005037  002234                  1$:     CLR     DATBUF                  ;INITIAL DATA
032446  005037  002236                          CLR     DATBUF+2                ;32 BITS WORTH
032452  012737  000001  002244          2$:     MOV     #1,SBEMSK               ;SINGLE ERROR MASK
032460  005037  002246                          CLR     SBEMSK+2                ;
032464  012737  000001  002250          3$:     MOV     #1,DBEMSK               ;DOUBLE ERROR MASK
032472  005037  002252                          CLR     DBEMSK+2                ;
032476  013737  002234  002240          4$:     MOV     DATBUF,TSTDAT           ;PRESERVE ORIG DATA
032504  013737  002236  002242                  MOV     DATBUF+2,TSTDAT+2
032512  105737  002256                  .       TSTB    PASFLG  ;WHICH PASS ?
032516  001404                                  BEQ     5$                      ;FIRST PASS, NO COMPLEMENTING
032520  005137  002240                          COM     TSTDAT
032524  005137  002242                          COM     TSTDAT+2                ;SECOND PASS, COMPLEMENT TSTDAT
032530  104503                          5$:     CLR1CSR                         ;CLEAR 1 SELECTED CSR
032532  023737  002244  002250                  CMP     SBEMSK,DBEMSK           ;CHECK FOR SAME MASKS
032540  001004                                  BNE     6$                      ;BR IF NOT EQUAL
032542  023737  002246  002252                  CMP     SBEMSK+2,DBEMSK+2               ;SECOND WORD ALSO
032550  001474                                  BEQ     11$                     ;BR TO MAKE THEM NOT EQUAL
032552  012737  002240  002272          6$:     MOV     #TSTDAT,SOURCE          ;ADDRESS FOR CHKGEN
032560  004737  041662                          CALL    CHKGEN                  ;GO GENERATE CHECK BITS
032564  013701  002244                          MOV     SBEMSK,R1
032570  074137  002240                          XOR     R1,TSTDAT
032574  013701  002246                          MOV     SBEMSK+2,R1
032600  074137  002242                          XOR     R1,TSTDAT+2
032604  013701  002250                          MOV     DBEMSK,R1
032610  074137  002240                          XOR     R1,TSTDAT
032614  013701  002252                          MOV     DBEMSK+2,R1
032620  074137  002242                          XOR     R1,TSTDAT+2
032624  012701  002362                  7$:     MOV     #TESTADD,R1             ;TEST LOCATION
032630  104471                                  ECC1DIS                         ;DISABLE ECC ON 1 SELECTED CSR
032632  013731  002240                          MOV     TSTDAT,@(R1)+           ;WRITE FIRST 16 BITS
                                                ;LOAD CSR WITH IMAGE FROM R2
032636  104475                                  CB1CSR                          ;WRITE GENERATED CHECKBITS IN 1 SELECTED CSR
032640  013771  002242  000000                  MOV     TSTDAT+2,@(R1)          ;WRITE SECOND 16 BITS + CHECKBITS
032646  104503                                  CLR1CSR                         ;CLEAR 1 SELECTED CSR
032650  013702  002362                          MOV     TESTADD,R2              ;GET ADDRESS OF TEST LOC
032654  010203                                  MOV     R2,R3                   ;R2 DESIGNATES FIRST BYTE
032656  062703  000003                          ADD     #3,R3                   ;R3 DESIGNATES LAST BYTE
032662  112722  000360                  8$:     MOVB    #360,(R2)+              ;TRY WRITING A BYTE
032666  012701  002362                          MOV     #TESTADD,R1
032672  017100  000000                          MOV     @(R1),R0
032676  023700  002240                          CMP     TSTDAT,R0               ;CHECK FOR UNCHANGED DATA
032702  001404                                  BEQ     9$                      ;BR IF OK
032704  017137  000000  002032                  MOV     @(R1),ADDRESS
032712  104455                                  PERR31

032714  017100  000002                  9$:     MOV     @2(R1),R0
032720  023700  002242                          CMP     TSTDAT+2,R0             ;READ SECOND WORD
032724  001404                                  BEQ     10$                     ;BR IF UNCHANGED
032726  017137  000002  002032                  MOV     @2(R1),ADDRESS
032734  104455                                  PERR31

```

```
032736  020203                          10$:    CMP     R2,R3                   ;TESTED LAST BYTE ?
032740  001350                                  BNE     8$                      ;BR IF NO
032742  105737  002256                  11$:    TSTB    PASFLG
032746  100452                                  BMI     15$                     ;BRANCH IF TESTING CHECK BITS
032750  005737  002252                          TST     DBEMSK+2                ;CHECKING FOR LAST ERROR BIT
032754  100405                                  BMI     12$                     ;BR IF DONE HERE
032756                                          DLEFT   DBEMSK
032766  000643                                  BR      4$
032770                                  12$:    IF #SW11 SET.IN @SWR THEN GOTO 13$
033000                                          IF QVFLAG IS TRUE THEN GOTO 13$
033006  005737  002246                          TST     SBEMSK+2                ;LAST SBE MASK
033012  100405                                  BMI     13$                     ;BR IF DONE WITH THIS PASS
033014                                          DLEFT   SBEMSK
033024  000617                                  BR      3$
033026  105737  002256                  13$:    TSTB    PASFLG  ;TEST PASS FLAG
033032  001003                                  BNE     14$                     ;NON ZERO MEANS WE'RE DONE
033034  105237  002256                          INCB    PASFLG  ;NOT DONR
033040  000600                                  BR      1$
033042  052737  000200  002256          14$:    BIS     #BIT7,PASFLG
033050  005037  002240                          CLR     TSTDAT
033054  005037  002242                          CLR     TSTDAT+2
033060  012737  000040  002244                  MOV     #40,SBEMSK
033066  012737  000100  002250                  MOV     #100,DBEMSK
033074  012737  003740  002274          15$:    MOV     #3740,CHECK
033102  013702  002244                          MOV     SBEMSK,R2
033106  074237  002274                          XOR     R2,CHECK
033112  013702  002250                          MOV     DBEMSK,R2
033116  074237  002274                          XOR     R2,CHECK
033122  006337  002250                          ASL     DBEMSK
033126  032737  020000  002250                  BIT     #BIT13,DBEMSK
033134  001633                                  BEQ     7$
033136  006337  002244                          ASL     SBEMSK
033142  032737  004000  002244                  BIT     #BIT11,SBEMSK
033150  001006                                  BNE     16$
033152  013737  002244  002250                  MOV     SBEMSK,DBEMSK
033160  006337  002250                          ASL     DBEMSK
033164  000743                                  BR      15$
033166  104471                          16$:    ECC1DIS                         ;DISABLE ECC ON 1 SELECTED CSR
033170  012701  002362                          MOV     #TESTADD,R1             ;TEST LOCATION
033174                                          CLEAR   @(R1)+,@(R1)            ;TO ERASE ANY DBE'S FROM TESTING
                                                ;RESTORE CSR
033202  104503                                  CLR1CSR                         ;CLEAR 1 SELECTED CSR
033204  000207                                  RETURN
```

```
033206                          MTP016: SUBTST  <<MTP016        WRITE INHIBIT OF WORD WITH DBE>>
                                ;*************************************************************************
                                ;*SUBTEST       MTP016  WRITE INHIBIT OF WORD WITH DBE
                                ;*************************************************************************
                                        ;DOUBLE BIT ERROR WRITE CANCEL WITH
                                        ;WORD WRITE.
                                        ;CHECKS WRITE INHIBIT WITH WORD WRITES TO
                                        ;WORD WITH DOUBLE ERROR.
033206  005037  002234          T12A:   CLR     DATBUF          ;BACKGROUND FOR DOUBLE ERRORS
033212  005037  002236                  CLR     DATBUF+2        ;2 WORDS WORTH
033216  012737  000001  002244          MOV     #1,SBEMSK       ;SINGLE ERROR MASK
033224  005037  002246                  CLR     SBEMSK+2        ;
033230  012737  000001  002250  T12B:   MOV     #1,DBEMSK       ;DOUBLE ERROR MASK
033236  005037  002252                  CLR     DBEMSK+2
033242  013737  002234  002240  1$:     MOV     DATBUF,TSTDAT   ;DATA FOR TEST
033250  013737  002236  002242          MOV     DATBUF+2,TSTDAT+2;BOTH WORDS
033256  105737  002256                  TSTB    PASFLG  ;COMP DATA ON SECOND PASS ONLY
033262  001404                          BEQ     2$              ;BR IF FIRST PASS
033264  005137  002240                  COM     TSTDAT          ;COMP FIRST WORD
033270  005137  002242                  COM     TSTDAT+2        ;NOW SECOND WORD
033274  023737  002244  002250  2$:     CMP     SBEMSK,DBEMSK   ;CHECK FOR IDENTICAL MASKS
033302  001004                          BNE     3$              ;BR IF DIFFERENT
033304  023737  002246  002252          CMP     SBEMSK+2,DBEMSK+2;UPPER WORD TOO
033312  001502                          BEQ     8$              ;BR TO MAKE THEM NOT EQUAL
033314  012737  002240  002272  3$:     MOV     #TSTDAT,SOURCE  ;NEED ADDR OF DATA FOR CHKGEN
033322  004737  041662                  CALL    CHKGEN          ;GO GENERATE CHECK BITS
033326  013701  002244                  MOV     SBEMSK,R1
033332  074137  002240                  XOR     R1,TSTDAT
033336  013701  002246                  MOV     SBEMSK+2,R1
033342  074137  002242                  XOR     R1,TSTDAT+2
033346  013701  002250                  MOV     DBEMSK,R1
033352  074137  002240                  XOR     R1,TSTDAT
033356  013701  002252                  MOV     DBEMSK+2,R1
033362  074137  002242                  XOR     R1,TSTDAT+2
033366  012701  002362          4$:     MOV     #TESTADD,R1     ;FIRST TEST ADRRESS
033372  104471                          ECC1DIS                 ;DISABLE ECC ON 1 SELECTED CSR
033374  013731  002240                  MOV     TSTDAT,@(R1)+   ;WRITE FIRST 16 BITS
033400  104475                          CB1CSR                  ;WRITE GENERATED CHECKBITS IN 1 SELECTED CSR
033402  013771  002242  000000          MOV     TSTDAT+2,@(R1)  ;WRITE SECOND 16 BITS + CHECKBITS
033410  105037  002257                  CLRB    UPPFLG          ;SET FOR 2 LOOPS
033414  162701  000002                  SUB     #2,R1           ;POINT TO LOW WORD
033420  104503                  5$:     CLR1CSR                 ;CLEAR 1 SELECTED CSR
033422  012771  177400  000000          MOV     #177400,@(R1)   ;TRY WRITING LOCATION
033430  012701  002362                  MOV     #TESTADD,R1
033434  017100  000000                  MOV     @(R1),R0
033440  023700  002240                  CMP     TSTDAT,R0       ;CHECK FOR ORIGINAL DATA
033444  001404                          BEQ     6$              ;SHOULD BE UNCHANGED
033446  017137  000000  002032          MOV     @(R1),ADDRESS
033454  104455                          PERR31

033456  062701  000002          6$:     ADD     #2,R1
033462  017100  000000                  MOV     @(R1),R0
033466  023700  002242                  CMP     TSTDAT+2,R0     ;THIS SHOULD BE UNCHANGED ALSO
033472  001404                          BEQ     7$
033474  017137  000000  002032          MOV     @(R1),ADDRESS
033502  104455                          PERR31
```

```
 5682 033504  105737  002257          7$:     TSTB    UPPFLG              ;WHICH LOOP ?
 5683 033510  001003                          BNE     8$                  ;SECOND, BR OUT
 5684 033512  105237  002257                  INCB    UPPFLG              ;FIRST, KEEP GOING
 5685 033516  000740                          BR      5$
 5686 033520  105737  002256          8$:     TSTB    PASFLG
 5687 033524  100454                          BMI     12$
 5688 033526  005737  002252                  TST     DBEMSK+2            ;LAST BIT ?
 5689 033532  100405                          BMI     9$                  ;MINUS = BIT 31
 5690 033534                                  DLEFT   DBEMSK
 5691 033544  000636                          BR      1$
 5692 033546                          9$:     IF #SW11 SET.IN @SWR THEN GOTO 10$
 5693 033556                                  IF QVFLAG IS TRUE THEN GOTO 10$
 5694 033564  005737  002246                  TST     SBEMSK+2            ;LAST BIT IN THIS MASK ?
 5695 033570  100406                          BMI     10$                 ;BR IF LAST BIT
 5696 033572                                  DLEFT   SBEMSK
 5697 033602  000137  033230                  JMP     T12B
 5698 033606  105737  002256          10$:    TSTB    PASFLG  ;FIRST PASS ?
 5699 033612  001004                          BNE     11$                 ;BR IF SECOND
 5700 033614  105237  002256                  INCB    PASFLG  ;INDICATE SECOND PASS COMING
 5701 033620  000137  033206                  JMP     T12A
 5702 033624  052737  000200  002256  11$:    BIS     #BIT7,PASFLG
 5703 033632  005037  002240                  CLR     TSTDAT
 5704 033636  005037  002242                  CLR     TSTDAT+2
 5705 033642  012737  000040  002244          MOV     #40,SBEMSK
 5706 033650  012737  000100  002250          MOV     #100,DBEMSK
 5707 033656  012737  003740  002274  12$:    MOV     #3740,CHECK
 5708 033664  013702  002244                  MOV     SBEMSK,R2
 5709 033670  074237  002274                  XOR     R2,CHECK
 5710 033674  013702  002250                  MOV     DBEMSK,R2
 5711 033700  074237  002274                  XOR     R2,CHECK
 5712 033704  006337  002250                  ASL     DBEMSK
 5713 033710  032737  020000  002250          BIT     #BIT13,DBEMSK
 5714 033716  001623                          BEQ     4$
 5715 033720  006337  002244                  ASL     SBEMSK
 5716 033724  032737  004000  002244          BIT     #BIT11,SBEMSK
 5717 033732  001006                          BNE     13$
 5718 033734  013737  002244  002250          MOV     SBEMSK,DBEMSK
 5719 033742  006337  002250                  ASL     DBEMSK
 5720 033746  000743                          BR      12$
 5721 033750  104471                  13$:    ECC1DIS                     ;DISABLE ECC ON 1 SELECTED CSR
 5722 033752  012701  002362                  MOV     #TESTADD,R1         ;RESTORE TEST ADDRESS
 5723 033756  005031                          CLR     @(R1)+              ;CLEAR ANY DBE'S FROM TEST
 5724 033760  005071  000000                  CLR     @(R1)
 5725 033764  104503                          CLR1CSR                     ;CLEAR 1 SELECTED MK11 CSR
 5726 033766  000207                          RETURN
```

```
033770                               MTP017: SUBTST  <<MTP017         HOLDING 1'S & 0'S TEST>>
                                     ;*************************************************************************
                                     ;*SUBTEST        MTP017  HOLDING 1'S & 0'S TEST
                                     ;*************************************************************************
                                             ;*(1)    THIS TEST CHECKS THE MEMORY FOR THE CAPABILITY
                                             ;*       OF HOLDING 1'S AND 0'S BY WRITING A BACKGROUND
                                             ;*       OF 000377 AND READING IT
                                             ;*(2)    MEMORY IS WRITTEN USING A BYTE AT A TIME
                                             ;*(3)    STEPS 1 & 2 ARE REPEATED WITH A SWAPPED BACKGROUND PATTERN
                                             ;NOTE:   THIS TEST WRITES BYTES & READS WORDS
033770  012701  060000                       MOV     #FIRST,R1
033774  010104                               MOV     R1,R4
033776  012705  160000                       MOV     #LAST+2,R5
034002  012700  000377                       MOV     #377,R0          ;GET THE PATTERN INTO R0
034006  010003                               MOV     R0,R3
034010  000303                               SWAB    R3
034012  110021                       1$:     MOVB    R0,(R1)+         ;WRITE A BYTE
034014  110321                               MOVB    R3,(R1)+         ;WRITE THE MEMORY WITH THE BYTE STORED IN BAKPAT+1
034016  020105                               CMP     R1,R5            ;COMPARE TEST LOC TO TOP + 2
034020  103774                               BLO     1$               ;BRANCH IF LOWER

034022  014102                       2$:     MOV     -(R1),R2
034024  020002                               CMP     R0,R2            ;TEST THE MEMORY TO SEE IF IT CONTAINS
                                                                      ;THE WORD STORED IN BAKPAT
034026  001401                               BEQ     3$
034030  104446                               PERR22

034032  020104                       3$:     CMP     R1,R4            ;KEEP ON TESTING THE MEMORY UNTIL
034034  101372                               BHI     2$               ;R1 EQUALS THE LOWEST ADDRESS
034036  000303                               SWAB    R3               ;CHANGE THE DATA PATTERN
034040  000300                               SWAB    R0
034042  001763                               BEQ     1$               ;IF THE DATA PATTERN DOES NOT HAVE LOW
                                                                      ; BYTE =0 THEN FALL THRU
034044  000207                               RETURN
```

```
034046                  MTP020: SUBTST  <<MTP020        MARCHING 1'S & 0'S IN CHECK BITS TEST>>
                        ;*******************************************************************************
                        ;*SUBTEST        MTP020  MARCHING 1'S & 0'S IN CHECK BITS TEST
                        ;*******************************************************************************
                                ;*THIS TEST IS CONCERNED ONLY WITH THE INTEGRITY
                                ;*OF THE MOS RAMS THAT STORE THE CHECKBITS.

                                ;077 --> 100 DOWN
034046  160201          MTPA20: SUB     R2,R1           ;V177640
034050  160204                  SUB     R2,R4           ;V177642
034052  005711                  TST     (R1)            ;V177644        ;1ST WORD OK?
034054  001002                  BNE     1$              ;V177646        ;NO - SKIP
034056  005714                  TST     (R4)            ;V177650        ;2ND WORD OK?
034060  001401                  BEQ     2$              ;V177652        ;YES - SKIP
034062  104453          1$:     PERR27                  ;V177654        ;GOOD=000000,,000000,,077
034064  010314          2$:     MOV     R3,(R4)         ;V177656        ;2ND WORD <= 100000
034066  005011                  CLR     (R1)            ;V177660        ;CLEAR 1ST WORD
034070  020100                  CMP     R1,R0           ;V177662        ;ARE WE DONE?
034072  101365                  BHI     MTPA20          ;V177664        ;BRANCH IF NOT
034074  000207                  RETURN                  ;V177666


                                ;100 --> 077 UP
034076  005711          MTPB20: TST     (R1)            ;V177640        ;1ST WORD OK?
034100  001002                  BNE     3$              ;V177642        ;NO - SKIP
034102  020314                  CMP     R3,(R4)         ;V177644        ;2ND WORD OK?
034104  001401                  BEQ     4$              ;V177646        ;YES - SKIP
034106  104452          3$:     PERR26                  ;V177650        ;GOOD=000000,,100000,,100
034110  005014          4$:     CLR     (R4)            ;V177652        ;CLEAR 2ND WORD
034112  005011                  CLR     (R1)            ;V177654        ;CLEAR 1ST WORD
034114  060201                  ADD     R2,R1           ;V177656
034116  060204                  ADD     R2,R4           ;V177660
034120  020405                  CMP     R4,R5           ;V177662        ;TOP + 2 YET?
034122  001365                  BNE     MTPB20          ;V177664        ;NO - LOOP
034124  000207                  RETURN                  ;V177666


                                ;077 --> 100 UP
034126  005711          MTPC20: TST     (R1)            ;V177640        ;1ST WORD OK?
034130  001002                  BNE     5$              ;V177642        ;NO - SKIP
034132  005714                  TST     (R4)            ;V177644        ;2ND WORD OK?
034134  001401                  BEQ     6$              ;V177646        ;YES - SKIP
034136  104453          5$:     PERR27                  ;V177650        ;GOOD=000000,,000000,,077
034140  010314          6$:     MOV     R3,(R4)         ;V177652        ;WRITE 1ST WORD
034142  005011                  CLR     (R1)            ;V177654        ;WRITE 2ND WORD
034144  060204                  ADD     R2,R4           ;V177656
034146  060201                  ADD     R2,R1           ;V177660
034150  020405                  CMP     R4,R5           ;V177662        ;TOP + 2 YET?
034152  001365                  BNE     MTPC20          ;V177664        ;NO - LOOP
034154  000207                  RETURN                  ;V177666
```

```
                                          ;100 --> 077 DOWN
034156  160201                  MTPD20:  SUB     R2,R1           ;V177640
034160  160204                           SUB     R2,R4           ;V177642
034162  020314                           CMP     R3,(R4)         ;V177644        ;2ND WORD OK?
034164  001002                           BNE     7$              ;V177646        ;NO - SKIP
034166  005711                           TST     (R1)            ;V177650        ;1ST WORD OK?
034170  001401                           BEQ     8$              ;V177652        ;YES - SKIP
034172  104452                  7$:      PERR26                  ;V177654        ;GOOD=000000,,100000,,100
034174  005014                  8$:      CLR     (R4)            ;V177656        ;WRITE 1ST WORD
034176  005011                           CLR     (R1)            ;V177660        ;WRITE 2ND WORD
034200  020100                           CMP     R1,R0           ;V177662
034202  101365                           BHI     MTPD20          ;V177664
034204  000207                           RETURN                  ;V177666

                                          ;077 UP
034206  005711                  MTPE20:  TST     (R1)            ;V177640        ;1ST WORD OK?
034210  001002                           BNE     9$              ;V177642        ;NO - SKIP
034212  005714                           TST     (R4)            ;V177644        ;2ND WORD OK?
034214  001401                           BEQ     10$             ;V177646        ;YES - SKIP
034216  104453                  9$:      PERR27                  ;V177650        ;GOOD=000000,,000000,,077
034220  060201                  10$:     ADD     R2,R1           ;V177652
034222  060204                           ADD     R2,R4           ;V177654
034224  020405                           CMP     R4,R5           ;V177656        ;TOP + 2 YET?
034226  001367                           BNE     MTPE20          ;V177660        ;NO - LOOP
034230  000207                           RETURN                  ;V177662
```

```
 5836 034232                                    MTPA21: SUBTST  <<MTPA21         MARCHING 1'S & 0'S PATTERN TEST>>
                                                ;*******************************************************************************
                                                ;*SUBTEST        MTPA21  MARCHING 1'S & 0'S PATTERN TEST
                                                ;*******************************************************************************
 5837                                                   ;READ,BYTESWAP-MODIFY,READ,DOWN
 5838 034232  014100                            1$:     MOV     -(R1),R0;V177640
 5839 034234  020200                                    CMP     R2,R0   ;V177642
 5840 034236  001401                                    BEQ     2$      ;V177644
 5841 034240  104443                                    PERR17          ;V177646
 5842
 5843 034242  000311                            2$:     SWAB    (R1)    ;V177650
 5844 034244  011100                                    MOV     (R1),R0 ;V177652
 5845 034246  020300                                    CMP     R3,R0   ;V177654
 5846 034250  001401                                    BEQ     3$      ;V177656
 5847 034252  104444                                    PERR20          ;V177660
 5848
 5849 034254  020401                            3$:     CMP     R4,R1   ;V177662        ;DONE?
 5850 034256  001365                                    BNE     1$      ;V177664        ;NO - LOOP
 5851 034260  000207                                    RETURN          ;V177666        ;YES - RETURN
 5852
 5853 034262                                    MTPB21: ;READ,BYTESWAP-MODIFY,READ,UP
 5854 034262  011100                            1$:     MOV     (R1),R0 ;V177640
 5855 034264  020300                                    CMP     R3,R0   ;V177642
 5856 034266  001401                                    BEQ     2$      ;V177644
 5857 034270  104444                                    PERR20          ;V177646
 5858
 5859 034272  000311                            2$:     SWAB    (R1)    ;V177650
 5860 034274  011100                                    MOV     (R1),R0 ;V177652
 5861 034276  020200                                    CMP     R2,R0   ;V177654
 5862 034300  001401                                    BEQ     3$      ;V177656
 5863 034302  104443                                    PERR17          ;V177660
 5864
 5865 034304  062701  000002                    3$:     ADD     #2,R1   ;V177662
 5866 034310  020501                                    CMP     R5,R1   ;V177666        ;DONE?
 5867 034312  001363                                    BNE     1$      ;V177670        ;NO - LOOP
 5868 034314  000207                                    RETURN          ;V177672        ;YES - RETURN
 5869
 5870 034316                                    MTPC21: ;READ,BYTESWAP-MODIFY,READ,UP
 5871 034316  011100                            1$:     MOV     (R1),R0 ;V177640
 5872 034320  020200                                    CMP     R2,R0   ;V177642
 5873 034322  001401                                    BEQ     2$      ;V177644
 5874 034324  104443                                    PERR17          ;V177646
 5875
 5876 034326  000311                            2$:     SWAB    (R1)    ;V177650
 5877 034330  011100                                    MOV     (R1),R0 ;V177652
 5878 034332  020300                                    CMP     R3,R0   ;V177654
 5879 034334  001401                                    BEQ     3$      ;V177656
 5880 034336  104444                                    PERR20          ;V177660
 5881
 5882 034340  062701  000002                    3$:     ADD     #2,R1   ;V177662
 5883 034344  020501                                    CMP     R5,R1   ;V177666        ;DONE?
 5884 034346  001363                                    BNE     1$      ;V177670        ;NO - LOOP
 5885 034350  000207                                    RETURN          ;V177672        ;YES - RETURN
```

```
034352                    MTPD21: ;READ,BYTESWAP-MODIFY,READ,DOWN
034352  014100            1$:     MOV     -(R1),R0;V177640
034354  020300                    CMP     R3,R0   ;V177642
034356  001401                    BEQ     2$      ;V177644
034360  104444                    PERR20          ;V177646

034362  000311            2$:     SWAB    (R1)    ;V177650
034364  011100                    MOV     (R1),R0 ;V177652
034366  020200                    CMP     R2,R0   ;V177654
034370  001401                    BEQ     3$      ;V177656
034372  104443                    PERR17          ;V177660

034374  020401            3$:     CMP     R4,R1   ;V177662        ;DONE?
034376  001365                    BNE     1$      ;V177664        ;NO - LOOP
034400  000207                    RETURN          ;V177666        ;YES - RETURN
```

```
034402                          MTP022: SUBTST  <<MTP022        REFRESH & SHIFTING DIAGONAL TEST>>
                                ;*******************************************************************************
                                ;*SUBTEST       MTP022  REFRESH & SHIFTING DIAGONAL TEST
                                ;*******************************************************************************
                                          ;(1)     WE WRITE A DIAGONAL PATTERN IN MEMORY (WITH CACHE ON).
                                          ;(2)     IF A REFRESH TEST WE DISTURB ALL ROWS FOR > 2 MS (WITH CACHE ON).
                                          ;(3)     WE READ & CHECK FOR CORRECTNESS THE DIAGONAL PATTERN
                                          ;        (WITH CACHE OFF).
        000010                  KDIAG=8.            ;HOW OFTEN A DIAGONAL STRIPE OCCURS (MUST BE A POWER OF 2)
034402                                  FOR EVEN := #1 TO #2                    ;FOR DATA & COMPLEMENT DATA
034410                                    IF EVEN EQ #1
034420                                      LET R2 := ZEROS
034424                                      LET R3 := ONES
034430                                    ELSE
034432                                      LET R2 := ONES
034436                                      LET R3 := ZEROS
034442                                    END ;OF IF EVEN
034442                                    FOR STRIPES := #0 TO #KDIAG-1         ;FOR THE NUMBER OF STRIPES

                                            ;WRITE LOOP
034446  104423                              CACHON                      ;TURN CACHE ON
034450                                      LET COUNT := STRIPES
034456                                      LET R1 := #FIRST
034462                                      WHILE R1 LOS #LAST
034470                                        IF COUNT LT #0 THEN LET COUNT := #KDIAG-1
034504                                        IF #374 OFF.IN R1 THEN LET COUNT := COUNT - #1
034516                                        IF COUNT NE #0
034524                                          LET (R1) := R2
034526                                          LET 2(R1) := R2
034532                                        ELSE
034534                                          LET (R1) := R3
034536                                          LET 2(R1) := R3
034542                                        END ;OF IF COUNT
034542                                        LET COUNT := COUNT - #1
034546                                        LET R1 := R1 + #4
034552                                      END ;OF WHILE
                                            ;END OF WRITE LOOP

034554                                      IF DIAGFLAG IS FALSE THEN $CALL REFRESH
                                            ;READ LOOP
034566                                      LET COUNT := STRIPES
034574                                      LET R1 := #FIRST
034600  104424                              CACHOFF                     ;TURN CACHE OFF
```

```
034602                                    WHILE R1 LOS #LAST
034610                                      IF COUNT LT #0 THEN LET COUNT := #KDIAG-1
034624                                      IF #374 OFF.IN R1 THEN LET COUNT := COUNT - #1
034636                                      IF COUNT NE #0
034644                                        LET R0 := (R1)
034646                                        IF R2 NE R0
034652  104443                                  PERR17
034654                                        END ;OF IF R2
034654                                        LET R0 := 2(R1)
034660                                        IF R2 NE R0
034664  104443                                  PERR17
034666                                        END ;OF IF R2
034666                                      ELSE
034670                                        LET R0 := (R1)
034672                                        IF R3 NE R0
034676  104444                                  PERR20
034700                                        END ;OF IF R3
034700                                        LET R0 := 2(R1)
034704                                        IF R3 NE R0
034710  104444                                  PERR20
034712                                        END ;OF IF R3
034712                                      END ;OF IF COUNT
034712                                      LET COUNT := COUNT - #1
034716                                      LET R1 := R1 + #4
034722                                    END ;OF WHILE
                                          ;END OF READ LOOP


034724                                  END ;OF FOR STRIPES
034740                                END ;OF FOR EVEN
034754  000207                          RETURN

034756                        REFRESH:SUBTST  <<SUBR  REFRESH DELAY>>
                              ;*******************************************************************************
                              ;*SUBTEST         SUBR    REFRESH DELAY
                              ;*******************************************************************************
                                        ;DISTURB EACH ROW FOR > 3.2 MS
034756                                  FOR R0 := #FIRST TO #FIRST+374 BY #4
034762  004737  035026                    CALL REFSUB
034766                                  END ;OF FOR R0
035000                                  LET R0 := #FIRST+BIT14
035004                                  WHILE R0 LOS #LAST+BIT14+374
035012  004737  035026                    CALL REFSUB
035016                                    LET R0 := R0 + #4
035022                                  END ;OF WHILE
035024  000207                          RETURN
035026  012704  000640        REFSUB:  MOV     #640,R4                  ;TIME FOR A > 3.2 MS LOOP
035032  062700  000002                 ADD     #2,R0
035036  005140                1$:      COM     -(R0)
035040  005120                         COM     (R0)+
035042  005110                         COM     (R0)
035044  005110                         COM     (R0)
035046  077405                         SOB     R4,1$
035050  162700  000002                 SUB     #2,R0
035054  000207                         RETURN
```

```
 035056                            MTPA24: SUBTST  <<MTPA24        FAST GALLOPING PATTERN TEST>>
                                   ;*******************************************************************************
                                   ;*SUBTEST        MTPA24  FAST GALLOPING PATTERN TEST
                                   ;*******************************************************************************
                                                   ;THE TOTAL TEST (INCLUDING SETUP) IS AS FOLLOWS
                                                   ;*(1)    THIS TEST WRITES THE MEMORY WITH A BACK GROUND PATTERN
                                                   ;*       STORED AT LOCATION BAKPAT
                                                   ;*(2)    TEST BEGINS AT LOWEST LOCATION BEING TESTED
                                                   ;*       (LETS NAME IT 'A')
                                                   ;*(3)    LETS NAME THE 1ST LOCATION IN THE ROW/COLUMN UNDER TEST AS 'B'.
                                                   ;*(4)    SWAPS BYTES FOR LOCATION 'A'.
                                                   ;*(5)    READS 'A', READS 'B'
                                                   ;*(6)    'B' = 'B'+400   (ADDS 64 DOUBLE WORDS TO 'B')
                                                   ;*(7)    REPEATS STEPS 5 AND 6 UNTIL 'B' IS GREATER THAN THE
                                                   ;*(8)    END OF THE BANK  A+2
                                                   ;*(9)    REPEATS  STEPS 3-8 UNTILL 'A' REACHES THE END OF THE BANK
                                                   ;*(10)   AFTER EXECUTING THE TEST  DATA IS COMPLEMENTED
                                                   ;*       AND STEPS 1-9 ARE REPEATED
                                                   ;REGISTERS ARE USED AS FOLLOWS
                                                   ;R0      TEST DATA
                                                   ;R1      'A'
                                                   ;R2      'B'
                                                   ;R3      BAKPAT
                                                   ;R4      SWAPAT
                                                   ;R5      LAST

                                                   ;NOTE THE PATTERN STARTS AT MTPB24!!!!!!!!!!!!!!!

                                                   ;UIPAR'S
 035056  011100                    1$:     MOV     (R1),R0                 ;V177640        ;READ 'A'
 035060  020004                            CMP     R0,R4                   ;V177642        ;CHECK 'A'
 035062  001401                            BEQ     2$                      ;V177644        ;BR IF OK
 035064  104447                            PERR23                          ;V177646        ;REPORT ERROR

 035066  011200                    2$:     MOV     (R2),R0                 ;V177650        ;READ 'B'
 035070  020003                            CMP     R0,R3                   ;V177652        ;CHECK 'B'
 035072  001401                            BEQ     3$                      ;V177654        ;BR IF OK
 035074  104450                            PERR24                          ;V177656        ;REPORT ERROR

 035076  062702  000400            3$:     ADD     #400,R2                 ;V177660        ;BUMP 'B'
 035102  020205                            CMP     R2,R5                   ;V177664        ;AT END YET?
 035104  101764                            BLOS    1$                      ;V177666        ;BR IF NO

 035106  062701  000002                    ADD     #2,R1                   ;V177670        ;BUMP 'A'
 035112  000137  035116                    JMP     @#MTPB24                ;V177674        ;GOTO V177260
```

```
035116                          MTPB24: SUBTST  <<MTPB24         FAST GALLOP PART B>>
                                ;*************************************************************************
                                ;*SUBTEST        MTPB24  FAST GALLOP PART B
                                ;*************************************************************************

                                        ;SDPAR'S
035116  010411                          MOV     R4,(R1)         ;V172260        ;WRITE 'A'
035120  020105                          CMP     R1,R5           ;V172262        ;DONE?
035122  001001                          BNE     1$              ;V172264        ;BR IF NO
035124  000207                          RETURN                  ;V172266        ;YES - RETURN
035126  000137  035132          1$:     JMP     @#MTPC24        ;V172270        ;GOTO V172360

035132                          MTPC24: SUBTST  <<MTPC24         FAST GALLOP PART C>>
                                ;*************************************************************************
                                ;*SUBTEST        MTPC24  FAST GALLOP PART C
                                ;*************************************************************************

                                        ;KDPAR'S
035132  010102                          MOV     R1,R2           ;V172360        ;RESET 'B' <--- 'A'
035134  011100                          MOV     (R1),R0         ;V172362        ;READ 'A'
035136  020004                          CMP     R0,R4           ;V172364        ;CHECK 'A'
035140  001401                          BEQ     1$              ;V172366        ;BR IF OK
035142  104447                          PERR23                  ;V172370        ;REPORT ERROR
035144  000137  035076          1$:     JMP     @#MTPA24+20     ;V172372        ;GOTO V177660
```

```
  6063 035150                                 MTP025: SUBTST  <<MTP025         INTERRUPT ENABLE TEST>>
                                              ;*****************************************************************************
                                              ;*SUBTEST        MTP025  INTERRUPT ENABLE TEST
                                              ;*****************************************************************************
  6064 035150  005037  002240                         CLR     TSTDAT           ;GENERATE CHECKBITS ON 0,,0
  6065 035154  005037  002242                         CLR     TSTDAT+2
  6066 035160  012737  002240  002272                 MOV     #TSTDAT,SOURCE
  6067 035166  004737  041662                         CALL    CHKGEN
  6068 035172  012737  000003  002074                 MOV     #3,NOPAR                  ;SETUP PARITY ACTION
  6069 035200  012701  002362                         MOV     #TESTADD,R1      ;FIRST TEST ADDRESS
  6070 035204  012737  035244  002264                 MOV     #1$,PARTHERE     ;SETUP TRAP DESTINATION
  6071 035212  004737  035466                         CALL    MTPA25           ;WRITE DATA & CHECKBITS
  6072 035216  104473                                 ECC1INIT                 ;INITIALIZE 1 SELECTED MK11 CSR
  6073 035220  005771  000000                         TST     @(R1)            ;ACCESS LOCATIONS FOR DBE TRAPS
  6074 035224  005771  000002                         TST     @2(R1)
  6075                                                ;NONE - GOOD - ACCESS FOR SBE TRAPS
  6076 035230  104507                                 ENA1SBE                  ;DISABLE TRAPS ON SBE'S FROM 1 SELECTED CSR
  6077 035232  005771  000000                         TST     @(R1)
  6078 035236  005771  000002                         TST     @2(R1)
  6079 035242  000404                                 BR      2$               ;NONE - GOOD - SKIP
  6080 035244  104426                        1$:      READCSR
  6081 035246                                         FATAL   27
  6082 035254  005237  002240                2$:      INC     TSTDAT           ;CHECK FOR CORRECT ACTION ON SBE'S
  6083 035260  004737  035414                         CALL    MTPD25           ;IN ALL 4 BYTES
  6084 035264  012737  000400  002240                 MOV     #400,TSTDAT
  6085 035272  004737  035414                         CALL    MTPD25
  6086 035276  005037  002240                         CLR     TSTDAT
  6087 035302  005237  002242                         INC     TSTDAT+2
  6088 035306  004737  035414                         CALL    MTPD25
  6089 035312  012737  000400  002242                 MOV     #400,TSTDAT+2
  6090 035320  004737  035414                         CALL    MTPD25
  6091
  6092 035324  005037  002242                         CLR     TSTDAT+2         ;CHECK FOR CORRECT ACTION ON DBE'S
  6093 035330  012737  000003  002240                 MOV     #3,TSTDAT        ;IN ALL 4 BYTES
  6094 035336  004737  035436                         CALL    MTPE25
  6095 035342  012737  001400  002240                 MOV     #1400,TSTDAT
  6096 035350  004737  035436                         CALL    MTPE25
  6097 035354  005037  002240                         CLR     TSTDAT
  6098 035360  012737  000003  002242                 MOV     #3,TSTDAT+2
  6099 035366  004737  035436                         CALL    MTPE25
  6100 035372  012737  001400  002242                 MOV     #1400,TSTDAT+2
  6101 035400  004737  035436                         CALL    MTPE25
  6102 035404  104503                                 CLR1CSR                  ;CLEAR 1 SELECTED MK11 CSR
  6103 035406  005037  002074                         CLR     NOPAR            ;INDICATE PARITY ACTION
  6104 035412  000207                                 RETURN
  6105
  6106 035414  004737  035466               MTPD25:   CALL    MTPA25           ;WRITE DATA & CHECKBITS
  6107 035420  104471                                 ECC1DIS                  ;DISABLE ECC ON 1 SELECTED CSR
  6108 035422  004737  035510                         CALL    MTPB25           ;CHECK FOR NO TRAPS
  6109 035426  104507                                 ENA1SBE                  ;DISABLE TRAPS ON SBE'S FROM 1 SELECTED CSR
  6110 035430  004737  035550                         CALL    MTPC25           ;CHECK FOR EXPECTED TRAP
  6111 035434  000207                                 RETURN
```

```
035436  004737  035466          MTPE25: CALL    MTPA25            ;WRITE DATA & CHECKBITS
035442  104471                          ECC1DIS                   ;DISABLE ECC ON 1 SELECTED CSR
035444  004737  035510                  CALL    MTPB25            ;CHECK FOR NO TRAPS
                                        ;ENABLE DBE TRAPS
035450  104473                          ECC1INIT                  ;INITIALIZE 1 SELECTED MK11 CSR
035452  004737  035550                  CALL    MTPC25            ;CHECK FOR EXPECTED TRAP
035456  104507                          ENA1SBE                   ;DISABLE TRAPS ON SBE'S FROM 1 SELECTED CSR
035460  004737  035550                  CALL    MTPC25            ;CHECK FOR EXPECTED TRAP
035464  000207                          RETURN

                                        ;WRITE TSTDAT & TSTDAT+2 & CHECKBITS
035466  104471                  MTPA25: ECC1DIS                   ;DISABLE ECC ON 1 SELECTED CSR
035470  013771  002240  000000          MOV     TSTDAT,@(R1)      ;WRITE FIRST 16 BITS
035476  104475                          CB1CSR                    ;WRITE GENERATED CHECKBITS IN 1 SELECTED CSR
035500  013771  002242  000002          MOV     TSTDAT+2,@2(R1) ;WRITE 2ND 16 BITS & CHECKBITS
035506  000207                          RETURN

                                        ;CHECK FOR NO TRAP OCCURING CONDITION
035510  012737  035530  002264  MTPB25: MOV     #1$,PARTHERE      ;SETUP TRAP DESTINATION
035516  005771  000000                  TST     @(R1)             ;ACCESS LOCATIONS
035522  005771  000002                  TST     @2(R1)
035526  000207                          RETURN                    ;NO TRAP - GOOD - RETURN

035530  104426                  1$:     READCSR
035532  011137  002032                  MOV     (R1),ADDRESS      ;SAVE VIRTUAL ADDRESS
035536  104024                          ERROR   +24
035540                                  SET     HEADER
035546  000207                          RETURN

                                        ;TRAP SHOULD OCCURE TEST
035550  012737  035564  002264  MTPC25: MOV     #1$,PARTHERE      ;SETUP TRAP DESTINATION
035556  005771  000000                  TST     @(R1)             ;ACCESS 1ST LOCATION
035562  000405                          BR      2$                ;NO TRAP - BAD NEWS - SKIP
035564  012737  035614  002264  1$:     MOV     #3$,PARTHERE      ;SETUP TRAP DESTINATION
035572  005771  000002                  TST     @2(R1)            ;ACCESS 2ND LOCATION
035576  104426                  2$:     READCSR                   ;NO TRAP - BAD NEWS
035600  011137  002032                  MOV     (R1),ADDRESS      ;SAVE VIRTUAL ADDRESS
035604  104025                          ERROR   +25
035606                                  SET     HEADER
035614  000207                  3$:     RETURN
```

```
6157 035616                       MTPA26: SUBTST  <<MTPA26        RANDOM DATA (WRITE)>>
                                  ;*************************************************************************
                                  ;*SUBTEST        MTPA26  RANDOM DATA (WRITE)
                                  ;*************************************************************************
6158 035616  000137  035666       1$:     JMP     @#MTPC26        ;V177640        GOTO V172360
6159 035622  010221                       MOV     R2,(R1)+        ;V177644
6160 035624  010321                       MOV     R3,(R1)+        ;V177646
6161 035626  077005                       SOB     R0,1$           ;V177650
6162 035630  000207                       RETURN                  ;V177652
6163
6164 035632                       MTPB26: SUBTST  <<MTPB26        RANDOM DATA (READ)>>
                                  ;*************************************************************************
                                  ;*SUBTEST        MTPB26  RANDOM DATA (READ)
                                  ;*************************************************************************
6165                                      .DSABL  AMA
6166                                      .ENABL  LSB
6167 035632  000137  035666       1$:     JMP     @#MTPC26        ;V177640        GOTO V172360
6168 035636  020221                       CMP     R2,(R1)+        ;V177644
6169 035640  001401                       BEQ     2$              ;V177646
6170 035642  104451                       PERR25                  ;V177650
6171 035644  005127                2$:    COM     (PC)+           ;V177652
6172 035646  000000               RANODD: 0                       ;V177654        FOR ERROR REPORTING
6173 035650  020321                       CMP     R3,(R1)+        ;V177656
6174 035652  001401                       BEQ     3$              ;V177660
6175 035654  104451                       PERR25                  ;V177662
6176 035656  005167  177764       3$:     COM     RANODD          ;V177664
6177 035662  077015                       SOB     R0,1$           ;V177670
6178 035664  000207                       RETURN                  ;V177672
6179                                      .DSABL  LSB
6180                                      .ENABL  AMA
6181
6182 035666                       MTPC26: SUBTST  <<RANDOM NUMBER SUBPROGRAM>>
                                  ;*************************************************************************
                                  ;*SUBTEST        RANDOM NUMBER SUBPROGRAM
                                  ;*************************************************************************
6183                                      ;CALLER MUST SETUP
6184                                      ;       MOV     SEEDLO,R3
6185                                      ;       MOV     SEEDHI,R2
6186                                      ;       MOV     R3,R5
6187                                      ;       MOV     R2,R4
6188 035666  073427  000007               ASHC    #7,R4           ;V172360
6189 035672  060305                       ADD     R3,R5           ;V172364
6190 035674  005504                       ADC     R4              ;V172366
6191 035676  060204                       ADD     R2,R4           ;V172370
6192 035700  062705  001057               ADD     #1057,R5        ;V172372
6193 035704  000240                       NOP                     ;V172376        GOTO V172260
6194
6195 035706                       MTPD26: SUBTST  <<RANDOM NUMBER SUBSUBPROGRAM>>
                                  ;*************************************************************************
                                  ;*SUBTEST        RANDOM NUMBER SUBSUBPROGRAM
                                  ;*************************************************************************
6196 035706  005504                       ADC     R4              ;V172260
6197 035710  062704  047401               ADD     #47401,R4       ;V172262
6198 035714  010503                       MOV     R5,R3           ;V172266
6199 035716  010402                       MOV     R4,R2           ;V172270
6200 035720  000137  035622               JMP     @#MTPA26+4      ;V172272        GOTO V177644
```

```
  6203 035724                       MTP030: SUBTST  <<MT0030        FLUSH OUT DBE'S>>
                                    ;*******************************************************************************
                                    ;*SUBTEST        MT0030  FLUSH OUT DBE'S
                                    ;*******************************************************************************
  6204 035724   011002              1$:     MOV     (R0),R2         ;V177640
  6205 035726   010220                      MOV     R2,(R0)+        ;V177642
  6206 035730   077103                      SOB     R1,1$           ;V177644
  6207 035732   000207                      RETURN                  ;V177646
  6208
  6209 035734                       MTP031: SUBTST  <<MTP031        SOB-A-LONG TEST>>
                                    ;*******************************************************************************
                                    ;*SUBTEST        MTP031  SOB-A-LONG TEST
                                    ;*******************************************************************************
  6210                                      .DSABL  AMA
  6211 035734   000000                      0                       ;MOVE TERMINATOR
  6212 035736   077001              1$:     SOB     R0,1$           ;SOB TILL R0 UNDERFLOWS
  6213 035740   005167  177772              COM     1$              ;WRITE COMPLEMENT OF SOB
  6214 035744   020167  177766              CMP     R1,1$           ;READ & CHECK FOR NOT "SOB R0,DOT"
  6215 035750   001403                      BEQ     2$              ;OK - SKIP
  6216 035752   104454                      PERR30
  6217 035754   010167  177756              MOV     R1,1$
  6218 035760   005167  177752      2$:     COM     1$              ;CORRECT SOB INSTRUCTION
  6219 035764   010200                      MOV     R2,R0           ;REINITIALIZE SOB CONSTANT
  6220                                      ;UPDATE MOVE REGISTERS
  6221 035766   010503                      MOV     R5,R3
  6222 035770   005725                      TST     (R5)+           ;BUMP (SAFELY) BY 2
  6223 035772   010504                      MOV     R5,R4
  6224 035774   020537  002472              CMP     R5,@#LINK1      ;DONE?
  6225 036000   001001                      BNE     3$              ;NO - SKIP
  6226 036002   000207                      RETURN                  ;YES
  6227
  6228 036004   014344              3$:     MOV     -(R3),-(R4)
  6229 036006   001376                      BNE     3$
  6230 036010   000752                      BR      1$
  6231          000056              SOBLENGTH=.-MTP031
  6232                                      .ENABL  AMA
```

```
036012                          MTP032: SUBTST  <<MTP032        WRITE RECOVERY TEST>>
                                ;******************************************************************************
                                ;*SUBTEST        MTP032  WRITE RECOVERY TEST
                                ;******************************************************************************
                                        ;THE TEST ACTUALLY EXECUTED ALREADY IN THE MEMORY UNDER TEST.
                                        ;THIS CODE INSURES THAT IT CHANGED MEMORY TO HAVE
                                        ;1/2 BANK OF #5141        WHICH IS A "COM -(R1)" INSTRUCTION AND
                                        ;1/2 BANK OF #110         WHICH IS A "JMP (R0)"   INSTRUCTION.

036012  012401                  1$:     MOV     (R4)+,R1        ;V177640        ;GET DATA FROM LOWER 1/2 BANK
036014  020102                          CMP     R1,R2           ;V177642        ;IS IT #5141?
036016  001401                          BEQ     2$              ;V177644        ;YES - SKIP
036020  104430                          PERR02                  ;V177646        ;NO - TAKE ERROR TRAP
036022  077305                  2$:     SOB     R3,1$           ;V177650        ;LOOP FOR 1/2 BANK
036024  013703  002472                  MOV     @#LINK1,R3      ;V177652        ;RESTORE LOOP SIZE
036030  012400                  3$:     MOV     (R4)+,R0        ;V177656        ;GET DATA FROM UPPER 1/2 BANK
036032  020005                          CMP     R0,R5           ;V177660        ;IS IT #110?
036034  001401                          BEQ     4$              ;V177662        ;YES - SKIP
036036  104427                          PERR01                  ;V177664        ;NO- TAKE ERROR TRAP
036040  077305                  4$:     SOB     R3,3$           ;V177666        ;LOOP FOR 1/2 BANK
036042  000207                          RETURN
```

```
036044                         MTP033: SUBTST  <<MTP033        BRANCH GOBBLE TEST>>
                               ;*****************************************************************************
                               ;*SUBTEST        MTP033  BRANCH GOBBLE TEST
                               ;*****************************************************************************
                                        .DSABL  AMA
036044  000000                          0                       ;MOVE TERMINATOR
036046  000000                 BGTEST:  0                       ;TEST WORD (TWO BYTES)
036050  000261                 BRGOBB:  SEC                     ;SET CARRY (TO BE ADDED TO ''BGTEST'')
036052  105511                          ADCB    (R1)            ;INCREMENT LOW BYTE OF ''BGTEST''
036054  100402                          BMI     1$              ;BRANCH WHEN BIT7 IS SET
036056  105212                          INCB    (R2)            ;INCREMENT HIGH BYTE OF ''BGTEST''
036060  000773                          BR      BRGOBB          ;LOOP 128 TIMES


                                        ;NOW CHECK FOR CORRECT CONDITION CODES
036062  102401                 1$:      BVS     2$              ;BR IF V-BIT SET (SHOULD BE)
036064  104461                          PERR35                  ;NO - REPORT ERROR AND ABORT TEST
                                                                ;COND CODES NOT EQUAL TO 1010
036066  000242                 2$:      CLV                     ;CLEAR V-BIT
036070  105212                          INCB    (R2)            ;INCREMENT HIGH BYTE OF ''BGTEST'' ONCE MORE
036072  103402                          BCS     3$              ;BR IF C-BIT SET (SHOULD NOT BE)
036074  102001                          BVC     3$              ;BR IF V-BIT CLEAR (SHOULD NOT BE)
036076  100401                          BMI     4$              ;BR IF N-BIT SET (SHOULD BE)
036100  104461                 3$:      PERR35                  ;NO - REPORT ERROR AND ABORT TEST
                                                                ;COND CODES NOT EQUAL TO 1010


                                        ;UPDATE TEST POINTERS
036102  010701                 4$:      MOV     PC,R1
036104  162701  000036         5$:      SUB     #5$-BGTEST,R1
036110  010102                          MOV     R1,R2
036112  005202                          INC     R2

                                        ;UPDATE MOVE REGISIERS
036114  010503                          MOV     R5,R3
036116  005725                          TST     (R5)+           ;BUMP (SAFELY) BY 2
036120  010504                          MOV     R5,R4


                                        ;DONE?
036122  020537  002472                  CMP     R5,@#LINK1      ;DONE?
036126  001001                          BNE     6$              ;NO - SKIP
036130  000207                          RETURN                  ;YES - RETURN

                                        ;MOVE CODE 1 LOCATION
036132  014344                 6$:      MOV     -(R3),-(R4)
036134  001376                          BNE     6$
036136  005011                          CLR     (R1)            ;CLEAR TEST WORD ''BGTEST''
036140  000743                          BR      BRGOBB          ;RUN MOVED CODE AGAIN
        000076                 GBLENGTH=.-MTP033
                                        .ENABL  AMA
```

```
   6326 036142                     MTP034: SUBTST  <<MTP034        SOFT ERROR - BACKROUND PATTERN TEST>>
                                   ;*****************************************************************************
                                   ;*SUBTEST        MTP034  SOFT ERROR - BACKROUND PATTERN TEST
                                   ;*****************************************************************************
   6327 036142  010220             1$:     MOV     R2,(R0)+                ;V177640
   6328 036144  077102                     SOB     R1,MTP034               ;V177642
   6329 036146  000207                     RETURN                          ;V177644
   6330 036150  012401             2$:     MOV     (R4)+,R1                ;V177646
   6331 036152  020102                     CMP     R1,R2                   ;V177650
   6332 036154  001402                     BEQ     3$                      ;V177652
   6333 036156  104430                     PERRO2                          ;V177654
   6334 036160  000240                     NOP                             ;V177656
   6335 036162  077306             3$:     SOB     R3,2$                   ;V177660
   6336 036164  000207                     RETURN                          ;V177662
```

```
036166                                   MTP035:SUBTST   <<MTP035        WORST CASE NOISE PARITY TEST>>
                                         ;*************************************************************************
                                         ;*SUBTEST        MTP035  WORST CASE NOISE PARITY TEST
                                         ;*************************************************************************
036166  012737  000003  002074                   MOV     #3,NOPAR        ;SET PARITY TRAPS TO RETURN TO "PARTHERE"

036174                                           FOR R0 := #FIRST TO #LAST BY #4000
036200  012737  000005  002144                     MOV   #BIT2!BIT0,CSR  ;SET WRITE WRONG PARITY & PAR. TRAPS INTO CSR
036206  104425                                     LOADCSR
036210  012737  036244  002264                     MOV   #1$,PARTHERE
036216  011010                                     MOV   (R0),(R0)       ;WWP TEST LOCATION
036220  005710                                     TST   (R0)
036222  010037  002032                             MOV   R0,ADDRESS
036226  104050                                     ERROR +50
036230  004737  054644                             CALL  PERBNK
036234  032763  002000  002626                     BIT   #BIT10,CONFIG+2(R3)
036242  001002                                     BNE   2$
036244  104426                             1$:     READCSR
036246  104512                                     ERRGEN

036250  104503                             2$:     CLR1CSR
036252  011010                                     MOV   (R0),(R0)       ;CLEAR WRONG PARITY IN MEMORY
036254  012737  000001  002144                     MOV   #BIT0,CSR
036262  104425                                     LOADCSR
036264  012737  036276  002264                     MOV   #3$,PARTHERE
036272  005710                                     TST   (R0)
036274  000405                                     BR    4$
036276  010037  002032                     3$:     MOV   R0,ADDRESS
036302  104050                                     ERROR +50
036304  004737  054644                             CALL  PERBNK
036310                                     4$:     END; OF FOR

036322  005037  002074                           CLR     NOPAR           ;RESET PARITY TRAP ACTION
036326  000207                                   RETURN
```

```
6370                                            .SBTTL  MISC SUBROUTINES
6371
6372 036330                             REGCOPY:SUBTST  <<SUBR  COPY R0 TO R4,R1 TO R3, & R2 TO R5>>
                                        ;*********************************************************************
                                        ;*SUBTEST       SUBR    COPY R0 TO R4,R1 TO R3, & R2 TO R5
                                        ;*********************************************************************
6373 036330  010004                             MOV     R0,R4
6374 036332  010103                             MOV     R1,R3
6375 036334  010205                             MOV     R2,R5
6376 036336  000207                             RETURN
6377
6378 036340                             FLIPWARN:SUBTST <<FLIP WARNING CONSTANTS IN WORST CASE NOISE TESTS>>
                                        ;*********************************************************************
                                        ;*SUBTEST       FLIP WARNING CONSTANTS IN WORST CASE NOISE TESTS
                                        ;*********************************************************************
6379 036340                                     PUSH    R0
6380 036342  005237  002556                     INC     FLIPLOC
6381 036346  042737  177774  002556             BIC     #^C3,FLIPLOC
6382 036354  022737  000001  002556             CMP     #1,FLIPLOC
6383 036362  001414                             BEQ     1$
6384 036364  022737  000002  002556             CMP     #2,FLIPLOC
6385 036372  001413                             BEQ     2$
6386 036374  022737  000003  002556             CMP     #3,FLIPLOC
6387 036402  001414                             BEQ     3$
6388 036404  005000                             CLR     R0
6389 036406  013704  002554                     MOV     ONES,R4
6390 036412  000414                             BR      4$
6391 036414                             1$:     CLEAR   R0,R4
6392 036420  000411                             BR      4$
6393 036422  012700  000401             2$:     MOV     #401,R0
6394 036426  013704  002554                     MOV     ONES,R4
6395 036432  000404                             BR      4$
6396 036434  012700  000401             3$:     MOV     #401,R0
6397 036440  012704  000401                     MOV     #401,R4
6398 036444  010037  027116             4$:     MOV     R0,WARN2
6399 036450  010037  027132                     MOV     R0,WARN3
6400 036454  010037  027156                     MOV     R0,WARN4
6401 036460  010037  027172                     MOV     R0,WARN5
6402 036464                                     POP     R0
6403 036466  000207                             RETURN
```

```
 6405 036470                              BACKGND:SUBTST  <<SUBR  WRITE BACKGROUND>>
                                          ;*******************************************************************************
                                          ;*SUBTEST        SUBR    WRITE BACKGROUND
                                          ;*******************************************************************************
 6406                                             ;WRITES DATA FROM R2
 6407 036470  104415                              SAVREG
 6408 036472  012700  060000                      MOV     #FIRST,R0
 6409 036476  012701  040000                      MOV     #SIZE,R1
 6410 036502  022737  000001  003710              CMP     #1,PROTYP
 6411 036510  001415                              BEQ     WARN6B
 6412 036512  012737  000207  027000  WARN6A:  MOV     #207,MTP000+4               ;WARNING PUTTING ''RETURN'' AFTER WRITE
 6413 036520  012737  026774  002254            MOV     #MTP000,SUPDOADD
 6414 036526  004737  026602                      CALL    SUPD03
 6415 036532  012737  000240  027000            MOV     #240,MTP000+4               ;RESTORE 'NOP' AFTER WRITE
 6416 036540  104416                              RESREG
 6417 036542  000207                              RETURN
 6418 036544                              WARN6B: BMOV    MTP000
 6419 036552  012737  000207  177644  WARN6:   MOV     #207,UIPAR2                 ;WARNING PUTTING ''RETURN'' INSTRUCTION AFTER WRITE
 6420 036560  004737  026424                      CALL    SUPD01
 6421 036564  104416                              RESREG
 6422 036566  000207                              RETURN
```

```
  6425 036570                               PCONFIG:SUBTST  <<SUBR  PRINT CONFIGURATION MAP>>
                                            ;*******************************************************************************
                                            ;*SUBTEST         SUBR    PRINT CONFIGURATION MAP
                                            ;*******************************************************************************
  6426 036570                                       PUSH    TKVEC,TKVEC+2,R0
  6427 036602  010637  037070                       MOV     SP,PCONFS                       ;SAVE LAST GOOD SP
  6428 036606  012737  037036  000060               MOV     #PCONF2,TKVEC
  6429 036614  012737  000340  000062               MOV     #340,TKVEC+2
  6430 036622  017700  143756                       MOV     @$TKB,R0                        ;KILL ANY OLD INTERRUPT
  6431 036626  042737  000200  177776               BIC     #BIT7,PSW                       ;LOWER CPU PRIORITY TO 140
  6432 036634  052777  000100  143740               BIS     #BIT6,@$TKS                     ;ENABLE KEYBOARD INTERRUPTS
  6433
  6434 036642                                       TYPE    MSG001
  6435 036646                                       TYPE    MSG002
  6436 036652                                       TYPE    MSG003
  6437 036656  022737  000060  002526               CMP     #60,LASTBANK
  6438 036664  002006                               BGE     NOOJ
  6439                                              ;IF FAT PAPER ON TERMINAL GOTO 1$
  6440 036666                                       IF #SW4 SET.IN @SWR THEN JUMPTO PCONF1
  6441 036702  012700  000074               NOOJ:   MOV     #60.,R0
  6442 036706  010004                               MOV     R0,R4
  6443 036710                                       CLEAR   R1,R3
  6444 036714                                       TYPE    MSG004
  6445 036720  004737  037072                       CALL    TCONFIG                         ;GO TYPE CONFIGURATION (1ST HALF)
  6446 036724  022737  000060  002526               CMP     #60,LASTBANK
  6447 036732  002041                               BGE     PCONF2
  6448 036734                                       TYPE    $CRLF
  6449 036740                                       TYPE    MSG017                          ;PRINT SPACE(S)
  6450 036744                                       TYPE    MSG011
  6451 036750                                       TYPE    $CRLF
  6452 036754                                       TYPE    MSG017                          ;PRINT SPACE(S)
  6453 036760                                       TYPE    MSG012
  6454 036764  012701  000360                       MOV     #60.*2*2,R1
  6455 036770  010103                               MOV     R1,R3
  6456 036772  004737  037072                       CALL    TCONFIG
  6457 036776  000417                               BR      PCONF2
  6458
  6459 037000  012700  000170               PCONF1: MOV     #120.,R0
  6460 037004  010004                               MOV     R0,R4
  6461 037006                                       CLEAR   R1,R3
  6462 037012                                       TYPE    MSG014                          ;SPACE
  6463 037016                                       TYPE    MSG011
  6464 037022                                       TYPE    MSG004
  6465 037026                                       TYPE    MSG012
  6466 037032  004737  037072                       CALL    TCONFIG
  6467
  6468 037036  013706  037070               PCONF2: MOV     PCONFS,SP                       ;RESTORE STACK
  6469 037042  042777  000100  143532               BIC     #BIT6,@$TKS
  6470 037050  117700  143530                       MOVB    @$TKB,R0                        ;READ CHAR TO KILL FLAG
  6471 037054                                       POP     R0,TKVEC+2,TKVEC
  6472 037066  000207                               RETURN
  6473
  6474 037070  000000                       PCONFS: 0                                       ;STACK SAVED HERE!
  6477 037072                                     SUBTST  <<SUBR   TYPE CONFIGURATION>>
                                        ;*******************************************************************************
                                        ;*SUBTEST          SUBR    TYPE CONFIGURATION
                                        ;*******************************************************************************
  6478                                  ;*******************************************************************************
  6479                                  ;CALL:   MOV     #N,R0            ;N=NUMBER OF CHARACTERS
  6480                                  ;        MOV     R0,R4            ;BACKUP
  6481                                  ;        MOV     #K.R1            ;INDEX CONSTANT
  6482                                  ;        MOV     R1,R3            ;BACKUP
  6483                                  ;        CALL    TCONFIG          ;ACTUAL CALL
  6484                                  ;        RETURN                   ;ONLY RETURN
  6485                                  ;*******************************************************************************
  6486
  6487                                          ;************
  6488                                          ;** ERROR **
  6489                                          ;************
  6490 037072                           TCONFIG:TYPE    MSG005
  6491 037076  032761  000001  002624  1$:     BIT     #BIT0,CONFIG(R1)            ;ERROR ON THIS BANK?
  6492 037104  001403                           BEQ     2$                          ;NO - SKIP
  6493 037106                                   TYPE    MSG013                      ;PRINT ''X''
  6494 037112  000402                           BR      3$
  6495 037114                           2$:     TYPE    MSG014                      ;PRINT SPACE
  6496 037120  062701  000004           3$:     ADD     #4,R1                       ;BUMP POINTER
  6497 037124  077014                           SOB     R0,1$                       ;LOOP TILL DONE
  6498 037126  010400                           MOV     R4,R0
  6499 037130  010301                           MOV     R3,R1
  6500
  6501                                          ;***********
  6502                                          ;** CPU'S **
  6503                                          ;***********
  6504 037132                                   TYPE    MSG008
  6505 037136  016105  002624           4$:     MOV     CONFIG(R1),R5
  6506 037142  006205                           ASR     R5                          ;GET CPU BITS
  6507 037144  042705  177760                   BIC     #^C17,R5                    ;CLEAR NON INTERESTING BITS
  6508 037150  005705                           TST     R5                          ;IS THERE ANYTHING THERE?
  6509 037152  001003                           BNE     8$                          ;YES - BRANCH.
  6510 037154  112705  000040                   MOVB    #' ,R5                      ;NO - MOVE A BLANK INTO R5
  6511 037160  000402                           BR      9$                          ;BRANCH OVER NEXT INSTRUCTION
  6517 037162  062705  000060           8$:     ADD     #60,R5                      ;MAKE ASCII
  6518 037166  110537  071234           9$:     MOVB    R5,MSG015                   ;PLUG INTO MEMORY
  6519 037172                                   TYPE    MSG015
  6520 037176  062701  000004                   ADD     #4,R1                       ;BUMP POINTER
  6521 037202  077023                           SOB     R0,4$                       ;LOOP TILL DONE
  6522 037204  010400                           MOV     R4,R0
  6523 037206  010301                           MOV     R3,R1
```

```
                                        ;******************
                                        ;** INTERLEAVE **
                                        ;******************
037210                                  TYPE    MSG007
                                        ;THIS IS AN ENTRY POINT FROM ERROR REPORTS
037214  032761  010000  002626  TCFIG1: BIT     #BIT12,CONFIG+2(R1)
037222  001014                          BNE     1$
037224  032761  000002  002624          BIT     #BIT1,CONFIG(R1)        ;IS THERE ANY MEMORY HERE?
037232  001004                          BNE     18$                     ;BRANCH IF MEMORY PRESENT.
037234  112737  000040  071234          MOVB    #' ,MSG015              ;MOVE A BLANK IN TO BE PRINTED
037242  000424                          BR      16$                     ;BRANCH TO TYPE ROUTINE
037244  112737  000055  071234  18$:    MOVB    #'-,MSG015
037252  000420                          BR      16$
037254  016105  002624          1$:     MOV     CONFIG(R1),R5
037260  042705  007777                  BIC     #^C170000,R5            ;GET CSR INTERLEAVE
037264  000305                          SWAB    R5
037266  072527  177774                  ASH     #-4,R5
037272  022705  000012                  CMP     #10.,R5
037276  100002                          BPL     2$
037300  062705  000007                  ADD     #7,R5
037304  062705  000060          2$:     ADD     #60,R5                  ;MAKE ASCII
037310  110537  071234                  MOVB    R5,MSG015               ;PLUG INTO MEMORY
037314                          16$:    TYPE    MSG015
037320                                  IF NOTAB NE #0 THEN $RETURN
037330  062701  000004                  ADD     #4,R1                   ;BUMP POINTER
037334  077051                          SOB     R0,TCFIG1               ;LOOP TILL DONE
037336  010400                          MOV     R4,R0
037340  010301                          MOV     R3,R1

                                        ;*******************
                                        ;** MEMORY TYPE **
                                        ;*******************
                                        .ENABL  LSB
037342                                  TYPE    MSG009
037346  033761  002104  002624  TCFIG2: BIT     CPUBIT,CONFIG(R1)
037354  001447                          BEQ     17$
037356  016105  002626                  MOV     CONFIG+2(R1),R5
037362  000305                          SWAB    R5                      ;GET MEMORY TYPE
037364  042705  177770                  BIC     #^C7,R5                 ;CLEAR NON INTERESTING BITS
037370  005705                          TST     R5
037372  001440                          BEQ     17$
037374  032705  000004                  BIT     #BIT2,R5
037400  001004                          BNE     4$
037402  112737  000102  071234          MOVB    #'B,MSG015
037410  000434                          BR      8$
037412  032705  000002          4$:     BIT     #BIT1,R5
037416  001013                          BNE     6$
037420  032705  000001                  BIT     #BIT0,R5
037424  001004                          BNE     5$
037426  112737  000115  071234          MOVB    #'M,MSG015
037434  000422                          BR      8$
037436  112737  000113  071234  5$:     MOVB    #'K,MSG015
037444  000416                          BR      8$
037446  032705  000001          6$:     BIT     #BIT0,R5
037452  001004                          BNE     7$
037454  112737  000114  071234          MOVB    #'L,MSG015
037462  000407                          BR      8$
```

```
037464  112737  000120  071234  7$:      MOVB    #'P,MSG015
037472  000403                           BR      8$
037474  112737  000040  071234  17$:     MOVB    #' ,MSG015
037502                          8$:      TYPE    MSG015
037506                                   IF NOTAB NE #0 THEN $RETURN
037516  062701  000004                   ADD     #4,R1                        ;BUMP POINTER
037522  077067                           SOB     R0,TCFIG2                    ;LOOP TILL DONE
037524  010400                           MOV     R4,R0
037526  010301                           MOV     R3,R1
                                         .DSABL  LSB

                                         ;*********
                                         ;** CSR **
                                         ;*********
037530                                   TYPE    MSG016
037534  112737  000040  071234  TCFIG3:  MOVB    #' ,MSG015
037542  016105  002624                   MOV     CONFIG(R1),R5
037546  032705  000002                   BIT     #BIT1,R5
037552  001414                           BEQ     16$
037554  042705  170377                   BIC     #^C7400,R5
037560  000305                           SWAB    R5
037562  022705  000012                   CMP     #10.,R5
037566  100002                           BPL     10$
037570  062705  000007                   ADD     #7,R5
037574  062705  000060          10$:     ADD     #60,R5                       ;MAKE ASCII
037600  110537  071234                   MOVB    R5,MSG015                    ;PLUG INTO MEMORY
037604                          16$:     TYPE    MSG015
037610                                   IF NOTAB NE #0 THEN $RETURN
037620  062701  000004                   ADD     #4,R1                        ;BUMP POINTER
037624  077035                           SOB     R0,TCFIG3                    ;LOOP TILL DONE
037626  010400                           MOV     R4,R0
037630  010301                           MOV     R3,R1

                                         ;***************
                                         ;** PROTECTED **
                                         ;***************
037632                                   TYPE    MSG010
037636  105761  002624          11$:     TSTB    CONFIG(R1)                   ;BANK PROTECTED?
037642  100004                           BPL     12$                          ;NO - SKIP
037644  112737  000120  071234           MOVB    #'P,MSG015
037652  000407                           BR      13$
037654  032761  000100  002624  12$:     BIT     #BIT6,CONFIG(R1)             ;PROTECTED REGION [illegible]?
037662  001406                           BEQ     14$                          ;NO - SKIP
037664  112737  000111  071234           MOVB    #'I,MSG015
037672                          13$:     TYPE    MSG015
037676  000402                           BR      15$
037700                          14$:     TYPE    MSG014                       ;PRINT SPACE
037704  062701  000004          15$:     ADD     #4,R1                        ;BUMP POINTER
037710  077026                           SOB     R0,11$                       ;LOOP TILL DONE
037712  010400                           MOV     R4,R0
037714  010301                           MOV     R3,R1
037716  000207                           RETURN
```

```
                                                .SBTTL  TRAP    PARITY ERROR HANDLER
                                        ;*******************************************************************************
                                        ;VECTOR TO HERE FROM TRAPS TO 114
                                        ;IGNORE ERRORS BUT COUNT IF NOPAR FLAG = 1.
                                        ;*******************************************************************************
                                        ;
                                        ;       CODE    ACTION
                                        ;
                                        ;       ----    ------
                                        ;         0     PRINT UNEXPECTED PARITY TRAP
                                        ;         1     COUNT ERROR
                                        ;         2     SET "ABORT" / SETUP "BADPC" / RETURN VIA PCBUMP
                                        ;         3     RETURN VIA "PARTHERE"
                                        ;
037720  022737  000001  002074  PARITY: CMP     #1,NOPAR                        ;COUNTING PARITY ERRORS?
037726  001003                          BNE     1$                              ;NO - SKIP
037730  005237  002070                  INC     PARCNT                          ;PARITY ERROR COUNTER + 1
037734  000002                          RTI
037736  022737  000002  002074  1$:     CMP     #2,NOPAR                        ;ACTION CODE = 2 ?
037744  001013                          BNE     2$                              ;NO - SKIP
037746                                  SET     ABORTFLAG                       ;YES
037754  004737  040126                  CALL    BADSTACK                        ;FIND BAD SP,PC,PSW OFF STACK
037760  063716  002276                  ADD     PCBUMP,(SP)                     ;UPDATE RETURN PC
037764  042766  000004  000002          BIC     #BIT2,2(SP)                     ;SHOW FAILURE BY .NE.
037772  000002                          RTI
037774  022737  000003  002074  ?$:     CMP     #3,NOPAR                        ;ACTION CODE = 3 ?
040002  001003                          BNE     3$                              ;NO - SKIP
040004  013716  002264                  MOV     PARTHERE,(SP)
040010  000002                          RTI
040012  004737  040126          3$:     CALL    BADSTACK                        ;FIND BAD SP,PC,PSW OFF STACK
040016                                  FATAL   32
```

```
                                                 .SBTTL  TRAP    NON-EXISTANT MEMORY (HOLES) HANDLER
                                         ;*******************************************************************************
                                         ;VECTOR TO HERE (SOMETIMES) FROM TRAPS TO 4
                                         ;       CODE IN NONEM DETERMINES ACTION AS FOLLOWS:
                                         ; 1)    IGNORE ERRORS BUT COUNT IF NONEM (NO NON-EXISTANT MEMORY) FLAG = 1.
                                         ; 2)    TO EXIT PATTERN 0 DURING SIZING IF NON-EXIST MEM ERROR
                                         ;*******************************************************************************

040024  022737  000001  002076  NONEXIST:CMP     #1,NONEM                ;COUNTING NON-EXISTANT MEMORY ERRORS?
040032  001011                           BNE     2$                      ;NO - SKIP
040034  005237  002066                   INC     NEMCNT                  ;BUMP NON-EXISTANT MEMORY COUNTER
040040  022737  000001  002066           CMP     #1,NEMCNT               ;FIRST ERROR?
040046  001002                           BNE     1$                      ;NO - SKIP
040050  010037  002032                   MOV     R0,ADDRESS              ;ASSUME R0 CONTAINS THE ADDRESS ACCESSED
040054  000002                  1$:      RTI
040056  005237  002066          2$:      INC     NEMCNT                  ;BUMP NON-EXISTANT MEMORY COUNTER
040062  012701  000001                   MOV     #1,R1                   ;DUMMY UP R1 FOR A FORCED SOB EXIT
040066  000002                           RTI

                                         ;*******************************************************************************
                                         .SBTTL  TRAP    TIMEOUT (TRAP TO 4) HANDLER
040070  004737  040126          TIMEOUT:CALL     BADSTACK                ;FIND BAD SP,PC,PSW OFF STACK
040074                                   FATAL   6
                                         ;*******************************************************************************
                                         .SBTTL  TRAP    MEMORY MANAGEMENT (TRAP TO 250) HANDLER
040102  004737  040126          MMTRAP: CALL     BADSTACK                ;FIND BAD SP,PC,PSW OFF STACK
040106                                   FATAL   7
                                         .SBTTL  TRAP    RESERVED INSTRUCTION HANDLER
040114  004737  040126          PDP1105:CALL     BADSTACK                ;FIND BAD SP,PC,PSW OFF STACK
040120                                   FATAL   5


040126                          BADSTACK:SUBTST <<FIND BAD SP, PC, & PSW FROM STACK>>
                                ;*******************************************************************************
                                ;*SUBTEST        FIND BAD SP, PC, & PSW FROM STACK
                                ;*******************************************************************************
040126  010637  002024                   MOV     SP,BADSP
040132  062737  000002  002024           ADD     #2,BADSP
040140  016637  000002  002020           MOV     2(SP),BADPC
040146  016637  000004  002030           MOV     4(SP),BADPSW
040154  000207                           RETURN
```

```
                                                 .SBTTL  TRAP    KERNEL TRAP HANDLER
                                        ;*****************************************************************
                                        ;KERNEL IS A TRAP THAT COMES HERE
                                        ;*****************************************************************

040156  042766  140000  000002  $KERNEL:        BIC     #140000,2(SP)
040164  000002                                  RTI
                                        ;*****************************************************************
                                                .SBTTL  TRAP    ENERGIZE TRAP HANDLER
040166  052737  000001  177572  $ENERGIZE:BIS   #BIT0,MMR0
040174  000002                                  RTI
                                        ;*****************************************************************
                                                .SBTTL  TRAP    DEENERGIZE TRAP HANDLER
040176  042737  000001  177572  $DEENERGIZE:BIC #BIT0,MMR0
040204  000002                                  RTI
                                        ;*****************************************************************
                                                .SBTTL  TRAP    CACHON TRAP HANDLER
040206  005737  002514          $CACHN: TST     CACHKN          ;IS THERE A CACHE
040212  001406                          BEQ     1$              ;NO - RETURN
040214  013737  002514  177746          MOV     CACHKN,CONTRL   ;SETUP CACHE AS PER CONSTANT (USUALLY 1 = FULLY ON)
040222  052737  000001  177746          BIS     #BIT0,CONTRL    ;DISABLE TRAPS (BUT NOT ABORTS)
040230  000002                  1$:     RTI
                                        ;*****************************************************************
                                                .SBTTL  TRAP    CACHOFF TRAP HANDLER
040232  005737  002514          $CACHF: TST     CACHKN          ;IS THERE A CACHE?
040236  001403                          BEQ     1$              ;NO - RETURN
                                        ;DISABLE TRAPS (NOT ABORTS), FORCE MISSES, FLUSH, BYPASS
040240  053737  002520  177746          BIS     CACHKF,CONTRL
040246  000002                  1$:     RTI
```

```
                                                 .SBTTL  TRAP    LOAD CSR TRAP HANDLER
                                                 ;LOAD CORRECT CSR WITH DATA IN CSR
                                                 ;PROGRAM CSR'S ASSERT INHIBIT MODE POINTER WHEN LOADED
 040250                                  $LOADC: PUSH    R0,R1                           ;SAVE REGISTERS
 040254  013700  002146                          MOV     CSRNO,R0                        ;CREATE CSR ADDRESS
 040260                                          IF INHECC IS TRUE THEN GOTO 3$  ;DON'T WANT INH. MODE POINTER ON
 040266  005737  002502                          TST     PGMCSR                          ;PROGRAM IN INTERLEAVED SPACE?
 040272  100007                                  BPL     1$                              ;BRANCH IF NOT
 040274  113701  002503                          MOVB    PGMCSR+1,R1                     ;CHECK SECOND CSR
 040300  042701  177740                          BIC     #^C37,R1                        ;CLEAR UNNECESSARY BITS
 040304  020137  002146                          CMP     R1,CSRNO                        ;IS THIS THE CURRENT CSR?
 040310  001404                                  BEQ     2$                              ;BRANCH IF IT IS
 040312  123737  002502  002146  1$:             CMPB    PGMCSR,CSRNO                    ;IS THIS THE CURRENT CSR?
 040320  001003                                  BNE     3$                              ;BRANCH IF NOT
 040322  052737  020000  002144  2$:             BIS     #BIT13,CSR                      ;SET THE INHIBIT MODE POINTER TO  1ST 16K
 040330  013760  002144  172100  3$:             MOV     CSR,CSRADD(R0)                  ;LOAD THE CSR
 040336                                          POP     R1,R0                           ;RESTORE REGISTERS
 040342  000002                                  RTI

                                                 .SBTTL  TRAP    READ CSR TRAP HANDLER
                                                 ;READ THE CORRECT CSR INTO LOCATIONS CSR
 040344                                  $READC: PUSH    R0
 040346  013700  002146                          MOV     CSRNO,R0
 040352  016037  172100  002144                  MOV     CSRADD(R0),CSR                  ;READ IT
 040360                                          POP     R0
 040362  000002                                  RTI
```

```
                                              .SBTTL  TRAP    TEST (R1) & READ CSR CAREFULLY
040364                               $TSTRD:  PUSH    R0,R2,R3
040372  012700  172100                        MOV     #CSRADD,R0               ;CREATE CSR ADDRESS
040376  063700  002146                        ADD     CSRNO,R0
040402  005002                                CLR     R2
040404  005737  002502                        TST     PGMCSR
040410  100007                                BPL     1$
040412  113703  002503                        MOVB    PGMCSR+1,R3
040416  042703  000200                        BIC     #BIT7,R3
040422  020337  002146                        CMP     R3,CSRNO
040426  001404                                BEQ     2$
040430  123737  002502  002146  1$:           CMPB    PGMCSR,CSRNO
040436  001002                                BNE     3$
040440  012702  020000          2$:           MOV     #BIT13,R2
040444  022737  000001  003710  3$:           CMP     #1,PROTYP        ;IS THIS AN 11/44?
040452  001403                                BEQ     4$               ;BRANCH IF IT IS
040454  004737  040542                        CALL    TSTRD1
040460  000405                                BR      5$
040462                          4$:           BMOV    TSTRD1
040470  004737  177640                        CALL    FASTCITY                 ;CALL TO THE USER INSTRUCTION PAR'S
                                              ;IF SINGLE BIT ERROR ONLY - SET CARRY BIT
040474                          5$:           POP     R3,R2,R0
040502                                        IF #BIT4 SET.IN CSR AND #BIT15 OFF.IN CSR
040522  052766  000001  000002                  BIS   #BIT0,2(SP)
040530                                        ELSE
040532  042766  000001  000002                  BIC   #BIT0,2(SP)
040540                                        END ;OF IF #BIT4
040540  000002                                RTI

040542  010210                       TSTRD1:  MOV     R2,(R0)                  ;V177640
040544                                        TESTAREA                         ;V177642 ;ENTER SUPERVISOR MODE
040552  105711                                TSTB    (R1)                     ;V177646
040554  042737  140000  177776                BIC     #BIT15!BIT14,PSW         ;V177650
040562  011037  002144                        MOV     (R0),CSR                 ;V177656
040566  000207                                RETURN                           ;V177662
```

```
                                                .SBTTL  TRAP    ECC DISABLE ALL CSR'S TRAP HANDLER
040570  012737  000002  002144  $ECCDIS:MOV     #BIT1,CSR
040576  004737  041314                  CALL    CSROUT
040602  000002                          RTI
                                        .SBTTL  TRAP    ECC DISABLE OF 1 SELECTED CSR TRAP HANDLER
040604  012737  000002  002144  $ECC1DIS:MOV    #BIT1,CSR
040612  104425                          LOADCSR
040614  000002                          RTI
                                        .SBTTL  TRAP    INITIALIZE ALL CSR'S TRAP HANDLER
040616  012737  000001  002144  $ECCINIT:MOV    #BIT0,CSR
040624  004737  041314                  CALL    CSROUT
040630  000002                          RTI
                                        .SBTTL  TRAP    INITIALIZE 1 SELECTED CSR TRAP HANDLER
040632  012737  000001  002144  $ECC1INIT:MOV   #BIT0,CSR
040640  104425                          LOADCSR
040642  000002                          RTI
                                        .SBTTL  TRAP    ENABLE SBE PARITY TRAPS ON ALL CSR'S
040644  012737  000003  002144  $ENASBE:MOV     #BIT0!BIT1,CSR
040652  004737  041314                  CALL    CSROUT
040656  000002                          RTI
                                        .SBTTL  TRAP    ENABLE SBE PARITY TRAPS ON 1 SELECTED CSR
040660  012737  000003  002144  $ENA1SBE:MOV    #BIT0!BIT1,CSR
040666  104425                          LOADCSR
040670  000002                          RTI
                                        .SBTTL  TRAP    WRITE CHECKBITS THRU ALL CSR'S TRAP HANDLER
040672  013737  002274  002144  $CBCSR: MOV     CHECK,CSR               ;BITS 11-5
040700  052737  000006  002144          BIS     #BIT1!BIT2,CSR          ;CHECK MODE
040706  004737  041314                  CALL CSROUT
040712  000002                          RTI
                                        .SBTTL  TRAP    WRITE CHECKBITS THRU 1 SELECTED CSR TRAP HANDLER
040714  013737  002274  002144  $CB1CSR:MOV     CHECK,CSR               ;BITS 11-5
040722  052737  000006  002144          BIS     #BIT1!BIT2,CSR          ;CHECK MODE
040730  104425                          LOADCSR
040732  000002                          RTI
```

```
                                                 .SBTTL  TRAP    WAS THERE A SBE ON ANY CSR TRAP HANDLER
     040734                           $WASSBE:PUSH    R1,R4
     040740  013701  002216                   MOV     TOTCSRS,R1        ;GET CSR'S BYTE
     040744  005004                           CLR     R4
     040746                                   BEGIN LWSBE
     040746                                     FOR CSRNO := #0 TO #36 BY #2
     040752  006301                               ASL           R1
     040754                                       ON.ERROR
     040756  104426                                 READCSR
     040760                                         IF #BIT4 SET.IN CSR
     040770                                           SET     R4
     040774                                           LEAVE LWSBE
     040776                                         END ;OF IF #BIT4
     040776                                       END ;OF ON.ERROR
     040776                                       IF R1 EQ #0 THEN LEAVE LWSBE
     041002                                     END ;OF FOR CSRNO
     041020                                   END LWSBE
     041020  006004                           ROR     R4                ;SEi C BIT FOR ERROR
     041022                                   POP     R4,R1
     041026                                   ON.ERROR
     041030  052766  000001  000002             BIS     #BIT0,2(SP)
     041036                                   ELSE
     041040  042766  000001  000002             BIC     #BIT0,2(SP)
     041046                                   END ;OF ON.ERROR
     041046  000002                           RTI
                                          .SBTTL  TRAP    WAS THERE A SBE IN 1 SELECTED CSR TRAP HANDLER
                                          ;ON RETURN IF CARRY IS SET THERE WAS A SBE
     041050  104426                   $WAS1SBE:READCSR
     041052  042766  000001  000002           BIC     #BIT0,2(SP)       ;CLR C BIT ON STACK
     041060  032737  000020  002144           BIT     #BIT4,CSR
     041066  001403                           BEQ     1$
     041070  052766  000001  000002           BIS     #BIT0,2(SP)       ;SET C BIT ON STACK
     041076  000002                   1$:     RTI
```

```
                                        .SBTTL  TRAP    WAS THERE A DBE ON ANY CSR TRAP HANDLER
041100                          $WASDBE:PUSH    R1,R4
041104  013701  002216                  MOV     TOTCSRS,R1      ;GET CSR'S BYTE
041110  005004                          CLR     R4
041112                                  BEGIN LWDBE
041112                                    FOR CSRNO := #0 TO #36 BY #2
041116  006301                              ASL         R1
041120                                      ON.ERROR
041122  104426                                READCSR
041124                                        IF #BIT15 SET.IN CSR
041134                                          SET     R4
041140                                          LEAVE LWDBE
041142                                        END ;OF IF #BIT4
041142                                      END ;OF ON.ERROR
041142                                    IF R1 EQ #0 THEN LEAVE LWDBE
041146                                    END ;OF FOR CSRNO
041164                                  END LWDBE
041164  006004                          ROR     R4              ;SET C BIT FOR ERROR
041166                                  POP     R4,R1
041172                                  ON.ERROR
041174  052766  000001  000002            BIS   #BIT0,2(SP)
041202                                  ELSE
041204  042766  000001  000002            BIC   #BIT0,2(SP)
041212                                  END ;OF ON.ERROR
041212  000002                          RTI
                                        .SBTTL  TRAP    WAS THERE A DBE ON 1 SELECTED CSR TRAP HANDLER
                                        ;ON RETURN IF CARRY IS SET THERE WAS A DBE
041214  104426                  $WAS1DBE:READCSR
041216  005737  002144                  TST     CSR             ;DBE?
041222  100004                          BPL     3$              ;NO - SKIP
041224  052766  000001  000002          BIS     #BIT0,2(SP)     ;SET C BIT ON STACK
041232  000002                          RTI
041234  042766  000001  000002  3$:     BIC     #BIT0,2(SP)     ;CLR C BIT ON STACK
041242  000002                          RTI
```

```
   6916                                                  .SBTTL  TRAP    CLEAR ALL ECC CSR'S TRAP HANDLER
   6917 041244                               $CLRCSR:CLEAR   CSR
   6918 041250  004737  041314                       CALL    CSROUT
   6919 041254  000002                               RTI
   6920                                              .SBTTL  TRAP    CLEAR 1 SELECTED CSR TRAP HANDLER
   6921 041256                               $CLR1CSR:CLEAR  CSR
   6922 041262  104425                               LOADCSR
   6923 041264  000002                               RTI
   6924                                              .SBTTL  TRAP    ECC DISABLE, CHECK MODE, & WRITE CHECKBITS IN ALL CSR'S TRAP HANDLER
   6925                                              ;CHECKBITS ALREADY IN LOC "CSR"
   6926 041266  052737  000006  002144       $CHKDIS:BIS     #BIT1!BIT2,CSR          ;ECC DISABLE & DIAG CHECK MODE
   6927 041274  004737  041314                       CALL    CSROUT
   6928 041300  000002                               RTI
   6929                                              .SBTTL  TRAP    ECC DISABLE, CHECK MODE, & WRITE CHECKBITS IN 1 SELECTED CSR
   6930                                              ;CHECKBITS ALREADY IN LOC "CSR"
   6931 041302  052737  000006  002144       $CHK1DIS:BIS    #BIT1!BIT2,CSR          ;ECC DISABLE & DIAG CHECK MODE
   6932 041310  104425                               LOADCSR
   6933 041312  000002                               RTI
```

```
  6936 041314                              CSROUT: SUBTST  <<SUBR  WRITE IN ALL CSR'S>>
                                           ;*****************************************************************************************
                                           ;*SUBTEST        SUBR    WRITE IN ALL CSR'S
                                           ;*****************************************************************************************
  6937 041314                                      PUSH    R1
  6938 041316  013701  002216                      MOV     TOTCSRS,R1      ;GET CSR'S BYTE
  6939 041322                                      BEGIN LCSROUT
  6940 041322                                        FOR CSRNO := #0 TO #36 BY #2
  6941 041326  006301                                  ASL           R1
  6942 041330                                          ON.ERROR
  6943 041332  104425                                          LOADCSR
  6944 041334                                          END ;OF ON.ERROR
  6945 041334                                        IF R1 EQ #0 THEN LEAVE LCSROUT
  6946 041340                                        END ;OF FOR CSRNO
  6947 041356                                      END LCSROUT
  6948 041356                                      POP     R1
  6949 041360  000207                              RETURN
  6950
  6951 041362                              $INVALID:       SUBTST  <<TRAP  INVALIDATE BACKGROUND PATTERN>>
                                           ;*****************************************************************************************
                                           ;*SUBTEST        TRAP    INVALIDATE BACKGROUND PATTERN
                                           ;*****************************************************************************************
  6952 041362                                      PUSH    R0,R1
  6953 041366  013701  002100                      MOV     BANK,R1
  6954 041372  006301                              ASL     R1
  6955 041374  006301                              ASL     R1
  6956 041376  042761  020000  002626              BIC     #BIT13,CONFIG+2(R1)
  6957 041404                                      POP     R1,R0
  6958 041410  000002                              RTI
```

```
041412                                  $ERRGEN:        SUBTST<<TRAP    GENERATE AND TEST ERROR ADDRESS>>
                                        ;*******************************************************************************
                                        ;*SUBTEST        TRAP    GENERATE AND TEST ERROR ADDRESS
                                        ;*******************************************************************************
041412                                          PUSH    R0,R1,R2,R3
041422  013703  002102                          MOV     BANKINDEX,R3
041426  005737  002426                          TST     NOSUPER
041432  001003                                  BNE     6$
041434  013700  172246                          MOV     SIPAR3,R0               ;GENERATE WHAT ERROR ADDR SHOULD BE
041440  000402                                  BR      7$
041442  013700  177646                  6$:     MOV     UIPAR3,R0
041446  072027  177773                  7$:     ASH     #-5,R0
041452  005737  002130                          TST     EUFLAG
041456  001002                                  BNE     1$
041460  042700  177600                          BIC     #^C177,R0
041464  000301                          1$:     SWAB    R1                      ;GET CURRENT ADDRESS BITS 11 AND 12
041466  006201                                  ASR     R1
041470  006201                                  ASR     R1
041472  006201                                  ASR     R1
041474  042701  177775                          BIC     #^C2,R1
041500  060100                                  ADD     R1,R0                   ;ADD THEM TO THE ADJUSTED PAR VALUE
                                        ;GET ERROR ADDRESS FROM CSR UNDER TEST
041502  013701  002144                          MOV     CSR,R1
041506  072127  177773                          ASH     #-5,R1
041512  042701  177600                          BIC     #^C177,R1
041516  005737  002424                          TST     NO22BIT                 ;IS THIS AN 11/44 OR 11/24?
041522  001024                                  BNE     2$                      ;BRANCH IF NOT NECESSARY
041524  005737  002130                          TST     EUFLAG                  ;IS IT EUB?
041530  001421                                  BEQ     2$                      ;BRANCH IF NOT
041532                                          PUSH    R0                      ;SAVE GENERATED ERROR ADDRESS
041534  013702  002146                          MOV     CSRNO,R2                ;GET CSR NUMBER
041540  052762  040000  172100                  BIS     #BIT14,CSRADD(R2)       ;TURN ON EUB BIT CAREFULLY
041546  016200  172100                          MOV     CSRADD(R2),R0           ;GET CSR CONTENTS
041552  042762  040000  172100                  BIC     #BIT14,CSRADD(R2)       ;TURN OFF EUB BIT CAREFULLY
041560  042700  177037                          BIC     #^C740,R0               ;CLEAR EVERYTHING BUT ERROR ADDR
041564  006300                                  ASL     R0
041566  006300                                  ASL     R0                      ;SHIFT ADDR BITS 18-21 INTO POSITION
041570  060001                                  ADD     R0,R1                   ;ADD TO CURRENT ERROR ADDRESS
041572                                          POP     R0
041574  020001                          2$:     CMP     R0,R1                   ;COMPARE REAL AND GENERATED ERR. ADDR.
041576  001420                                  BEQ     5$                      ;BRANCH IF THEY ARE THE SAME
041600  005737  002134                          TST     INTFLAG                 ;INTERLEAVED?
041604  001411                                  BEQ     3$                      ;NO - WE HAVE AN ERROR
041606  062700  000100                          ADD     #100,R0
041612  005737  002136                          TST     INT64K                  ;64K INTERLEAVED MEMORY?
041616  001002                                  BNE     4$
041620  062700  000100                          ADD     #100,R0
041624  020001                          4$:     CMP     R0,R1
041626  001404                                  BEQ     5$
041630  005737  002064                  3$:     TST     SKPERR                  ;ARE WE SUPPOSED TO SKIP ERROR P.O.?
041634  001001                                  BNE     5$                      ;YES - SKIP ERROR PRINTOUT
041636  104462                                  PERR36                          ;ELSE PRINT ERROR ADDRESS ERROR
041640  010137  002430                  5$:     MOV     R1,ERRADD               ;SAVE CSR'S ERROR ADDRESS
041644  005037  002064                          CLR     SKPERR                  ;ENABLE THE ERROR PRINTOUT AGAIN
041650                                          POP     R3,R2,R1,R0             ;RESTORE REGISTERS
041660  000002                                  RTI
```

```
  7015 041662                               CHKGEN: SUBTST<<SUBR    GENERATE CHECK BITS>>
                                            ;*************************************************************************
                                            ;*SUBTEST        SUBR    GENERATE CHECK BITS
                                            ;*************************************************************************
  7016                                              ;CHECK BIT GENERATOR ROUTINE
  7017                                              ;CALLING SEQUENCE IS:
  7018                                              ;       MOV     #WORD1,SOURCE   ;SOURCE = ADDRESS OF DATA
  7019                                              ;       CALL    CHKGEN
  7020                                              ;
  7021                                              ;CHECK BITS RETURNED IN BITS 11-5 OF LOCATION CHECK
  7022                                              ;
  7023 041662                                       PUSH    R0,R1,R2,R3,R4,R5
  7024 041676  012702  000077                       MOV     #77,R2          ;DEFAULT CHECKBITS FOR DOUBLE WORD OF ZEROS
  7025 041702  012703  041770                       MOV     #CHKTAB,R3      ;ADDRESS OF CHECKBIT TABLE
  7026 041706  013705  002272                       MOV     SOURCE,R5       ;GET SOURCE ADDRESS
  7027 041712  012501                               MOV     (R5)+,R1        ;GET LSB'S
  7028 041714  011500                               MOV     (R5),R0         ;GET MSB'S
  7029
  7030 041716  006704                       1$:     SXT     R4              ;EXTEND SIGN OF DOUBLE WORD TO R4
  7031 041720  142304                               BICB    (R3)+,R4        ;ELIMINATE BITS THAT DON'T COUNT
  7032 041722  074402                               XOR     R4,R2           ;COMPLEMENT MASKED BITS IN CHECKBITS
  7033 041724  073027  000001                       ASHC    #1,R0           ;DOUBLE PRECISION LEFT SHIFT R0,,R1
  7034 041730  001372                               BNE     1$              ;LOOP TILL ALL BITS ARE CHECKED
  7035
  7036 041732  042702  177600                       BIC     #^C177,R2               ;KILL ALL JUNK BITS
  7037 041736  000302                               SWAB    R2              ;POSITION CHECKBITS IN BITS 11-5
  7038 041740  006202                               ASR     R2
  7039 041742  006202                               ASR     R2
  7040 041744  006202                               ASR     R2
  7041 041746  010237  002274                       MOV     R2,CHECK
  7042 041752                                       POP     R5,R4,R3,R2,R1,R0
  7043 041766  000207                               RETURN
```

```
 041770                     CHKTAB: ;BYTE #3
 041770       200                   .BYTE    ^C177            ;BIT 31
 041771       301                   .BYTE    ^C076            ;BIT 30
 041772       302                   .BYTE    ^C075            ;BIT 29
 041773       203                   .BYTE    ^C174            ;BIT 28
 041774       304                   .BYTE    ^C073            ;BIT 27
 041775       205                   .BYTE    ^C172            ;BIT 26
 041776       206                   .BYTE    ^C171            ;BIT 25
 041777       307                   .BYTE    ^C070            ;BIT 24
                                    ;BYTE #2
 042000       310                   .BYTE    ^C067            ;BIT 23
 042001       211                   .BYTE    ^C166            ;BIT 22
 042002       212                   .BYTE    ^C165            ;BIT 21
 042003       313                   .BYTE    ^C064            ;BIT 20
 042004       214                   .BYTE    ^C163            ;BIT 19
 042005       315                   .BYTE    ^C062            ;BIT 18
 042006       316                   .BYTE    ^C061            ;BIT 17
 042007       217                   .BYTE    ^C160            ;BIT 16
                                    ;BYTE #1
 042010       320                   .BYTE    ^C057            ;BIT 15
 042011       221                   .BYTE    ^C156            ;BIT 14
 042012       222                   .BYTE    ^C155            ;BIT 13
 042013       323                   .BYTE    ^C054            ;BIT 12
 042014       224                   .BYTE    ^C153            ;BIT 11
 042015       325                   .BYTE    ^C052            ;BIT 10
 042016       326                   .BYTE    ^C051            ;BIT  9
 042017       227                   .BYTE    ^C150            ;BIT  8
                                    ;BYTE #0
 042020       340                   .BYTE    ^C037            ;BIT  7
 042021       241                   .BYTE    ^C136            ;BIT  6
 042022       242                   .BYTE    ^C135            ;BIT  5
 042023       343                   .BYTE    ^C034            ;BIT  4
 042024       244                   .BYTE    ^C133            ;BIT  3
 042025       345                   .BYTE    ^C032            ;BIT  2
 042026       346                   .BYTE    ^C031            ;BIT  1
 042027       247                   .BYTE    ^C130            ;BIT  0
```

```
042030                                    SUBTST<<SUBR    MAPPER>>
                                ;*******************************************************************************************
                                ;*SUBTEST         SUBR    MAPPER
                                ;*******************************************************************************************
                                ;THIS SUBROUTINE MAPS THE MEMORY BANK (16K WORDS = 1 BANK)
                                ;IN R3 TO THE TEST PATTERN AREA (SUPERVISOR VIRTUAL (60000 - 157777) FOR
                                ;THE 11/44 AND 11/45-55; USER VIRTUAL (60000 - 157777) FOR ALL OTHER
                                ;PDP-11'S).
                                ;
                                ;CALL     MOV     BANKNO,R3               ;SET UP BANK ARGUEMENT
                                ;         CALL    MAPPER                  ;ACTUAL CALL
                                ;         RETURN                          ;ONLY RETURN

                                          ;SET SUPERVISOR/USER UP FOR 1 TO 1 MAP
042030                          MAPPER:   PUSH    R0,R1,R2,R4,R5
042042  012700  172340                    MOV     #KIPAR0,R0              ;FIRST AREA TO MAP TO
042046  012701  172240                    MOV     #SIPAR0,R1              ;FIRST ADDRESS REGISTER
042052  012704  172200                    MOV     #SIPDR0,R4              ;FIRST DESCRIPTOR REGISTER
042056  005737  002426                    TST     NOSUPER                 ;CAN WE USE SUPERVISOR MODE?
042062  001404                            BEQ     4$                      ;YES, BRANCH
042064  012701  177640                    MOV     #UIPAR0,R1              ;FIRST ADDRESS REGISTER
042070  012704  177600                    MOV     #UIPDR0,R4              ;FIRST DESCRIPTOR REGISTER
042074  012702  077406          4$:       MOV     #77406,R2               ;CONSTANT FOR 4K PAGE, UP, R/W
042100  012705  000010                    MOV     #8.,R5                  ;COUNTER
042104  012021                  1$:       MOV     (R0)+,(R1)+             ;PUT IN SUPERVISOR ADDRESS
042106  010224                            MOV     R2,(R4)+                ;PUT IN SUPERVISOR DESCRIPTOR
042110  077503                            SOB     R5,1$                   ;LOOP TILL DONE
042112  012741  177600                    MOV     #177600,-(R1)           ;CORRECT LAST FIELD FOR PERIPHERALS PAGE

                                          ;SET UP SUPERVISOR/USER FOR TEST AREA
042116  022703  000170                    CMP     #120.,R3                ;MAP NOTHING (1 TO 1)?
042122  001516                            BEQ     3$                      ;YES - SKIP
042124  072327  000011                    ASH     #9.,R3                  ;BANK 1 STARTS AT 100,000 LESS 6 LSB'S
                                                                          ;FOR MEMORY MANAGEMENT = 1000
042130  012701  172246                    MOV     #SIPAR3,R1              ;SETUP FOR AUTO INCREMENTING
042134  005737  002426                    TST     NOSUPER                 ;DO WE HAVE SUPERVISOR MODE?
042140  001402                            BEQ     5$                      ;YES - BRANCH
042142  012701  177646                    MOV     #UIPAR3,R1              ;SETUP FOR AUTO INCREMENTING
042146  012702  000004          5$:       MOV     #4,R2                   ;COUNTER
042152  010321                  2$:       MOV     R3,(R1)+                ;PLUG IN PAR INFO
042154  062703  000200                    ADD     #200,R3                 ;BUMP ADDRESS 4K
042160  077204                            SOB     R2,2$                   ;LOOP TILL DONE
042162  005737  002232                    TST     SPLTCSR
042166  001442                            BEQ     9$
042170  162701  000010                    SUB     #10,R1
042174  010102                            MOV     R1,R2
042176  062702  000004                    ADD     #4,R2
042202  022737  000001  002232            CMP     #1,SPLTCSR
042210  001403                            BEQ     10$
042212  010200                            MOV     R2,R0
042214  010102                            MOV     R1,R2
042216  010001                            MOV     R0,R1
042220  012122                  10$:      MOV     (R1)+,(R2)+
042222  011112                            MOV     (R1),(R2)
042224  013700  002102                    MOV     BANKINDEX,R0
042230  005737  002136                    TST     INT64K
042234  001403                            BEQ     11$
```

```
042236  012700  004000                          MOV     #4000,R0
042242  000402                                  BR      12$
042244  012700  010000                  11$:    MOV     #10000,R0
042250  005737  002426                  12$:    TST     NOSUPER
042254  001403                                  BEQ     13$
042256  012701  177652                          MOV     #UIPAR5,R1
042262  000402                                  BR      14$
042264  012701  172252                  13$:    MOV     #SIPAR5,R1
042270  060021                          14$:    ADD     R0,(R1)+
042272  060011                                  ADD     R0,(R1)
                                                ;IF WE ONLY HAVE AN 124K SYSTEM, WE DON'T WANT TO TEST THE
                                                ;LAST 4K, WHERE THE UNIBUS DEVICE PAGE IS. INSTEAD, THE
                                                ;PROGRAM WILL REMAP THE LAST 4K TO 8-12K. ALSO, IF THERE
                                                ;IS A BANK 177 ON AN 11/44, THE PROGRAM WILL REMAP THE LAST
                                                ;4K TO 8-12K FOR THE SAME REASON.
042274  022737  000007  002526          9$:     CMP     #7,LASTBANK
042302  001010                                  BNE     7$
042304  005737  002424                          TST     NO22BIT                 ;11/44 OR 24?
042310  001423                                  BEQ     3$                      ;BRANCH IF SO
042312  022737  000007  002100                  CMP     #7,BANK                 ;BANK 7?
042320  001017                                  BNE     3$                      ;NO - BRANCH
042322  000404                                  BR      8$
042324  022737  000177  002526          7$:     CMP     #177,LASTBANK
042332  001012                                  BNE     3$
042334  005737  002426                  8$:     TST     NOSUPER
042340  001404                                  BEQ     6$
042342  013737  177652  177654                  MOV     UIPAR5,UIPAR6
042350  000403                                  BR      3$
042352  013737  172252  172254          6$:     MOV     SIPAR5,SIPAR6
042360                                  3$:     POP     R5,R4,R2,R1,R0
042372  000207                                  RETURN
                                                .SBTTL  TRAP    MAP KERNEL (ALMOST 1 TO 1) TRAP HANDLER
042374                                  $KMAP:  PUSH    R0,R1,R2,R3,R4
042406  005000                                  CLR     R0                      ;1ST AREA TO MAP TO
042410  012701  172340                          MOV     #KIPAR0,R1              ;FIRST ADDRESS
042414  012702  077406                          MOV     #77406,R2               ;CONSTANT FOR 4K PAGE,UP,R/W
042420  012703  172300                          MOV     #KIPDR0,R3              ;1ST PAGE DESCRIPTOR REGISTER
042424  012704  000010                          MOV     #8.,R4                  ;COUNTER
042430  010021                          1$:     MOV     R0,(R1)+                ;PUT IN KERNEL ADDRESS
042432  010223                                  MOV     R2,(R3)+                ;PUT IN KERNEL DISCRIPTOR
042434  062700  000200                          ADD     #200,R0                 ;ADD ADDRESS CONSTANT FOR 4K CHANGE
042440  077405                                  SOB     R4,1$                   ;LOOP TILL DONE
042442  012741  177600                          MOV     #177600,-(R1)           ;THE PERIPHERALS PAGE TO KIPAR7
042446  012741  177400                          MOV     #177400,-(R1)           ;AND NEXT LOWER PAGE TO KIPAR6
042452                                          POP     R4,R3,R2,R1,R0
042464  000002                                  RTI
```

```
7192 042466                          RELOCATE:SUBTST <<RELOCATE PROGRAM>>
                                     ;*******************************************************************************
                                     ;*SUBTEST       RELOCATE PROGRAM
                                     ;*******************************************************************************
7193 042466                                  IF #SW12 SET.IN @SWR THEN $RETURN ERROR
7194 042502                                  IF APTFLAG IS TRUE OR ACTFLAG IS TRUE
7195 042516                                    IF $PASS NE #0 THEN $RETURN ERROR
7196 042530                                  END; OF IF APTFLAG
7197 042530                                  BEGIN LOADERBANK
7198 042530                                    FOR BANK := #1 TO LASTBANK
7199 042536  004737  044240                      CALL EXBANK
7200 042542                                      IF ACFLAG IS TRUE AND PFLAG IS FALSE AND BMFLAG IS FALSE
7201 042564  013700  002100                        MOV       BANK,R0
7202 042570  010037  002402                        MOV       R0,LOADBANK
7203 042574  013701  002536                        MOV       LOADHOME,R1
7204 042600  004737  043710                        CALL      BANKMOV
7205 042604  004737  044206                        CALL      NEWLOAD         ;MAP NEW LOADER BANK IN KERNEL
7206 042610  013701  002102                        MOV       BANKINDEX,R1
7207 042614  052761  100000  002626                BIS       #BIT15,CONFIG+2(R1)     ;MARK LOADER
7208 042622  042761  020000  002626                BIC       #BIT13,CONFIG+2(R1)     ;INVALIDATE BACKGROUND PATTERN
7209 042630                                        LEAVE LOADERBANK
7210 042632                                      END ;OF IF ACFLAG
7211 042632                                    END ;OF FOR BANK
7212 042646                                    IF #SW13 OFF.IN @SWR
7213 042656                                      TYPE        MSG075          ;RELOCATION NOT POSSIBLE
7214 042662                                    END ;OF IF #SW13
7215 042662                                    $RETURN ERROR
7216 042666                                  END LOADERBANK
7217 042666                                  BEGIN FINDBANK
7218 042666  013702  002526                  MOV     LASTBANK,R2
7219 042672  006302                          ASL     R2
7220 042674  006302                          ASL     R2              ;R2 <- R2 * 4
7221 042676                                    FOR R1 := #2*2 TO R2 BY #4
7222 042702                                      IF #BIT7!BIT0 OFF.IN CONFIG(R1) ;IF NO ERRORS & NOT PROGRAM SPACE
7223 042712                                        IF #BIT15 OFF.IN CONFIG+2(R1) ;IF NOT LOADER BANK
7224 042722                                          IF CPUBIT SET.IN CONFIG(R1) ;IF ACCESSABLE
7225 042732                                            IF #BIT9 SET.IN CONFIG+2(R1) THEN LEAVE FINDBANK ;IF PARITY
7226 042742                                              IF #BIT6 SET.IN CONFIG(R1) AND #BIT7 OFF.IN CONFIG(R1)
7227                                                       ;IF 1ST PROTECTABLE ECC BANK
7228 042762                                                LEAVE FINDBANK
7229 042764                                              END ;OF IF #BIT6
7230 042764                                              IF INHECC IS FALSE
7231 042772                                                SET   INHECC
7232 043000  010137  002510                                MOV  R1,INHBANK
7233 043004                                              END; OF IF INHECC
7234 043004                                          END ;OF IF CPUBIT
7235 043004                                        END ;OF IF #BIT15
7236 043004                                      END ;OF IF #BIT7
7237 043004                                    END ;OF FOR
7238 043014                                    IF FULLREL IS FALSE
7239 043022                                      IF INHECC IS TRUE
7240 043030  013701  002510                        MOV    INHBANK,R1
7241 043034  023727  002260  000030                CMP    REALPAT,#30       ;IS THIS PATTERN 30?
7242 043042  001423                                BEQ    RELENT1           ;YES - SKIP MESSAGE
7243 043044                                        TYPE   MSG123
7244 043050  000420                                BR     RELENT1
7245 043052                                      END; OF IF INHECC
```

```
043052                                     END; OF IF FULLREL
043052  005037  002506                     CLR   INHECC                   ;MAKE SURE FLAG IS TURNED OFF!
043056                                     IF #SW13 OFF.IN @SWR
043066  023727  002260  000030               CMP          REALPAT,#30     ;IS THIS PATTERN 30?
043074  001402                               BEQ          SKUB            ;YES - SKIP MESSAGE
043076                                       TYPE         MSG075          ;RELOCATION NOT POSSIBLE
043102                                     END ;OF IF #SW13
043102                           SKUB:     $RETURN ERROR
043106                                   END FINDBANK
043106                                   CLEAR   INHECC                   ;IF WE RELOCATED PROPERLY, THIS SHOULD BE OFF!
043112  042761  020000  002626   RELENT1: BIC     #BIT13,CONFIG+2(R1)      ;INVALIDATE BACKGROUND PATTERN
043120  005000                           CLR     R0
043122  071027  000004                   DIV     #4,R0
043126                           RELOC1: LET NEWBANK := R0
043132  013737  002502  002504           MOV     PGMCSR,PGMCSR+2          ;SAVE CURRENT PGM. CSR
043140  004737  044056                   CALL    USERMAP                  ;MAP NEWBANK TO USER PAR
043144                                   USER                             ;ENTER USER MODE
043152                                   BMOV    0,100000,SIZE            ;MOVE PROGRAM
043164  104417                           KERNEL                           ;ENTER KERNEL MODE
043166  022737  000001  003710           CMP     #1,PROTYP                ;IS THIS AN 11/44 ?
043174  001021                           BNE     JMPRL1                   ;JUMP IF NOT
043176  042737  000040  172516           BIC     #BIT5,MMR3               ;TURN OFF UNIBUS MAP
043204  013700  002270                   MOV     NEWBANK,R0
043210  006200                           ASR     R0
043212                                   ON.ERROR
043214  012737  100000  170200             MOV    #BIT15,MAPLO
043222                                   END ;OF ON.ERROR
043222  010037  170202                   MOV     R0,MAPHO
043226  004737  043644                   CALL    LOWMAP                   ;SETUP LOWER 16K IN UNIBUS MAP
043232  052737  000040  172516           BIS     #BIT5,MMR3               ;ENERGIZE UNIBUS MAP
043240  042737  000001  177572   JMPRL1: BIC     #BIT0,MMR0               ;DEENERGIZE MEMORY MANAGEMENT
043246  004737  044140                   CALL    NEWKERNEL
043252  013700  002270                   MOV     NEWBANK,R0
043256  006300                           ASL     R0
043260  006300                           ASL     R0                       ;R0 <- R0 * 4
043262  016002  002624                   MOV     CONFIG(R0),R2
043266  000302                           SWAB    R2
043270  042702  177760                   BIC     #^C17,R2
043274  006302                           ASL     R2
043276  052737  000001  177572           BIS     #BIT0,MMR0               ;ENERGIZE MEMORY MANAGEMENT
043304  010237  002502                   MOV     R2,PGMCSR                ;PUT NEW PGM. CSR INTO PGMCSR
043310  032760  010000  002626           BIT     #BIT12,CONFIG+2(R0)      ;IS THE NEW BANK INTERLEAVED?
043316  001412                           BEQ     1$                       ;BRANCH IF NOT INTERLEAVED
043320  016002  002624                   MOV     CONFIG(R0),R2
043324  042702  007777                   BIC     #^C170000,R2
043330  072227  177775                   ASH     #-3,R2
043334  052702  100000                   BIS     #BIT15,R2
043340  050237  002502                   BIS     R2,PGMCSR
043344                           1$:     SET     RLFLAG
043352                                   $RETURN NOERROR
```

```
  7298 043356                               UNRELOCATE:SUBTST       <<UNRELOCATE PROGRAM>>
                                            ;*************************************************************************
                                            ;*SUBTEST        UNRELOCATE PROGRAM
                                            ;*************************************************************************
  7299                                               ;RESTORE LOADERS
  7300 043356                                        PUSH    R0
  7301 043360  013701  002402                        MOV     LOADBANK,R1
  7302 043364  013700  002536                        MOV     LOADHOME,R0
  7303 043370  004737  043710                        CALL    BANKMOV
  7304 043374  004737  044206                        CALL    NEWLOAD                       ;MAP NEW LOADER BANK IN KERNEL SPACE
  7305 043400                                        PUSH    BANK
  7306 043404  013737  002402  002100                MOV     LOADBANK,BANK
  7307 043412  004737  044240                        CALL    EXBANK
  7308 043416  013701  002102                        MOV     BANKINDEX,R1
  7309 043422  042761  100000  002626                BIC     #BIT15,CONFIG+2(R1)           ;CLEAR LOADER FLAG
  7310 043430  013737  002536  002100                MOV     LOADHOME,BANK
  7311 043436  004737  044240                        CALL    EXBANK
  7312 043442  013701  002102                        MOV     BANKINDEX,R1
  7313 043446  042761  020000  002626                BIC     #BIT13,CONFIG+2(R1)           ;INVALIDATE BACKGROUND PATTERN
  7314 043454                                        POP     BANK
  7315 043460                                        CLEAR   INHECC                        ;MAKE SURE ECC TESTS ARE NOT INHIBITED!
  7316
  7317                                               ;RESTORE BANK 0
  7318 043464  042737  020000  002626                BIC     #BIT13,CONFIG+2               ;INVALIDATE BACKGROUND PATTERN
  7319 043472                                        LET NEWBANK := #0
  7320 043476  004737  044056                        CALL    USERMAP                       ;MAP NEWBANK TO USER PAR
  7321 043502                                        USER                                  ;ENTER USER MODE
  7322 043510                                        BMOV    0,100000,SIZE                 ;MOVE PROGRAM
  7323 043522  104417                                KERNEL                                ;ENTER KERNEL MODE
  7324 043524  042737  000001  177572                BIC     #BIT0,MMR0                    ;DEENERGIZE MEMORY MANAGEMENT
  7325 043532  004737  044140                        CALL    NEWKERNEL
  7326 043536  013737  002504  002502                MOV     PGMCSR+2,PGMCSR               ;RESTORE PREVIOUS PGM. CSR
  7327 043544  052737  000001  177572                BIS     #BIT0,MMR0                    ;ENERGIZE MEMORY MANAGEMENT
  7328 043552  005037  002124                        CLR     RLFLAG
  7329 043556  022737  000001  003710                CMP     #1,PROTYP                     ;IS THIS AN 11/44 ?
  7330 043564  001014                                BNE     1$
  7331 043566  042737  000040  172516                BIC     #BIT5,MMR3                    ;TURN OFF UNIBUS MAP
  7332 043574                                        CLEAR   MAPLO,MAPHO
  7333 043604  004737  043644                        CALL    LOWMAP                        ;SETUP LOWER 16K OF UNIBUS MAP
  7334 043610  052737  000040  172516                BIS     #BIT5,MMR3                    ;ENERGIZE UNIBUS MAP
  7335 043616  012700  002626          1$:           MOV     #CONFIG+2,R0        ;MOVE 2ND WORD OF CONFIG TO R0
  7336 043622  042710  020000          2$:           BIC     #BIT13,(R0)         ;CLEAR BACKGROUND VALID BIT
  7337 043626  062700  000004                        ADD     #4,R0               ;INCREMENT TO NEXT BANK
  7338 043632  020027  003620                        CMP     R0,#3620            ;DONE?
  7339 043636  003771                                BLE     2$                  ;NO - BRANCH
  7340 043640                                        POP     R0
  7341 043642  000207                                RETURN
  7342
  7343 043644                               LOWMAP: SUBTST  <<SETUP LOWER 16K OF UNIBUS MAP>>
                                            ;*************************************************************************
                                            ;*SUBTEST        SETUP LOWER 16K OF UNIBUS MAP
                                            ;*************************************************************************
  7344 043644                                        PUSH    R0,R1,R2
  7345 043652  012700  170200                        MOV     #MAPL0,R0
  7346 043656  012701  170204                        MOV     #MAPL1,R1
  7347 043662  012702  000003                        MOV     #3,R2
  7348 043666  012011                        1$:     MOV     (R0)+,(R1)
```

```
   7349 043670  062721  020000              ADD     #BIT13,(R1)+
   7350 043674  012021                      MOV     (R0)+,(R1)+
   7351 043676  077205                      SOB     R2,1$
   7352 043700                              POP     R2,R1,R0
   7353 043706  000207                      RETURN
```

```
043710                                  BANKMOV:SUBTST  <<MOVE BANKS>>
                                        ;*************************************************************************
                                        ;*SUBTEST       MOVE BANKS
                                        ;*************************************************************************
                                                ;MOVE 3/4 OF A BANK
                                                ;CALLING SEQUENCE
                                                ;R0 = DESTINATION BANK
                                                ;R1 = SOURCE BANK
043710  104415                                  SAVREG
043712  004737  044056                          CALL    USERMAP
043716  104416                                  RESREG
043720  104415                                  SAVREG
043722  072027  000011                          ASH     #9.,R0
043726  072127  000011                          ASH     #9.,R1
043732  012702  177650                          MOV     #UIPAR4,R2
043736  012703  000200                          MOV     #200,R3

043742  010122                                  MOV     R1,(R2)+                ;MAP 1ST HALF BANK
043744  060301                                  ADD     R3,R1                   ;BUMP BY 4K
043746  010122                                  MOV     R1,(R2)+
043750  060301                                  ADD     R3,R1

043752  010022                                  MOV     R0,(R2)+
043754  060300                                  ADD     R3,R0
043756  010022                                  MOV     R0,(R2)+
043760  060300                                  ADD     R3,R0

043762                                          USER
043770                                          BMOV    100000,140000,SIZE/2    ;MOV 1ST HALF BANK
044002  104417                                  KERNEL                  ;ENTER KERNEL MODE

044004  012702  177650                          MOV     #UIPAR4,R2

044010  010122                                  MOV     R1,(R2)+                ;MAP 2ND HALF BANK
044012  060301                                  ADD     R3,R1                   ;BUMP BY 4K
044014  010122                                  MOV     R1,(R2)+
044016  060301                                  ADD     R3,R1

044020  010022                                  MOV     R0,(R2)+
044022  060300                                  ADD     R3,R0
044024  010022                                  MOV     R0,(R2)+
044026  060300                                  ADD     R3,R0

044030                                          USER
044036                                          BMOV    100000,140000,SIZE/4    ;MOV 3ND FOURTH OF BANK
044050  104417                                  KERNEL                  ;ENTER KERNEL MODE

044052  104416                                  RESREG
044054  000207                                  RETURN
```

```
044056                          USERMAP:SUBTST  <<SUBR  MAP USER TO NEW BANK>>
                                ;*************************************************************************
                                ;*SUBTEST        SUBR    MAP USER TO NEW BANK
                                ;*************************************************************************
044056  012701  177640                          MOV     #UIPAR0,R1                      ;COPY KERNEL PAR'S & PDR'S (0-3)
044062  012702  172340                          MOV     #KIPAR0,R2
044066  012703  177600                          MOV     #UIPDR0,R3
044072  012704  172300                          MOV     #KIPDR0,R4
044076  012705  000004                          MOV     #4,R5
044102  012221                  1$:             MOV     (R2)+,(R1;+
044104  011423                                  MOV     (R4),(R3)+
044106  077503                                  SOB     R5,1$

044110  013700  002270                          MOV     NEWBANK,R0
044114  072027  000011                          ASH     #9.,R0                          ;BANK 1 STARTS AT 100,000 LESS 6 LSB'S
                                                                                ;FOR MEMORY MANAGEMENT = 1000
044120  012705  000004                          MOV     #4,R5
044124  010021                  2$:             MOV     R0,(R1)+                        ;SETUP UIPAR(4-7)
044126  062700  000200                          ADD     #200,R0                         ;BUMP ADDRESS 4K
044132  011423                                  MOV     (R4),(R3)+                      ;SETUP UIPDR(4-7)
044134  077505                                  SOB     R5,2$
044136  000207                                  RETURN

044140                          NEWKERNEL:SUBTST        <<SUBR  SETUP KERNEL PAR'S FOR NEW BANK>>
                                ;*************************************************************************
                                ;*SUBTEST        SUBR    SETUP KERNEL PAR'S FOR NEW BANK
                                ;*************************************************************************
044140                                          PUSH    R0,R1,R5
044146  012700  172340                          MOV     #KIPAR0,R0
044152  013701  002270                          MOV     NEWBANK,R1
044156  072127  000011                          ASH     #9.,R1                          ;BANK 1 STARTS AT 100,000 LESS 6 LSB'S
                                                                                ;FOR MEMORY MANAGEMENT = 1000
044162  012705  000004                          MOV     #4,R5
044166  010120                  1$:             MOV     R1,(R0)+                        ;SETUP KIPAR(0-3)
044170  062701  000200                          ADD     #200,R1
044174  077504                                  SOB     R5,1$
044176                                          POP     R5,R1,R0
044204  000207                                  RETURN

044206                          NEWLOAD:SUBTST  <<SUBR  SETUP KERNEL PAR'S FOR NEW LOADER BANK>>
                                ;*************************************************************************
                                ;*SUBTEST        SUBR    SETUP KERNEL PAR'S FOR NEW LOADER BANK
                                ;*************************************************************************
                                                ;R0 CONTAINS THE DESTINATION BANK
044206                                          PUSH    R0,R1
044212  012701  172350                          MOV     #KIPAR4,R1
044216  072027  000011                          ASH     #9.,R0                          ;BANK 1 STARTS AT 100000 LESS 6 LSB'S (1000)
044222  010021                                  MOV     R0,(R1)+                        ;SETUP KIPAR4
044224  062700  000200                          ADD     #200,R0
044230  010021                                  MOV     R0,(R1)+                        ;SETUP KIPAR5
044232                                          POP     R1,R0
044236  000207                                  RETURN
```

```
044240                                  EXBANK: SUBTST  <<SUBR  EXAMINE BANK>>
                                        ;*******************************************************************************************
                                        ;*SUBTEST        SUBR    EXAMINE BANK
                                        ;*******************************************************************************************
                                        ;DOES THE FOLLOWING:
                                        ;(1)   SETS UP "BANKINDEX" AND R1 BASED ON VALUE OF "BANK".
                                        ;(2)   SETS THE "MKFLAG" IF THE BANK IS ECC.
                                        ;(3)   SETS THE "KFLAG" IF THE BANK IS MF11S-K.
                                        ;(4)   SETS THE "KPFLAG" IF THE BANK IS THE PROTECTED REGION OF ECC MEMORY.
                                        ;(5)   SETS THE "ACFLAG" IF THE BANK CAN BE ACCESSED BY THIS CPU.
                                        ;(6)   SETS THE "PFLAG" IF THE BANK IS IN PROGRAM SPACE.
                                        ;(7)   SETS THE  "RRFLAG" IF RELOCATION IS REQUIRED TO TEST THIS BANK; HOWEVER,
                                        ;      IT COMPLEMENTS THIS FLAG IF THE RELOCATION FLAG "RLFLAG" IS SET (THIS IS
                                        ;      NECESSARY FOR THE USE OF THE RECURSIVE "MODE" SUBROUTINES). THE "RRFLAG"
                                        ;      IS ALWAYS SET TO DISABLE TESTING IF FIELD SERVICE MODE  "SELECTED BANKS"
                                        ;      ARE BEING TESTED AND THIS BANK IS NOT SELECTED.
                                        ;(8)   SETS THE "BMFLAG" IF THE BANK IS A BAD MEMORY;  HOWEVER, IT  COMPLEMENTS
                                        ;      THIS FLAG IF THE "WORST" FLAG IS NOT SET  (THIS IS NECESSARY FOR THE USE
                                        ;      OF THE RECURSIVE "MODE" SUBROUTINES).
                                        ;(9)   SETS THE "EUFLAG" IF THE BANK HAS EXTENDED UNIBUS MEMORY.
                                        ;(10) SETS THE "INTFLAG" IF THE BANK IS INTERLEAVED.
                                        ;(11) SETS THE "INT64K" FLAG IF THE BANK IS INTERLEAVED ON 64K WORD BOUNDS.
                                        ;(12) SETS THE "SKIPMK" FLAG IF THIS BANK IS INTERLEAVED, AND HAS ALREADY
                                        ;      BEEN TESTED.
                                        ;
044240                                          PUSH    R0,R1,R2
044246                                          CLEAR   MKFLAG,KPFLAG,KFLAG,EUFLAG
044266                                          SET     ACFLAG
044274                                          CLEAR   PFLAG,RRFLAG,BMFLAG
044310                                          CLEAR   INTFLAG,INT64K,SKIPMK
044324  013701  002100                          MOV     BANK,R1
044330  006301                                  ASL     R1
044332  006301                                  ASL     R1                      ;R1 <- R1 * 4
044334  010137  002102                          MOV     R1,BANKINDEX
044340  032761  000100  002624                  BIT     #BIT6,CONFIG(R1)                ;PROTECTED REGION OF ECC MEMORY?
044346  001403                                  BEQ     1$                              ;NO - SKIP
044350                                          SET     KPFLAG
044356  012700  000002          1$:             MOV     #BIT1,R0
044362                                          IF R0 SET.IN CPUBIT AND R0 OFF.IN CONFIG(R1)
044376  005037  002114                            CLR   ACFLAG
044402                                          END ;OF IF R0
044402  005737  002114                          TST     ACFLAG                          ;ACTIVE MEMORY?
044406  001415                                  BEQ     12$                             ;BRANCH IF NOT
044410  016102  002626                          MOV     CONFIG+2(R1),R2
044414  000302                                  SWAB    R2
044416  042702  177770                          BIC     #^C7,R2                         ;ISOLATE MEM TYPE BITS
044422  020227  000002                          CMP     R2,#2                           ;IS THIS AN ILLEGAL MEM TYPE?
044426  003005                                  BGT     12$                             ;BRANCH IF NOT
044430                                          SET     BMFLAG                          ;SET BAD BANK FLAG
044436  000137  044700                          JMP     ENEXBK                          ;JUMP OVER REST OF FLAG TESTS
044442  032761  000400  002626  12$:            BIT     #BIT8,CONFIG+2(R1)              ;IS THIS EUB?
044450  001003                                  BNE     2$                              ;BRANCH IF NOT
044452                                          SET     EUFLAG                          ;YES - SET EUB FLAG
044460  032761  001000  002626  2$:             BIT     #BIT9,CONFIG+2(R1)              ;IS THERE ECC THERE?
044466  001012                                  BNE     3$                              ;NO - SKIP
044470                                          SET     MKFLAG          ;YES - SET MKFLAG
044476  032761  000400  002626                  BIT     #BIT8,CONFIG+2(R1)              ;IS THIS MF11S-K MEMORY
```

```
044504  001403                           BEQ     3$                            ;NO - IT'S MS11-M
044506                                   SET     KFLAG                         ;YES - SET KFLAG
044514  032761  000200  002624  3$:      BIT     #BIT7,CONFIG(R1)              ;BANK = PROGRAM SPACE?
044522  001406                           BEQ     5$                            ;NO - SKIP
044524                                   SET     PFLAG,RRFLAG
044540  005737  002124          5$:      TST     RLFLAG                        ;IS PROGRAM RELOCATED?
044544  001402                           BEQ     6$                            ;NO - SKIP
044546  005137  002122                   COM     RRFLAG                        ;YES - COMPLEMENT RELOCATION REQUIRED FLAG
044552  032761  000001  002624  6$:      BIT     #BIT0,CONFIG(R1)              ;ERRORS PRESENT IN THIS BANK?
044560  001403                           BEQ     8$                            ;NO - SKIP
044562                                   SET     BMFLAG
044570  005737  002540          8$:      TST     WORST                         ;IS THIS A WORST FIRST PASS?
044574  001002                           BNE     9$                            ;YES - SKIP
044576  005137  002126                   COM     BMFLAG                        ;NO - COMPLEMENT BAD MEMORY FLAG
044602                          9$:      IF SELONLY IS TRUE AND #BIT14 OFF.IN CONFIG+2(R1)
044620                                     SET   RRFLAG
044626                                   END ;OF IF SELONLY
044626  032761  010000  002626           BIT     #BIT12,CONFIG+2(R1)           ;IS THIS BANK INTERLEAVED?
044634  001421                           BEQ     ENEXBK                        ;BRANCH IF IT IS NOT
044636                                   SET     INTFLAG
044644  032761  004000  002626           BIT     #BIT11,CONFIG+2(R1)           ;IS THIS BANK INTERLEAVED WITH 64K BOARDS?
044652  001403                           BEQ     10$                           ;BRANCH IF IT IS NOT
044654                                   SET     INT64K
044662  032761  000040  002624  10$:     BIT     #BIT5,CONFIG(R1)              ;SHOULD THIS BANK BE TESTED?
044670  001403                           BEQ     ENEXBK                        ;BRANCH IF IT SHOULD
044672                                   SET     SKIPMK
044700                          ENEXBK:  POP     R2,R1,R0                      ;RESTORE REGISTERS
044706  000207                           RETURN
```

```
 7540 044710                                 BANKOK: SUBTST  <<SUBR  BANK OK?>>
                                             ;*******************************************************************************
                                             ;*SUBTEST        SUBR    BANK OK?
                                             ;*******************************************************************************
 7541                                                ;TEST TO INSURE THAT THE TYPE OF MEMORY IN THE PRESENT BANK
 7542                                                ;IS OF THE TYPE WE ARE TESTING "TMFLAG".
 7543                                                ;RESULT IS RETURNED IN THE CONDITION CODES (OK = (=0)).
 7544 044710  013700  002132                         MOV     TMFLAG,R0
 7545 044714  005100                                 COM     R0
 7546 044716  013701  002116                         MOV     MKFLAG,R1
 7547 044722  074001                                 XOR     R0,R1
 7548 044724  000207                                 RETURN                          ;OK = (=OK)
 7549
 7550 044726                                 INCRPT:
 7551 044726                                 INCPAT: SUBTST  <<SUBR  INCREMENT PATTERN TESTING >>
                                             ;*******************************************************************************
                                             ;*SUBTEST        SUBR    INCREMENT PATTERN TESTING
                                             ;*******************************************************************************
 7552                                                ;INCREMENT THE PATTERN & SET UP THE CONDITION CODES
 7553                                                ;RESULT - Z BIT SET INDICATES OVERFLOW
 7554 044726  005237  002110                         INC     PATTERN
 7555 044732  022737  000030  002110                 CMP     #30,PATTERN             ;SET UP CONDITION CODES
 7556 044740  000207                                 RETURN                          ;NOT EQUAL TO ZERO IS GOOD (NO OVERFLOW)
 7557
 7558 044742                                 SETPAT:
 7559 044742                                 HIPAT:  SUBTST  <<SUBR  SET HIGHEST PATTERN TESTING TYPE>>
                                             ;*******************************************************************************
                                             ;*SUBTEST        SUBR    SET HIGHEST PATTERN TESTING TYPE
                                             ;*******************************************************************************
 7560 044742  012737  000027  002110                 MOV     #27,PATTERN             ;SET HIGHEST PATTERN
 7561 044750  000207                                 RETURN
 7562
 7563 044752                                 INCBNK: SUBTST  <<SUBR  INCREMENT BANK & TEST>>
                                             ;*******************************************************************************
                                             ;*SUBTEST        SUBR    INCREMENT BANK & TEST
                                             ;*******************************************************************************
 7564                                                ;RESULTS RETURNED IN CONDITION CODES
 7565 044752  005237  002100                         INC     BANK
 7566 044756  023737  002526  002100                 CMP     LASTBANK,BANK           ;TOO FAR?
 7567 044764  000207                                 RETURN
```

```
044766                           BOOT:   SUBTST  <<BOOTSTRAP ROUTINE>>
                                 ;*******************************************************************************
                                 ;*SUBTEST        BOOTSTRAP ROUTINE
                                 ;*******************************************************************************
                                         ;INITIALIZE ALL CSR'S
                                         ;UNRELOCATE IF NECESSARY
                                         ;FLUSH OUT ANY DBE'S
                                         ;TURN OFF MEMORY MANAGEMENT
                                         ;TURN OFF THE UNIBUS MAP
                                         ;BOOT RK0 OR RK1
044766  104472                           ECCINIT            ;TRAP ON DOUBLE BIT ERRORS (NORMAL)
044770                                   SET4    #BOOT1         ;TRAPS TO 4 GOTO BOOT1
044776                                   IF RLFLAG IS TRUE THEN $CALL UNRELOCATE
045010  004737  024656                   CALL    MT0030         ;FLUSH OUT DBE'S
045014  104421                           DEENERGIZE             ;TURN OFF MEMORY MANAGEMENT
045016  005737  002424                   TST     NO22BIT                  ;IS THIS AN 11/44 OR 11/24?
045022  001003                           BNE     BOOT1
045024  042737  000040  172516           BIC     #BIT5,MMR3                        ;TURN OFF THE UNIBUS MAP
045032  005001                   BOOT1:  CLR     R1
045034  000005                   1$:     RESET
045036  012700  177406                   MOV     #177406,R0
045042  010160  000004                   MOV     R1,4(R0)
045046  012710  177400                   MOV     #177400,(R0)
045052  012740  000005                   MOV     #5,-(R0)
045056  105710                   2$:     TSTB    (R0)
045060  100376                           BPL     2$
045062  062701  020000                   ADD     #BIT13,R1
045066  005710                           TST     (R0)
045070  100761                           BMI     1$
045072  005007                           CLR     PC
```

```
 7599 045074                                EXIT:   SUBTST  <<HALT PROGRAM>>
                                            ;*******************************************************************************
                                            ;*SUBTEST        HALT PROGRAM
                                            ;*******************************************************************************
 7600 045074  004737  045126                        CALL    SHUTUP
 7601 045100                                EXIT2:  IF APTFLAG IS TRUE OR ACTFLAG IS TRUE
 7602 045114  000777                                  BR     .
 7603 045116                                        ELSE
 7604 045120  000000                        $EXHALT:  HALT
 7605 045122  000137  003630                          JMP    START
 7606 045126                                        END ;OF IF APTFLAG
 7607
 7608 045126                                SHUTUP: SUBTST  <<SHUTDOWN DIAGNOSTIC>>
                                            ;*******************************************************************************
                                            ;*SUBTEST        SHUTDOWN DIAGNOSTIC
                                            ;*******************************************************************************
 7609                                               ;INITIALIZE ALL CSR'S
 7610                                               ;UNRELOCATE
 7611                                               ;FLUSH OUT DBE'S
 7612                                               ;RESTORE LOADERS
 7613                                               ;TURN OFF MEMORY MANAGEMENT
 7614                                               ;UNMAP THE UNIBUS MAP
 7618 045126  104472                                ECCINIT         ;TRAP ON DOUBLE BIT ERRORS (NORMAL)
 7619 045130                                        IF RLFLAG IS TRUE THEN $CALL UNRELOCATE
 7620 045142                                        IF QUICK IS FALSE
 7621 045150  004737  024656                          CALL  MT0030            ;FLUSH OUT DBE'S
 7622 045154                                        END ;OF IF QUICK
 7623 045154  012700  000001                        MOV     #1,R0           ;DESTINATION BANK
 7624 045160  013701  002536                        MOV     LOADHOME,R1     ;SOURCE BANK
 7625 045164  004737  043710                        CALL    BANKMOV
 7626 045170  104421                                DEENERGIZE              ;TURN OFF MEMORY MANAGEMENT
 7627 045172  005737  002424                        TST     NO22BIT         ;DOES THIS PDP-11 HAVE 22-BIT ADDR?
 7628 045176  001003                                BNE     1$              ;BRANCH IF NOT
 7629 045200  042737  000040  172516                BIC     #BIT5,MMR3              ;TURN OFF UNIBUS MAP
 7633 045206  000207                        1$:     RETURN
 7634
 7635 045210                                APTDOWN:SUBTST  <<APT SHUTDOWN SEQUENCE>>
                                            ;*******************************************************************************
                                            ;*SUBTEST        APT SHUTDOWN SEQUENCE
                                            ;*******************************************************************************
 7636 045210                                        MAP     #0                      ;MAP SUPERVISOR SPACE (TEST AREA) TO BANK #0
 7637 045224                                        TESTAREA                ;ENTER TEST MODE
 7638 045232  012737  045210  060024                MOV     #APTDOWN,FIRST+24
 7639 045240  012737  000340  060026                MOV     #340,FIRST+26
 7640 045246  012737  000000  125210                MOV     #0,FIRST+APTDOWN
 7641 045254  104417                                KERNEL                  ;ENTER KERNEL MODE
 7642 045256  000000                        APTHLT: HALT
```

```
   7645 045260                                     SUBTST  <<BLOCK MOVE SUBROUTINE>>
                                          ;*******************************************************************************
                                          ;*SUBTEST         BLOCK MOVE SUBROUTINE
                                          ;*******************************************************************************
   7646                                            ;BLOCK3 HAS 3 ARGUEMENTS
   7647                                            ;BLOCK2 HAS 2 ARGUEMENTS
   7648                                            ;BLOCK1 HAS 1 ARGUEMENTS
   7649                                            ;
   7650                                            ;ALL ARE CALLED BY THE BMOV MACRO
   7651                                            .ENABL  LSB
   7652 045260                            BLOCK1: PUSH    R0,R1,R2
   7653 045266  012702  177640                     MOV     #FASTCITY,R2
   7654 045272  012701  000020                     MOV     #16.,R1
   7655 045276  000413                             BR      3$
   7656
   7657 045300                            BLOCK2: PUSH    R0,R1,R2
   7658 045306  012701  000020                     MOV     #16.,R1
   7659 045312  000404                             BR      2$
   7660
   7661 045314                            BLOCK3: PUSH    R0,R1,R2
   7662 045322  012501                             MOV     (R5)+,R1
   7663 045324  012502                    2$:      MOV     (R5)+,R2
   7664 045326  012500                    3$:      MOV     (R5)+,R0
   7665
   7666 045330  012022                    1$:      MOV     (R0)+,(R2)+
   7667 045332  077102                             SOB     R1,1$
   7668 045334                                     POP     R2,R1,R0
   7669 045342  000205                             RTS     R5
   7670                                            .DSABL  LSB
```

```
                                        .SBTTL  FIELD SERVICE MODE

045344                          FIELDSERVICE:SUBTST      <<SUBR  FIELD SERVICE COMMAND MODE>>
                                ;*************************************************************************
                                ;*SUBTEST        SUBR    FIELD SERVICE COMMAND MODE
                                ;*************************************************************************
045344  104415                          SAVREG
045346                                  TYPE    MSG020          ;FIELD SERVICE COMMAND MODE

045352                                  IF RLFLAG IS TRUE OR NOFSMODE IS TRUE
045366                                    TYPE  MSG048          ;NOT AVAILABLE NOW - TRY LATER!
045372  104416                            RESREG
045374  000207                            RETURN
045376                                  END ;OF IF RLFLAG
045376  005737  002514                  TST     CACHKN
045402  001402                          BEQ     1$
045404                                  PUSH    CONTRL          ;SAVE CACHE STATUS
045410                          1$:     PUSH    CSRNO,KAMIKAZE  ;SAVE CSR & KAMIKAZE STATUS
045420  104424                          CACHOFF                 ;TURN CACHE OFF
045422                                  SET     KAMIKAZE
045430                          FS1:    TYPE    MSG026          ;COMMAND:
045434  104414                          RDDEC                   ;READ A DECIMAL NUMBER
045436                                  POP     R0              ;COMMAND --> R0
045440  020027  000022                  CMP     R0,#18.
045444  101403                          BLOS    1$
045446                                  TYPE    MSG021
045452  000766                          BR      FS1
045454                          1$:     CASE R0
045464  045542                            FSCMD0                ;EXIT FIELD SERVICE COMMANDS
045466  045644                            FSCMD1                ;READ CSR
045470  045754                            FSCMD2                ;LOAD CSR
045472  046122                            FSCMD3                ;EXAMINE MEMORY
045474  046376                            FSCMD4                ;MODIFY MEMORY
045476  046716                            FSCMD5                ;SELECT BANK & PATTERN
045500  047540                            FSCMD6                ;TYPE CONFIGURATION MAP
045502  047546                            FSCMD7                ;SOB-A-LONG TEST
045504  050040                            FSCMD8                ;ERROR SUMMARY
045506  050244                            FSCMD9                ;REFRESH TEST
045510  050536                            FCMD10                ;SET FILL COUNT
045512  050564                            FCMD11                ;ENTER KAMIKAZE MODE
045514  050606                            FCMD12                ;EXIT KAMIKAZE MODE
045516  050626                            FCMD13                ;TURN CACHE OFF
045520  050650                            FCMD14                ;TURN CACHE ON
045522  050666                            FCMD15                ;TEST ONLY SELECTED BANKS
045524  050752                            FCMD16                ;RESUME TESTING ALL BANKS
045526  051014                            FCMD17                ;ENABLE TRACE
045530  051030                            FCMD18                ;DISABLE TRACE
045532                                  END ;OF CASE
045540  000733                          BR      FS1
```

```
045542                        FSCMD0: SUBTST  <<COMMAND 0     EXIT>>
                              ;***********************************************************************
                              ;*SUBTEST         COMMAND 0       EXIT
                              ;***********************************************************************
045542                                TYPE    MSG103          ;LEAVING FIELD SERVICE MODE
045546  062706  000002                ADD     #2,SP
045552                                IF SKIPKAMI IS TRUE
045560  062706  000002                  ADD   #2,SP                     ;THROW AWAY OLD KAMIKAZE FLAG
045564  005037  002006                  CLR   SKIPKAMI
045570                                ELSE
045572                                  POP   KAMIKAZE                  ;RESTORE OLD KAMIKAZE FLAG
045576                                END ;OF IF SKIPKAMI
045576                                POP     CSRNO
045602  005737  002514                TST     CACHKN
045606  001414                        BEQ     RESO
045610                                IF CACHKN EQ CACHKF               ;IF CACHE IS OFF
045620  062706  000002                  ADD   #2,SP                     ;THROW AWAY CACHE STATUS
045624                                ELSE
045626  005737  002514                  TST   CACHKN
045632  001402                          BEQ   RESO
045634                                  POP   CONTRL                    ;RESTORE CACHE STATUS
045640                                END ;OF IF CACHKN
045640  104416                RESO:   RESREG
045642  000207                        RETURN

045644                        FSCMD1: SUBTST  <<FS    COMMAND 1       READ CSR>>
                              ;***********************************************************************
                              ;*SUBTEST         FS      COMMAND 1       READ CSR
                              ;***********************************************************************
045644  004737  051042                CALL    WHICHCSR
045650  010637  002266                MOV     SP,FSSTACK
045654                                SET4    #RES1           ;TRAPS TO 4 GOTO RES1
045662  104426                        READCSR
045664                                SET     NOERROR
045672  104026                        ERROR   +26             ;USE ERROR ROUTINE FOR PRINTOUT
045674                                RES4                    ;RESET TRAPS TO 4 TO DEFAULT
045716  000207                        RETURN
045720                        RES1:   TYPE    MSG025          ;THIS CSR DOES NOT EXIST
045724  013706  002266                MOV     FSSTACK,SP
045730                                RES4                    ;RESET TRAPS TO 4 TO DEFAULT
045752  000207                        RETURN
```

```
   7761 045754                           FSCMD2: SUBTST  <<FS     COMMAND 2        LOAD CSR>>
                                         ;***********************************************************************
                                         ;*SUBTEST        FS       COMMAND 2        LOAD CSR
                                         ;***********************************************************************
   7762 045754  004737  051042                   CALL    WHICHCSR
   7763 045760  010637  002266                   MOV     SP,FSSTACK
   7764 045764                                   SET4    #RES2                   ;TRAPS TO 4 GOTO RES2
   7765 045772  104426                           READCSR
   7766 045774                                   TYPE    MSG027
   7767 046000                                   SET     NOERROR
   7768 046006  104026                           ERROR   +26                     ;USE ERROR ROUTINE FOR PRINTOUT
   7769 046010                                   RES4                            ;RESET TRAPS TO 4 TO DEFAULT
   7770 046032                                   TYPE    MSG023                  ;FIRST CSR WORD
   7771 046036  104413                           RDOCT                           ;READ AN OCTAL NUMBER
   7772 046040                                   POP     CSR                     ;PUT IN IN LOC "CSR"
   7773 046044  104425                           LOADCSR
   7774 046046  104426                           READCSR
   7775 046050                                   TYPE    MSG028
   7776 046054                                   SET     NOERROR
   7777 046062  104026                           ERROR   +26                     ;USE FOR PRINTOUT - NOT AN ERROR
   7778 046064  000207                           RETURN
   7779 046066                           RES2:   TYPE    MSG025                  ;THIS CSR DOES NOT EXIST
   7780 046072  013706  002266                   MOV     FSSTACK,SP
   7781 046076                                   RES4                            ;RESET TRAPS TO 4 TO DEFAULT
   7782 046120  000207                           RETURN
   7785 046122                             FSCMD3: SUBTST  <<FS    COMMAND 3       EXAMINE MEMORY>>
                                           ;*****************************************************************************
                                           ;*SUBTEST        FS      COMMAND 3       EXAMINE MEMORY
                                           ;*****************************************************************************
   7786 046122                                     PUSH    BANK,NOPAR,PARTHERE,4
   7787 046142  012737  000002  002074             MOV     #2,NOPAR        ;INDICATE PARITY ACTION
   7788 046150                                     TYPE    MSG029          ;EXAMINE MEMORY
   7789 046154                             1$:     TYPE    MSG031          ;PHYSICAL ADDRESS (0-17775776)??
   7790 046160  104413                             RDOCT                   ;READ OCTAL NUMBER ONTO STACK & $HIOCT
   7791 046162  013737  061766  002100             MOV     $HIOCT,BANK     ;PUT MSB'S IN BANK
   7792 046170                                     POP     R0              ;PUT LSB'S IN R0
   7793 046172  000241                             CLC
   7794 046174  006100                             ROL     R0
   7795 046176  006137  002100                     ROL     BANK
   7796 046202  000241                             CLC
   7797 046204  006000                             ROR     R0
   7798 046206  023737  002100  002526             CMP     BANK,LASTBANK   ;CHECK FOR BANK TOO HIGH
   7799 046214  003357                             BGT     1$              ;BRANCH IF TRUE
   7800 046216  062700  060000                     ADD     #FIRST,R0
   7801 046222  032700  000001                     BIT     #BIT0,R0        ;CHECK FOR ODD ADDRESS
   7802 046226  001352                             BNE     1$              ;BRANCH IF ODD ADDRESS
   7803 046230  020027  157776                     CMP     R0,#LAST        ;CHECK FOR ADDRESS OVER 16K
   7804 046234  101347                             BHI     1$              ;BRANCH IF OVER 16K
   7805 046236  012737  046310  002264             MOV     #3$,PARTHERE    ;INCASE OF ABORTS
   7806 046244                                     SET4    #4$             ;TRAPS TO 4 GOTO 4$
   7807 046252                                     MAP     BANK                    ;MAP SUPERVISOR SPACE (TEST AREA) TO BANK
   7808 046266                                     TESTAREA                ;ENTER TEST MODE
   7809 046274  011001                             MOV     (R0),R1
   7810 046276  104417                             KERNEL                  ;ENTER KERNEL MODE
   7811 046300                                     TYPOCS  R1
   7812 046306  000410                             BR      EXCMD3
   7813
   7814 046310                             3$:     TYPE    MSG032                  ;PARITY ABORT
   7815 046314  000405                             BR      EXCMD3
   7816
   7817 046316  062706  000004             4$:     ADD     #4,SP                   ;FIX STACK
   7818 046322                                     TYPE    MSG033                  ;TIMEOUT TRAP
   7819 046326  000400                             BR      EXCMD3
   7820
   7821 046330  104417                     EXCMD3: KERNEL                  ;ENTER KERNEL MODE
   7822 046332                                     POP     4,PARTHERE,NOPAR,BANK
   7823 046352                                     RES4                    ;RESET TRAPS TO 4 TO DEFAULT
   7824 046374  000207                             RETURN
```

```
046376                                  FSCMD4: SUBTST  <<FS     COMMAND 4        MODIFY MEMORY>>
                                        ;*********************************************************************************
                                        ;*SUBTEST        FS      COMMAND 4        MODIFY MEMORY
                                        ;*********************************************************************************
046376                                          PUSH    BANK,NOPAR,PARTHERE,4
046416  012737  000003  002074                  MOV     #3,NOPAR                ;INDICATE PARITY ACTION
046424                                          TYPE    MSG036                  ;MODIFY MEMORY
046430                                  1$:     TYPE    MSG031                  ;PHYSICAL ADDRESS (0-17775776)??
046434  104413                                  RDOCT                           ;READ OCTAL NUMBER ONTO STACK & $HIOCT
046436  013737  061766  002100                  MOV     $HIOCT,BANK             ;PUT MSB'S IN BANK
046444                                          POP     R0                      ;PUT LSB'S IN R0
046446  000241                                  CLC
046450  006100                                  ROL     R0
046452  006137  002100                          ROL     BANK
046456  000241                                  CLC
046460  006000                                  ROR     R0
046462                                          IF BANK GT LASTBANK THEN GOTO 1$  ;CHECK FOR BANK TOO HIGH
046472  062700  060000                          ADD     #FIRST,R0
046476                                          IF #BIT0 SET.IN R0 THEN GOTO 1$ ;CHECK FOR ODD ADDRESS
046504                                          IF R0 HI #LAST THEN GOTO 1$     ;CHECK FOR ADDRESS OVER 16K
046512  012737  046560  002264                  MOV     #3$,PARTHERE            ;INCASE OF ABORTS
046520                                          SET4    #4$                     ;TRAPS TO 4 GOTO 4$
046526                                          MAP     BANK                    ;MAP SUPERVISOR SPACE (TEST AREA) TO BANK
046542  104511                                  INVALIDATE
046544                                          TESTAREA                        ;ENTER TEST MODE
046552  011001                                  MOV     (R0),R1
                                                ;GETTING HERE MEANS WE GOT LUCKY - NO TRAPS
046554  104417                                  KERNEL                          ;ENTER KERNEL MODE
046556  000410                                  BR      5$

046560                                  3$:     TYPE    MSG032                  ;PARITY ABORT
046564  000431                                  BR      EXCMD4                  ;EXIT

046566  062706  000004                  4$:     ADD     #4,SP                   ;FIX STACK
046572                                          TYPE    MSG033                  ;TIMEOUT TRAP
046576  000424                                  BR      EXCMD4                  ;EXIT

046600                                  5$:     TYPE    MSG037                  ;OLD DATA WAS
046604                                          TYPOCS  R1                      ;PRINT IT
046612                                          TYPE    MSG039                  ;INPUT NEW DATA
046616  104413                                  RDOCT                           ;READ ON OCTAL NUMBER ONTO THE STACK
046620                                          POP     R1                      ;GET NEW NUMBER
046622                                          TESTAREA                        ;ENTER TEST MODE
046630  010110                                  MOV     R1,(R0)                 ;PUT IT IN MEMORY
046632  011001                                  MOV     (R0),R1                 ;READ IT AGAIN
046634  104417                                  KERNEL                          ;ENTER KERNEL MODE
046636                                          TYPE    MSG038                  ;DATA IS NOW
046642                                          TYPOCS  R1                      ;PRINT IT

046650  104417                          EXCMD4: KERNEL                          ;ENTER KERNEL MODE
046652                                          POP     4,PARTHERE,NOPAR,BANK
046672                                          RES4                            ;RESET TRAPS TO 4 TO DEFAULT
046714  000207                                  RETURN
```

```
 7879 046716                                 FSCMD5: SUBTST  <<FS     COMMAND 5         SELECT BANK & PATTERN>>
                                             ;*******************************************************************************
                                             ;*SUBTEST        FS      COMMAND 5         SELECT BANK & PATTERN
                                             ;*******************************************************************************
 7880 046716                                         PUSH    BANK,PATTERN,TESTADD,PCBUMP,TKVEC,TKVEC+2
 7881 046746  010637  002266                         MOV     SP,FSSTACK                 ;SAVE LAST GOOD STACK POINTER
 7882 046752                                         TYPE    MSG040                     ;SELECT BANK & PATTERN TEST
 7883 046756                                 1$:     TYPE    MSG030                     ;BANK(0-177)?
 7884 046762  104413                                 RDOCT                              ;READ AN OCTAL NUMBER ONTO THE STACK
 7885 046764                                         POP     BANK                       ;PUT IT IN BANK
 7886 046770                                         IF BANK GT LASTBANK THEN GOTO 1$   ;CHECK FOR BANK TOO HIGH
 7887
 7888 047000  013701  002100                         MOV     BANK,R1
 7889 047004  006301                                 ASL     R1
 7890 047006  006301                                 ASL     R1
 7891 047010                                         IF CPUBIT OFF.IN CONFIG(R1)
 7892 047020                                           TYPE  MSG041                     ;BANK NOT ACCESSABLE
 7893 047024                                           GOTO 1$
 7894 047026                                         END ;OF IF
 7895
 7896 047026                                 2$:     TYPE    MSG042                     ;PATTERN(0-35)?
 7897 047032  104413                                 RDOCT                              ;READ AN OCTAL NUMBER ONTO THE STACK
 7898 047034                                         POP     PATTERN                    ;PUT IT IN PATTERN
 7899 047040                                         IF PATTERN GT #35 THEN GOTO 2$     ;CHECK FOR PATTERN TO HIGH
 7900 047050                                         IF PATTERN EQ #0
 7901 047056                                           TYPE  MSG043                     ;PATTERN 0 DATA IS?
 7902 047062  104413                                   RDOCT                            ;READ AN OCTAL NUMBER ONTO THE STACK
 7903 047064                                           POP   R2                         ;PUT IT IN R2
 7904 047066                                         END ;OF IF
 7905
 7906
 7907 047066                                         MAP     BANK                       ;MAP SUPERVISOR SPACE (TEST AREA) TO BANK
 7908 047102  104511                                 INVALIDATE
 7909 047104  004737  044240                         CALL    EXBANK                     ;SET NEW MARGINS
 7910 047110                                         IF RRFLAG IS TRUE
 7911 047116                                           TYPE  MSG049                     ;BANK REQUIRES RELOCATION
 7912 047122                                           GOTO CMD5C
 7913 047124                                         END ;OF IF RRFLAG
 7914 047124                                         TYPE    MSG046            ;TO ESCAPE TYPE ANY KEY!
 7915 047130  012737  047450  000060                 MOV     #CMD5C,TKVEC
 7916 047136  012737  000340  000062                 MOV     #340,TKVEC+2
 7917 047144  017700  133434                         MOV     @$TKB,R0                   ;KILL ANY OLD INTERRUPT
 7918 047150  042737  000200  177776                 BIC     #BIT7,PSW                  ;LOWER CPU PRIORITY TO 140
 7919 047156  052777  000100  133416                 BIS     #BIT6,@$TKS                ;ENABLE KEYBOARD INTERRUPTS
 7920
 7921
 7922 047164                                         SET     HEADER,MUT
 7923 047200  013701  002100                 CMD5B:  MOV     BANK,R1
 7924 047204  006301                                 ASL     R1
 7925 047206  006301                                 ASL     R1
 7926 047210  005037  002232                         CLR     SPLTCSR
 7927 047214  005037  002256                         CLR     PASFLG
 7928 047220  012737  060000  002362                 MOV     #FIRST,TESTADD
 7929 047226  012737  060002  002364                 MOV     #FIRST+2,TESTADD+2
 7930 047234                                         IF #BIT12 SET.IN CONFIG+2(R1)
 7931 047244  005237  002232                           INC   SPLTCSR
 7932 047250                                           MAP   BANK
```

```
047264  012737  120000  002364                     MOV     #120000,TESTADD+2
047272                                           END; OF IF #BIT12
047272                                           IF #SW0 SET.IN @SWR
047302  104470                                     ECCDIS                          ;DISABLE ERROR CORRECTION
047304                                           ELSE
047306                                             PUSH   CSRNO
047312  104502                                     CLRCSR                          ;CLEAR CSRS
047314                                             POP    CSRNO
047320                                           END ;OF IF
047320  012737  000002  002074                   MOV     #2,NOPAR                  ;PARITY ACTION
047326  012737  000002  002276                   MOV     #2,PCBUMP                 ;TRAPS ADD 2 TO PC
047334  013700  002110                           MOV     PATTERN,R0
047340  006300                                   ASL     R0
047342  004770  047354                           CALL    @FSPAT(R0)
047346  005037  002074                           CLR     NOPAR
047352  000712                                   BR      CMD5B                     ;LOOP TILL KEYBOARD INTERRUPT

047354  020062                         FSPAT:    MT0000  ;<1 SEC       DATA PATTERN TEST
047356  020142                                   MT0001  ;<1 SEC       ADDRESS TEST
047360  020262                                   MT0002  ;<1 SEC       COMPLEMENT ADDRESS TEST
047362  020422                                   MT0003  ; 1 SEC       3 XOR 9 WORST CASE NOISE TEST
047364  020654                                   MT0004  ; 1 SEC       ROTATING ZEROS TEST
047366  020776                                   MT0005  ; 1 SEC       ROTATING ONES TEST
047370  021132                                   MT0006  ;<1 SEC       INITIAL DATA TEST
047372  021166                                   MT0007  ;<1 SEC       ADDRESS BIT TEST
047374  021230                                   MT0010  ;<1 SEC       BYTE ADDRESSING TEST
047376  021264                                   MT0011  ;<2 SEC       CREATE SINGLE BIT ERROR TEST
047400  021332                                   MT0012  ;<1 SEC       WRITE BYTE CLEARS SBE TEST
047402  021426                                   MT0013  ; 1 SEC       CREATE DOUBLE BIT ERROR TEST
047404  021502                                   MT0014  ; 1 SEC       WRITE INHIBIT DURING DATIP WITH DBE
047406  021560                                   MT0015  ; 1 SEC       WRITE INHIBIT OF BYTE WITH DBE
047410  021626                                   MT0016  ;<1 SEC       WRITE INHIBIT OF WORD WITH DBE
047412  021674                                   MT0017  ;<1 SEC       HOLDING 1'S & 0'S TEST
047414  021716                                   MT0020  ;<1 SEC       MARCHING 1'S & 0'S IN CHECK BITS
047416  023006                                   MT0021  ; 1 SEC       MARCHING 0'S & 1'S TEST
047420  023260                                   MT0022  ;10 SEC       REFRESH & SHIFTING DIAGONAL TEST
047422  023312                                   MT0023  ;10 SEC       SHIFTING DIAGONAL TEST
047424  023356                                   MT0024  ;20 SEC       FAST GALLOPING PATTERN TEST
047426  023622                                   MT0025  ;<1 SEC       INTERRUPT ENABLE TEST
047430  023670                                   MT0026  ;<1 SEC       RANDOM DATA TEST
047432  024172                                   MT0027  ; 1 SEC       UNIQUE BANK TEST
047434  024656                                   MT0030  ; 1 SEC       FLUSH OUT DBE'S TEST
047436  025160                                   MT0031  ; 3 SEC       SOB-A-LONG TEST
047440  025350                                   MT0032  ;<1 SEC       WRITE RECOVERY TEST
047442  025702                                   MT0033  ;35 SEC       BRANCH GOBBLE TEST
047444  026070                                   MT0034  ; 1 SEC       SOFT ERROR TEST
047446  026242                                   MT0035  ;<1 SEC       WORST CASE NOISE PARITY TEST

047450  013706  002266                 CMD5C:    MOV     FSSTACK,SP                ;RECOVER OLD STACK POINTER
047454  042777  000100  133120                   BIC     #BIT6,@$TKS
047462                                           POP     TKVEC+2,TKVEC
047472  117700  133106                           MOVB    @$TKB,R0                  ;GET CHARACTER TO GET RID OF FLAG
047476                                           POP     PCBUMP,TESTADD
047506                                           POP     PATTERN,BANK
047516                                           MAP     BANK                      ;REMAP OLD BANK
047532  004737  044240                           CALL    EXBANK
047536  000207                                   RETURN
```

```
   7991 047540                               FSCMD6: SUBTST  <<FS     COMMAND 6        TYPE CONFIGURATION MAP>>
                                             ;*****************************************************************************
                                             ;*SUBTEST        FS     COMMAND 6        TYPE CONFIGURATION MAP
                                             ;*****************************************************************************
   7992 047540  004737  036570                       CALL    PCONFIG
   7993 047544  000207                               RETURN
   7994
```

```
047546                           FSCMD7: SUBTST  <<FS     COMMAND 7       SOB-A-LONG TEST>>
                                 ;*******************************************************************************
                                 ;*SUBTEST        FS      COMMAND 7       SOB-A-LONG TEST
                                 ;*******************************************************************************
047546                                   PUSH    BANK,PATTERN,TKVEC,TKVEC+2,NOPAR
047572  010637  002266                   MOV     SP,FSSTACK              ;SAVE LAST GOOD STACK POINTER
047576                                   TYPE    MSG055                  ;SOB-A-LONG TEST

047602                                   IF #SW0 SET.IN @SWR
047612  104470                             ECCDIS                        ;DISABLE ERROR CORRECTION
047614                                   ELSE
047616  104502                             CLRCSR                        ;CLEAR CSRS
047620                                   END ;OF IF
047620                                   TYPE    MSG056                  ;BELL = EACH PASS COMPLETE

047624                                   TYPE    MSG046                  ;TO ESCAPE TYPE ANY KEY!
047630  012737  047754  000060           MOV     #CMD7C,TKVEC
047636  012737  000340  000062           MOV     #340,TKVEC+2
047644  017700  132734                   MOV     @$TKB,R0                ;KILL ANY OLD INTERRUPT
047650  042737  000200  177776           BIC     #BIT7,PSW               ;LOWER CPU PRIORITY TO 140
047656  052777  000100  132716           BIS     #BIT6,@$TKS             ;ENABLE KEYBOARD INTERRUPTS


047664                                   SET     HEADER,MUT

047700                           CMD7B:  FOR BANK := #0 TO LASTBANK
047704  004737  044240                     CALL EXBANK
047710                                     IF ACFLAG IS TRUE AND RRFLAG IS FALSE
047724  104511                               INVALIDATE
047726  004737  025160                       CALL MT0031
047732                                     END ;OF IF ACFLAG
047732                                   END ;OF FOR BANK
047746                                   TYPE    $BELL                   ;RING BELL
047752                                   GOTO    CMD7B

047754  013706  002266           CMD7C:  MOV     FSSTACK,SP                      ;RECOVER OLD STACK POINTER
047760  042777  000100  132614           BIC     #BIT6,@$TKS
047766  117700  132612                   MOVB    @$TKB,R0                        ;READ CHAR TO KILL FLAG
047772                                   POP     NOPAR,TKVEC+2,TKVEC,PATTERN,BANK
050016                                   MAP     BANK                    ;MAP SUPERVISOR SPACE (TEST AREA) TO BANK
050032  004737  044240                   CALL    EXBANK
050036  000207                           RETURN
```

```
  8038 050040                           FSCMD8: SUBTST  <<FS     COMMAND 8       ERROR SUMMARY>>
                                        ;*********************************************************************************
                                        ;*SUBTEST        FS      COMMAND 8       ERROR SUMMARY
                                        ;*********************************************************************************
  8039 050040                                   PUSH    R0,R2,R3,BANK
  8040 050052  013737  062474  002404           MOV     $PASS,TEMP
  8041 050060  005337  002404                   DEC     TEMP
  8042 050064                                   TYPDEC  TEMP
  8043 050072                                   TYPE    MSG125                  ;PASSES COMPLETED
  8044 050076                                   TYPDEC  $ERTTL
  8045 050104                                   TYPE    MSG079                  ;ERROR(S) DETECTED
  8046 050110                                   IF $ERTTL NE #0
  8047 050116  005037  002304                     CLR   SUCCESS
  8048 050122                                     FOR BANK := #0 TO LASTBANK
  8049 050126  013703  002100                       MOV BANK,R3
  8050 050132  070327  000004                       MUL #4,R3
  8051 050136                                       IFB CONFIG+2(R3) NE #0
  8052 050144                                         IF SUCCESS IS FALSE
  8053 050152                                           TYPE    MSG076          ;BANK  ERRORS
  8054 050156                                           SET SUCCESS
  8055 050164                                         END ;OF IF SUCCESS
  8056 050164                                         TYPOCS    BANK,3
  8057 050174  116300  002626                         MOVB      CONFIG+2(R3),R0
  8058 050200  042700  177400                         BIC       #^C377,R0
  8059 050204                                         TYPDEC    R0
  8060 050210                                         TYPE      $CRLF
  8061 050214                                       END ;OF IFB CONFIG(R3)
  8062 050214                                     END ;OF FOR BANK
  8063 050230                                   END ;OF IF $ERTTL
  8064 050230                                   POP     BANK,R3,R2,R0
  8065 050242  000207                           RETURN
```

```
050244                          FSCMD9: SUBTST  <<FS    COMMAND 9               REFRESH TEST>>
                                ;*******************************************************************************
                                ;*SUBTEST       FS      COMMAND 9               REFRESH TEST
                                ;*******************************************************************************
050244                                  PUSH    BANK,PATTERN,TKVEC,TKVEC+2,NOPAR
050270  010637  002266                  MOV     SP,FSSTACK              ;SAVE LAST GOOD STACK POINTER
050274                                  TYPE    MSG073                  ;REFRESH TEST

050300                                  IF #SW0 SET.IN @SWR
050310  104470                            ECCDIS                        ;DISABLE ERROR CORRECTION
050312                                  ELSE
050314  104502                            CLRCSR                        ;CLEAR CSRS
050316                                  END ;OF IF
050316                                  TYPE    MSG056                  ;BELL = EACH PASS COMPLETE

050322                                  TYPE    MSG046                  ;TO ESCAPE TYPE ANY KEY!
050326  012737  050452  000060          MOV     #CMD9C,TKVEC
050334  012737  000340  000062          MOV     #340,TKVEC+2
050342  017700  132236                  MOV     @$TKB,R0                ;KILL ANY OLD INTERRUPT
050346  042737  000200  177776          BIC     #BIT7,PSW               ;LOWER CPU PRIORITY TO 140
050354  052777  000100  132220          BIS     #BIT6,@$TKS             ;ENABLE KEYBOARD INTERRUPTS

050362                                  SET     HEADER,MUT

050376                          CMD9B:  FOR BANK := #0 TO LASTBANK
050402  004737  044240                    CALL EXBANK
050406                                    IF ACFLAG IS TRUE AND RRFLAG IS FALSE
050422  104511                              INVALIDATE
050424  004737  023260                      CALL MT0022
050430                                    END ;OF IF ACFLAG
050430                                  END ;OF FOR BANK
050444                                  TYPE    $BELL                   ;RING BELL
050450                                  GOTO    CMD9B

050452  013706  002266          CMD9C:  MOV     FSSTACK,SP              ;RECOVER OLD STACK POINTER
050456  042777  000100  132116          BIC     #BIT6,@$TKS
050464  117700  132114                  MOVB    @$TKB,R0                ;READ CHAR TO KILL FLAG
050470                                  POP     NOPAR,TKVEC+2,TKVEC,PATTERN,BANK
050514                                  MAP     BANK                    ;MAP SUPERVISOR SPACE (TEST AREA) TO BANK
050530  004737  044240                  CALL    EXBANK
050534  000207                          RETURN

```

```
050536                                 FCMD10: SUBTST  <<FS    COMMAND 10      SET FILL COUNT>>
                                       ;*******************************************************************************
                                       ;*SUBTEST        FS      COMMAND 10      SET FILL COUNT
                                       ;*******************************************************************************
050536                                         PUSH    R0
050540                                         TYPE    MSG085                  ;FILL COUNT(OCTAL)?
050544  104413                                 RDOCT
050546                                         POP     R0
050550  042700  177760                         BIC     #^C17,R0
050554  110037  002327                         MOVB    R0,$FILLS
050560                                         POP     R0
050562  000207                                 RETURN

050564                                 FCMD11: SUBTST  <<FS    COMMAND 11      ENTER KAMIKAZE MODE>>
                                       ;*******************************************************************************
                                       ;*SUBTEST        FS      COMMAND 11      ENTER KAMIKAZE MODE
                                       ;*******************************************************************************
050564                                         TYPE    MSG101          ;ENTERING KAMIKAZE MODE
050570                                         SET     KAMIKAZE,SKIPKAMI
050604  000207                                 RETURN

050606                                 FCMD12: SUBTST  <<FS    COMMAND 12      EXIT KAMIKAZE MODE>>
                                       ;*******************************************************************************
                                       ;*SUBTEST        FS      COMMAND 12      EXIT KAMIKAZE MODE
                                       ;*******************************************************************************
050606                                         TYPE    MSG102          ;LEAVING KAMIKAZE MODE
050612  005037  002004                         CLR     KAMIKAZE
050616                                         SET     SKIPKAMI
050624  000207                                 RETURN

050626                                 FCMD13: SUBTST  <<FS    COMMAND 13      TURN CACHE OFF>>
                                       ;*******************************************************************************
                                       ;*SUBTEST        FS      COMMAND 13      TURN CACHE OFF
                                       ;*******************************************************************************
050626                                         TYPE    MSG106                  ;CACHE IS OFF
050632  104424                                 CACHOFF                         ;TURN CACHE OFF
050634  013737  002514  002516                 MOV     CACHKN,CACHKN+2         ;SAVE OLD CACHE ON STATE
050642  005037  002514                         CLR     CACHKN                  ;KEEP CACHE OFF
050646  000207                                 RETURN

050650                                 FCMD14: SUBTST  <<FS    COMMAND 14      TURN CACHE ON>>
                                       ;*******************************************************************************
                                       ;*SUBTEST        FS      COMMAND 14      TURN CACHE ON
                                       ;*******************************************************************************
050650                                         TYPE    MSG107                  ;CACHE IS ON (EXCEPT DURING ACTUAL PATTERNS)
050654  013737  002516  002514                 MOV     CACHKN+2,CACHKN         ;RESTORE OLD CACHE ON STATE
050662  104423                                 CACHON                          ;TURN CACHE ON
050664  000207                                 RETURN
```

```
  8155
  8156 050666                         FCMD15: SUBTST  <<FS    COMMAND 15      TEST ONLY SELECTED BANKS>>
                                      ;*******************************************************************************
                                      ;*SUBTEST        FS      COMMAND 15      TEST ONLY SELECTED BANKS
                                      ;*******************************************************************************
  8157 050666                                 TYPE    MSG105          ;ENTER BANKS IN OCTAL - USE NUMBER OUTSIDE RANGE TO TERMINAT
  8158 050672  004737  050762                 CALL    CMD16A                  ;ERASE OLD SELECTIONS
  8159 050676                                 BEGIN CMD16LOOP
  8160 050676                                   REPEAT
  8161 050676                                     TYPE          MSG030        ;BANK(0-177)?
  8162 050702  104413                             RDOCT                       ;READ AN OCTAL NUMBER ONTO THE STACK
  8163 050704                                     POP R1                      ;PUT IT IN R1
  8164 050706                                     IF R1 GT #177 OR R1 LT #0
  8165 050720                                       LEAVE CMD16LOOP
  8166 050722                                     END ;OF IF R1
  8167 050722  006301                             ASL  R1
  8168 050724  006301                             ASL  R1                     ;R1 <- R1 * 4
  8169 050726  052761  040000  002626             BIS #BIT14,CONFIG+2(R1)
  8170 050734                                   END ;OF REPEAT
  8171 050736                                 END CMD16LOOP
  8172 050736                                 TYPE    MSG110                  ;ONLY SELECTED BANKS WILL BE TESTED
  8173 050742                                 SET     SELONLY
  8174 050750  000207                         RETURN
  8175
  8176 050752                         FCMD16: SUBTST  <<FS    COMMAND 16      RESUME TESTING ALL BANKS>>
                                      ;*******************************************************************************
                                      ;*SUBTEST        FS      COMMAND 16      RESUME TESTING ALL BANKS
                                      ;*******************************************************************************
  8177 050752                                 TYPE    MSG111                  ;ALL BANKS WILL BE TESTED
  8178 050756  005037  002000                 CLR     SELONLY
  8179
  8180                                        ;ENTRY POINT FROM CMD15
  8181 050762  013702  002526         CMD16A: MOV     LASTBANK,R2
  8182 050766  006302                         ASL     R2
  8183 050770  006302                         ASL     R2
  8184 050772                                 FOR R1 := #0 TO R2 BY #4
  8185 050774  042761  040000  002626           BIC   #BIT14,CONFIG+2(R1)
  8186 051002                                 END ;OF FOR R1
  8187 051012  000207                         RETURN
```

```
   8190 051014                                       FCMD17: SUBTST  <<FS    COMMAND 17      ENABLE TRACE>>
                                                     ;*****************************************************************************
                                                     ;*SUBTEST        FS      COMMAND 17      ENABLE TRACE
                                                     ;*****************************************************************************
   8191 051014                                                TYPE    MSG127
   8192 051020  012737  177777  006104                        MOV     #-1,TRACE
   8193 051026  000207                                        RETURN
   8196 051030                                FCMD18: SUBTST  <<FS     COMMAND 18      DISABLE TRACE>>
                                              ;*****************************************************************************
                                              ;*SUBTEST         FS      COMMAND 18      DISABLE TRACE
                                              ;*****************************************************************************
   8197 051030                                        TYPE    MSG128
   8198 051034  005037  006104                        CLR     TRACE
   8199 051040  000207                                RETURN
```

```
051042                         WHICHCSR:SUBTST <<SUBR  DETERMINE CORRECT CSR>>
                               ;*************************************************************************
                               ;*SUBTEST        SUBR    DETERMINE CORRECT CSR
                               ;*************************************************************************
051042  013700  002216                  MOV     TOTCSRS,R0          ;GET CSR'S FLAG
051046  022700  100000                  CMP     #BIT15,R0           ;CSR 0?
051052  001003                          BNE     1$                  ;NO - SKIP
051054  005037  002146                  CLR     CSRNO               ;YES - SET IT UP
051060  000207                          RETURN

051062                         1$:      TYPE    MSG022              ;WHICH CSR(0-F)
051066  104412                          RDLIN                       ;GET CHARACTER
051070                                  POP     R0                  ;PUT IN R0
051072  011000                          MOV     (R0),R0             ;PUT CHAR IN R0
051074  020027  000106                  CMP     R0,#106             ;CHECK LIMIT
051100  101370                          BHI     1$                  ;IF BAD LOOP TILL HE TYPES IT RIGHT
051102  022700  000101                  CMP     #'A,R0
051106  103002                          BHIS    2$
051110  162700  000007                  SUB     #7,R0
051114  162700  000060         2$:      SUB     #60,R0
051120  006300                          ASL     R0
051122  010037  002146                  MOV     R0,CSRNO
051126  000207                          RETURN
```

```
                                      .SBTTL  ERROR DATA (SUPERVISOR) SETUP STUFF
051130                        $PER25: LET ADDRESS := R1 - #2
051142                                IF ABORTFLAG IS FALSE
051150                                  TESTAREA                ;ENTER TEST MODE
051156                                  LET BAD := -2(R1)
051164  104417                          KERNEL                  ;ENTER KERNEL MODE
051166                                END ;OF IF ABORTFLAG
051166                                IF 177654 EQ #0
051174                                  LET GOOD := R2
051200                                ELSE
051202                                  LET GOOD := R3
051206                                END ;OF IF
051206  000137  054410                JMP     PERRAW

051212                        PERRA3: SUBTST  <<DATA WAS 3 WORDS>>
                              ;*******************************************************************'
                              ;*SUBTEST        DATA WAS 3 WORDS
                              ;*******************************************************************
051212                                IF BADPC EQ #0 THEN $CALL BADSTACK
051224                                PUSH    R0
051226  005037  002144                CLR     CSR             ;MAKE SURE CSR BIT HOLDER IS CLEAR
051232  104505                        CHK1DIS                 ;DISABLE ECC & WRITE CHECKBITS FROM 1 SELECTED CSR
051234                                TESTAREA
051242  005711                        TST     (R1)            ;READ LOCATION TO READ CHECKBITS INTO CSR
051244  104417                        KERNEL
051246  104426                        READCSR                 ;GET CSR CONTENTS
051250  013700  002144                MOV     CSR,R0          ;SAVE CSR CONTENTS IN R0
051254  104503                        CLR1CSR                 ;RETURN CSR TO NORMAL MODE
051256  072027  177773                ASH     #-5,R0          ;MOVE CHECK BITS TO BOTTOM OF WORD
051262  042700  177600                BIC     #^C177,R0       ;CLEAR OFF EXTRANEOUS GARBAGE
051266                                LET ADDRESS := R1       ;SAVE VIRTUAL ADDRESS FOR PRINTOUT
051272  005037  002042                CLR     GOOD            ;FIRST TEST WORD WRITTEN SHOULD ALWAYS BE ZERO
051276                                TESTAREA                ;ENTER TEST MODE
051304  011137  002050                MOV     (R1),BAD        ;GET BAD DATA FROM MUT - FIRST WORD
051310  011437  002052                MOV     (R4),BAD2       ;AND SECOND WORD
051314  104417                        KERNEL                  ;ENTER KERNEL MODE
051316  110037  002054                MOVB    R0,BAD3         ;MOVE BAD CHECKBITS FOR PRINTOUT
051322  105037  002055                CLRB    BAD3+1          ;CLEAR OFF THE OTHER UNUSED BITS
051326  004737  054644                CALL    PERBNK          ;MARK BANK AS BAD IN CONFIG TABLE
051332  104033                        ERROR   +33
051334                                POP     R0              ;RESTORE R0
051336                                IF #SW0 SET.IN @SWR
051346  104506                          ENASBE                ;TRAP ON SINGLE BIT ERRORS
051350                                ELSE
051352  104472                          ECCINIT               ;TRAP ON UNCORRECTABLE ERRORS
051354                                END; OF IF #SW0
051354  000002                        RTI
```

```
 8816 051356                              $PER30: LET GOOD := R1
 8817 051362                                      LET ADDRESS := (SP) - 16
 8818 051374                                      IF ABORTFLAG IS FALSE
 8819 051402                                        TESTAREA              ;ENTER TEST MODE
 8820 051410                                        LET BAD := @ADDRESS
 8821 051416 104417                                 KERNEL                        ;ENTER KERNEL MODE
 8822 051420                                      END ;OF IF ABORTFLAG
 8823 051420 000137 054410                        JMP PERRAW
 8824
 8825 051424                              GETDATA:SUBTST  <<GET DATA FROM ABORTED AREA IF POSSIBLE>>
                                          ;*******************************************************************************
                                          ;*SUBTEST        GET DATA FROM ABORTED AREA IF POSSIBLE
                                          ;*******************************************************************************
 8826 051424                                      PUSH    R0,4,114
 8827 051436 010637 051522                        MOV     SP,GETDA1
 8828 051442 012737 051502 000004                 MOV     #1$,4
 8829 051450 012737 051502 000114                 MOV     #1$,114
 8830 051456 013700 002032                        MOV     ADDRESS,R0
 8831 051462                                      TESTAREA
 8832 051470 011037 002050                        MOV     (R0),BAD
 8833 051474 104417                               KERNEL
 8834 051476 005037 002140                        CLR     ABORTFLAG
 8835 051502 013706 051522                1$:     MOV     GETDA1,SP               ;RESTORE KNOWN GOOD STACK POINTER
 8836 051506                                      POP     114,4,R0
 8837 051520 000207                               RETURN
 8838 051522 000000                       GETDA1: 0
```

```
                                                .SBTTL  POWER FAIL AUTO RESTART
                                                .SBTTL  ROUTINE POWER DOWN AND UP
                                        ;*************************************************************************
                                        ;POWER DOWN ROUTINE
051524                                  $PWRDN:
                                                ;SAVE CACHE STATUS
051524  005737  002514                          TST     CACHKN
051530  001403                                  BEQ     5$
051532                                          PUSH    CONTRL
051536  104423                                  CACHON                  ;TURN CACHE ON
051540  012737  052476  000024          5$:     MOV     #$ILLUP,PWRVEC ;;SET FOR FAST UP
051546  012737  000340  000026                  MOV     #340,PWRVEC+2   ;;PRIO:7
051554                                          PUSH    R0,R1,R2,R3,R4,R5,CSRNO
                                                ;SAVE USER PAR'S & PDR7
051574  012700  177700                          MOV     #177700,R0
051600  012701  000021                          MOV     #17.,R1
051604                                  1$:     PUSH    -(R0)
051606  077102                                  SOB     R1,1$
                                                ;SAVE SUPERVISOR PAR'S
051610  005737  002426                          TST     NOSUPER
051614  001013                                  BNE     PD1
051616  012700  172300                          MOV     #172300,R0
051622  012701  000020                          MOV     #16.,R1
051626                                  2$:     PUSH    -(R0)
051630  077102                                  SOB     R1,2$
051632                                          IF RLFLAG IS TRUE THEN $CALL WOOPS
                                                ;COPY KERNEL MAP TO USER & SUPERVISOR
051644  012700  172300                  PD1:    MOV     #KIPDR0,R0
051650  012701  177600                          MOV     #UIPDR0,R1
051654  012702  172200                          MOV     #SIPDR0,R2
051660  012703  000040                          MOV     #32.,R3
051664  011021                          3$:     MOV     (R0),(R1)+
051666  012022                                  MOV     (R0)+,(R2)+
051670  077303                                  SOB     R3,3$
```

```
                                        ;SAVE USER & SUPERVISOR STACK POINTERS
051672                                  USER
051700  010600                          MOV     USP,R0
051702  104417                          KERNEL                  ;ENTER KERNEL MODE
051704                                  PUSH    R0
051706  005737  002426                  TST     NOSUPER
051712  001006                          BNE     7$
051714                                  SUPERVISOR              ;ENTER SUPERVISOR MODE
051722  010600                          MOV     SSP,R0
051724  104417                          KERNEL                  ;ENTER KERNEL MODE
051726                                  PUSH    R0
                                        ;SAVE ECC REGISTERS
051730  013701  002216          7$:     MOV     TOTCSRS,R1      ;GET CSR'S
051734                                  BEGIN   LCSRSAVE
051734                                    FOR CSRNO := #0 TO #36 BY #2
051740  006301                              ASL         R1
051742                                      ON.ERROR
051744  104426                                READCSR
051746                                        PUSH      CSR
051752                                      END ;OF ON.ERROR
051752                                    IF R1 EQ #0 THEN LEAVE LCSRSAVE
051756                                    END ;OF FOR CSRNO
051774                                  END LCSRSAVE
                                        ;SAVE MMR0,1,2,3
051774                                  PUSH    MMR0,MMR1,MMR2
052010  005737  002426                  TST     NOSUPER
052014  001002                          BNE     8$
052016                                  PUSH    MMR3
                                        ;SAVE KERNEL PAR'S
052022  012700  172400          8$:     MOV     #172400,R0
052026  012701  000020                  MOV     #16.,R1
052032                          4$:     PUSH    -(R0)
052034  077102                          SOB     R1,4$
                                        ;SAVE UNIBUS MAP REGISTERS
052036  022737  000001  003710          CMP     #1,PROTYP               ;IS THIS AN 11/44?
052044  001004                          BNE     9$                      ;BRANCH IF NOT
052046                                  PUSH    MAPH0,MAPL0
                                        ;SAVE POSSIBLE SOFTWARE SWITCH REGISTER
052056                          9$:     PUSH    @SWR
                                        ;SAVE STACK POINTER
052062  010637  052502                  MOV     SP,$SAVR6       ;;SAVE SP
                                        ;NOW SET UP REAL VECTOR
052066  012737  052100  000024          MOV     #$PWRUP,PWRVEC  ;;SET UP VECTOR
052074  000000                  $DOWN:  HALT
052076  000776                          BR      $DOWN           ;;HANG UP
```

```
                                        ;*******************************************************************************
                                        ;POWER UP ROUTINE
052100                                  $PWRUP:
052100  012737  052476  000024                  MOV     #$ILLUP,PWRVEC ;;SET FOR FAST DOWN
                                                ;RESTORE STACK POINTER
052106  013706  052502                          MOV     $SAVR6,SP       ;;GET SP
052112  005037  052502                          CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
052116  005237  052502                  1$:     INC     $SAVR6          ;;WAIT FOR THE INC
052122  001375                                  BNE     1$              ;;OF A WORD
                                                ;RESTORE POSSIBLE SOFTWARE SWITCH REGISTER
052124                                          POP     @SWR
                                                ;RESTORE UNIBUS MAP
052130  022737  000001  003710                  CMP     #1,PROTYP               ;IS THIS AN 11/44?
052136  001006                                  BNE     10$
052140                                          POP     MAPLO,MAPHO
052150  004737  043644                          CALL    LOWMAP          ;SETUP LOWER 16K OF UNIBUS MAP
                                                ;RESTORE KERNEL PAR'S & PDR'S
052154  012700  172340                  10$:    MOV     #172340,R0
052160  012702  172300                          MOV     #KIPDR0,R2
052164  012701  000020                          MOV     #16.,R1
052170                                  6$:     POP     (R0)+
052172  012722  077406                          MOV     #77406,(R2)+
052176  077104                                  SOB     R1,6$
                                                ;RESTORE MMR3,2,1,0
052200  005737  002426                          TST     NOSUPER
052204  001002                                  BNE     11$
052206                                          POP     MMR3
052212                                  11$:    POP     MMR2,MMR1,MMR0
                                                ;RESTORE ECC REGISTERS
052226  013701  002216                          MOV     TOTCSRS,R1      ;GET CSR'S
052232  042701  177400                          BIC     #177400,R1
052236                                          BEGIN LCSRRESTORE
052236                                            FOR CSRNO := #36 DOWNTO #0 BY #2
052244  006201                                      ASR         R1
052246                                              ON.ERROR
052250                                                POP         CSR
052254  104425                                        LOADCSR
052256                                              END ;OF ON.ERROR
052256                                            IF R1 EQ #0 THEN LEAVE LCSRRESTORE
052262                                            END ;OF FOR CSRNO
052300                                          END LCSRRESTORE
                                                ;COPY KERNEL MAP TO USER & SUPERVISOR
052300  012700  172300                          MOV     #KIPDR0,R0
052304  012701  177600                          MOV     #UIPDR0,R1
052310  012702  172200                          MOV     #SIPDR0,R2
052314  012703  000040                          MOV     #32.,R3
052320  011021                          3$:     MOV     (R0),(R1)+
052322  012022                                  MOV     (R0)+,(R2)+
052324  077303                                  SOB     R3,3$
```

```
                                                  ;RESTORE SUPERVISOR & USER STACK POINTERS
052326  005737  002426                            TST     NOSUPER
052332  001006                                    BNE     13$
052334                                            POP     R0
052336                                            SUPERVISOR              ;ENTER SUPERVISOR MODE
052344  010006                                    MOV     R0,SSP
052346  104417                                    KERNEL                  ;ENTER KERNEL MODE
052350                                   13$:     POP     R0
052352                                            USER
052360  010006                                    MOV     R0,USP
052362  104417                                    KERNEL                  ;ENTER KERNEL MODE
                                                  ;RESTORE SUPERVISOR PAR'S
052364  012700  172240                            MOV     #172240,R0
052370  012701  000020                            MOV     #16.,R1
052374                                   7$:      POP     (R0)+
052376  077102                                    SOB     R1,7$
                                                  ;RESTORE USER PAR'S & PDR7
052400  012700  177636                            MOV     #177636,R0
052404  012701  000021                            MOV     #17.,R1
052410                                   8$:      POP     (R0)+
052412  077102                                    SOB     R1,8$
                                                  ;RESTORE POSSIBLE SOFTWARE DISPLAY REGISTER
052414  013777  002010  130156                    MOV     $PATMAR,@DISPLAY
052422  013737  002010  000174                    MOV     $PATMAR,DISPREG
052430                                            POP     CSRNO,R5,R4,R3,R2,R1,R0
052450  012737  051524  000024                    MOV     #$PWRDN,PWRVEC  ;;SET UP THE POWER DOWN VECTOR
052456                                            TYPE    MSG051          ;REPORT THE POWER FAILURE
                                                  ;RESTORE CACHE STATUS
052462  005737  002514                            TST     CACHKN
052466  001402                                    BEQ     9$
052470                                            POP     CONTRL
052474  000002                           9$:      RTI
052476  000000                           $ILLUP:  HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
052500  000776                                    BR      $ILLUP          ;; BEFORE THE POWER DOWN WAS COMPLETE
052502  000000                           $SAVR6:  0                       ;;PUT THE SP HERE
                                                  .EVEN
```

```
052504                                 WOOPS:  SUBTST  <<POWER FAIL WHILE RELOCATED>>
                                       ;*****************************************************************************
                                       ;*SUBTEST        POWER FAIL WHILE RELOCATED
                                       ;*****************************************************************************
052504                                         PUSH    BANK
052510  005037  002100                         CLR     BANK
052514                                         MAP     BANK                    ;MAP SUPERVISOR SPACE (TEST AREA) TO BANK
052530                                         SUPERVISOR              ;ENTER SUPERVISOR MODE
052536  013737  060024  053102                 MOV     FIRST+PWRVEC,WOOPSAV
052544  013737  060026  053104                 MOV     FIRST+PWRVEC+2,WOOPSAV+2
052552                                         BMOV    FIRST+WOOPUP,WOOPSAV+4,WOOPEND-WOOPUP/2+12.
052564  012737  052670  060024                 MOV     #WOOPUP,FIRST+PWRVEC
052572  012737  000340  060026                 MOV     #340,FIRST+PWRVEC+2
052600                                         BMOV    WOOPUP,FIRST+WOOPUP,WOOPEND-WOOPUP/2
052612  012700  172340                         MOV     #KIPAR0,R0
052616  012701  133052                         MOV     #FIRST+WOOPEND,R1
052622  012702  000010                         MOV     #8.,R2
052626  012021                         1$:     MOV     (R0)+,(R1)+
052630  077202                                 SOB     R2,1$
052632  005737  002426                         TST     NOSUPER
052636  001002                                 BNE     2$
052640  013721  172516                         MOV     MMR3,(R1)+
052644  013721  177576                 2$:     MOV     MMR2,(R1)+
052650  013721  177574                         MOV     MMR1,(R1)+
052654  013721  177572                         MOV     MMR0,(R1)+
052660  104417                                 KERNEL                  ;ENTER KERNEL MODE
052662                                         POP     BANK
052666  000207                                 RETURN
```

```
 9057 052670                              WOOPUP: SUBTST  <<POWER UP FROM BANK 0 TO RELOCATION>>
                                          ;*****************************************************************************
                                          ;*SUBTEST        POWER UP FROM BANK 0 TO RELOCATION
                                          ;*****************************************************************************
 9058 052670  012700  053052                      MOV     #WOOPEND,R0
 9059 052674  012701  172340                      MOV     #KIPAR0,R1
 9060 052700  012703  172300                      MOV     #KIPDR0,R3
 9061 052704  012702  000010                      MOV     #8.,R2
 9062 052710  012021                      1$:     MOV     (R0)+,(R1)+
 9063 052712  012723  077406                      MOV     #77406,(R3)+
 9064 052716  077204                              SOB     R2,1$
 9065 052720  005737  002426                      TST     NOSUPER
 9066 052724  001002                              BNE     3$
 9067 052726  012037  172516                      MOV     (R0)+,MMR3
 9068 052732  012037  177576              3$:     MOV     (R0)+,MMR2
 9069 052736  012037  177574                      MOV     (R0)+,MMR1
 9070 052742  012037  177572                      MOV     (R0)+,MMR0
 9071 052746  013706  052502                      MOV     $SAVR6,SP
 9072 052752                                      PUSH    BANK
 9073 052756  005037  002100                      CLR     BANK
 9074 052762                                      MAP     BANK                    ;MAP SUPERVISOR SPACE (TEST AREA) TO BANK
 9075 052776                                      SUPERVISOR              ;ENTER SUPERVISOR MODE
 9076 053004  013737  053102  060024              MOV     WOOPSAV,FIRST+PWRVEC
 9077 053012  013737  053104  060026              MOV     WOOPSAV+2,FIRST+PWRVEC+2
 9078                                             ;SIMULATE THE FOLLOWING BLOCK MOV BUT WITH NO STACK ACCESSES
 9079                                             ;BMOV   WOOPSAV+4,FIRST+WOOPUP,WOOPEND-WOOPUP/2+12.
 9080 053020  012700  053106                      MOV     #WOOPSAV+4,R0
 9081 053024  012701  000105                      MOV     #WOOPEND-WOOPUP/2+12.,R1
 9082 053030  012702  132670                      MOV     #FIRST+WOOPUP,R2
 9083 053034  012022                      2$:     MOV     (R0)+,(R2)+
 9084 053036  077102                              SOB     R1,2$
 9085
 9086 053040  104417                              KERNEL                  ;ENTER KERNEL MODE
 9087 053042                                      POP     BANK
 9088 053046  000137  052100                      JMP     $PWRUP
 9089 053052  000014                      WOOPEND:.REPT   12.
 9092 053102  000107                      WOOPSAV:.REPT   WOOPEND-WOOPUP/2+12.+2
```

```
                                 .SBTTL  IO SUBROUTINES

                                 .SBTTL  ROUTINE TYPE

                        ;*************************************************************************
                        ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
                        ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
                        ;*NOTE1:         $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
                        ;*NOTE2:         $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
                        ;*NOTE3:         $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
                        ;*
                        ;*CALL:
                        ;*1) USING A TRAP INSTRUCTION
                        ;*       TYPE    MESADR              ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
                        ;*OR
                        ;*       TYPE
                        ;*       MESADR
                        ;*

053320  105737  002330  $TYPE:   TSTB    $TPFLG              ;;IS THERE A TERMINAL?
053324  100407                   BMI     6$                  ;BR IF NO
053326  010046          1$:      MOV     R0,-(SP)            ;;SAVE R0
053330  017600  000002           MOV     @2(SP),R0           ;;GET ADDRESS OF ASCIZ STRING
053334  112046          4$:      MOVB    (R0)+,-(SP)         ;;PUSH CHARACTER TO BE TYPED ONTO STACK
053336  001005                   BNE     7$                  ;;BR IF IT ISN'T THE TERMINATOR
053340  005726                   TST     (SP)+               ;;IF TERMINATOR POP IT OFF THE STACK
053342  012600          5$:      MOV     (SP)+,R0            ;;RESTORE R0
053344  062716  000002  6$:      ADD     #2,(SP)             ;;ADJUST RETURN PC
053350  000002                   RTI                         ;;RETURN
053352  122716  000011  7$:      CMPB    #HT,(SP)            ;BRANCH IF NOT <HT>
053356  001002                   BNE     11$
053360  112716  000040           MOVB    #' ,(SP)            ;REPLACE TAB WITH SPACE
053364  122716  000200  11$:     CMPB    #CRLF,(SP)          ;;BRANCH IF NOT <CRLF>
053370  001006                   BNE     8$
053372  005726                   TST     (SP)+               ;;POP  <CR><LF> EQUIV
053374                           TYPE                        ;;TYPE A CR AND LF
053376  002620                   $CRLF
053400  105037  053632           CLRB    $CHARCNT            ;;CLEAR CHARACTER COUNT
053404  000753                   BR      4$                  ;;GET NEXT CHARACTER
053406  004737  053446  8$:      CALL    $TYPEC              ;;GO TYPE THIS CHARACTER
053412  123726  002612  9$:      CMPB    $FILLC,(SP)+        ;;IS IT TIME FOR FILLER CHARS.?
053416  001346                   BNE     4$                  ;;IF NO GO GET NEXT CHAR.
053420  013746  002326           MOV     $NULL,-(SP)         ;;GET # OF FILLER CHARS. NEEDED
                                                             ;;AND THE NULL CHAR.
053424  105366  000001  10$:     DECB    1(SP)               ;;DOES A NULL NEED TO BE TYPED?
053430  002770                   BLT     9$                  ;;BR IF NO--GO POP THE NULL OFF OF STACK
053432  004737  053446           CALL    $TYPEC              ;;GO TYPF A NULL
053436  105337  053632           DECB    $CHARCNT            ;;DO NOT COUNT AS A COUNT
053442  000770                   BR      10$                 ;;LOOP
053444  000000          XOCHAR:  .WORD 0
053446                  $TYPEC:  PUSH    R1
053450  116601  000004           MOVB    4(SP),R1
053454  005737  002514           TST     CACHKN
053460  001402                   BEQ     2$
053462                           PUSH    CONTRL
053466                  2$:      PUSH    R0
053470  104424                   CACHOFF                     ;TURN CACHE OFF
```

```
 9178 053472  105777  127110          3$:        TSTB   @$TPS              ;;WAIT UNTIL PRINTER IS READY
 9179 053476  100375                               BPL    3$
 9180 053500  005037  053444                     CLR    XOCHAR
 9181 053504  105777  127072                     TSTB   @$TKS              ;;CHECK FOR XOFF
 9182 053510  100032                             BPL    NC                 ;;SKIP IF NO CHARACTER
 9183 053512  117737  127066  053444             MOVB   @$TKB,XOCHAR       ;;SAVE THE CHARACTER
 9184 053520  042737  177600  053444             BIC    #^C177,XOCHAR      ;;STRIP OFF ASCII
 9185 053526  023727  053444  000023             CMP    XOCHAR,#023        ;;WAS IT A CONTROL S?
 9186 053534  001020                             BNE    NC                 ;;BRANCH IF NOT
 9187 053536  105777  127040          CONTS3:    TSTB   @$TKS              ;;WAIT FOR CHARACTER
 9188 053542  100375                             BPL    CONTS3
 9189 053544  117737  127034  053444             MOVB   @$TKB,XOCHAR       ;;GET CHARACTER
 9190 053552  042737  177600  053444             BIC    #^C177,XOCHAR      ;;STRIP OFF ASCII
 9191 053560                                     IF XOCHAR EQ #21          ;; IF IT IS A ^Q
 9192 053570  000402                               BR     NC
 9193 053572                                     ELSE
 9194 053574  000760                               BR     CONTS3
 9195 053576                                     END ;OF  IF XOCHAR
 9196 053576  110177  127006          NC:          MOVB  R1,@$TPB           ;;LOAD CHAR TO BE TYPED INTO DATA REG.
 9200 053602  122766  000015  000002             CMPB   #CR,2(SP)          ;;IS CHARACTER A CARRIAGE RETURN?
 9201 053610  001003                             BNE    1$                 ;;BRANCH IF NO
 9202 053612  105037  053632                     CLRB   $CHARCNT           ;;YES--CLEAR CHARACTER COUNT
 9203 053616  000406                             BR     $TYPEX             ;;EXIT
 9204 053620  122766  000012  000002  1$:        CMPB   #LF,2(SP)          ;;IS CHARACTER A LINE FEED?
 9205 053626  001402                             BEQ    $TYPEX             ;;BRANCH IF YES
 9206 053630  105227                             INCB   (PC)+              ;;COUNT THE CHARACTER
 9207 053632  000000                  $CHARCNT:.WORD  0                  ;;CHARACTER COUNT STORAGE
 9208 053634                          $TYPEX: POP    R0
 9209 053636  005737  002514                     TST    CACHKN             ;IS THERE A CACHE?
 9210 053642  001402                             BEQ    2$                 ;BRANCH IF NOT
 9211 053644                                     POP    CONTRL             ;POP CACHE STATUS
 9212 053650                          2$:        POP    R1
 9213 053652  000207                             RETURN
 9214 053654                          SUPLIMIT:;!!!!!!!!!!!!THIS IS THE LIMIT ON SUPERVISOR MAPPED TO MUT SPACE
```

```
                .SBTTL  ERROR DATA SETUP

        ;       USE THIS        IF THIS CONDITION DISCRIBES THE ERROR
        ;
        ;       PERR01          TRAP
        ;                       BAD DATA IN R0 UNLESS ABORTED
        ;                               THEN BAD DATA IS POINTED TO BY -(R4)
        ;                       GOOD DATA IN R5
        ;
        ;       PERR02          TRAP
        ;                       BAD DATA IN R1 UNLESS ABORTED
        ;                               THEN BAD DATA IS POINTED TO BY -(R4)
        ;                       GOOD DATA IN R2
        ;
        ;       PERR03          TRAP
        ;                       BAD DATA IS POINTED TO BY -(R1)
        ;                       GOOD DATA IN R4
        ;
        ;       PERR04          TRAP
        ;                       BAD DATA IN R4 UNLESS ABORTED
        ;                               THEN BAD DATA IS POINTED TO BY -2(R0)
        ;                       GOOD DATA IN R2
        ;
        ;       PERR05          JSR     PC
        ;                       BAD DATA IS POINTED TO BY -(R0)
        ;                       GOOD DATA IN R2
        ;                       RETURN AFTER SETTING UP GOOD,BAD,ADDRESS
        ;
        ;       PERR06          JSR     PC
        ;                       BAD DATA IS POINTED TO BY -(R0)
        ;                       GOOD DATA IS ZERO
        ;                       RETURN AFTER SETTING UP GOOD,BAD,ADDRESS
        ;
        ;       PERR07          TRAP
        ;                       BAD DATA IN R2 UNLESS ABORTED
        ;                               THEN BAD DATA IS POINTED TO BY (R1)
        ;                       GOOD DATA IN DATBUF
        ;
        ;       PERR10          TRAP
        ;                       BAD DATA IN R2 UNLESS ABORTED
        ;                               THEN BAD DATA IS POINTED TO BY 2(R1)
        ;                       GOOD DATA IN DATBUF+2
        ;
        ;       PERR11          TRAP
        ;                       BYTE TEST
        ;                       BAD DATA IN RIGHT BYTE OF R0 UNLESS ABORTED
        ;                               THEN BAD DATA IS POINTED TO BY (R1)
        ;                       GOOD DATA IS A ZERO BYTE
        ;
        ;       PERR12          TRAP
        ;                       BYTE TEST
        ;                       BAD DATA IN RIGHT BYTE OF R0 UNLESS ABORTED
        ;                               THEN BAD DATA IS POINTED TO BY (R1)
        ;                       GOOD DATA IS A BYTE OF ONES
        ;
        ;       PERR13          TRAP
        ;                       BAD DATA IN R0 UNLESS ABORTED
```

```
;                               THEN BAD DATA IS POINTED TO BY (R1)
;                       GOOD DATA IS ZERO
;
;       PERR14          TRAP
;                       BAD DATA IN RO UNLESS ABORTED
;                               THEN BAD DATA IS POINTED TO BY (R1)
;                       GOOD DATA IS ONES
;
;       PERR15          TRAP
;                       BAD DATA IN RO UNLESS ABORTED
;                               THEN BAD DATA IS POINTED TO BY (R1)
;                       GOOD DATA IN TSTDAT
;
;       PERR16          TRAP
;                       BAD DATA IN RO UNLESS ABORTED
;                               THEN BAD DATA IS POINTED TO BY (R1)
;                       GOOD DATA IN TSTDAT+2
;
;       PERR17          TRAP
;                       BAD DATA IN RO UNLESS ABORTED
;                               THEN BAD DATA IS POINTED TO BY (R1)
;                       GOOD DATA IN R2
;
;       PERR20          TRAP
;                       BAD DATA IN RO UNLESS ABORTED
;                               THEN BAD DATA IS POINTED TO BY (R1)
;                       GOOD DATA IN R3
;
;       PERR21          TRAP
;                       7 BIT BYTE TEST
;                       BAD DATA IN RIGHT BYTE OF RO UNLESS ABORTED
;                               THEN BAD DATA IS POINTED TO BY (R1)
;                       GOOD DATA IS A 7 BIT BYTE ON ONES
;
;       PERR22          TRAP
;                       BAD DATA IN R2 UNLESS ABORTED
;                               THEN BAD DATA IS POINTED TO BY (R1)
;                       GOOD DATA IN RO
;
;       PERR23          TRAP
;                       BAD DATA IN RO UNLESS ABORTED
;                               THEN BAD DATA IS POINTED TO BY (R1)
;                       GOOD DATA IN R4
;
;       PERR24          TRAP
;                       BAD DATA IN RO UNLESS ABORTED
;                               THEN BAD DATA IS POINTED TO BY (R2)
;                       GOOD DATA IN R3
;
;       PERR25          TRAP
;                       BAD DATA POINTED TO BY -(R1)
;                       GOOD DATA IN R2 UNLESS LOC V177654 IS SET
;                               THEN GOOD DATA IS IN R3
;
;       PERR26          TRAP
;                       BAD DATA IS DOUBLE WORD POINTED TO BY R1 AND IN LOW 7 BITS OF RO
;                       GOOD DATA IS 000000,,100000,,100
```

```
;
;  PERR27      TRAP
;              BAD DATA IS DOUBLE WORD POINTED TO BY R1 AND IN LOW 7 BITS OF RO
;              GOOD DATA IS 000000,,000000,,077
;
;  PERR30      TRAP
;              BAD DATA IS POINTED TO BY -16(SP)
;              GOOD DATA IS IN R1
;
;  PERR31      TRAP
;              SPECIAL ECC FAILURE HANDLER
;
;  PERR32      TRAP
;              SPECIAL ECC FAILURE HANDLER
;
;  PERR33      TRAP
;              SPECIAL ECC FAILURE HANDLER
;
;  PERR34      TRAP
;              SPECIAL ECC FAILURE HANDLER
;
;  PERR35      TRAP
;              SPECIAL BRANCH GOBBLE FAILURE HANDLER.
;
;       CALLING SEQUENCE FOR TRAP TYPES
;       BEQ     2$            ;NO - ERROR,BRANCH FOR CARD
;       PERRXX                ;TRAP TO ERROR ROUTINE
;2$:    NEXT    INSTRUCTION   ;CONTINUE TESTING
```

```
053654  010437  002032          $PER01: MOV     R4,ADDRESS
053660  162737  000002  002032          SUB     #2,ADDRESS
053666  010037  002050                  MOV     R0,BAD
053672  010537  002042                  MOV     R5,GOOD
053676  000137  054410                  JMP PERRAW

053702  010437  002032          $PER02: MOV     R4,ADDRESS
053706  162737  000002  002032          SUB     #2,ADDRESS
053714  010137  002050                  MOV     R1,BAD
053720  010237  002042                  MOV     R2,GOOD
053724  000137  054410                  JMP PERRAW

053730  010137  002032          $PER03: MOV     R1,ADDRESS
053734  162737  000002  002032          SUB     #2,ADDRESS
053742  010437  002042                  MOV     R4,GOOD
053746  016137  177776  002050          MOV     -2(R1),BAD
053754  000137  054410                  JMP PERRAW

053760  010037  002032          $PER04: MOV     R0,ADDRESS
053764  162737  000002  002032          SUB     #2,ADDRESS
053772  010437  002050                  MOV     R4,BAD
053776  010237  002042                  MOV     R2,GOOD
054002  000137  054410                  JMP     PERRAW

054006  010237  002042          PERR05: MOV     R2,GOOD
054012  014037  002050          PERA05: MOV     -(R0),BAD
054016  010037  002032                  MOV     R0,ADDRESS
054022  062700  000002                  ADD     #2,R0                   ;RESTORE R0
054026  004737  040126                  CALL    BADSTACK
054032  000207                          RETURN

054034  005037  002042          PERR06: CLR     GOOD
054040  000764                          BR      PERA05

054042  010137  002032          $PER07: MOV     R1,ADDRESS
054046  010237  002050                  MOV     R2,BAD
054052  013737  002234  002042          MOV     DATBUF,GOOD
054060  000137  054410                  JMP     PERRAW

054064                          $PER10: LET ADDRESS := R1 + #2
054076                                  LET BAD := R2
054102                                  LET GOOD := DATBUF+2
054110  000137  054410                  JMP     PERRAW

054114                          $PER11: LET ADDRESS := R1
054120                                  LET BAD := R0
054124                                  LET GOOD := #0
054130  000137  054462                  JMP     PERRAB

054134                          $PER12: LET ADDRESS := R1
054140                                  LET BAD := R0
054144                                  LET GOOD := #377
054152  000137  054462                  JMP     PERRAB
```

```
 10091 054156                           $PER13: LET ADDRESS := R1
 10092 054162                                   LET BAD := R0
 10093 054166                                   LET GOOD := #0
 10094 054172  000137  054410                   JMP     PERRAW
 10095
 10096 054176                           $PER14: LET ADDRESS := R1
 10097 054202                                   LET BAD := R0
 10098 054206                                   LET GOOD := ONES
 10099 054214  000137  054410                   JMP     PERRAW
 10100
 10101 054220                           $PER15: LET ADDRESS := R1
 10102 054224                                   LET BAD := R0
 10103 054230                                   LET GOOD := TSTDAT
 10104 054236  000137  054410                   JMP     PERRAW
 10105
 10106 054242                           $PER16: LET ADDRESS := R1
 10107 054246                                   LET BAD := R0
 10108 054252                                   LET GOOD := TSTDAT+2
 10109 054260  000453                           BR      PERRAW
 10110
 10111 054262                           $PER17: LET ADDRESS := R1
 10112 054266                                   LET BAD := R0
 10113 054272                                   LET GOOD := R2
 10114 054276  000444                           BR      PERRAW
 10115
 10116 054300                           $PER20: LET ADDRESS := R1
 10117 054304                                   LET BAD := R0
 10118 054310                                   LET GOOD := R3
 10119 054314  000435                           BR      PERRAW
 10120
 10121 054316                           $PER21: LET ADDRESS := R1
 10122 054322                                   LET BAD := R0
 10123 054326                                   LET GOOD := #177
 10124 054334  000477                           BR      PERRA7
 10125
 10126 054336                           $PER22: LET ADDRESS := R1
 10127 054342                                   LET BAD := R2
 10128 054346                                   LET GOOD := R0
 10129 054352  000416                           BR      PERRAW
 10130
 10131 054354                           $PER23: LET ADDRESS := R1
 10132 054360                                   LET BAD := R0
 10133 054364                                   LET GOOD := R4
 10134 054370  000407                           BR      PERRAW
 10135
 10136 054372                           $PER24: LET ADDRESS := R2
 10137 054376                                   LET BAD := R0
 10138 054402                                   LET GOOD := R3
 10139 054406  000400                           BR      PERRAW
 10141 054410                            PERRAW: SUBTST  <<DATA WAS A WORD>>
                                         ;*******************************************************************************
                                         ;*SUBTEST        DATA WAS A WORD
                                         ;*******************************************************************************
 10142 054410  004737  054644                    CALL    PERBNK
 10143 054414                                    IF ABORTFLAG IS TRUE THEN $CALL GETDATA
 10144 054426                                    IF BADPC EQ #0 THEN $CALL BADSTACK
 10145 054440  004737  054620                    CALL    PERXOR
 10146 054444                                    IF ABORTFLAG IS FALSE
 10147 054452  104011                              ERROR +11
 10148 054454                                    ELSE
 10149 054456  104012                              ERROR +12
 10150 054460                                    END ;OF IF ABORTFLAG
 10151 054460  000002                            RTI
 10152
 10153 054462                            PERRAB: SUBTST  <<DATA WAS A BYTE>>
                                         ;*******************************************************************************
                                         ;*SUBTEST        DATA WAS A BYTE
                                         ;*******************************************************************************
 10154 054462  004737  054644                    CALL    PERBNK
 10155 054466                                    IF ABORTFLAG IS TRUE THEN $CALL GETDATA
 10156 054500                                    IF BADPC EQ #0 THEN $CALL BADSTACK
 10157 054512  004737  054620                    CALL    PERXOR
 10158 054516                                    IF ABORTFLAG IS FALSE
 10159 054524  104014                              ERROR +14
 10160 054526                                    ELSE
 10161 054530  104015                              ERROR +15
 10162 054532                                    END ;OF IF ABORTFLAG
 10163 054532  000002                            RTI
 10166 054534                                PERRA7: SUBTST  <<DATA WAS A 7 BIT BYTE>>
                                             ;*******************************************************************************
                                             ;*SUBTEST        DATA WAS A 7 BIT BYTE
                                             ;*******************************************************************************
 10167 054534                                        IF BADPC EQ #0 THEN $CALL BADSTACK
 10168 054546  004737  054620                        CALL    PERXOR
 10169 054552  004737  054644                        CALL    PERBNK
 10170 054556  104022                                ERROR   +22
 10171 054560  000002                                RTI
 10172
 10173 054562                                $PER26: LET GOOD2 := #100000
 10174 054570                                        LET GOOD3 := #100
 10175 054576  000137  051212                        JMP PERRA3
 10176
 10177 054602  005037  002044                $PER27: CLR     GOOD2
 10178 054606                                        LET GOOD3 := #077
 10179 054614  000137  051212                        JMP PERRA3
 10180
 10181 054620                                PERXOR: SUBTST  <<DETERMINE XOR OF GOOD & BAD>>
                                             ;*******************************************************************************
                                             ;*SUBTEST        DETERMINE XOR OF GOOD & BAD
                                             ;*******************************************************************************
 10182 054620                                        PUSH    R0
 10183 054622  013700  002042                        MOV     GOOD,R0
 10184 054626  013737  002050  002056                MOV     BAD,BADXOR
 10185 054634  074037  002056                        XOR     R0,BADXOR
 10186 054640                                        POP     R0
 10187 054642  000207                                RETURN
```

```
054644                          PERBNK: SUBTST<<LOG ERROR ON BAD BANK>>
                                ;******************************************************************************
                                ;*SUBTEST        LOG ERROR ON BAD BANK
                                ;******************************************************************************
                                        ;WHILE WE'RE HERE LET'S MARK THE BAD BANK IN THE CONFIGURATION TABLE
054644                                  PUSH    R0,R1
054650  013701  002100                  MOV     BANK,R1
054654  006301                          ASL     R1
054656  006301                          ASL     R1
054660  052761  000001  002624          BIS     #BIT0,CONFIG(R1)
054666  105261  002626                  INCB    CONFIG+2(R1)            ;BUMP BANK COUNTER
054672  001002                          BNE     12$                     ;NO OVERFLOW - SKIP
054674  105361  002626                  DECB    CONFIG+2(R1)            ;SET BACK TO 255.
054700  126137  002626  002524  12$:    CMPB    CONFIG+2(R1),ERRMAX     ;IS IT PAST MAX?
054706  101403                          BLOS    11$                     ;NO - SKIP
054710                                  SET     TOOMANY                 ;YES
054716                          11$:    POP     R1,R0
054722  000207                          RETURN

054724  010037  002050          PERECC: MOV     R0,BAD
054730                                  IF ADDRESS EQ TESTADD
054740  013737  002240  002042            MOV TSTDAT,GOOD
054746                                  ELSE
054750  013737  002242  002042            MOV    TSTDAT+2,GOOD
054756                                  END ;OF IF (R1)
054756  004737  054620                  CALL    PERXOR
054762                                  SET     HEADER
054770  000207                          RETURN

054772                          $PER31: IF BADPC EQ #0 THEN $CALL BADSTACK
055004  004737  054724                  CALL    PERECC
055010                                  IF REALPAT EQ #11
055020  104037                            ERROR +37
055022                                  END ;OF IF REALPAT
055022                                  IF REALPAT EQ #14
055032  104042                            ERROR +42
055034                                  END ;OF IF REALPAT
055034                                  IF REALPAT EQ #15
055044  104043                            ERROR +43
055046                                  END ;OF IF REALPAT
055046                                  IF REALPAT EQ #16
055056  104044                            ERROR +44
055060                                  END ;OF IF REALPAT
055060                                  SET HEADER
055066  000002                          RTI
```

```
 10234 055070                              $PER32: IF BADPC EQ #0 THEN $CALL BADSTACK
 10235 055102  010137  002032                      MOV     R1,ADDRESS
 10236 055106  010037  002050                      MOV     R0,BAD
 10237 055112  010237  002042                      MOV     R2,GOOD
 10238 055116                                      SET     HEADER
 10239 055124  104040                              ERROR   +40
 10240 055126                                      SET     HEADER
 10241 055134  000002                              RTI
 10242
 10243 055136                              $PER33: IF BADPC EQ #0 THEN $CALL BADSTACK
 10244 055150  010137  002032                      MOV     R1,ADDRESS
 10245 055154  010037  002050                      MOV     R0,BAD
 10246 055160  105037  002051                      CLRB    BAD+1
 10247 055164  012737  000377  002042              MOV     #377,GOOD
 10248 055172  004737  054620                      CALL    PERXOR
 10249 055176                                      SET     HEADER
 10250 055204  104041                              ERROR   +41
 10251 055206                                      SET     HEADER
 10252 055214  000002                              RTI
 10253
 10254 055216                              $PER34:IF BADPC EQ #0 THEN $CALL BADSTACK
 10255 055230                                      IF #BIT15!BIT4 OFF.IN CSR
 10256 055240  104016                                ERROR +16                          ;NO SBE OR DBE
 10257 055242                                      ELSE
 10258 055244  104001                                ERROR +1                           ;EXPECTED SBE SO DBE MUST HAVE GOTTEN SET
 10259 055246                                      END ;OF IF #BIT15!BIT4
 10260 055246  000002                              RTI
 10261
 10262                                             ;DURING BRANCH GOBBLE THE CONDITION CODES WERE WRONG
 10263 055250  004737  054644              $PER35: CALL    PERBNK
 10264 055254  004737  040126                      CALL    BADSTACK
 10265 055260  013737  002030  0C2050              MOV     BADPSW,BAD
 10266 055266  012737  000012  002042              MOV     #12,GOOD
 10267 055274  104047                              ERROR   +47
 10268 055276  062706  000004                      ADD     #4,SP                        ;FIX STACK FROM TRAP
 10269 055302  000207                              RETURN                               ;ABORTING TEST
 10270
 10271 055304  010037  002042              $PER36: MOV     R0,GOOD
 10272 055310  010137  002050                      MOV     R1,BAD
 10273 055314                                      SET     HEADER
 10274 055322  104023                              ERROR   +23
 10275 055324                                      SET     HEADER
 10276 055332  000002                              RTI
```

```
                                            .SBTTL  ROUTINE SCOPE HANDLER
                                    ;*******************************************************************************
                                    ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
                                    ;*AND LOAD THE DISPLAY DATA INTO THE DISPLAY REGISTER
                                    ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
                                    ;*SW14=1        LOOP ON TEST
                                    ;*SW9=1 LOOP ON ERROR
                                    ;*CALL
                                    ;*      SCOPE           ;;SCOPE=IOT
055334  005237  062476              $SCOPE: INC     $DEVCT          ;TELL APT WE ARE ALIVE
055340                                      IF RESULT IS LT
055342  005037  062476                        CLR   $DEVCT
055346  105237  062500                        INCB  $UNIT
055352                                      END ;OF IF RESULT
055352  104410                              CKSWR                   ;;TEST  FOR CHANGE IN SOFT-SWR
055354  005737  006104                      TST     TRACE
055360  001402                              BEQ     NOTRCE
055362  004737  061002                      CALL    CONTT           ;TRACE
055366                              NOTRCE:
055366                                      IF STOPOK IS TRUE AND #SW8 SET.IN @SWR
055404  005037  002370                        CLR   STOPOK
055410  000137  045074                        JMP   EXIT
055414                                      END ;OF IF STOPOK
055414                                      IF NOSCOPE IS TRUE
055422  000002                                RTI
055424                                      END ;OF IF NOSCOPE
055424                              1$:     IF #SW14 SET.IN @SWR THEN GOTO $OVER
                                    ;#####START OF CODE FOR THE XOR TESTER#####
055434  000425                      $XTSTR: BR      2$                      ;;IF RUNNING ON THE ''XOR'' TESTER CHANGE
                                                                            ;;THIS INSTRUCTION TO A ''NOP'' (NOP=240)
055436  013746  000004                      MOV     ERRVEC,-(SP)    ;;SAVE THE CONTENTS OF THE ERROR VECTOR
055442  012737  055462  000004              MOV     #1$,ERRVEC      ;;SET FOR TIMEOUT
055450  005737  177060                      TST     177060  ;;TIME OUT ON XOR?
055454  012637  000004                      MOV     (SP)+,ERRVEC    ;;RESTORE THE ERROR VECTOR
055460  000430                              BR      $SVLAD          ;;GO TO THE NEXT TEST
055462  062706  000004              1$:     ADD     #4,SP           ;FIX STACK FROM TRAP
055466  022737  000001  003710              CMP     #1,PROTYP       ;IS THIS AN 11/44?
055474  001002                              BNE     6$              ;BRANCH IF NOT
055476  005037  177766                      CLR     CPUERR          ;RESET CPU ERROR REGISTER
055502  012637  000004              6$:     MOV     (SP)+,ERRVEC    ;;RESTORE THE ERROR VECTOR
055506  000407                              BR      4$              ;;LOOP ON THE PRESENT TEST
055510                              2$:;#####END OF CODE FOR THE XOR TESTER#####
055510  105737  002012              3$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
055514  001412                              BEQ     $SVLAD          ;;BR IF NO
055516  032777  001000  125052              BIT     #SW9,@SWR       ;;LOOP ON ERROR?
055524  001404                              BEQ     5$              ;;BR IF NO
055526  013737  002564  002562      4$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
055534  000410                              BR      $OVER
055536  105037  002012              5$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
055542  011637  002562              $SVLAD: MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
055546  011637  002564                      MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
055552  005037  002332                      CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
055556  004737  055570              $OVER:  CALL    GETDIS
055562  013716  002562                      MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
055566  000002                              RTI                     ;;FIXES PS
```

```
10343 055570                           GETDIS: SUBTST  <<SUBR  DISPLAY>>
                                       ;*****************************************************************
                                       ;*SUBTEST         SUBR    DISPLAY
                                       ;*****************************************************************
10344 055570  113737  002100  002011           MOVB    BANK,$BANK
10345 055576  113737  002260  002010           MOVB    REALPAT,$PATMAR
10346 055604                                   PUSH    R0
10347 055606  005737  002124                   TST     RLFLAG                  ;ARE WE RELOCATED?
10348 055612  001403                           BEQ     1$                      ;NO - SKIP
10349 055614  052737  100000  002010           BIS     #BIT15,$PATMAR          ;YES - SET MSB
10350 055622                           1$:
10354 055622  013777  002010  124750           MOV     $PATMAR,@DISPLAY
10355 055630  013737  002010  000174           MOV     $PATMAR,DISPREG         ;SOFTWARE DISPLAY REGISTER
10356 055636                                   POP     R0
10357 055640  000207                           RETURN
```

```
                                          .SBTTL  ROUTINE ERROR HANDLER
                                  ;*********************************************************************************
                                  ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
                                  ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
                                  ;*AND GO TO $ERRTYP ON ERROR
                                  ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
                                  ;*SW15=1         HALT ON ERROR
                                  ;*SW13=1         INHIBIT ERROR TYPEOUTS
                                  ;*SW10=1         BELL ON ERROR
                                  ;*SW9=1 LOOP ON ERROR
                                  ;*CALL
                                  ;*        ERROR   N          ;;ERROR=EMT AND N=ERROR ITEM NUMBER

                                          .ENABL  LSB
055642                            $ERROR: IF NOERROR IS FALSE
055650  104410                              CKSWR                     ;;TEST FOR CHANGE IN SOFT-SWR
055652  105237  002012            1$:       INCB  $ERFLG              ;;SET THE ERROR FLAG
055656  001775                              BEQ   1$                  ;;DON'T LET THE FLAG GO TO ZERO
055660  004737  055570                      CALL  GETDIS              ;SETUP DISPLAY STUFF
055664  013737  002010  062472              MOV   $PATMAR,$TESTN      ;FOR APT
055672  032777  002000  124676              BIT   #SW10,@SWR          ;;BELL ON ERROR?
055700  001404                              BEQ   2$                  ;;NO - SKIP
055702                                      TYPE  $BELL               ;;RING BELL
055706                                      TYPE  MSG014              ;CONTROL Z
055712  005237  002570            ?$:       INC   $ERTTL              ;;COUNT THE NUMBER OF ERRORS
055716                                      IF RESULT IS MI
055720  012737  077777  002570                MOV #77777,$ERTTL
055726                                      END ;OF IF RESULT
055726                                    END ;OF IF NOERROR
055726  011637  002016                    MOV     (SP),ERRPC          ;;GET ADDRESS OF ERROR INSTRUCTION
055732  162737  000002  002016            SUB     #2,ERRPC
055740  010637  002022                    MOV     SP,ERRSP
055744  016637  000002  002026            MOV     2(SP),ERRPSW
055752  117737  124040  002013            MOVB    @ERRPC,$ITEMB       ;;STRIP AND SAVE THE ERROR ITEM CODE
055760                                    IF NOERROR IS FALSE
055766                                      IF BADPC NE #0
055774  013737  002020  002016                MOV BADPC,ERRPC
056002  162737  000002  002016                SUB #2,ERRPC
056010  013737  002024  002022                MOV BADSP,ERRSP
056016  013737  002030  002026                MOV BADPSW,ERRPSW
056024  005037  002020                        CLR BADPC
056030                                      END ;IF
056030  013737  002016  062470              MOV   ERRPC,$FATAL        ;FOR APT
056036                                      IF #SW13 SET.IN @SWR
056046  000412                                BR  3$
056050                                      END ;OF IF #SW13
056050                                      IF #SW5 SET.IN @SWR AND TOOMANY IS TRUE
056066                                        GOTO 3$
056070                                      END ;OF IF #SW5
056070                                    END ;OF IF NOERROR
056070  004737  056306                    CALL     $ERRTYP            ;;GO TO USER ERROR ROUTINE
```

```
056074                            3$:      IF NOERROR IS FALSE
056102  005777  124470                       TST    @SWR                ;;HALT ON ERROR
056106  100002                               BPL    7$                  ;;SKIP IF CONTINUE
056110  000000                    $HALT:     HALT                       ;;HALT ON ERROR!
056112  104410                               CKSWR                      ;;TEST FOR CHANGE IN SOFT-SWR
056114                            7$:        IF NOSCOPE IS FALSE AND #SW9 SET.IN @SWR
056132  013716  002564                         MOV $LPERR,(SP)          ;;FUDGE RETURN FOR LOOPING
056136                                       END ;OF IF NOSCOPE
056136  005737  002332                       TST    $ESCAPE             ;;CHECK FOR AN ESCAPE ADDRESS
056142  001402                               BEQ    9$                  ;;BR IF NONE
056144  013716  002332                       MOV    $ESCAPE,(SP)        ;;FUDGE RETURN ADDRESS FOR ESCAPE
056150                            9$:        IF DETFLAG IS FALSE
056156  022737  000001  003710                 CMP #1,PROTYP                      ;IS THIS AN 11/44?
056164  001002                                 BNE 11$
056166  005037  177766                         CLR CPUERR
056172                                  11$: IF ACTFLAG IS TRUE OR APTFLAG IS TRUE OR FATAL$ IS TRUE
056214  012737  000001  062466                   MOV       #1,$MSGTY          ;FOR APT
056222  000137  045074                           JMP       EXIT
056226                                         END ;OF IF ACTFLAG
056226                                         IF XXDPCHAIN IS TRUE AND $ERTTL HI #20
056244                                           TYPE      MSG066             ;ERROR COUNT EXCEEDED 20 - ABORTING FOR XXDP CHAIN
056250  013700  000042                           MOV       42,R0
056254  005037  000042                           CLR       42
056260  000137  014410                           JMP       $ZAP42
056264                                         END ;OF IF XXDPCHAIN
056264                                       END ;OF IF DETFLAG
056264                                     ELSE
056266                                       SET    HEADER
056274                                     END ;OF IF NOERROR
056274                            10$:     CLEAR    TOOMANY,NOERROR
056304  000002                             RTI                          ;;RETURN
                                           .DSABL  LSB
                                           .SBTTL  ROUTINE ERROR MESSAGE TYPEOUT

                                   ;*******************************************************************************
                                   ;*THIS ROUTINE USES THE ''ITEM CONTROL BYTE'' ($ITEMB) TO DETERMINE WHICH
                                   ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE ''ERROR TABLE'' ($ERRTB),
                                   ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.

056306  104415                     $ERRTYP:SAVREG
056310                                     TYPE    $CRLF               ;;''CARRIAGE RETURN'' & ''LINE FEED''
056314  005000                             CLR     R0                  ;;PICKUP THE ITEM INDEX
056316  153700  002013                     BISB    $ITEMB,R0
056322  001004                             BNE     1$                  ;;IF ITEM NUMBER IS ZERO, JUST
                                                                       ;;TYPE THE PC OF THE ERROR
056324                                     TYPOCT  ERRPC,<ERROR ADDRESS>
056332  000503                             BR      11$                 ;;GET OUT
056334  005300                     1$:     DEC     R0                  ;;ADJUST THE INDEX SO THAT IT WILL
056336  006300                             ASL     R0                  ;;       WORK FOR THE ERROR TABLE
056340  006300                             ASL     R0
056342  006300                             ASL     R0
056344  062700  063110                     ADD     #$ERRTB,R0          ;;FORM TABLE POINTER
056350  012037  056406                     MOV     (R0)+,3$            ;;PICKUP ''ERROR MESSAGE'' POINTER
056354  001417                             BEQ     4$                  ;;SKIP TYPEOUT IF NO POINTER
056356  005737  002400                     TST     NOERROR             ;IS THIS REALLY AN ERROR?
056362  001003                             BNE     12$                 ;YES - SKIP
056364  005737  002552                     TST     HEADER              ;TYPE HEADER?
056370  100011                             BPL     4$                  ;NO - SKIP
056372  005737  002062             12$:    TST     FATAL$              ;WAS IT A FATAL ERROR?
056376  001402                             BEQ     2$                  ;NO - SKIP
056400                                     TYPE    MSG067              ;FATAL
056404                             2$:     TYPE                        ;;TYPE THE ''ERROR MESSAGE''
056406  000000                     3$:     .WORD   0                   ;;''ERROR MESSAGE'' POINTER GOES HERE
056410                                     TYPE    $CRLF               ;;''CARRIAGE RETURN'' & ''LINE FEED''
056414  012037  056440             4$:     MOV     (R0)+,5$            ;;PICKUP ''DATA HEADER'' POINTER
056420  001412                             BEQ     6$                  ;;SKIP TYPEOUT IF 0
056422  005737  002400                     TST     NOERROR             ;IS THIS REALLY AN ERROR?
056426  001003                             BNE     13$                 ;YES - SKIP
056430  005737  002552                     TST     HEADER              ;TYPE HEADER?
056434  100004                             BPL     6$                  ;NO - SKIP
056436                             13$:    TYPE                        ;;TYPE THE ''DATA HEADER''
056440  000000                     5$:     .WORD   0                   ;;''DATA HEADER'' POINTER GOES HERE
056442                                     TYPE    $CRLF               ;;''CARRIAGE RETURN'' & ''LINE FEED''
056446  012001                     6$:     MOV     (R0)+,R1            ;;PICKUP ''DATA TABLE'' POINTER
056450  001427                             BEQ     10$                 ;;BR IF NO DATA TO BE TYPED
056452  012002                             MOV     (R0)+,R2            ;;PICKUP ''DATA FORMAT'' POINTER
```

```
 10497 056454   112203                  7$:     MOVB    (R2)+,R3
 10498 056456   006303                          ASL     R3                      ;MAKE IT A WORD ADDRESS
 10499 056460   004773  056466                  CALL    @8$(R3)
 10500 056464   000412                          BR      9$
 10501 056466   056602                  8$:     TAG70$
 10502 056470   056612                          TAG71$
 10503 056472   056622                          TAG72$
 10504 056474   056672                          TAG73$
 10505 056476   056732                          TAG74$
 10506 056500   056744                          TAG75$
 10507 056502   056756                          TAG76$
 10508 056504   057022                          TAG77$
 10509 056506   057030                          TAG78$
 10510 056510   057110                          TAG79$
 10515 056512   062701  000002          9$:     ADD     #2,R1                   ;UPDATE DATA TABLE POINTER
 10516 056516   005711                          TST     (R1)                    ;;IS THERE ANOTHER NUMBER?
 10517 056520   001403                          BEQ     10$                     ;;BR IF NO
 10518 056522                                   TYPE    MSG018                  ;TYPE 2 SPACES
 10519 056526   000752                          BR      7$                      ;;LOOP
 10520
 10521 056530   005737  002106          10$:    TST     MUT                     ;IS THERE A MEMORY UNDER TEST
 10522 056534   001402                          BEQ     11$                     ;NO - SKIP
 10523 056536   005237  002552                  INC     HEADER                  ;YES - BUMP HEADER FLAG
 10524 056542   104416                  11$:    RESREG
 10525 056544                                   IF #SW7 SET.IN @SWR AND DETFLAG IS FALSE AND NOERROR IS FALSE
 10526 056570   004737  057132                    CALL  DETAIL
 10527 056574                                   END ;OF IF #SW7
 10528 056574                                   TYPE    MSG104                  ;CONTROL Z
 10529 056600   000207                          RETURN
```

```
                                ;***********************************************************************************
                                ;*** OCTAL ***
                                ;***********************************************************************************
056602                          TAG70$: TYPOCT  @(R1)             ;;TYPE AN OCTAL NUMBER
056610  000207                          RETURN

                                ;***********************************************************************************
                                ;*** DECIMAL ***
                                ;***********************************************************************************
056612                          TAG71$: TYPDEC  @(R1)             ;;TYPE A DECIMAL NUMBER
056620  000207                          RETURN

                                ;***********************************************************************************
                                ;*** INTERLEAVE ***
                                ;***********************************************************************************
056622                          TAG72$: PUSH    R1,R5
056626  013701  002100                  MOV     BANK,R1
056632  070127  000004                  MUL     #4,R1
056636                                  SET     NOTAB                     ;INDICATE NO TABLE TO BE PRINTED - NOW
056644                                  TYPE    MSG014
056650  004737  037214                  CALL    TCFIG1
056654  005037  002342                  CLR     NOTAB
056660                                  POP     R5,R1
056664                                  TYPE    MSG014                    ;1 SPACE
056670  000207                          RETURN

                                ;***********************************************************************************
                                ;*** CSR ***
                                ;***********************************************************************************
056672                          TAG73$: PUSH    R1,R5
056676  013701  002100                  MOV     BANK,R1
056702  070127  000004                  MUL     #4,R1
056706                                  SET     NOTAB
056714  004737  037534                  CALL    TCFIG3
056720  005037  002342                  CLR     NOTAB
056724                                  POP     R5,R1
056730  000207                          RETURN

                                ;***********************************************************************************
                                ;*** PATTERN ***
                                ;***********************************************************************************
056732                          TAG74$: TYPOCS  REALPAT,<TYPE (0-77)>,2,Z
056742  000207                          RETURN

                                ;***********************************************************************************
                                ;*** BANK ***
                                ;***********************************************************************************
056744                          TAG75$: TYPOCS  BANK,<TYPE (0-167)>,3
056754  000207                          RETURN
```

```
 10582                                      ;*******************************************************************************
 10583                                      ;*** MTYPE ***
 10584                                      ;*******************************************************************************
 10585 056756                               TAG76$: PUSH    R1,R5
 10586 056762  013701  002100                       MOV     BANK,R1
 10587 056766  070127  000004                       MUL     #4,R1
 10588 056772                                       SET     NOTAB
 10589 057000                                       TYPE    MSG019
 10590 057004  004737  037346                       CALL    TCFIG2
 10591 057010  005037  002342                       CLR     NOTAB
 10592 057014                                       POP     R5,R1
 10593 057020  000207                               RETURN
 10594
 10595                                      ;*******************************************************************************
 10596                                      ;*** UNKNOWN DATA ***
 10597                                      ;*******************************************************************************
 10598 057022                               TAG77$: TYPE    MSG061
 10599 057026  000207                               RETURN
 10600
 10601                                      ;*******************************************************************************
 10602                                      ;*** PHYSICAL ADDRESS ***
 10603                                      ;*******************************************************************************
 10604 057030  013737  002032  002036       TAG78$: MOV     ADDRESS,PHYADD
 10605 057036  162737  060000  002036               SUB     #FIRST,PHYADD
 10606 057044  013737  002100  002040               MOV     BANK,PHYADD+2
 10607 057052  006237  002040                       ASR     PHYADD+2
 10608 057056  103003                               BCC     1$
 10609 057060  052737  100000  002036               BIS     #BIT15,PHYADD
 10610 057066  012746  002036               1$:     MOV     #PHYADD,-(SP)           ;POINTER TO DOUBLE WORD ON STACK
 10611 057072  004737  062346                       CALL    $DB20                   ;CALL DOUBLE PRECISION CONVERSION ROUTINE
 10612 057076  062706  000002                       ADD     #2,SP                   ;FIX STACK
 10613 057102                                       TYPE    $OCT8
 10614 057106  000207                               RETURN
 10615
 10616                                      ;*******************************************************************************
 10617                                      ;*** OCTAL BYTE ***
 10618                                      ;*******************************************************************************
 10619 057110                               TAG79$: TYPE    MSG018          ;2 SPACES
 10620 057114                                       TYPOCS  @(R1),<TYPE BYTE>,3,Z
 10621 057124                                       TYPE    MSG014          ;SPACE
 10622 057130  000207                               RETURN
 10666 057132                                   DETAIL: SUBTST  <<SUBR  DETAILED ERROR REPORT>>
                                                ;*********************************************************************
                                                ;*SUBTEST        SUBR    DETAILED ERROR REPORT
                                                ;*********************************************************************
 10667 057132   005237  002212                          INC     DETFLAG
 10668 057136   022737  000003  002212                  CMP     #3,DETFLAG
 10669 057144   101473                                  BLOS    4$
 10670 057146   022737  000002  002212                  CMP     #2,DETFLAG
 10671 057154   001435                                  BEQ     2$
 10672 057156                                           PUSH    HEADER,MUT
 10673 057166                                           SET     HEADER
 10674 057174   005037  002106                          CLR     MUT
 10675 057200   010037  002172                          MOV     R0,DETR0
 10676 057204   012700  002174                          MOV     #DETR1,R0
 10677 057210   010120                                  MOV     R1,(R0)+
 10678 057212   010220                                  MOV     R2,(R0)+
 10679 057214   010320                                  MOV     R3,(R0)+
 10680 057216   010420                                  MOV     R4,(R0)+
 10681 057220   010520                                  MOV     R5,(R0)+
 10682 057222   013720  002022                          MOV     ERRSP,(R0)+
 10683 057226   013720  002026                          MOV     ERRPSW,(R0)+
 10684 057232   013700  002172                          MOV     DETR0,R0
 10685 057236                                           SET     NOERROR
 10686 057244   104013                                  ERROR   +13
 10687 057246   000423                                  BR      1$
 10688 057250                                   2$:     PUSH    HEADER,MUT
 10689 057260                                           SET     HEADER
 10690 057266   005037  002106                          CLR     MUT
 10691 057272                                           SET     NOERROR
 10692 057300   104031                                  ERROR   +31
 10693 057302   022737  000001  003710                  CMP     #1,PROTYP                       ;IS THIS AN 11/44?
 10694 057310   001002                                  BNE     1$
 10695 057312   005037  177766                          CLR     CPUERR
 10696 057316                                   1$:     POP     MUT,HEADER
 10697                                                   ;WARNING RECURSIVE
 10698 057326   004737  057132                          CALL    DETAIL
 10699 057332   000207                                  RETURN
```

```
                                        ;SIMULATE CONTROL "T"
057334  004737  061002          4$:     CALL    CONTT                   ;DISPLAY "DISPLAY" INFO

                                        ;TYPE CONTENTS OF ALL CSR'S
057340                                  PUSH    CSR,CSRNO,R1
057352                                  TYPE    MSG058
057356                                  TYPE    $CRLF
057362  013701  002216                  MOV     TOTCSRS,R1
057366                                  BEGIN DUMPCSRLOOP
057366                                    FOR CSRNO := #0 TO #36 BY #2
057372  006301                                ASL         R1
057374                                        ON.ERROR
057376  104426                                  READCSR
057400                                          TYPOCT      CSR
057406                                          TYPE        MSG018          ;2 SPACES
057412                                        END ;OF ON.ERROR
057412                                        IF R1 EQ #0 THEN LEAVE DUMPCSRLOOP
057416                                    END ;OF FOR CSRNO
057434                                  END DUMPCSRLOOP
057434                                  POP     R1,CSRNO,CSR

                                        ;TYPE STACKS
057446                                  PUSH    R0,R1
057452                                  TYPE    MSG088          ;KERNEL STACK
057456  013701  002534                  MOV     KSTACK,R1
057462  162701  000002                  SUB     #2,R1
057466                                  FOR     R0 := SP TO R1 BY #2
057470                                    TYPE  $CRLF
057474                                    TYPOCT        R0
057500                                    TYPE          MSG018          ;2 SPACES
057504                                    TYPOCT        (R0)
057510                                  END ;OF FOR R0
                                        ;SET PREVIOUS MODE TO SUPERVISOR
057520  005737  002426                  TST     NOSUPER
057524  001036                          BNE     DET1
057526  042737  030000  177776          BIC     #BIT13!BIT12,PSW
057534  052737  010000  177776          BIS     #BIT12,PSW
057542  006506                          MFPI    SSP
057544                                  POP     R1,R0
057550                                  TYPE    MSG089          ;SUPERVISOR STACK
057554                                  IF R0 LT #SUPSTK
057562                                    FOR R0 := R0 TO #SUPSTK-2 BY #2
057562                                      TYPE        $CRLF
057566                                      TYPOCT      R0
057572                                      TYPE        MSG018          ;2 SPACES
057576                                      TYPOCT      (R0)
057602                                    END ;OF FOR R0
057614                                  ELSE
057616                                    TYPE  MSG091          ;IS EMPTY
057622                                  END ;OF IF R0
                                        ;SET PREVIOUS MODE TO USER
057622  052737  030000  177776  DET1:   BIS     #BIT13!BIT12,PSW
057630  006506                          MFPI    USP
057632                                  POP     R0
057634                                  TYPE    MSG090          ;USER STACK
057640                                  IF R0 LT #USESTK
057646                                    FOR R0 := R0 TO #USESTK-2 BY #2
```

```
 10759 057646                                       TYPE        $CRLF
 10760 057652                                       TYPOCT      R0
 10761 057656                                       TYPE        MSG018          ;2 SPACES
 10762 057662                                       TYPOCT      (R0)
 10763 057666                                     END ;OF FOR R0
 10764 057700                                   ELSE
 10765 057702                                     TYPE  MSG091          ;IS EMPTY
 10766 057706                                   END ;OF IF R0
 10767 057706                                   TYPE    $CRLF
 10768 057712   005037  002212                  CLR     DETFLAG
 10769 057716                                   POP     R0
 10770 057720   000207                          RETURN
```

```
                                          .SBTTL  ROUTINE 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

057722  017646  000000            $TYPOS: MOV     @(SP),-(SP)        ;;PICKUP THE MODE
057726  116637  000001  060145            MOVB    1(SP),$OFILL       ;;LOAD ZERO FILL SWITCH
057734  112637  060147                    MOVB    (SP)+,$OMODE+1     ;;NUMBER OF DIGITS TO TYPE
057740  062716  000002                    ADD     #2,(SP)            ;;ADJUST RETURN ADDRESS
057744  000406                            BR      $TYPON
057746  112737  000001  060145    $TYPOC: MOVB    #1,$OFILL          ;;SET THE ZERO FILL SWITCH
057754  112737  000006  060147            MOVB    #6,$OMODE+1        ;;SET FOR SIX(6) DIGITS
057762  112737  000005  060144    $TYPON: MOVB    #5,$OCNT           ;;SET THE ITERATION COUNT
057770  010346                            MOV     R3,-(SP)           ;;SAVE R3
057772  010446                            MOV     R4,-(SP)           ;;SAVE R4
057774  010546                            MOV     R5,-(SP)           ;;SAVE R5
057776  113704  060147                    MOVB    $OMODE+1,R4        ;;GET THE NUMBER OF DIGITS TO TYPE
060002  005404                            NEG     R4
060004  062704  000006                    ADD     #6,R4              ;;SUBTRACT IT FOR MAX. ALLOWED
060010  110437  060146                    MOVB    R4,$OMODE          ;;SAVE IT FOR USE
060014  113704  060145                    MOVB    $OFILL,R4          ;;GET THE ZERO FILL SWITCH
060020  016605  000012                    MOV     12(SP),R5          ;;PICKUP THE INPUT NUMBER
060024  005003                            CLR     R3                 ;;CLEAR THE OUTPUT WORD
060026  006105                    1$:     ROL     R5                 ;;ROTATE MSB INTO "C"
060030  000404                            BR      3$                 ;;GO DO MSB
060032  006105                    2$:     ROL     R5                 ;;FORM THIS DIGIT
060034  006105                            ROL     R5
060036  006105                            ROL     R5
060040  010503                            MOV     R5,R3
060042  006103                    3$:     ROL     R3                 ;;GET LSB OF THIS DIGIT
060044  105337  060146                    DECB    $OMODE             ;;TYPE THIS DIGIT?
060050  100016                            BPL     6$                 ;;BR IF NO
060052  042703  177770                    BIC     #177770,R3         ;;GET RID OF JUNK
060056  001002                            BNE     4$                 ;;TEST FOR 0
060060  005704                            TST     R4                 ;;SUPPRESS THIS 0?
060062  001403                            BEQ     5$                 ;;BR IF YES
060064  005204                    4$:     INC     R4                 ;;DON'T SUPPRESS ANYMORE 0'S
```

```
060066  052703  000060                 BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
060072  052703  000040         5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
060076  110337  060142                 MOVB    R3,8$           ;;SAVE FOR TYPING
060102                                 TYPE    8$              ;;GO TYPE THIS DIGIT
060106  105337  060144         6$:     DECB    $OCNT           ;;COUNT BY 1
060112  003347                         BGT     2$              ;;BR IF MORE TO DO
060114  002402                         BLT     7$              ;;BR IF DONE
060116  005204                         INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
060120  000744                         BR      2$              ;;GO DO THE LAST DIGIT
060122  012605                 7$:     MOV     (SP)+,R5        ;;RESTORE R5
060124  012604                         MOV     (SP)+,R4        ;;RESTORE R4
060126  012603                         MOV     (SP)+,R3        ;;RESTORE R3
060130  016666  000002  000004         MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
060136  012616                         MOV     (SP)+,(SP)
060140  000002                         RTI                     ;;RETURN
060142     000                 8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
060143     000                         .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
060144     000                 $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
060145     000                 $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
060146  000000                 $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
```

```
                                          .SBTTL  ROUTINE CONVERT BINARY TO DECIMAL AND TYPE
                                ;*******************************************************************************
                                ;*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:
                                ;*        MOV     NUM,-(SP)           ;;PUT THE BINARY NUMBER ON THE STACK
                                ;*        TYPDS                       ;;GO TO THE ROUTINE
060150                          $TYPDS:   PUSH    R0,R1,R2,R3,R5
060162  012746  020200                    MOV     #20200,-(SP)        ;;SET BLANK SWITCH AND SIGN
060166  016605  000020                    MOV     20(SP),R5           ;;GET THE INPUT NUMBER
060172  100004                            BPL     1$                  ;;BR IF INPUT IS POS.
060174  005405                            NEG     R5                  ;;MAKE THE BINARY NUMBER POS.
060176  112766  000055  000001            MOVB    #'-,1(SP)           ;;MAKE THE ASCII NUMBER NEG.
060204  005000                  1$:       CLR     R0                  ;;ZERO THE CONSTANTS INDEX
060206  012703  060364                    MOV     #$DBLK,R3           ;;SETUP THE OUTPUT POINTER
060212  112723  000040                    MOVB    #' ,(R3)+           ;;SET THE FIRST CHARACTER TO A BLANK
060216  005002                  2$:       CLR     R2                  ;;CLEAR THE BCD NUMBER
060220  016001  060354                    MOV     $DTBL(R0),R1        ;;GET THE CONSTANT
060224  160105                  3$:       SUB     R1,R5               ;;FORM THIS BCD DIGIT
060226  002402                            BLT     4$                  ;;BR IF DONE
060230  005202                            INC     R2                  ;;INCREASE THE BCD DIGIT BY 1
060232  000774                            BR      3$
060234  060105                  4$:       ADD     R1,R5               ;;ADD BACK THE CONSTANT
060236  005702                            TST     R2                  ;;CHECK IF BCD DIGIT=0
060240  001002                            BNE     5$                  ;;FALL THROUGH IF 0
060242  105716                            TSTB    (SP)                ;;STILL DOING LEADING 0'S?
060244  100407                            BMI     7$                  ;;BR IF YES
060246  106316                  5$:       ASLB    (SP)                ;;MSD?
060250  103003                            BCC     6$                  ;;BR IF NO
060252  116663  000001  177777            MOVB    1(SP),-1(R3)        ;;YES--SET THE SIGN
060260  052702  000060          6$:       BIS     #'0,R2              ;;MAKE THE BCD DIGIT ASCII
060264  052702  000040          7$:       BIS     #' ,R2              ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
060270  110223                            MOVB    R2,(R3)+            ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
060272  005720                            TST     (R0)+               ;;JUST INCREMENTING
060274  020027  000010                    CMP     R0,#10              ;;CHECK THE TABLE INDEX
060300  002746                            BLT     2$                  ;;GO DO THE NEXT DIGIT
060302  003002                            BGT     8$                  ;;GO TO EXIT
060304  010502                            MOV     R5,R2               ;;GET THE LSD
060306  000764                            BR      6$                  ;;GO CHANGE TO ASCII
060310  105726                  8$:       TSTB    (SP)+               ;;WAS THE LSD THE FIRST NON-ZERO?
060312  100003                            BPL     9$                  ;;BR IF NO
060314  116663  177777  177776            MOVB    -1(SP),-2(R3)       ;;YES--SET THE SIGN FOR TYPING
060322  105013                  9$:       CLRB    (R3)                ;;SET THE TERMINATOR
060324                                    POP     R5,R3,R2,R1,R0
060336                                    TYPE    $DBLK               ;;NOW TYPE THE NUMBER
060342  016666  000002  000004            MOV     2(SP),4(SP)         ;;ADJUST THE STACK
060350  012616                            MOV     (SP)+,(SP)
060352  000002                            RTI                         ;;RETURN TO USER
060354  023420                  $DTBL:    10000.
060356  001750                            1000.
060360  000144                            100.
060362  000012                            10.
060364  000000  000000  000000  $DBLK:    .WORD   0,0,0,0
060372  000000
```

```
                                          .SBTTL  ROUTINE TTY INPUT
                                  ;*******************************************************************
                                  ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
                                  ;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
                                  ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
                                  ;*WHEN OPERATING IN TTY FLAG MODE.
                                          .ENABLE LSB
060374                            $CKSWR:
060374  005737  053444                    TST     XOCHAR          ;;SOMETHING THERE?
060400  001406                            BEQ     NOCH            ;; GO ON IF NOT
060402  013746  053444                    MOV     XOCHAR,-(SP)    ;; USE IT
060406  005037  053444                    CLR     XOCHAR
060412  000137  060434                    JMP     CONTS1
060416  105777  122160            NOCH:   TSTB    @$TKS           ;;CHAR THERE?
060422  100130                            BPL     12$             ;;IF NO, DON'T WAIT AROUND
060424  117746  122154                    MOVB    @$TKB,-(SP)     ;;SAVE THE CHAR
060430  042716  177600                    BIC     #^C177,(SP)     ;;STRIP-OFF THE ASCII
060434  022716  000006            CONTS1: CMP     #6,(SP)         ;IS IT CONTROL F?
060440  001002                            BNE     1$              ;NO SKIP
060442  004737  045344                    CALL    FIELDSERVICE
060446  022716  000024            1$:     CMP     #24,(SP)        ;IS IT CONTROL T?
060452  001002                            BNE     16$             ;NO - SKIP
060454  004737  061002                    CALL    CONTT           ;YES - CALL CONTROL T ROUTINE
060460  022716  000003            16$:    CMP     #3,(SP)         ;IS IT CONTROL C?
060464  001454                            BEQ     5$              ;YES EXIT *****NOTE***** STACK IS SCREWED UP!
060466  022716  000023            2$:     CMP     #23,(SP)        ;IS IT CONTROL S?
060472  001002                            BNE     17$             ;NO - SKIP
060474  004737  061056                    CALL    CONTS           ;YES - CALL CONTROL S ROUTINE
060500  022716  000013            17$:    CMP     #13,(SP)        ;IS IT CONTROL K?
060504  001005                            BNE     6$              ;NO - SKIP
060506                                    TYPE    $CNTLK          ;TYPE A ^K
060512  013706  002142                    MOV     CTLKVEC,SP      ;RESET KSP TO AFTER PATTERN EXEC ROUTINE
060516  000207                            RETURN                  ;RETURN TO PATTERN EXEC ROUTINE
060520  022737  000176  002576    6$:     CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED?
060526  001067                            BNE     CKEND           ;;BRANCH IF NO
060530  022716  000007                    CMP     #7,(SP)         ;;IS IT A CONTROL G?
060534  001064                            BNE     CKEND           ;;NO, RETURN TO USER
060536  005737  002060                    TST     $AUTO           ;ARE WE RUNNING IN AUTO-MODE?
060542  001061                            BNE     CKEND           ;BRANCH IF YES
060544                                    TYPE    $CNTLG          ;;ECHO THE CONTROL-G (^G)
060550                            $GTSWR: TYPE    $MSWR           ;;TYPE CURRENT CONTENTS
060554                                    TYPOCT  @SWR            ;;OF THE SWR
060562                                    TYPE    $MNEW           ;;PROMPT FOR NEW SWR
060566  005046                    3$:     CLR     -(SP)           ;;CLEAR COUNTER
060570  005046                            CLR     -(SP)           ;;THE NEW SWR
060572  105777  122004            4$:     TSTB    @$TKS           ;;CHAR THERE?
060576  100375                            BPL     4$              ;;IF NOT TRY AGAIN
060600  117746  122000                    MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
060604  042716  177600                    BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
060610  021627  000003                    CMP     (SP),#3         ;;IS IT A CONTROL-C?
060614  001006                            BNE     7$              ;;BRANCH IF NOT
060616                            5$:     TYPE    $CNTLC          ;;YES, ECHO CONTROL-C (^C)
060622  062706  000006                    ADD     #6,SP           ;;CLEAN UP STACK
060626  000137  044766                    JMP     BOOT            ;;CONTROL-C RESTART
060632  021627  000025            7$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
060636  001005                            BNE     9$              ;;BRANCH IF NOT
060640                                    TYPE    $CNTLU          ;;YES, ECHO CONTROL-U (^U)
```

```
 11005 060644  062706  000006          8$:     ADD     #6,SP                  ;;IGNORE PREVIOUS INPUT
 11006 060650  000746                          BR      3$                     ;;LET'S TRY IT AGAIN
 11007 060652  021627  000015          9$:     CMP     (SP),#15               ;;IS IT A <CR>?
 11008 060656  001016                          BNE     13$                    ;;BRANCH IF NO
 11009 060660  005766  000004                  TST     4(SP)                  ;;YES, IS IT THE FIRST CHAR?
 11010 060664  001403                          BEQ     10$                    ;;BRANCH IF YES
 11011 060666  016677  000002  121702          MOV     2(SP),@SWR             ;;SAVE NEW SWR
 11012 060674  062706  000006          10$:    ADD     #6,SP                  ;;CLEAR UP STACK
 11013 060700                                  TYPE    $CRLF                  ;;ECHO <CR> AND <LF>
 11014 060704  000002                  12$:    RTI                            ;;RETURN
 11015 060706  062706  000002          CKEND:  ADD     #2,SP                  ;FIX STACK
 11016 060712  000002                          RTI                            ;RETURN
 11017 060714  004737  053446          13$:    CALL    $TYPEC                 ;;ECHO CHAR
 11018 060720  021627  000060                  CMP     (SP),#60               ;;CHAR < 0?
 11019 060724  002420                          BLT     15$                    ;;BRANCH IF YES
 11020 060726  021627  000067                  CMP     (SP),#67               ;;CHAR > 7?
 11021 060732  003015                          BGT     15$                    ;;BRANCH IF YES
 11022 060734  042726  000060                  BIC     #60,(SP)+              ;;STRIP-OFF ASCII
 11023 060740  005766  000002                  TST     2(SP)                  ;;IS THIS THE FIRST CHAR
 11024 060744  001403                          BEQ     14$                    ;;BRANCH IF YES
 11025 060746  006316                          ASL     (SP)                   ;;NO, SHIFT PRESENT
 11026 060750  006316                          ASL     (SP)                   ;;    CHAR OVER TO MAKE
 11027 060752  006316                          ASL     (SP)                   ;;    ROOM FOR NEW ONE.
 11028 060754  005266  000002          14$:    INC     2(SP)                  ;;KEEP COUNT OF CHAR
 11029 060760  056616  177776                  BIS     -2(SP),(SP)            ;;SET IN NEW CHAR
 11030 060764  000702                          BR      4$                     ;;GET THE NEXT ONE
 11031 060766                          15$:    TYPE    $QUES                  ;;TYPE ?<CR><LF>
 11032 060772  000724                          BR      8$                     ;;SIMULATE CONTROL-U
 11033 060774     136     113     015  $CNTLK: .ASCIZ  /^K/<15><12>           ;CONTROL K ASCII STRING
       060777     012     000
 11034                                         .EVEN
 11035                                         .DSABL  LSB
```

```
  11038 061002                          CONTT:  SUBTST  <<CONTROL T>>
                                        ;*************************************************************************
                                        ;*SUBTEST        CONTROL T
                                        ;*************************************************************************
  11039 061002                                  PUSH    R0
  11040 061004                                  TYPE    $CRLF
  11050 061010                                    IF RLFLAG IS TRUE
  11051 061016                                      TYPE          MSG092          ;RELOCATED
  11052 061022                                    END ;OF IF RLFLAG
  11053 061022                                    TYPE  MSG093                    ;BANK=
  11054 061026                                    TYPOCS          BANK,,3         ;TYPE 3 DIGITS
  11055 061036                                    TYPE  MSG095                    ;PAT=
  11056 061042                                    TYPOCS          REALPAT,,2      ;TYPE 2 DIGITS
  11060 061052                                  POP     R0
  11061 061054  000207                          RETURN
  11062
  11063 061056                          CONTS:  SUBTST  <<CONTROL S & CONTROL Q>>
                                        ;*************************************************************************
                                        ;*SUBTEST        CONTROL S & CONTROL Q
                                        ;*************************************************************************
  11064 061056                                  POP     R0                      ;GET RID OF RETURN ADDRESS FROM STACK
  11065 061060  105777  121516          CONTS2: TSTB    @$TKS                   ;WAIT FOR CHARACTER
  11066 061064  100375                          BPL     CONTS2
  11067 061066  117716  121512                  MOVB    @$TKB,(SP)              ;REPLACE OVER OLD CHARACTER ON STACK
  11068 061072  042716  177600                  BIC     #^C177,(SP)             ;STRIP ALL BUT ASCII
  11069 061076                                  IF (SP) EQ #21                  ;IF IT IS A CONTROL Q
  11070 061104  000137  060434                    JMP   CONTS1
  11071 061110                                  ELSE
  11072 061112  000762                            BR    CONTS2
  11073 061114                                  END ;OF IF (SP)
```

```
                                          ;*******************************************************************************
                                          ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
                                          ;*CALL:
                                          ;*      RDCHR                           ;;INPUT A SINGLE CHARACTER FROM THE TTY
                                          ;*      RETURN HERE                     ;;CHARACTER IS ON THE STACK
                                          ;*                                      ;;WITH PARITY BIT STRIPPED OFF
                                          ;

061114  011646                            $RDCHR: MOV    (SP),-(SP)             ;;PUSH DOWN THE PC
061116  016666  000004  000002                    MOV    4(SP),2(SP)            ;;SAVE THE PS
061124  105777  121452                    1$:     TSTB   @$TKS                  ;;WAIT FOR
061130  100375                                    BPL    1$                     ;;A CHARACTER
061132  117766  121446  000004                    MOVB   @$TKB,4(SP)            ;;READ THE TTY
061140  042766  177600  000004                    BIC    #^C<177>,4(SP)         ;;GET RID OF JUNK IF ANY
061146  026627  000004  000023                    CMP    4(SP),#23              ;;IS IT A CONTROL-S?
061154  001013                                    BNE    3$                     ;;BRANCH IF NO
061156  105777  121420                    2$:     TSTB   @$TKS                  ;;WAIT FOR A CHARACTER
061162  100375                                    BPL    2$                     ;;LOOP UNTIL ITS THERE
061164  117746  121414                            MOVB   @$TKB,-(SP)            ;;GET CHARACTER
061170  042716  177600                            BIC    #^C177,(SP)            ;;MAKE IT 7-BIT ASCII
061174  022627  000021                            CMP    (SP)+,#21              ;;IS IT A CONTROL-Q?
061200  001366                                    BNE    2$                     ;;IF NOT DISCARD IT
061202  000750                                    BR     1$                     ;;YES, RESUME
061204  026627  000004  000140            3$:     CMP    4(SP),#140             ;;IS IT UPPER CASE?
061212  002407                                    BLT    4$                     ;;BRANCH IF YES
061214  026627  000004  000175                    CMP    4(SP),#175             ;;IS IT A SPECIAL CHAR?
061222  003003                                    BGT    4$                     ;;BRANCH IF YES
061224  042766  000040  000004                    BIC    #40,4(SP)              ;;MAKE IT UPPER CASE
061232  000002                            4$:     RTI                           ;;GO BACK TO USER
                                          ;*******************************************************************************
                                          ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
                                          ;*CALL:
                                          ;*      RDLIN                           ;;INPUT A STRING FROM THE TTY
                                          ;*      RETURN HERE                     ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
                                          ;*                                      ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
061234  010346                            $RDLIN: MOV    R3,-(SP)               ;;SAVE R3
061236  005046                                    CLR    -(SP)                  ;;CLEAR THE RUBOUT KEY
061240  012703  061532                    1$:     MOV    #$TTYIN,R3             ;;GET ADDRESS
061244  022703  061556                    2$:     CMP    #$TTYIN+20.,R3         ;;BUFFER FULL?
061250  101477                                    BLOS   8$                     ;;BR IF YES
061252  104411                                    RDCHR                         ;;GO READ ONE CHARACTER FROM THE TTY
061254  112613                                    MOVB   (SP)+,(R3)             ;;GET CHARACTER
061256  122713  000003                            CMPB   #3,(R3)                ;;IS IT A CONTROL-C?
061262  001016                                    BNE    3$                     ;;BRANCH IF NO
061264                                            TYPE   $CNTLC                 ;;TYPE A CONTROL-C (^C)
061270  005726                                    TST    (SP)+                  ;;CLEAN RUBOUT KEY OFF OF THE STACK
061272  012603                                    MOV    (SP)+,R3               ;;RESTORE R3
061274  032777  000400  121274                    BIT    #BIT8,@SWR             ;;IS THERE A HALT FLAG SET IN THE SWR?
061302  001404                                    BEQ    11$                    ;;BRANCH IF NOT TO BOOT ROUTINE
061304  005037  002370                            CLR    STOPOK                 ;;GET READY TO HALT PROGRAM
061310  000137  045074                            JMP    EXIT                   ;;GO HALT PROGRAM
061314  000137  044766                    11$:    JMP    BOOT                   ;;GOTO CONTROL-C RESTART
061320  122713  000177                    3$:     CMPB   #177,(R3)              ;;IS IT A RUBOUT
061324  001022                                    BNE    5$                     ;;BR IF NO
061326  005716                                    TST    (SP)                   ;;IS THIS THE FIRST RUBOUT?
061330  001007                                    BNE    4$                     ;;BR IF NO
061332  112737  000134  061530                    MOVB   #'\,10$                ;;TYPE A BACK SLASH
```

```
061340                                      TYPE    10$
061344  012716  177777                      MOV     #-1,(SP)         ;;SET THE RUBOUT KEY
061350  005303                      4$:     DEC     R3               ;;BACKUP BY ONE
061352  020327  061532                      CMP     R3,#$TTYIN       ;;STACK EMPTY?
061356  103434                              BLO     8$               ;;BR IF YES
061360  111337  061530                      MOVB    (R3),10$                 ;;SETUP TO TYPEOUT THE DELETED CHAR.
061364                                      TYPE    10$              ;;GO TYPE
061370  000725                              BR      2$               ;;GO READ ANOTHER CHAR.
061372  005716                      5$:     TST     (SP)             ;;RUBOUT KEY SET?
061374  001406                              BEQ     6$               ;;BR IF NO
061376  112737  000134  061530              MOVB    #'\,10$          ;;TYPE A BACK SLASH
061404                                      TYPE    10$
061410  005016                              CLR     (SP)             ;;CLEAR THE RUBOUT KEY
061412  122713  000025              6$:     CMPB    #25,(R3)         ;;IS CHARACTER A CTRL U?
061416  001003                              BNE     7$               ;;BR IF NO
061420                                      TYPE    $CNTLU           ;;TYPE A CONTROL ''U''
061424  000705                              BR      1$               ;;GO START OVER
061426  122713  000022              7$:     CMPB    #22,(R3)         ;;IS CHARACTER A ''^R''?
061432  001011                              BNE     9$               ;;BRANCH IF NO
061434  105013                              CLRB    (R3)             ;;CLEAR THE CHARACTER
061436                                      TYPE    $CRLF            ;;TYPE A ''CR'' & ''LF''
061442                                      TYPE    $TTYIN           ;;TYPE THE INPUT STRING
061446  000676                              BR      2$               ;;GO PICKUP ANOTHER CHACTER
061450                              8$:     TYPE    $QUES            ;;TYPE A '?'
061454  000671                              BR      1$               ;;CLEAR THE BUFFER AND LOOP
061456  111337  061530              9$:     MOVB    (R3),10$                 ;;ECHO THE CHARACTER
061462                                      TYPE    10$
061466  122723  000015                      CMPB    #15,(R3)+        ;;CHECK FOR RETURN
061472  001264                              BNE     2$               ;;LOOP IF NOT RETURN
061474  105063  177777                      CLRB    -1(R3)           ;;CLEAR RETURN (THE 15)
061500                                      TYPE    $LF              ;;TYPE A LINE FEED
061504  005726                              TST     (SP)+            ;;CLEAN RUBOUT KEY FROM THE STACK
061506  012603                              MOV     (SP)+,R3         ;;RESTORE R3
061510  011646                              MOV     (SP),-(SP)       ;;ADJUST THE STACK AND PUT ADDRESS OF THE
061512  016666  000004  000002              MOV     4(SP),2(SP)      ;;       FIRST ASCII CHARACTER ON IT
061520  012766  061532  000004              MOV     #$TTYIN,4(SP)
061526  000002                              RTI                      ;;RETURN
061530     000                      10$:    .BYTE   0                ;;STORAGE FOR ASCII CHAR. TO TYPE
061531     000                              .BYTE   0                ;;TERMINATOR
061532  000024                      $TTYIN: .REPT   20.              ;;RESERVE SIZE BYTES FOR TTY INPUT
061556     136     103     015      $CNTLC: .ASCIZ  /^C/<15><12>     ;;CONTROL ''C''
061561     012     000
061563     136     125     015      $CNTLU: .ASCIZ  /^U/<15><12>     ;;CONTROL ''U''
061566     012     000
061570     136     107     015      $CNTLG: .ASCIZ  /^G/<15><12>     ;;CONTROL ''G''
061573     012     000
061575     015     012     123      $MSWR:  .ASCIZ  <15><12>/SWR = /
061600     127     122     040
061603     075     040     000
061606     040     040     116      $MNEW:  .ASCIZ  /  NEW = /
061611     105     127     040
061614     075     040     000
                                            .EVEN
```

```
                                        .SBTTL  ROUTINE READ AN OCTAL NUMBER FROM THE TTY
                                ;*************************************************************************
                                ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
                                ;*CHANGE IT TO BINARY.
                                ;*THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL
                                ;*OCTAL DIGITS. IF AN ILLEGAL CHARACTER IS READ A "?" WILL BE TYPED
                                ;*FOLLOWED BY A CARRIAGE RETURN-LINE FEED. THE COMPLETE NUMBER MUST
                                ;*THEN BE RETYPED. THE INPUT IS TERMINATED BY TYPING A CARRIAGE RETURN.
                                ;*CALL:
                                ;*      RDOCT                   ;;READ AN OCTAL NUMBER
                                ;*      RETURN HERE             ;;LOW ORDER BITS ARE ON TOP OF THE STACK
                                ;*                              ;;HIGH ORDER BITS ARE IN $HIOCT
061620  011646                  $RDOCT: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR THE
061622  016666  000004  000002          MOV     4(SP),2(SP)     ;;INPUT NUMBER
061630                                  PUSH    R0,R1,R2
061636  104412                  1$:     RDLIN                   ;;READ AN ASCIZ LINE
061640  012600                          MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
061642  010037  061746                  MOV     R0,5$           ;;AND SAVE IT
061646  005001                          CLR     R1              ;;CLEAR DATA WORD
061650  005002                          CLR     R2
061652  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
061654  001420                          BEQ     3$              ;;IF ZERO GET OUT
061656  122716  000060                  CMPB    #'0,(SP)        ;;MAKE SURE THIS CHARACTER
061662  003026                          BGT     4$              ;;IS AN OCTAL DIGIT
061664  122716  000067                  CMPB    #'7,(SP)
061670  002423                          BLT     4$
061672  006301                          ASL     R1              ;;*2
061674  006102                          ROL     R2
061676  006301                          ASL     R1              ;;*4
061700  006102                          ROL     R2
061702  006301                          ASL     R1              ;;*8
061704  006102                          ROL     R2
061706  042716  177770                  BIC     #^C7,(SP)       ;;STRIP THE ASCII JUNK
061712  062601                          ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
061714  000756                          BR      2$              ;;LOOP
061716  005726                  3$:     TST     (SP)+           ;;CLEAN TERMINATOR FROM STACK
061720  010166  000012                  MOV     R1,12(SP)       ;;SAVE THE RESULT
061724  010237  061766                  MOV     R2,$HIOCT
061730                                  POP     R2,R1,R0
061736  000002                          RTI                     ;;RETURN
061740  005726                  4$:     TST     (SP)+           ;;CLEAN PARTIAL FROM STACK
061742  105010                          CLRB    (R0)            ;;SET A TERMINATOR
061744                                  TYPE                    ;;TYPE UP THRU THE BAD CHAR.
061746  000000                  5$:     .WORD   0
061750                                  TYPE    MSG062          ;INPUT MUST BE A
061754                                  TYPE    MSG063          ;N OCTAL
061760                                  TYPE    MSG064          ;NUMBER
061764  000724                          BR      1$              ;;TRY AGAIN
061766  000000                  $HIOCT: .WORD   0               ;;HIGH ORDER BITS GO HERE
                                        .SBTTL  ROUTINE READ A DECIMAL NUMBER FROM THE TTY

                                ;*************************************************************************
                                ;*THIS ROUTINE WILL READ A DECIMAL (ASCII) NUMBER FROM THE TTY AND
                                ;*CHANGE IT TO BINARY. IF TOO MANY CHARACTERS OR ANY ILLEGAL CHARACTERS
                                ;*ARE READ A "?" FOLLOWED BY A CARRIAGE RETURN-LINE FEED WILL BE TYPED.
                                ;*THE COMPLETE NUMBER MUST BE RETYPED. THE INPUT IS TERMINATED BY THE
                                ;*USER TYPING A CARRIAGE RETURN. THE RANGE OF THE INPUT NUMBER IS
```

```
                                      ;*POSITIVE 32767 TO NEGATIVE 32768.
                                      ;*CALL:
                                      ;*      RDDEC                         ;;READ A DECIMAL NUMBER
                                      ;*      RETURN HERE                   ;;NUMBER IS ON TOP OF THE STACK
                                      ;

061770  011646                        $RDDEC: MOV     (SP),-(SP)            ;;PROVIDE SPACE FOR
061772  016666  000004  000002                MOV     4(SP),2(SP)           ;;THE INPUT NUMBER
062000                                        PUSH    R0,R1,R2
062006  104412                        1$:     RDLIN                         ;;READ AN ASCIZ LINE
062010  012600                                MOV     (SP)+,R0              ;;ADDRESS OF 1ST CHAR.
062012  010037  062136                        MOV     R0,6$                 ;;SAVE INCASE OF BAD INPUT
062016  005046                                CLR     -(SP)                 ;;CLEAR DATA WORD
062020  005002                                CLR     R2                    ;;SIGN SET POSITIVE
062022  122710  000055                        CMPB    #'-,(R0)              ;;SEE IF A MINUS SIGN WAS TYPED
062026  001001                                BNE     2$                    ;;BR IF NO MINUS SIGN
062030  112002                                MOVB    (R0)+,R2              ;;SAVE FOR LATER USE
062032  112001                        2$:     MOVB    (R0)+,R1              ;;PICKUP THIS CHARACTER
062034  001424                                BEQ     3$                    ;;GET OUT IF ZERO
062036  122701  000060                        CMPB    #'0,R1                ;;MAKE SURE THIS CHARACTER
062042  003032                                BGT     5$                    ;;IS A DIGIT BETWEEN 0 & 9
062044  122701  000071                        CMPB    #'9,R1
062050  002427                                BLT     5$
062052  032716  170000                        BIT     #^C7777,(SP)          ;;DON'T LET NUMBER GET TO BIG
062056  001024                                BNE     5$                    ;;BR IF NUMBER WOULD OVERFLOW
062060  006316                                ASL     (SP)                  ;;*2
062062  011646                                MOV     (SP),-(SP)            ;;SAVE FOR LATER
062064  006316                                ASL     (SP)                  ;;*4
062066  006316                                ASL     (SP)                  ;;*8.
062070  062616                                ADD     (SP)+,(SP)            ;;*10.
062072  102416                                BVS     5$                    ;;OVERFLOW ISN'T ALLOWED
062074  162701  000060                        SUB     #'0,R1                ;;STRIP AWAY THE ASCII JUNK
062100  060116                                ADD     R1,(SP)               ;;ADD IN THIS DIGIT
062102  102412                                BVS     5$                    ;;OVERFLOW ISN'T ALLOWED
062104  000752                                BR      2$                    ;;LOOP
062106  005702                        3$:     TST     R2                    ;;CHECK IF NUMBER IS NEG
062110  001401                                BEQ     4$                    ;;BR IF NO
062112  005416                                NEG     (SP)                  ;;YES--NEGATE THE NUMBER
062114  012666  000012                4$:     MOV     (SP)+,12(SP)          ;;SAVE THE RESULT
062120                                        POP     R2,R1,R0
062126  000002                                RTI                           ;;RETURN

062130  005726                        5$:     TST     (SP)+                 ;;CLEAN PARTIAL NUMBER FROM STACK
062132  105010                                CLRB    (R0)                  ;;SET A TERMINATOR
062134                                        TYPE                          ;;TYPE THE INPUT UP TO BAD CHAR.
062136  000000                        6$:     .WORD   0                     ;;POINTER GOES HERE
062140                                        TYPE    MSG062                ;INPUT MUSST BE A
062144                                        TYPE    MSG065                ;DECIMAL
062150                                        TYPE    MSG064                ;NUMBER
062154  000714                                BR      1$                    ;;TRY AGAIN
```

```
                                        .SBTTL  ROUTINE SAVE AND RESTORE R0-R5

                                ;*******************************************************************************
                                ;*SAVE R0-R5
                                ;*CALL:
                                ;*      SAVREG
                                ;*UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
                                ;*
                                ;*TOP---(+16)
                                ;* +2---(+18)
                                ;* +4---R5
                                ;* +6---R4
                                ;* +8---R3
                                ;*+10---R2
                                ;*+12---R1
                                ;*+14---R0

062156                          $SAVREG:
062156                                  PUSH    R0,R1,R2,R3,R4,R5
062172  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PS OF MAIN FLOW
062176  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PC OF MAIN FLOW
062202  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PS OF CALL
062206  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PC OF CALL
062212  000002                          RTI

                                ;*RESTORE R0-R5
                                ;*CALL:
                                ;*      RESREG
062214                          $RESREG:
062214  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF CALL
062220  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF CALL
062224  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
062230  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
062234                                  POP     R5,R4,R3,R2,R1,R0
062250  000002                          RTI
```

```
                                                .SBTTL  ROUTINE RANDOM NUMBER GENERATOR

                                ;*******************************************************************************
                                ;*THIS ROUTINE IS A DOUBLE PRECISION PSEUDO RANDOM NUMBER GENERATOR
                                ;*WITH A RANGE OF 0 TO 2**(+33)-1.
                                ;*CALL:
                                ;*      CALL    $RAND                   ;;CALL THE ROUTINE
                                ;*      RETURN                          ;;RETURN HERE THE RANDOM
                                ;*                                      ;;NUMBER WILL BE IN
                                ;*                                      ;;$HINUM,$LONUM

062252                          $RAND:  PUSH    R0,R1,R2
062260  013700  002544                  MOV     SEEDLO,R0               ;SET R0 WITH LOW
062264  013701  002542                  MOV     SEEDHI,R1               ;SET R1 WITH HIGH
062270  012702  000007                  MOV     #7,R2                   ;SET SHIFT COUNT
062274  006300                  1$:     ASL     R0                      ;;SHIFT R0 LEFT AND
062276  006101                          ROL     R1                      ;;ROTATE CARRY INTO R1 AND
062300  077203                          SOB     R2,1$
062302  063700  002544                  ADD     SEEDLO,R0               ;ADD NUMBER TO MAKE X 129
062306  005501                          ADC     R1                      ;;PROPOGATE CARRY
062310  063701  002542                  ADD     SEEDHI,R1               ;ADD NUMBER TO MAKE X 129
062314  062700  001057                  ADD     #1057,R0                ;;ADD LOW CONSTANT
062320  005501                          ADC     R1                      ;;PROPOGATE CARRY
062322  062701  047401                  ADD     #47401,R1               ;;ADD HIGH CONSTANT
062326  010037  002544                  MOV     R0,SEEDLO               ;SAVE R0
062332  010137  002542                  MOV     R1,SEEDHI               ;SAVE R1
062336                                  POP     R2,R1,R0
062344  000207                          RETURN
```

```
                                          .SBTTL  ROUTINE DOUBLE LENGTH BINARY TO OCTAL ASCII CONVERT
                                  ;*********************************************************************
                                  ;*THIS ROUTINE WILL CONVERT A 32-BIT UNSIGNED BINARY NUMBER TO AN
                                  ;*UNSIGNED OCTAL ASCIZ NUMBER.
                                  ;*CALL
                                  ;*       MOV     #PNTR,-(SP)       ;;POINTER TO LOW WORD OF BINARY NUMBER
                                  ;*       CALL    $DB20             ;;CALL THE ROUTINE
                                  ;*       RETURN                    ;;THE ADDRESS OF THE FIRST ASCIZ CHAR. IS ON THE STACK


062346  104415                    $DB20:  SAVREG                    ;;SAVE ALL REGISTERS
062350  016601  000002                    MOV     2(SP),R1          ;;PICKUP THE POINTER TO LOW WORD
062354  012705  062465                    MOV     #$OCTVL+13.,R5    ;;POINTER TO DATA TABLE
062360  012704  000014                    MOV     #12.,R4           ;;DO ELEVEN CHARACTERS
062364  012703  177770                    MOV     #^C7,R3           ;;MASK
062370  012100                            MOV     (R1)+,R0          ;;LOWER WORD
062372  012101                            MOV     (R1)+,R1          ;;HIGH WORD
062374  005002                            CLR     R2                ;;TERMINATOR
062376  110245                    1$:     MOVB    R2,-(R5)          ;;PUT CHARACTER IN DATA TABLE
062400  010002                            MOV     R0,R2             ;;GET THIS DIGIT
062402  005304                            DEC     R4                ;;COUNT THIS CHARACTER
062404  003007                            BGT     3$                ;;BR IF NOT THE LAST DIGIT
062406  001405                            BEQ     2$                ;;BR IF IT IS THE LAST DIGIT
062410  005205                            INC     R5                ;;ALL DIGITS DONE-ADJUST POINTER FOR FIRST
062412  010566  000002                    MOV     R5,2(SP)          ;;ASCIZ CHAR. & PUT IT ON THE STACK
062416  104416                            RESREG                    ;;RESTORE ALL REGISTERS
062420  000207                            RETURN                    ;;RETURN TO USER
062422  006203                    2$:     ASR     R3                ;;POSITION THE MASK FOR THE LAST DIGIT
062424  006001                    3$:     ROR     R1                ;;POSITION THE BINARY NUMBER FOR
062426  006000                            ROR     R0                ;;        THE NEXT OCTAL DIGIT
062430  006001                            ROR     R1
062432  006000                            ROR     R0
062434  006001                            ROR     R1
062436  006000                            ROR     R0
062440  040302                            BIC     R3,R2             ;;MASK OUT ALL JUNK
062442  062702  000060                    ADD     #'0,R2            ;;MAKE THIS CHAR. ASCII
062446  000753                            BR      1$                ;;GO PUT IT IN THE DATA TABLE
062450  000016                    $OCTVL: .REPT   14.               ;;RESERVE DATA TABLE
        062454                    $OCT8=$OCTVL+4                    ;POINTER TO 11 DIGIT NUMBER
```

```
                                .SBTTL  TABLES

                                .SBTTL  APT MAILBOX-ETABLE
062466                  $MAIL:
062466  000000          $MSGTY: .WORD   0       ;;MESSAGE TYPE CODE
062470  000000          $FATAL: .WORD   0       ;;FATAL ERROR NUMBER (ERROR PC)
062472  000000          $TESTN: .WORD   0       ;;TEST PATTERN NUMBER
062474  000000          $PASS:  .WORD   0       ;;PASS COUNT
062476  000000          $DEVCT: .WORD   0       ;;DEVICE COUNT
062500  000000          $UNIT:  .WORD   0       ;;I/O UNIT NUMBER
062502  000000          $MSGAD: .WORD   0       ;;MESSAGE ADDRESS
062504  000000          $MSGLG: .WORD   0       ;;MESSAGE LENGTH
062506                  $ETABLE:                ;;APT ENVIRONMENT TABLE
062506     000          $ENV:   .BYTE   0       ;;ENVIRONMENT BYTE      ;SET TO A 1 FOR APT AUTO MODE
                        ;NOTE:  IF BIT #7 IS SET IN $ENVM THE TABLE BELOW (BEGINNING AT $MAMS1 AND
                        ;       ENDING AT $MADR4) MUST BE FILLED IN TO INDICATE THE PROPER AMOUNT OF
                        ;       EACH TYPE OF MEMORY.
062507     000          $ENVM:  .BYTE   0       ;ENVIRONMENT MODE
                                                ;BIT7(200)=USE APT SIZE INFO ;BIT5(40)=NO CONSOLE
062510  000101          $SWREG: .WORD   101     ;;APT SWITCH REGISTER
062512  000000          $USWR:  .WORD   0       ;USED TO LIMIT THE NUMBER OF PASSES
062514  000000          $CPUOP: .WORD   0       ;;CPU TYPE,OPTIONS
                        ;*                      BITS 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
062516     001          $MAMS1: .BYTE   1       ;;HIGH ADDRESS,M.S. BYTE ;DEFAULT = 64K
062517     004          $MTYP1: .BYTE   4       ;;MEM. TYPE,BLK#1
                        ;*                      MEM.TYPE BYTE   --  (HIGH BYTE)
                        ;*                              900 NSEC CORE=001
                        ;*                              300 NSEC BIPOLAR=002
                        ;*                              PARITY MOS=003
                        ;*                              ERROR CORRECTING MOS=004
062520  177776          $MADR1: .WORD   177776  ;;HIGH ADDRESS,BLK#1
                        ;*                        MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF ''TYPE'' ABOVE
062522     000          $MAMS2: .BYTE   0       ;;HIGH ADDRESS,M.S. BYTE
062523     000          $MTYP2: .BYTE   0       ;;MEM.TYPE,BLK#2
062524  000000          $MADR2: .WORD   0       ;;MEM.LAST ADDRESS,BLK#2
062526     000          $MAMS3: .BYTE   0       ;;HIGH ADDRESS,M.S.BYTE
062527     000          $MTYP3: .BYTE   0       ;;MEM.TYPE,BLK#3
062530  000000          $MADR3: .WORD   0       ;;MEM.LAST ADDRESS,BLK#3
062532     000          $MAMS4: .BYTE   0       ;;HIGH ADDRESS,M.S.BYTE
062533     000          $MTYP4: .BYTE   0       ;;MEM.TYPE,BLK#4
062534  000000          $MADR4: .WORD   0       ;;MEM.LAST ADDRESS,BLK#4
062536  000000          $VECT1: .WORD   0       ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
062540  000000          $VECT2: .WORD   0       ;;INTERRUPT VECTOR#2BUS PRIORITY#2
062542  000000          $BASE:  .WORD   0       ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
062544  000000          $DEVM:  .WORD   0       ;;DEVICE MAP

062546  000000          $CDW1:  .WORD   0
062550  000000          $CDW2:  .WORD   0
```

```
                                ;THE FOLLOWING LOCATIONS SPECIFY WHICH PATTERNS
                                ;ARE TO BE RUN FOR PARTICULAR MEMORIES
                                ;
                                ;REFERENCE THE TABLE LISTED BELOW TO RELATE BITS TO PATTERNS.
                                ;BIT0 SET WILL RUN THE FIRST ENTRY IN THE TABLE, BIT0 SET
                                ;IN THE SECOND WORD WILL RUN THE 17TH ENTRY IN THE TABLE ...
                                ;
                                ;NOTE** NULL TESTS DO NOT TAKE ANY TIME
                                ;                                        ;FIELD SERVICE VALUE
062552  177777                  $DDW0:  .WORD   177777  ;ECC CSR TESTS         177777          TABLE = MKCSRT:
062554  177777                  $DDW1:  .WORD   177777  ;ECC CSR TESTS         177777          TABLE = MKCSRT:
062556  177777                  $DDW2:  .WORD   177777  ;ECC PATTERNS          103777          TABLE = MKPAT:
062560  177777                  $DDW3:  .WORD   177777  ;ECC PATTERNS          177777          TABLE = MKPAT:
062562  177777                  $DDW4:  .WORD   177777  ;PARITY PATTERNS       003777          TABLE = MJPAT:
062564  177777                  $DDW5:  .WORD   177777  ;PARITY PATTERNS       177774          TABLE = MJPAT:
062566                          $ETEND:
                                ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
                                ;INTERFACE SPEC.

062566                          $APTHD:
062566  000000                  $HIBTS: .WORD   0               ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
062570  062466                  $MBADR: .WORD   $MAIL           ;;ADDRESS OF APT MAILBOX (BITS 0-15)
062572  000043                  $TSTM:  .WORD   35.             ;;RUN TIM OF LONGEST TEST
062574  001274                  $PASTM: .WORD   700.            ;;RUN TIME IN SECS. OF 1ST PASS ON 128K (QUICK VERIFY)
062576  000000                  $UNITM: .WORD   0.              ;;EXTRA RUN TIME OF A PASS FOR EACH ADDITIONAL 128K (QV)
062600  000040                          .WORD   $ETEND-$MAIL/2  ;;LENGTH MAILBOX-ETABLE(WORDS)
```

```
                                                .SBTTL  ROUTINE 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.

062602  010046                          $TRAP:  MOV     R0,-(SP)                ;;SAVE R0
062604  016600  000002                          MOV     2(SP),R0                ;;GET TRAP ADDRESS
062610  005740                                  TST     -(R0)                   ;;BACKUP BY 2
062612  111000                                  MOVB    (R0),R0                 ;;GET RIGHT BYTE OF TRAP
062614  006300                                  ASL     R0                      ;;POSITION FOR INDEXING
062616  016000  062644                          MOV     $TRPAD(R0),R0           ;;INDEX TO TABLE
062622  000200                                  RTS     R0                      ;;GO TO ROUTINE


                                        ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO

062624  011646                          $TRAP2: MOV     (SP),-(SP)              ;;MOVE THE PC DOWN
062626  016666  000004  000002                  MOV     4(SP),2(SP)             ;;MOVE THE PSW DOWN
062634  000002                                  RTI                             ;;RESTORE THE PSW

062636                                  $NOTRAP:TYPE    MSG006                  ;UNDEFINED TRAP INSTRUCTION
062642  000000                          $HALT2: HALT
```

```
                                        .SBTTL  TRAP TABLE

                               ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
                               ;*BY THE "TRAP" INSTRUCTION.

                               ;        ROUTINE
                               ;        -------
062644  062624                 $TRPAD:  .WORD    $TRAP2
062646  053320                          $TYPE    ;CALL=TYPEIT     TRAP+1(104401)  TTY TYPEOUT ROUTINE
062650  057746                          $TYPOC   ;CALL=TYPOC      TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
062652  057722                          $TYPOS   ;CALL=TYPOS      TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
062654  062636                          $NOTRAP;$TYPON   ;CALL=TYPON      TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
062656  060150                          $TYPDS   ;CALL=TYPDS      TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
062660  062636                          $NOTRAP;$TYPBN   ;CALL=TYPBN      TRAP+6(104406)  TYPE BINARY (ASCII) NUMBER

062662  060550                          $GTSWR   ;CALL=GTSWR      TRAP+7(104407)  GET SOFT-SWR SETTING
062664  060374                          $CKSWR   ;CALL=CKSWR      TRAP+10(104410) TEST FOR CHANGE IN SOFT-SWR

062666  061114                          $RDCHR   ;CALL=RDCHR      TRAP+11(104411) TTY TYPEIN CHARACTER ROUTINE
062670  061234                          $RDLIN   ;CALL=RDLIN      TRAP+12(104412) TTY TYPEIN STRING ROUTINE
062672  061620                          $RDOCT   ;CALL=RDOCT      TRAP+13(104413) READ AN OCTAL NUMBER FROM TTY
062674  061770                          $RDDEC   ;CALL=RDDEC      TRAP+14(104414) READ A DECIMAL NUMBER FROM TTY

062676  062156                          $SAVREG ;CALL=SAVREG      TRAP+15(104415) SAVE R0-R5 ROUTINE
062700  062214                          $RESREG ;CALL=RESREG      TRAP+16(104406) RESTORE R0-R5 ROUTINE

062702  040156                          $KERNEL   ;CALL=KERNEL    TRAP+17(104417) ENTER KERNEL MODE
062704  040166                          $ENERGIZE;CALL=ENERGIZETRAP+20(104420) TURN ON MEMORY MANAGEMENT & TRAPS
062706  040176                          $DEENERGI;CALL=DEENERGITRAP+21(104421) TURN OFF MEMORY MENAGEMENT & TRAPS

062710  042374                          $KMAP     ;CALL=KMAP      TRAP+22(104422) MAP KERNEL 1 TO 1

062712  040206                          $CACHN   ;CALL=CACHON     TRAP+23(104423) TURN CACHE ON
062714  040232                          $CACHF   ;CALL=CACHOFF    TRAP+24(104424) TURN CACHE OFF

062716  040250                          $LOADC   ;CALL=LOADCSR    TRAP+25(104425) LOAD CORRECT CSR
062720  040344                          $READC   ;CALL=READCSR    TRAP+26(104426) READ CORRECT CSR

062722  053654                          $PER01   ;CALL=PERR01     TRAP+27(104427) PROGRAM DETECTED ERROR
062724  053702                          $PER02   ;CALL=PERR02     TRAP+30(104430) PROGRAM DETECTED ERROR
062726  053730                          $PER03   ;CALL=PERR03     TRAP+31(104431) PROGRAM DETECTED ERROR
062730  053760                          $PER04   ;CALL=PERR04     TRAP+32(104432) PROGRAM DETECTED ERROR
062732  054042                          $PER07   ;CALL=PERR07     TRAP+33(104433) PROGRAM DETECTED ERROR
062734  054064                          $PER10   ;CALL=PERR10     TRAP+34(104434) PROGRAM DETECTED ERROR
062736  054114                          $PER11   ;CALL=PERR11     TRAP+35(104435) PROGRAM DETECTED ERROR
062740  054134                          $PER12   ;CALL=PERR12     TRAP+36(104436) PROGRAM DETECTED ERROR
062742  054156                          $PER13   ;CALL=PERR13     TRAP+37(104437) PROGRAM DETECTED ERROR
062744  054176                          $PER14   ;CALL=PERR14     TRAP+40(104440) PROGRAM DETECTED ERROR
062746  054220                          $PER15   ;CALL=PERR15     TRAP+41(104441) PROGRAM DETECTED ERROR
062750  054242                          $PER16   ;CALL=PERR16     TRAP+42(104442) PROGRAM DETECTED ERROR
062752  054262                          $PER17   ;CALL=PERR17     TRAP+43(104443) PROGRAM DETECTED ERROR
062754  054300                          $PER20   ;CALL=PERR20     TRAP+44(104444) PROGRAM DETECTED ERROR
062756  054316                          $PER21   ;CALL=PERR21     TRAP+45(104445) PROGRAM DETECTED ERROR
062760  054336                          $PER22   ;CALL=PERR22     TRAP+46(104446) PROGRAM DETECTED ERROR
062762  054354                          $PER23   ;CALL=PERR23     TRAP+47(104447) PROGRAM DETECTED ERROR
062764  054372                          $PER24   ;CALL=PERR24     TRAP+50(104450) PROGRAM DETECTED ERROR
062766  051130                          $PER25   ;CALL=PERR25     TRAP+51(104451) PROGRAM DETECTED ERROR
```

```
 11565 062770  054562                 $PER26   ;CALL=PERR26     TRAP+52(104452) PROGRAM DETECTED ERROR
 11566 062772  054602                 $PER27   ;CALL=PERR27     TRAP+53(104453) PROGRAM DETECTED ERROR
 11567 062774  051356                 $PER30   ;CALL=PERR30     TRAP+54(104454) PROGRAM DETECTED ERROR
 11568 062776  054772                 $PER31   ;CALL=PERR31     TRAP+55(104455) PROGRAM DETECTED ERROR
 11569 063000  055070                 $PER32   ;CALL=PERR32     TRAP+56(104456) PROGRAM DETECTED ERROR
 11570 063002  055136                 $PER33   ;CALL=PERR33     TRAP+57(104457) PROGRAM DETECTED ERROR
 11571 063004  055216                 $PER34   ;CALL=PERR34     TRAP+60(104460) PROGRAM DETECTED ERROR
 11572 063006  055250                 $PER35   ;CALL=PERR35     TRAP+61(104461) PROGRAM DETECTED ERROR
 11573 063010  055304                 $PER36   ;CALL=PERR36     TRAP+62(104462) PROGRAM DETECTED ERROR
 11574 063012  062636                 $NOTRAP  ;CALL=PERR37     TRAP+63(104463) PROGRAM DETECTED ERROR
 11575 063014  062636                 $NOTRAP  ;CALL=PERR40     TRAP+64(104464) PROGRAM DETECTED ERROR
 11576 063016  062636                 $NOTRAP  ;CALL=PERR41     TRAP+65(104465) PROGRAM DETECTED ERROR
 11577 063020  062636                 $NOTRAP  ;CALL=PERR42     TRAP+66(104466) PROGRAM DETECTED ERROR
 11578 063022  062636                 $NOTRAP  ;CALL=PERR43     TRAP+67(104467) PROGRAM DETECTED ERROR
 11579
 11580 063024  040570                 $ECCDIS  ;CALL=ECCDIS     TRAP+70(104470) DISABLE ECC ON ALL CSR'S
 11581 063026  040604                 $ECC1DIS;CALL=ECC1DIS     TRAP+71(104471) DISABLE ECC ON 1 SELECTED CSR
 11582 063030  040616                 $ECCINIT;CALL=ECCINIT     TRAP+72(104472) INITIALIZE ALL MK11 CSR'S
 11583 063032  040632                 $ECC1INIT;CALL=ECC1INIT   TRAP+73(104473) INITIALIZE 1 SELECTED MK11 CSR
 11584 063034  040672                 $CBCSR   ;CALL=CBCSR      TRAP+74(104474) WRITE GENERATED CHECKBITS IN ALL CSR'S
 11585 063036  040714                 $CB1CSR  ;CALL=CB1CSR     TRAP+75(104475) WRITE GENERATED CHECKBITS IN 1 SELECTED CSR
 11586 063040  040734                 $WASSBE  ;CALL=WASSBE     TRAP+76(104476) WAS THERE A SBE ON ANY CSR?
 11587 063042  041050                 $WAS1SBE;CALL=WAS1SBE     TRAP+77(104477) WAS THERE A SBE ON 1 SELECTED CSR?
 11588 063044  041100                 $WASDBE  ;CALL=WASDBE     TRAP+100(104500)WAS THERE A DBE ON ANY CSR?
 11589 063046  041214                 $WAS1DBE;CALL=WAS1DBE     TRAP+101(104501)WAS THERE A DBE ON 1 SELECTED CSR?
 11590 063050  041244                 $CLRCSR  ;CALL=CLRCSR     TRAP+102(104502)CLEAR ALL CSR'S
 11591 063052  041256                 $CLR1CSR;CALL=CLR1CSR     TRAP+103(104503)CLEAR 1 SELECTED CSR
 11592 063054  041266                 $CHKDIS  ;CALL=CHKDIS     TRAP+104(104504)DISABLE ECC & WRITE CKBITS FROM ALL CSR'S
 11593 063056  041302                 $CHK1DIS;CALL=CHK1DIS     TRAP+105(104505)DISABLE ECC & WRITE CKBITS FROM 1 CSR
 11594 063060  040644                 $ENASBE  ;CALL=ENASBE     TRAP+106(104506)ENABLE TRAPS ON SBE'S FROM ALL CSR'S
 11595 063062  040660                 $ENA1SBE;CALL=ENA1SBE     TRAP+107(104507)ENABLE TRAPS ON SBE'S FROM 1 SELECTED CSR
 11596 063064  040364                 $TSTRD   ;CALL=TSTREAD    TRAP+110(104510)TEST LOC (R1) & TST FOR SBE (WITHOUT FETCHES
 11597 063066  041362                 $INVALID;CALL=INVALID     TRAP+111(104511)INVALIDATE BACKGROUND PATTERN ON BANK
 11598 063070  041412                 $ERRGEN  ;CALL=ERRGEN     TRAP+114(104512)TEST ERROR ADDRESS
 11599 063072  062636                 $NOTRAP
 11600 063074  062636                 $NOTRAP
 11601 063076  062636                 $NOTRAP
 11602 063100  062636                 $NOTRAP
 11603 063102  062636                 $NOTRAP
 11604 063104  062636                 $NOTRAP
 11605 063106  062636                 $NOTRAP
 11608        177776                  ST      =       177776          ;STATUS REGISTER
```

```
                                        .SBTTL  TABLE    ERROR POINTER

                                ;*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
                                ;*      DF              ;;POINTS TO THE DATA FORMAT


063110                          $ERRTB: ;ERROR  1
063110  065401                          EM24
063112  067526                          DH13
063114  063760                          DT13
063116  064316                          DF11
                                        ;ERROR  2
063120  064355                          EM2
063122  067035                          DH1
063124  063610                          DT1
063126  064174                          DF2
                                        ;ERROR  3
063130  064413                          EM3
063132  067115                          DH3
063134  063622                          DT3
063136  064311                          DF9
                                        ;ERROR  4
063140  064445                          EM4
063142  067115                          DH3
063144  063632                          DT4
063146  064311                          DF9
                                        ;ERROR  5
063150  064513                          EM5
063152  067151                          DH5
063154  063642                          DT5
063156  064174                          DF2
                                        ;ERROR  6
063160  064570                          EM6
063162  067151                          DH5
063164  063642                          DT5
063166  064174                          DF2
                                        ;ERROR  7
063170  064615                          EM7
063172  067151                          DH5
063174  063642                          DT5
063176  064174                          DF2
                                        ;ERROR  10
063200  066733                          EM53
063202  070171                          DH25
063204  064136                          DT25
063206  064174                          DF2
```

```
 11667                                              ;ERROR  11
 11668 063210   064655                              EM11
 11669 063212   067275                              DH7
 11670 063214   063674                              DT7
 11671 063216   064220                              DF3
 11672                                              ;ERROR  12
 11673 063220   064655                              EM11
 11674 063222   067275                              DH7
 11675 063224   063674                              DT7
 11676 063226   064233                              DF4
 11677                                              ;ERROR  13
 11678 063230   064677                              EM12
 11679 063232   067405                              DH10
 11680 063234   063724                              DT10
 11681 063236   064174                              DF2
 11682                                              ;ERROR  14
 11683 063240   064655                              EM11
 11684 063242   067275                              DH7
 11685 063244   063674                              DT7
 11686 063246   064246                              DF5
 11687                                              ;ERROR  15
 11688 063250   064655                              EM11
 11689 063252   067275                              DH7
 11690 063254   063674                              DT7
 11691 063256   064261                              DF6
 11692                                              ;ERROR  16
 11693 063260   064723                              EM13
 11694 063262   067526                              DH13
 11695 063264   063760                              DT13
 11696 063266   064316                              DF11
 11697                                              ;ERROR  17
 11698 063270   064755                              EM14
 11699 063272   067526                              DH13
 11700 063274   063760                              DT13
 11701 063276   064316                              DF11
 11702                                              ;ERROR  20
 11703 063300   065021                              EM15
 11704 063302   067526                              DH13
 11705 063304   063760                              DT13
 11706 063306   064316                              DF11
 11707                                              ;ERROR  21
 11708 063310   066762                              EM55
 11709 063312   070215                              DH26
 11710 063314   064146                              DT26
 11711 063316   064174                              DF2
 11712                                              ;ERROR  22
 11713 063320   065067                              EM17
 11714 063322   067275                              DH7
 11715 063324   063674                              DT7
 11716 063326   064246                              DF5
 11717                                              ;ERROR  23
 11718 063330   066570                              EM50
 11719 063332   070043                              DH23
 11720 063334   064074                              DT23
 11721 063336   064327                              DF13
 11724                                          ;ERROR  24
 11725 063340   065127                          EM19
 11726 063342   067526                          DH13
 11727 063344   063760                          DT13
 11728 063346   064316                          DF11
 11729                                          ;ERROR  25
 11730 063350   065204                          EM20
 11731 063352   067526                          DH13
 11732 063354   063760                          DT13
 11733 063356   064316                          DF11
 11734                                          ;ERROR  26
 11735 063360   000000                          0                   ;NO MESSAGE
 11736 063362   067521                          DH12
 11737 063364   063754                          DT12
 11738 063366   064174                          DF2
 11739                                          ;ERROR  27
 11740 063370   065266                          EM21
 11741 063372   067503                          DH11
 11742 063374   063746                          DT11
 11743 063376   064174                          DF2
 11744                                          ;ERROR  30
 11745 063400   065325                          EM22
 11746 063402   067526                          DH13
 11747 063404   063760                          DT13
 11748 063406   064316                          DF11
 11749                                          ;ERROR  31
 11750 063410   000000                          0                   ;NO MESSAGE
 11751 063412   067623                          DH14
 11752 063414   064002                          DT14
 11753 063416   064174                          DF2
 11754                                          ;ERROR  32
 11755 063420   065352                          EM23
 11756 063422   067151                          DH5
 11757 063424   063642                          DT5
 11758 063426   064174                          DF2
 11766                                          ;ERROR  33
 11767 063430   065460                          EM25
 11768 063432   067702                          DH15
 11769 063434   064020                          DT16
 11770 063436   064274                          DF7
 11771                                          ;ERROR  34
 11772 063440   065505                          EM26
 11773 063442   070021                          DH16
 11774 063444   064050                          DT17
 11775 063446   064220                          DF3
```

```
                                  ;ERROR  35
063450   066706                   EM52
063452   070171                   DH25
063454   064136                   DT25
063456   064174                   DF2
                                  ;ERROR  36
063460   065556                   EM27
063462   070021                   DH16
063464   064050                   DT17
063466   064307                   DF8
                                  ;ERROR  37
063470   066362                   EM35
063472   067275                   DH7
063474   063674                   DT7
063476   064220                   DF3
                                  ;ERROR  40
063500   065646                   EM29
063502   067275                   DH7
063504   063674                   DT7
063506   064220                   DF3
                                  ;ERROR  41
063510   065730                   EM30
063512   067275                   DH7
063514   063674                   DT7
063516   064246                   DF5
                                  ;ERROR  42
063520   066047                   EM31
063522   067275                   DH7
063524   063674                   DT7
063526   064220                   DF3
                                  ;ERROR  43
063530   066147                   EM32
063532   067275                   DH7
063534   063674                   DT7
063536   064220                   DF3
                                  ;ERROR  44
063540   066254                   EM33
063542   067275                   DH7
063544   063674                   DT7
063546   064220                   DF3
                                  ;ERROR  45
063550   066624                   EM51
063552   070122                   DH24
063554   064116                   DT24
063556   064337                   DF14
                                  ;ERROR  46
063560   066447                   EM36
063562   067230                   DH6
063564   063660                   DT6
063566   064174                   DF2
```

```
 11856                                         ;ERROR  47
 11857 063570   066516                         EM40
 11858 063572   067072                         DH2
 11859 063574   064056                         DT20
 11860 063576   064174                         DF2
 11898                                         ;ERROR  50
 11899 063600   067003                         EM56
 11900 063602   070233                         DH27
 11901 063604   064154                         DT27
 11902 063606   064346                         DF15
 11912                                                 .SBTTL  ERROR DATA TAGS (DT)
 11913 063610   002016   002032   002042   DT1:        .WORD   ERRPC,ADDRESS,GOOD,BAD,0
       063616   002050   000000
 11917 063622   002016   002034   002070   DT3:        .WORD   ERRPC,PADDRESS,PARCNT,0
       063630   000000
 11918 063632   002016   002032   002066   DT4:        .WORD   ERRPC,ADDRESS,NEMCNT,0
       063640   000000
 11919 063642   002016   177572   177574   DT5:        .WORD   ERRPC,MMRO,MMR1,MMR2,MMR3,CPUERR,0
       063650   177576   172516   177766
       063656   000000
 11920 063660   002016   002372   002350   DT6:        .WORD   ERRPC,APTPAR,LSIZE,APTECC,MSIZE,0
       063666   002374   002352   000000
 11921 063674   002016   002170   002032   DT7:        .WORD   ERRPC,DUMMY,ADDRESS,DUMMY,GOOD,BAD,BADXOR
       063702   002170   002042   002050
       063710   002056
 11922 063712   002170   002170   002170               .WORD   DUMMY,DUMMY,DUMMY,DUMMY,0
       063720   002170   000000
 11923 063724   002172   002174   002176   DT10:       .WORD   DETRO,DETR1,DETR2,DETR3,DETR4,DETR5,DETSP,DETPSW,0
       063732   002200   002202   002204
       063740   002206   002210   000000
 11924 063746   002016   002144   000000   DT11:       .WORD   ERRPC,CSR,0
 11925 063754   002144   000000            DT12:       .WORD   CSR,0
 11926 063760   002016   002170   002032   DT13:       .WORD   ERRPC,DUMMY,ADDRESS,DUMMY,TSTDAT,TSTDAT+2,CHECK,CSR,0
       063766   002170   002240   002242
       063774   002274   002144   000000
 11927 064002   177746   177572   177574   DT14:       .WORD   CONTRL,MMRO,MMR1,MMR2,MMR3,CPUERR,0
       064010   177576   172516   177766
       064016   000000
 11928 064020   002016   002170   002170   DT16:       .WORD   ERRPC,DUMMY,DUMMY,GOOD,GOOD2,GOOD3
       064026   002042   002044   002046
 11929 064034   002050   002052   002054               .WORD   BAD,BAD2,BAD3,DUMMY,DUMMY,0
       064042   002170   002170   000000
 11930 064050   002016   002170   000000   DT17:       .WORD   ERRPC,DUMMY,0
 11935 064056   002016   002042   002050   DT20:       .WORD   ERRPC,GOOD,BAD,0
       064064   000000
 11939 064066   002016   002170   000000   DT22:       .WORD   ERRPC,DUMMY,0
 11940 064074   002016   002170   002042   DT23:       .WORD   ERRPC,DUMMY,GOOD,BAD,DUMMY,DUMMY,DUMMY,DUMMY,0
       064102   002050   002170   002170
       064110   002170   002170   000000
 11941 064116   002016   002170   002144   DT24:       .WORD   ERRPC,DUMMY,CSR,DUMMY,DUMMY,DUMMY,DUMMY,0
       064124   002170   002170   002170
       064132   002170   000000
 11942 064136   002016   002042   002144   DT25:       .WORD   ERRPC,GOOD,CSR,0
       064144   000000
 11943 064146   002016   002050   000000   DT26:       .WORD   ERRPC,BAD,0
 11944 064154   002016   002170   002032   DT27:       .WORD   ERRPC,DUMMY,ADDRESS,DUMMY,DUMMY,DUMMY,DUMMY,0
       064162   002170   002170   002170
       064170   002170   000000
 11951                                               .SBTTL  ERROR DATA FORMATS (DF)
 11952 064174     000      000      000   DF2:      .BYTE   0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
       064177     000      000      000
       064202     000      000      000
       064205     000      000      000
       064210     000      000      000
       064213     000      000      000
       064216     000      000
 11953 064220     000      005      000   DF3:      .BYTE   0,5,0,8.,0,0,0,3,6,2,4
       064223     010      000      000
       064226     000      003      006
       064231     002      004
 11954 064233     000      005      000   DF4:      .BYTE   0,5,0,8.,0,8.,8.,3,6,2,4
       064236     010      000      010
       064241     010      003      006
       064244     002      004
 11955 064246     000      005      000   DF5:      .BYTE   0,5,0,8.,9.,9.,9.,3,6,2,4
       064251     010      011      011
       064254     011      003      006
       064257     002      004
 11956 064261     000      005      000   DF6:      .BYTE   0,5,0,8.,9.,8.,8.,3,6,2,4
       064264     010      011      010
       064267     010      003      006
       064272     002      004
 11957 064274     000      005      010   DF7:      .BYTE   0,5,8.,0,0,9.,0,0,9.,2,4
       064277     000      000      011
       064302     000      000      011
       064305     002      004
 11958 064307     000      005            DF8:      .BYTE   0,5
 11959 064311     000      001      001   DF9:      .BYTE   0,1,1,1,1
       064314     001      001
 11960 064316     000      005      000   DF11:     .BYTE   0,5,0,8.,0,0,0,0,0
       064321     010      000      000
       064324     000      000      000
 11961 064327     000      005      000   DF13:     .BYTE   0,5,0,0,3,6,2,4
       064332     000      003      006
       064335     002      004
 11962 064337     000      005      000   DF14:     .BYTE   0,5,0,3,6,2,4
       064342     003      006      002
       064345     004
 11963 064346     000      005      000   DF15:     .BYTE   0,5,0,8.,3,6,4
       064351     010      003      006
       064354     004
 11969                                            .SBTTL  ERROR MESSAGES (EM)
 11978 064355      103      101      116  EM2:    .ASCIZ  /CAN'T SET 22 BIT MODE IN MMR3/
 11979 064413      120      101      122  EM3:    .ASCIZ  /PARITY ERROR(S) IN BANK 0/
 11980 064445      116      117      116  EM4:    .ASCIZ  /NON-EXISTANT MEMORY (HOLES) IN BANK 0/
 11981 064513      111      114      114  EM5:    .ASCIZ  /ILLEGAL OR RESERVED INSTRUCTION (TRAP TO 10)/
 11982 064570      125      116      105  EM6:    .ASCIZ  /UNEXPECTED TRAP TO 4/
 11983 064615      115      105      115  EM7:    .ASCIZ  /MEMORY MANAGEMENT (TRAP TO 250)/
 11987
 11988 064655      115      105      115  EM11:   .ASCIZ  /MEMORY DATA ERROR/
 11989 064677      104      105      124  EM12:   .ASCIZ  /DETAILED ERROR DUMP/
 11990 064723      115      111      123  EM13:   .ASCIZ  /MISSING EXPECTED SBE FLAG/
 11991 064755      127      122      111  EM14:   .ASCIZ  /WRITE BYTE FAILED TO CLEAR SBE FLAG/
 11992 065021      106      101      111  EM15:   .ASCIZ  /FAILED TO GET INTERRUPT WITH DBE FLAG/
 11993 065067      115      105      115  EM17:   .ASCIZ  /MEMORY DATA ERROR IN CHECK BITS/
 11994 065127      123      102      105  EM19:   .ASCIZ  /SBE OR DBE CAUSED PARITY TRAP WHEN INHIBITED/
 11995 065204      123      102      105  EM20:   .ASCIZ  /SBE OR DBE DID NOT CAUSE PARITY TRAP WHEN ENABLED/
 11996 065266      123      102      105  EM21:   .ASCIZ  /SBE OR DBE ON MASTER TEST WORD/
 11997 065325      115      111      123  EM22:   .ASCIZ  /MISSING EXPECTED DBE/
 11998 065352      125      116      105  EM23:   .ASCIZ  /UNEXPECTED PARITY TRAP/
 11999 065401      122      105      103  EM24:   .ASCIZ  /RECEIVED DBE FLAG WHEN EXPECTING ONLY SBE FLAG/
 12000 065460      103      110      105  EM25:   .ASCIZ  /CHECK BIT DATA ERROR/
 12001 065505      101      104      104  EM26:   .ASCIZ  /ADDRESS PARITY ERROR DID NOT CAUSE ABORT/
 12002 065556      105      103      103  EM27:   .ASCIZ  /ECC INHIBIT MODE POINTER FAILURE - DID NOT PROTECT BANK/
 12006 065646      103      117      122  EM29:   .ASCIZ  /CORRECTION FAILURE WITH ECC ENABLED ON FORCED SBE/
 12007 065730      127      122      111  FM30:   .ASCII  /WRITE BYTE (MOVB) WITH ECC ENABLED FAILED TO CLEAR DATA AT/<CRLF>
 12008 066023      106      117      122          .ASCIZ  /FORCED SBE LOCATION/
 12009 066047      101      123      122  EM31:   .ASCIZ  /ASRB (R3)+ WITH ECC ENABLED CHANGED DATA AT FORCED DBE LOCATION/
 12010 066147      115      117      126  EM32:   .ASCIZ  /MOVB #360,(R2)+ WITH ECC ENABLED CHANGED DATA AT FORCED DBE LOCATION/
 12011 066254      115      117      126  EM33:   .ASCIZ  /MOV #177400,(R1) WITH ECC ENABLED CHANGED DATA AT FORCED DBE LOCATION/
 12012 066362      125      116      105  EM35:   .ASCIZ  /UNEXPECTED CORRECTION WITH ECC DISABLE ON FORCED SBE/
 12013 066447      101      120      124  EM36:   .ASCIZ  /APT SIZE DISAGREES WITH PROGRAM SIZING/
 12019 066516      102      122      101  EM40:   .ASCIZ  /BRANCH GOBBLE FAILED CONDITION CODES TEST/
 12028 066570      102      101      104  EM50:   .ASCIZ  /BAD ERROR ADDRESS GENERATED/
 12029 066624      123      102      105  EM51:   .ASCIZ  /SBE & DBE FLAGS NOT SET ON FORCED UNCORRECTED SBE/
 12030 066706      102      111      124  EM52:   .ASCIZ  /BIT SET ERROR IN CSR/
 12031 066733      102      111      124  EM53:   .ASCIZ  /BIT CLEAR ERROR IN CSR/
 12032 066762      111      114      114  EM55:   .ASCIZ  /ILLEGAL CSR TYPE/
 12033 067003      102      101      104  EM56:   .ASCIZ  /BAD PARITY TRAP GENERATED/
 12039                                              .SBTTL  ERROR DATA HEADERS (DH)
 12040 067035       040      040      120  DH1:     .ASCIZ  /  PC    DEV ADD   GOOD     BAD/
 12041 067072       040      040      120  DH2:     .ASCIZ  /  PC    GD-CC BD-CC/
 12042 067115       040      040      120  DH3:     .ASCIZ  /  PC    1ST ADD  # OF ERRORS/
 12043 067151       040      040      120  DH5:     .ASCIZ  /  PC      MMR0     MMR1     MMR2     MMR3    CPUERR/
 12044 067230       040      040      120  DH6:     .ASCIZ  /  PC    APTPAR   LSIZE  APTECC  MSIZE/
 12045 067275       040      040      120  DH7:     .ASCII  /  PC     BANK  VADD      PADD     GOOD/
 12046 067341       040      040      040           .ASCIZ  /      BAD      XOR  CSR  MTYP INT PAT/
 12047 067405       040      040      122  DH10:    .ASCIZ  /  R0        R1        R2        R3        R4        R5        SP        PSW/
 12048 067503       040      040      120  DH11:    .ASCIZ  /  PC         CSR/
 12049 067521       040      103      123  DH12:    .ASCIZ  / CSR/
 12050 067526       040      040      120  DH13:    .ASCII  /  PC     BANK  VADD      PADD     WROTE1  WROTE2/
 12051 067603       040      103      110           .ASCIZ  / CHKBITS    CSR/
 12052 067623       103      117      116  DH14:    .ASCIZ  /CONTRL   MMR0     MMR1     MMR2     MMR3    CPUERR/
 12053 067702       040      040      120  DH15:    .ASCII  /  PC     BANK     PADD   GD-WD1  GD-WD2  GD-CHK/
 12054 067757       040      102      101           .ASCIZ  / BAD-WD1 BAD-WD2  BAD-CHK INT PAT/
 12055 070021       040      040      120  DH16:    .ASCIZ  /  PC     BANK/
 12060 070036       040      040      120  DH19:    .ASCIZ  /  PC/
 12066 070043       040      040      120  DH23:    .ASCIZ  /  PC     BANK GD-ERR BAD-ERR CSR  MTYP INT  PAT/
 12067 070122       040      040      120  DH24:    .ASCIZ  /  PC     BANK  (CSR) CSR  MTYP INT  PAT/
 12068 070171       040      040      120  DH25:    .ASCIZ  /  PC  GD-DAT  (CSR)/
 12069 070215       040      040      120  DH26:    .ASCIZ  /  PC  BADCODE/
 12070 070233       040      040      120  DH27:    .ASCIZ  /  PC     BANK  VADD      PADD   CSR  MTYP PAT/
```

```
12073                                   .SBTTL  MESSAGES
12074 070307     200     040     103  MSG000: .ASCIZ  <CRLF>" CZMSDB - MS11L/M MEMORY DIAGNOSTIC"
12075 070354     200     040     040  MSG001: .ASCIZ  <CRLF>/                          MEMORY CONFIGURATION MAP/
12076 070436     200     040     040  MSG002: .ASCIZ  <CRLF>/                                16K WORD BANKS/
12077 070513     200     040     040  MSG003: .ASCII <CRLF>/                  1       2        3/
12078 070555     040     040     040          .ASCIZ  /        4        5        6        7  /
12079 070620     200     040     040  MSG004: .ASCII <CRLF>/          012345670123456701234567/
12080 070661     060     061     062          .ASCIZ  /01234567012345670123456701234567012З/
12081 070726     200     105     122  MSG005: .ASCIZ  <CRLF>/ERRORS  /
12082 070740     200     125     116  MSG006: .ASCIZ  <CRLF>/UNDEFINED TRAP INSTRUCTION/<32>
12083 070775     200     111     116  MSG007: .ASCIZ  <CRLF>/INTRLV  /                ;INTERLEAVED CSR #
12084 071007     200     103     120  MSG008: .ASCIZ  <CRLF>/CPU MAP /                ;CPU ACCESSED BANK
12085 071021     200     115     105  MSG009: .ASCIZ  <CRLF>/MEMTYPE /                ;MEMORY TYPE
12086 071033     200     120     122  MSG010: .ASCIZ  <CRLF>/PROTECT /                ;MEMORY PROTECTED
12087 071045     040     040     040  MSG011: .ASCIZ  /    0        1        2        3        4        5       6/
12088 071133     064     065     066  MSG012: .ASCIZ  /4567012345670123456701234567012345670123456701234567/
12089 071230     130     000          MSG013: .ASCIZ  /X/
12090 071232     040     000          MSG014: .ASCIZ  / /                 ;SPACE
12091 071234     000     000          MSG015: .BYTE   0,0                 ;FOR SINGLE ASCII CHARACTERS & TERMINATOR
12092 071236     200     103     123  MSG016: .ASCIZ  <CRLF>/CSR      /
12093 071250     040     040     040  MSG017: .ASCIZ  /        /                  ;8 SPACES
12094 071261     040     040     000  MSG018: .ASCIZ  /  /                        ;2 SPACES
12095 071264     040     040     040  MSG019: .ASCIZ  /   /                       ;3 SPACES
12096 071270     200     106     123  MSG020: .ASCIZ  <CRLF>/FS COMMAND MODE/
12097 071311     200     103     117  MSG021: .ASCII  <CRLF>/COMMANDS AVAILABLE:/
12098 071335     200     060     040          .ASCII  <CRLF>/0 = EXIT/
12099 071346     200     061     040          .ASCII  <CRLF>/1 = READ CSR/
12100 071363     200     062     040          .ASCII  <CRLF>/2 = LOAD CSR/
12101 071400     200     063     040          .ASCII  <CRLF>/3 = EXAMINE MEMORY/
12102 071423     200     064     040          .ASCII  <CRLF>/4 = MODIFY MEMORY/
12103 071445     200     065     040          .ASCII  <CRLF>/5 = SELECT BANK & PATTERN/
12104 071477     200     066     040          .ASCII  <CRLF>/6 = TYPE CONFIG MAP/
12105 071523     200     067     040          .ASCII  <CRLF>/7 = SOB-A-LONG TEST/
12106 071547     200     070     040          .ASCII  <CRLF>/8 = ERROR SUMMARY/
12107 071571     200     071     075          .ASCII  <CRLF>/9= REFRESH TEST/
12108 071611     200     061     060          .ASCII  <CRLF>/10= SET FILL COUNT/
12109 071634     200     061     061          .ASCII  <CRLF>/11= ENTER KAMIKAZE MODE/
12110 071664     200     061     062          .ASCII  <CRLF>/12= EXIT KAMIKAZE MODE/
12111 071713     200     061     063          .ASCII  <CRLF>/13= TURN CACHE OFF/
12112 071736     200     061     064          .ASCII  <CRLF>/14= TURN CACHE ON/
12117 071760     200     061     065          .ASCII  <CRLF>/15= TEST SELECTED BANKS/
12118 072010     200     061     066          .ASCII  <CRLF>/16= TEST ALL BANKS/
12119 072033     200     061     067          .ASCII  <CRLF>/17= ENABLE TRACE/
12120 072054     200     061     070          .ASCII  <CRLF>/18= DISABLE TRACE/
12121 072076     015     012     000          .BYTE   15,12,0
12122 072101     200     127     110  MSG022: .ASCIZ  <CRLF>/WHICH CSR(0-F)? /
12123 072123     200     103     123  MSG023: .ASCIZ  <CRLF>/CSR WORD? /
12124 072137     200     103     123  MSG025: .ASCIZ  <CRLF>/CSR DOES NOT EXIST/
12125 072163     200     103     117  MSG026: .ASCIZ  <CRLF>/COMMAND:/
12126 072175     200     117     114  MSG027: .ASCIZ  <CRLF>/OLD CSR WAS/
12127 072212     200     103     123  MSG028: .ASCIZ  <CRLF>/CSR IS NOW/
12128 072226     200     105     130  MSG029: .ASCIZ  <CRLF>/EXAMINE MEMORY/
12129 072246     200     102     101  MSG030: .ASCIZ  <CRLF>/BANK(0-177)? /
12130 072265     200     120     110  MSG031: .ASCIZ  <CRLF>/PHYSICAL ADDRESS(0-17757776)? /
12131 072325     200     120     101  MSG032: .ASCIZ  <CRLF>/PARITY ABORT/<32>
12132 072344     200     124     111  MSG033: .ASCIZ  <CRLF>/TIMEOUT TRAP/<32>
12133 072363     200     102     131  MSGA34: .ASCIZ  <CRLF>/BYPASSING ECC LOGIC TESTS ON BANK /
```

```
12134 072427    040    104    125  MSGB34: .ASCIZ  / DUE TO LACK OF SBE FREE LOCATIONS/
12135 072472    121    126    000  MSG035: .ASCIZ  /QV/
12136 072475    200    115    117  MSG036: .ASCIZ  <CRLF>/MODIFY MEMORY/
12137 072514    200    117    114  MSG037: .ASCIZ  <CRLF>/OLD DATA WAS /
12138 072533    200    104    101  MSG038: .ASCIZ  <CRLF>/DATA IS NOW /
12139 072551    200    111    116  MSG039: .ASCIZ  <CRLF>/INPUT NEW DATA? /
12140 072573    200    123    105  MSG040: .ASCIZ  <CRLF>/SELECT BANK & PATTERN TEST/
12141 072627    200    102    101  MSG041: .ASCIZ  <CRLF>/BANK NOT ACCESSABLE/
12142 072654    200    120    101  MSG042: .ASCIZ  <CRLF>/PATTERN(0-35)? /
12143 072675    200    120    101  MSG043: .ASCIZ  <CRLF>/PATTERN 0 DATA IS? /
12144 072722    200    124    117  MSG046: .ASCIZ  <CRLF>/TO ESCAPE TYPE ANY KEY/<CRLF><12><12>
12145 072755    200    124    105  MSG047: .ASCIZ  <CRLF>/TEST COMPLETE/
12146 072774    040    116    117  MSG048: .ASCIZ  / NOT AVAILABLE NOW - TRY LATER!/
12147 073034    200    102    101  MSG049: .ASCIZ  <CRLF>/BANK REQUIRES RELOCATION/
12148 073066    200    102    101  MSG050: .ASCII  <CRLF>/BATTERY BACKUP TEST/
12149 073112    200    127    122           .ASCIZ  <CRLF>/WRITING & CHECKING ADDRESS PATTERN AS BACKGROUND/
12150 073174    200    120    117  MSG051: .ASCIZ  <CRLF>/POWER RECOVERY/
12151 073214    200    122    105  MSG052: .ASCIZ  <CRLF>/REMOVE SYSTEM POWER FOR/
12152 073245    040    123    105  MSG053: .ASCIZ  / SECONDS MAX!/
12153 073263    200    116    117  MSG054: .ASCIZ  <CRLF>/NOW STARTING READ TEST OF MEMORY BANKS/
12154 073333    200    123    117  MSG055: .ASCIZ  <CRLF>/SOB-A-LONG TEST/
12155 073354    200    102    105  MSG056: .ASCIZ  <CRLF>/BELL = EACH PASS COMPLETE/
12156 073407    200    040    040  MSG058: .ASCIZ  <CRLF>/  CSR    CSR .../
12157 073431    077    077    077  MSG061: .ASCIZ  /??????/
12158 073440    111    116    120  MSG062: .ASCIZ  /INPUT MUST BE A/
12159 073460    116    040    117  MSG063: .ASCIZ  /N OCTAL /
12160 073471    116    125    115  MSG064: .ASCIZ  /NUMBER/<CRLF>
12161 073501    040    104    105  MSG065: .ASCIZ  / DECIMAL /
12162 073513    200    105    122  MSG066: .ASCIZ  <CRLF>/ERROR COUNT EXCEEDED 20 - ABORTING FOR XXDP CHAIN/
12163 073576    106    101    124  MSG067: .ASCIZ  /FATAL /
12164 073605    113    040    127  MSG070: .ASCIZ  /K WORDS OF MEMORY TOTAL/<CRLF>
12165 073636    113    040    117  MSG071: .ASCIZ  /K OF BIPOLAR/<CRLF>
12166 073654    113    040    117  MSG072: .ASCIZ  /K OF MF11S-K/<CRLF>
12167 073672    200    122    105  MSG073: .ASCIZ  <CRLF>/REFRESH TEST/
12168 073710    200    122    105  MSG075: .ASCIZ  <CRLF>/RELOCATION NOT POSSIBLE/<32>
12169 073742    200    040    040  MSG076: .ASCIZ  <CRLF>/  BANK  ERRORS/<CRLF>
12170 073763    200    105    116  MSG077: .ASCIZ  <CRLF>/END PASS #/
12171 073777    040    105    122  MSG079: .ASCIZ  / ERROR(S) DETECTED/<CRLF>
12178 074023    200    106    111  MSG085: .ASCIZ  <CRLF>/FILL COUNT(OCTAL)? /
12186 074050    200    113    105  MSG088: .ASCIZ  <CRLF>/KERNEL STACK/
12187 074066    200    123    125  MSG089: .ASCIZ  <CRLF>/SUPERVISOR STACK/
12188 074110    200    125    123  MSG090: .ASCIZ  <CRLF>/USER STACK/
12189 074124    040    111    123  MSG091: .ASCIZ  / IS EMPTY/
12190 074136    122    105    114  MSG092: .ASCIZ  /RELOCATED  /
12191 074152    102    101    116  MSG093: .ASCIZ  /BANK=/
12192 074160    040    040    120  MSG095: .ASCIZ  /  PAT=/
12200 074167    200    105    116  MSG101: .ASCIZ  <CRLF>/ENTERING KAMIKAZE MODE/
12201 074217    200    114    105  MSG102: .ASCIZ  <CRLF>/LEAVING KAMIKAZE MODE/
12202 074246    200    114    105  MSG103: .ASCIZ  <CRLF>/LEAVING FS MODE/<CRLF>
12203 074270    032    000         MSG104: .BYTE   32,0                    ;CONTROL Z
12204 074272    200    105    116  MSG105: .ASCIZ  <CRLF>/ENTER BANKS IN OCTAL - USE NUMBER OUTSIDE RANGE TO TERMINATE (177)/
12205 074376    200    103    101  MSG106: .ASCIZ  <CRLF>/CACHE IS OFF/
12206 074414    200    103    101  MSG107: .ASCIZ  <CRLF>/CACHE IS ON (EXCEPT DURING ACTUAL PATTERNS)/
12211 074471    200    117    116  MSG110: .ASCIZ  <CRLF>/ONLY SELECTED BANKS WILL BE TESTED/
12212 074535    200    101    114  MSG111: .ASCIZ  <CRLF>/ALL BANKS WILL BE TESTED/
12213 074567    113    040    117  MSG112: .ASCIZ  /K OF MS11-L/<CRLF>
12214 074604    113    040    117  MSG113: .ASCIZ  /K OF MS11-M/<CRLF>
```

```
 12215 074621      113      040      117  MSG114: .ASCIZ   /K OF UNIBUS PARITY/<CRLF>
 12216 074645      200      040      040  MSG116: .ASCIZ   <CRLF>"    11/34"
 12217 074657      200      040      040  MSG117: .ASCIZ   <CRLF>"    11/44"
 12218 074671      200      040      040  MSG118: .ASCIZ   <CRLF>"    11/60"
 12219 074703      200      040      040  MSG119: .ASCIZ   <CRLF>/    NO/
 12220 074712      040      103      101  MSG120: .ASCIZ   / CACHE AVAILABLE/
 12221 074733      040      103      101  MSG121: .ASCIZ   / CACHE BYPASSED/
 12222 074753      200      103      123  MSG122: .ASCII   <CRLF>/CSR NUMBER /
 12223 074767      000                    MSGA122:.BYTE    0
 12224 074770      040      103      117           .ASCIZ   / CONTROLS TOO MANY BANKS/
 12225 075021      200      120      122  MSG123: .ASCIZ   <CRLF>/PROGRAM RELOCATED - ECC TESTS INHIBITED/
 12226 075072      200      116      125  MSG124: .ASCIZ   <CRLF>/NUMBER OF CSR'S IS WRONG IN BANK /
 12227 075135      040      120      101  MSG125: .ASCIZ   / PASSES COMPLETED/
 12228 075157      200      120      122  MSG126: .ASCIZ   <CRLF>/PROGRAM CSR COULD NOT BE DETERMINED/
 12229 075224      200      124      122  MSG127: .ASCIZ   <CRLF>/TRACE ENABLED/
 12230 075243      200      124      122  MSG128: .ASCIZ   <CRLF>/TRACE DISABLED/
 12231                                             .EVEN
 12237 075264                                      $$END
 12238 075264                             END:
 12239 075264   004124                             .PRINT  60000-SUPLIMIT   ;SUPERVISOR ADDRESSES LEFT
 12251 075264   002514                             .PRINT  100000-END       ;ADDRESSES LEFT IN 16K
 12255          000200                             .END START3
```

```
ABORTF  002140      B10     014560      CBCSR = 104474      DEENER= 104421      DT20    064056
ACFLAG  002114      B11     016176      CB1CSR= 104475      DETAIL  057132      DT22    064066
ACTFLA  002320      B12     016444      CHECK   002274      DETFLA  002212      DT23    064074
ADDRES  002032      B13     016452      CHKDIS= 104504      DETPSW  002210      DT24    064116
ANA2    007134      B14     016470      CHKGEN  041662      DETR0   002172      DT25    064136
APTDOW  045210      B15     016532      CHKTAB  041770      DETR1   002174      DT26    064146
APTECC  002374      B16     016604      CHK1DI= 104505      DETR2   002176      DT27    064154
APTFLA  002322      B17     022000      CKEND   060706      DETR3   002200      DT3     063622
APTHAN  014514      B2      012450      CKSWR = 104410      DETR4   002202      DT4     063632
APTHLT  045256      B20     022240      CLRCSR= 104502      DETR5   002204      DT5     063642
APTPAR  002372      B21     024260      CLREX   007104      DETSP   002206      DT6     063660
APTSIZ  002414      B22     024264      CLRMEM  007004      DET1    057622      DT7     063674
BACKGN  036470      B23     024464      CLR1CS= 104503      DF11    064316      DUMMY   002170
BAD     002050      B24     024472      CMD16A  050762      DF13    064327      DUMPCS= 000061
BADPC   002020      B25     024770      CMD16L= 000052      DF14    064337      ECCDIS= 104470
BADPSW  002030      B26     034410      CMD5B   047200      DF15    064346      ECCINI= 104472
BADSP   002024      B27     034446      CMD5C   047450      DF2     064174      ECC1DI= 104471
BADSTA  040126      B3      012512      CMD7B   047700      DF3     064220      ECC1IN= 104473
BADXOR  002056      B30     034462      CMD7C   047754      DF4     064233      EMTVEC= 000030
BAD2    002052      B31     034602      CMD9B   050376      DF5     064246      EM11    064655
BAD3    002054      B32     034762      CMD9C   050452      DF6     064261      EM12    064677
BAFPAF  014650      B33     035004      CONFGE  002420      DF7     064274      EM13    064723
BAFPAR  014756      B34     036200      CONFIE  003630      DF8     064307      EM14    064755
BAKPAT  002572      B35     040746      CONFIG  002624      DF9     064311      EM15    065021
BANK    002100      B36     040752      CONTFL  002214      DH1     067035      EM17    065067
BANKIN  002102      B37     041112      CONTRL= 177746      DH10    067405      EM19    065127
BANKMO  043710      B4      013316      CONTS   061056      DH11    067503      EM2     064355
BANKOK  044710      B40     041116      CONTS1  060434      DH12    067521      EM20    065204
BAWPAF  015064      B41     041322      CONTS2  061060      DH13    067526      EM21    065266
BAWPAR  015214      B42     041326      CONTS3  053536      DH14    067623      EM22    065325
BGTEST  036046      B43     042530      CONTT   061002      DH15    067702      EM23    065352
BIT0  = 000001      B44     042536      COUNT   002340      DH16    070021      EM24    065401
BIT1  = 000002      B45     042666      CPUBIT  002104      DH19    070036      EM25    065460
BIT10 = 002000      B46     042702      CPUERR= 177766      DH2     067072      EM26    065505
BIT11 = 004000      B47     047704      CR    = 000015      DH23    070043      EM27    065556
BIT12 = 010000      B5      013460      CRLF  = 000200      DH24    070122      EM29    065646
BIT13 = 020000      B50     050126      CSR     002144      DH25    070171      EM3     064413
BIT14 = 040000      B51     050402      CSRADD= 172100      DH26    070215      EM30    065730
BIT15 = 100000      B52     050676      CSRCAS  017356      DH27    070233      EM31    066047
BIT2  = 000004      B53     050676      CSRFBA  002224      DH3     067115      EM32    066147
BIT3  = 000010      B54     050774      CSRFIR  002220      DH5     067151      EM33    066254
BIT4  = 000020      B55     051734      CSRHOL  002476      DH6     067230      EM35    066362
BIT5  = 000040      B56     051740      CSRINC  002300      DH7     067275      EM36    066447
BIT6  = 000100      B57     052236      CSRINF  002432      DIAGFL  002002      EM4     064445
BIT7  = 000200      B6      013522      CSRINT  002230      DISPLA  002600      EM40    066516
BIT8  = 000400      B60     052244      CSRLAS  002222      DISPRE= 000174      EM5     064513
BIT9  = 001000      B61     057366      CSRLBA  002226      DISPTB  014176      EM50    066570
BLOCK1  045260      B62     057372      CSRLOO  002302      DOBACK  014550      EM51    066624
BLOCK2  045300      B63     057470      CSRNO   002146      DSWR  = 177570      EM52    066706
BLOCK3  045314      B64     057562      CSROUT  041314      DT1     063610      EM53    066733
BMFLAG  002126      B65     057646      CSRSTU= 000014      DT10    063724      EM55    066762
BOOT    044766      B7      014006      CTEST   006106      DT11    063746      EM56    067003
BOOT1   045032      CACHKF  002520      CTLKVE  002142      DT12    063754      EM6     064570
BRGOBB  036050      CACHKN  002514      DATARG= 177754      DT13    063760      EM7     064615
BSIZE   002344      CACHOF= 104424      DATBUF  002234      DT14    064002      ENASBE= 104506
B0      004522      CACHON= 104423      DBEMSK  002250      DT16    064020      ENA1SB= 104507
B1      011670      CACHVE= 000114      DDISP = 177570      DT17    064050      END     075264
```

```
ENERGI= 104420      E52     050736      HIPAT   044742      LOADHO  002536      L153    021452
ENEXBK  044700      E53     050736      HOLDLO= 000011      LOOP    014150      L154    021516
ERRADD  002430      E54     051012      HT    = 000011      LOWMAP  043644      L155    021526
ERRGEN= 104512      E55     051774      I       002422      LSIZE   002350      L156    021526
ERRMAX  002524      E56     051774      IIII  = 177777      LWDBE = 000037      L157    021536
ERROR = 104000      E57     052300      ILLCSR  013216      LWSBE = 000035      L16     010662
ERRPC   002016      E6      013550      IMPTES  012160      L0      004532      L160    021574
ERRPSW  002026      E60     052300      INCBNK  044752      L1      004574      L161    021604
ERRSP   002022      E61     057434      INCPAT  044726      L10     005072      L162    021604
ERRVEC= 000004      E62     057434      INCRPT  044726      L100    014150      L163    021642
EUFLAG  002130      E63     057520      INHBAN  002510      L101    014264      L164    021652
EVEN    002334      E64     057614      INHECC  002506      L102    014364      L165    021652
EXBANK  044240      E65     057700      INTFLA  002134      L103    014374      L166    021732
EXCMD3  046330      E7      014126      INT64K  002136      L104    014544      L167    021742
EXCMD4  046650      FASTCI= 177640      INVALI= 104511      L105    014544      L17     010664
EXIT    045074      FATAL$  002062      IOTVEC= 000020      L106    014632      L170    021742
EXIT2   045100      FCMD10  050536      JMPRL1  043240      L107    014744      L171    022104
E0      004550      FCMD11  050564      KAMIKA  002004      L11     005100      L172    022032
E1      012106      FCMD12  050606      KAMITE  026370      L110    015052      L173    022032
E10     014646      FCMD13  050626      KDIAG = 000010      L111    015202      L174    022064
E11     016216      FCMD14  050650      KDPAR0= 172360      L112    015332      L175    022064
E12     016770      FCMD15  050666      KDPAR6= 172374      L113    015462      L176    022142
E13     016754      FCMD16  050752      KDPAR7= 172376      L114    015634      L177    022176
E14     016710      FCMD17  051014      KERNEL= 104417      L115    015764      L2      004702
E15     016710      FCMD18  051030      KERSTK= 002000      L116    016136      L20     011476
E16     016660      FIELDS  045344      KFLAG   002500      L117    016216      L200    022200
E17     022076      FINDBA= 000045      KIPAR0= 172340      L12     005130      L201    022226
E2      012652      FINT    006404      KIPAR4= 172350      L120    016254      L202    022230
E20     022340      FIRST = 060000      KIPAR5= 172352      L121    016264      L203    022314
E21     024442      FLIPLO  002556      KIPAR6= 172354      L122    016710      L204    022426
E22     024426      FLIPWA  036340      KIPDR0= 172300      L123    016672      L205    022430
E23     024650      FLUSH   014270      KMAP  = 104422      L124    016736      L206    023270
E24     024634      FSCMD0  045542      KPFLAG  002112      L125    017200      L207    023322
E25     025046      FSCMD1  045644      KSIZE   002346      L126    017310      L21     011512
E26     034754      FSCMD2  045754      KSTACK  002534      L127    017446      L210    023366
E27     034740      FSCMD3  046122      LAST  = 157776      L13     005362      L211    023636
E3      012636      FSCMD4  046376      LASTBA  002526      L130    017474      L212    023646
E30     034554      FSCMD5  046716      LASTBL  002530      L131    017500      L213    023646
E31     034724      FSCMD6  047540      LASTER  002014      L132    017502      L214    024412
E32     035000      FSCMD7  047546      LBLS0 = 000455      L133    017546      L215    024360
E33     035024      FSCMD8  050040      LBLS1 = 000065      L134    017602      L216    024412
E34     036322      FSCMD9  050244      LBLS2 = 000447      L135    017606      L217    024456
E35     041020      FSINFL  002412      LBLS3 = 000442      L136    017610      L22     011636
E36     041020      FSPAT   047354      LBLS4 = 000315      L137    017756      L220    024620
E37     041164      FSSTAC  002266      LBLS5 = 000317      L14     005362      L221    024566
E4      013452      FS1     045430      LBLS6 = 000016      L140    020432      L222    024620
E40     041164      FS7FLA  002416      LCSROU= 000041      L141    021300      L223    025032
E41     041356      FULLRE  002512      LCSRRE= 000057      L142    021310      L224    025032
E42     041356      GBLENG= 000076      LCSRSA= 000055      L143    021310      L225    025140
E43     042666      GETDAT  051424      LEGALC= 000003      L144    021346      L226    025112
E44     042646      GETDA1  051522      LF    = 000012      L145    021356      L227    025170
E45     043106      GETDIS  055570      LINK1   002472      L146    021356      L23     011614
E46     043014      GOOD    002042      LINK2   002474      L147    021406      L230    025360
E47     047746      GOOD2   002044      LKS   = 177546      L15     010656      L231    025712
E5      013520      GOOD3   002046      LOADBA  002402      L150    021412      L232    026202
E50     050230      GTSWR = 104407      LOADCS= 104425      L151    021442      L233    026240
E51     050444      HEADER  002552      LOADER= 000043      L152    021452      L234    026340
```

```
L235     026412          L317     043004          L400     054426          L53      013442          MSG007   070775
L236     026420          L32      011754          L401     054440          L54      013416          MSG008   071007
L237     026424          L320     043052          L402     054456          L55      013372          MSG009   071021
L24      011626          L321     043052          L403     054460          L56      013364          MSG010   071033
L240     030242          L322     043102          L404     054500          L57      013370          MSG011   071045
L241     030240          L323     043222          L405     054512          L6       005100          MSG012   071133
L242     030242          L324     044402          L406     054530          L60      013414          MSG013   071230
L243     031670          L325     044626          L407     054532          L61      013410          MSG014   071232
L244     034432          L326     045010          L41      012354          L62      013414          MSG015   071234
L245     034442          L327     045114          L410     054546          L63      013442          MSG016   071236
L246     034554          L33      012072          L411     054750          L64      013442          MSG017   071250
L247     034504          L330     045120          L412     054756          L65      013442          MSG018   071261
L25      011642          L331     045126          L413     055004          L66      013536          MSG019   071264
L250     034516          L332     045142          L414     055022          L67      013606          MSG020   071270
L251     034534          L333     045154          L415     055034          L7       005100          MSG021   071311
L252     034542          L334     045366          L416     055046          L70      013760          MSG022   072101
L253     034566          L335     045376          L417     055060          L71      014150          MSG023   072123
L254     034724          L336     045532          L42      012354          L72      014116          MSG025   072137
L255     034624          L337     045572          L420     055102          L73      014034          MSG026   072163
L256     034636          L34      012024          L421     055150          L74      014036          MSG027   072175
L257     034670          L340     045576          L422     055230          L75      014076          MSG028   072212
L26      012072          L341     045626          L423     055244          L76      014112          MSG029   072226
L260     034654          L342     045640          L424     055246          L77      014146          MSG030   072246
L261     034666          L343     047026          L425     055352          MAINT  = 177750          MSG031   072265
L262     034712          L344     047066          L426     055414          MAPH0  = 170202          MSG032   072325
L263     034700          L345     047124          L427     055424          MAPL0  = 170200          MSG033   072344
L264     034712          L346     047272          L43      012354          MAPL1  = 170204          MSG035   072472
L265     035024          L347     047306          L430     055726          MAPPER   042030          MSG036   072475
L266     036702          L35      012072          L431     055726          MBERR    013074          MSG037   072514
L267     037330          L350     047320          L432     056070          MEMDON   014216          MSG038   072533
L27      012072          L351     047616          L433     056030          MFPT   = 000007          MSG039   072551
L270     037516          L352     047620          L434     056050          MJPAT    020002          MSG040   072573
L271     037620          L353     047732          L435     056070          MJTEST   017676          MSG041   072627
L272     040532          L354     050230          L436     056266          MKCONT   016244          MSG042   072654
L273     040540          L355     050214          L437     056136          MKCSRT   017366          MSG043   072675
L274     040776          L356     050164          L44      012360          MKFLAG   002116          MSG046   072722
L275     040776          L357     050314          L440     056264          MKLOOP   016426          MSG047   072755
L276     041040          L36      012144          L441     056214          MKPAT    017616          MSG048   072774
L277     041046          L360     050316          L442     056226          MKTEST   017456          MSG049   073034
L3       004724          L361     050430          L443     056264          MMR0   = 177572          MSG050   073066
L30      012072          L362     050720          L444     056274          MMR1   = 177574          MSG051   073174
L300     041142          L363     050722          L445     056574          MMR2   = 177576          MSG052   073214
L301     041142          L364     051166          L446     057412          MMR3   = 172516          MSG053   073245
L302     041204          L365     051202          L447     057616          MMTRAP   040102          MSG054   073263
L303     041212          L366     051206          L45      012636          MMVEC  = 000250          MSG055   073333
L304     041334          L367     051224          L450     057622          MSEEDH   002546          MSG056   073354
L305     042502          L37      012150          L451     057702          MSEEDL   002550          MSG058   073407
L306     042516          L370     051352          L452     057706          MSGA12   074767          MSG061   073431
L307     042530          L371     051354          L453     061022          MSGA34   072363          MSG062   073440
L31      011750          L372     051420          L454     061112          MSGB34   072427          MSG063   073460
L310     042530          L373     051644          L455     061114          MSG000   070307          MSG064   073471
L311     042632          L374     051752          L46      012636          MSG001   070354          MSG065   073501
L312     042662          L375     052256          L47      012612          MSG002   070436          MSG066   073513
L313     043004          L376     053574          L5       005020          MSG003   070513          MSG067   073576
L314     043004          L377     053576          L50      012564          MSG004   070620          MSG070   073605
L315     043004          L4       004742          L51      012574          MSG005   070726          MSG071   073636
L316     042764          L40      012152          L52      012614          MSG006   070740          MSG072   073654
```

SYMBOL TABLE

```
MSG073  073672      MTPB21  034262      MT0017  021674      PCONFS  037070      QVFLAG  002316
MSG075  073710      MTPB24  035116      MT0020  021716      PCONF1  037000      RANODD  035646
MSG076  073742      MTPB25  035510      MT0021  023006      PCONF2  037036      RDCHR = 104411
MSG077  073763      MTPB26  035632      MT0022  023260      PDP110  040114      RDDEC = 104414
MSG079  073777      MTPC03  027204      MT0023  023312      PD1     051644      RDLIN = 104412
MSG085  074023      MTPC20  034126      MT0024  023356      PERA05  054012      RDOCT = 104413
MSG088  074050      MTPC21  034316      MT0025  023622      PERBNK  054644      READCS= 104426
MSG089  074066      MTPC24  035132      MT0026  023670      PERECC  054724      READON  002360
MSG090  074110      MTPC25  035550      MT0027  024172      PERRAB  054462      REALPA  002260
MSG091  074124      MTPC26  035666      MT0030  024656      PERRAW  054410      REFRES  034756
MSG092  074136      MTPD03  027222      MT0031  025160      PERRA3  051212      REFSUB  035026
MSG093  074152      MTPD20  034156      MT0032  025350      PERRA7  054534      REGCOP  036330
MSG095  074160      MTPD21  034352      MT0033  025702      PERR01= 104427      RELENT  043112
MSG101  074167      MTPD25  035414      MT0034  026070      PERR02= 104430      RELOCA  042466
MSG102  074217      MTPD26  035706      MT0035  026242      PERR03= 104431      RELOC1  043126
MSG103  074246      MTPE20  034206      MT020Y  022554      PERR04= 104432      RESREG= 104416
MSG104  074270      MTPE25  035436      MT020Z  022370      PERR05  054006      RESTAR  002566
MSG105  074272      MTP000  026774      MT0999  026354      PERR06  054034      RESVEC= 000010
MSG106  074376      MTP001  027020      MT1     016224      PERR07= 104433      RES0    045640
MSG107  074414      MTP002  027052      MT2     016230      PERR10= 104434      RES1    045720
MSG110  074471      MTP005  027316      MUT     002106      PERR11= 104435      RES2    046066
MSG111  074535      MTP006  027352      NC      053576      PERR12= 104436      RLFLAG  002124
MSG112  074567      MTP007  027552      NEMCNT  002066      PERR13= 104437      RRFLAG  002122
MSG113  074604      MTP010  027652      NEWBAN  002270      PERR14= 104440      RTNVAL=%000000
MSG114  074621      MTP011  027760      NEWKER  044140      PERR15= 104441      SAVREG= 104415
MSG116  074645      MTP012  030556      NEWLOA  044206      PERR16= 104442      SBEMSK  002244
MSG117  074657      MTP013  031144      NOCH    060416      PERR17= 104443      SBENT   017330
MSG118  074671      MTP014  031660      NOERRO  002400      PERR20= 104444      SBETES  017052
MSG119  074703      MTP015  032442      NOFSMO  002376      PERR21= 104445      SCOPE = 000004
MSG120  074712      MTP016  033206      NONEM   002076      PERR22= 104446      SDPAR0= 172260
MSG121  074733      MTP017  033770      NONEXI  040024      PERR23= 104447      SDPAR5= 172272
MSG122  074753      MTP020  034046      NOOJ    036702      PERR24= 104450      SDPAR6= 172274
MSG123  075021      MTP022  034402      NOPAR   002074      PERR25= 104451      SDPAR7= 172276
MSG124  075072      MTP025  035150      NOSCOP  002410      PERR26= 104452      SEEDHI  002542
MSG125  075135      MTP030  035724      NOSUPE  002426      PERR27= 104453      SEEDLO  002544
MSG126  075157      MTP031  035734      NOTAB   002342      PERR30= 104454      SELONL  002000
MSG127  075224      MTP032  036012      NOTRCE  055366      PERR31= 104455      SETPAT  044742
MSG128  075243      MTP033  036044      NO22BI  002424      PERR32= 104456      SHADL1  011546
MSIZE   002352      MTP034  036142      NULLFL  002314      PERR33= 104457      SHUTUP  045126
MTA030  024670      MTP035  036166      OLDCAC  002262      PERR34= 104460      SIPAR0= 172240
MTB020= 000017      MTST3   011516      OLDCSR  002150      PERR35= 104461      SIPAR3= 172246
MTEST   016150      MTV020  022366      ONES    002554      PERR36= 104462      SIPAR5= 172252
MTLA11  030006      MT0000  020062      PADDRE  002034      PERR37= 104463      SIPAR6= 172254
MTLB11  030020      MT0001  020142      PAFBAF  015344      PERR40= 104464      SIPDR0= 172200
MTLC11  030032      MT0002  020262      PAFBAW  015474      PERR41= 104465      SIZE  = 040000
MTLD11  030126      MT0003  020422      PARBAF  015646      PERR42= 104466      SKIPKA  002006
MTL020  021772      MT0004  020654      PARBAW  015776      PERR43= 104467      SKIPMK  002312
MTPA03  027104      MT0005  020776      PARCNT  002070      PERXOR  054620      SKPERR  002064
MTPA04  027242      MT0006  021132      PARITY  037720      PFLAG   002120      SKUB    043102
MTPA20  034046      MT0007  021166      PARTHE  002264      PGMCSR  002502      SKUJ    013100
MTPA21  034232      MT0010  021230      PARVEC= 000114      PHEBE   013076      SOBK    002532
MTPA24  035056      MT0011  021264      PASFLG  002256      PHYADD  002036      SOBLEN= 000056
MTPA25  035466      MT0012  021332      PATERR  002072      PROTYP  003710      SOFTPA  002560
MTPA26  035616      MT0013  021426      PATPLU  004514      PSIZE   002354      SOURCE  002272
MTPB03  027144      MT0014  021502      PATTER  002110      PSW   = 177776      SPLTCS  002232
MTPB04  027276      MT0015  021560      PCBUMP  002276      PWRVEC= 000024      SSP   =%000006
MTPB20  034076      MT0016  021626      PCONFI  036570      QUICK   002406      ST    = 177776
```

```
STACK = 002000          TAG75$  056744          UNRELO  043356          $DDW3   062560          $MADR2  062524
START   003630          TAG76$  056756          UPPFLG  002257          $DDW4   062562          $MADR3  062530
START1  000300          TAG77$  057022          USERMA  044056          $DDW5   062564          $MADR4  062534
START2  000310          TAG78$  057030          USESTK= 000700          $DEENE  040176          $MAIL   062466
START3  000200          TAG79$  057110          USP    =%000006         $DEVCT  062476          $MAMS1  062516
STAR27  024252          TAG9$   010764          WARN1   011052          $DEVM   062544          $MAMS2  062522
STOPOK  002370          TBG4$   026676          WARN2   027116          $DIDDO= 000000          $MAMS3  062526
STRIPE  002336          TCFIG1  037214          WARN3   027132          $DOAGA  014544          $MAMS4  062532
SUBAAA  004552          TCFIG2  037346          WARN4   027156          $DOAGN  014440          $MBADR  062570
SUBAAB  004702          TCFIG3  037534          WARN5   027172          $DOWN   052074          $MNEW   061606
SUBAAI  011542          TCONFI  037072          WARN6   036552          $DTBL   060354          $MSGAD  062502
SUBAAP  013260          TEMP    002404          WARN6A  036512          $ECCDI  040570          $MSGLG  062504
SUBAAR  012422          TEST    006010          WARN6B  036544          $ECCIN  040616          $MSGTY  062466
SUBAAS  010474          TESTAD  002362          WARN7   024230          $ECC1D  040604          $MSWR   061575
SUCCES  002304          TESTMO  002522          WASDBE= 104500          $ECC1I  040632          $MTYP1  062517
SUPDOA  002254          TIME    002310          WASSBE= 104476          $ENASB  040644          $MTYP2  062523
SUPDO1  026424          TIMEOU  040070          WAS1DB= 104501          $ENA1S  040660          $MTYP3  062527
SUPDO2  026440          TKVEC = 000060          WAS1SB= 104477          $ENDAD  014430          $MTYP4  062533
SUPDO3  026602          TMFLAG  002132          WHICHC  051042          $ENERG  040166          $NOTRA  062636
SUPDO4  026616          TOOMAN  002356          WOOPEN  053052          $ENV    062506          $NULL   002326
SUPDR0  002152          TOTCSR  002216          WOOPS   052504          $ENVM   062507          $NWTST= 000001
SUPDR1  002154          TRACE   006104          WOOPSA  053102          $EOP    014274          $OCNT   060144
SUPDR2  002156          TRAPVE= 000034          WOOPUP  052670          $ERFLG  002012          $OCTVL  062450
SUPDR3  002160          TSTBAN  011404          WORST   002540          $ERRGE  041412          $OCT8 = 062454
SUPDR4  002162          TSTDAT  002240          XOCHAR  053444          $ERROR  055642          $OMODE  060146
SUPDR5  002164          TSTRD1  040542          XXDPCH  002324          $ERRTB  063110          $OVER   055556
SUPDR6  002166          TSTREA= 104510          ZEROS   002306          $ERRTY  056306          $O$   = 000000
SUPLIM  053654          TST1    005402          $APTHD  062566          $ERTTL  002570          $PASS   062474
SUPSTK= 000740          TST2    010500          $AUTO   002060          $ESCAP  002332          $PASTM  062574
SWAPAT  002574          TST3    010664          $BANK   002011          $ETABL  062506          $PATMA  002010
SWR     002576          TST4    011652          $BASE   062542          $ETEND  062566          $PER01  053654
SWREG = 000176          TST5    014150          $BELL   002613          $EXHAL  045120          $PER02  053702
SW0   = 000001          TST6    014222          $CACHF  040232          $E$   = 000001          $PER03  053730
SW1   = 000002          TYPDS = 104405          $CACHN  040206          $FATAL  062470          $PER04  053760
SW10  = 002000          TYPEIT= 104401          $CBCSR  040672          $FILLC  002612          $PER07  054042
SW11  = 004000          TYPOC = 104402          $CB1CS  040714          $FILLS  002327          $PER10  054064
SW12  = 010000          TYPOS = 104403          $CDW1   062546          $F$   = 000000          $PER11  054114
SW13  = 020000          TYPS0 = 000000          $CDW2   062550          $GTSWR  060550          $PER12  054134
SW14  = 040000          TYPS1 = 000002          $CHARC  053632          $HALT   056110          $PER13  054156
SW15  = 100000          TYPS2 = 000000          $CHKDI  041266          $HALT2  062642          $PER14  054176
SW2   = 000004          TYPS3 = 000000          $CHK1D  041302          $HIBTS  062566          $PER15  054220
SW3   = 000010          TYPS4 = 000000          $CKSWR  060374          $HIOCT  061766          $PER16  054242
SW4   = 000020          TYPS5 = 000000          $CLRCS  041244          $ILLUP  052476          $PER17  054262
SW5   = 000040          TYPS6 = 000002          $CLR1C  041256          $INVAL  041362          $PER20  054300
SW6   = 000100          T12A    033206          $CMTAG  002000          $ITEMB  002013          $PER21  054316
SW7   = 000200          T12B    033230          $CMTGE  002514          $I$   = 000001          $PER22  054336
SW8   = 000400          UDPAR0= 177660          $CNTLC  061556          $KERNE  040156          $PER23  054354
SW9   = 001000          UDPAR7= 177676          $CNTLG  061570          $KMAP   042374          $PER24  054372
SYSSIZ  003712          UIPAR0= 177640          $CNTLK  060774          $K$   = 000061          $PER25  051130
TAG2$   011136          UIPAR1= 177642          $CNTLU  061563          $L    = 000066          $PER26  054562
TAG3$   011172          UIPAR2= 177644          $CPUOP  062514          $LF     002621          $PER27  054602
TAG4$   026520          UIPAR3= 177646          $CRLF   002620          $LL   = 000064          $PER30  051356
TAG70$  056602          UIPAR4= 177650          $DBLK   060364          $LOADC  040250          $PER31  054772
TAG71$  056612          UIPAR5= 177652          $DB20   062346          $LPADR  002562          $PER32  055070
TAG72$  056622          UIPAR6= 177654          $DDW0   062552          $LPERR  002564          $PER33  055136
TAG73$  056672          UIPDR0= 177600          $DDW1   062554          $L$   = 000000          $PER34  055216
TAG74$  056732          UNITOP  002366          $DDW2   062556          $MADR1  062520          $PER35  055250
```

```
$PER36  055304          $SAVRE  062156          $TN    = 000007         $TYPEC  053446          $WASDB  041100
$PWRDN  051524          $SAVR6  052502          $TPB     002610         $TYPEX  053634          $WASSB  040734
$PWRUP  052100          $SCOPE  055334          $TPFLG   002330         $TYPOC  057746          $WAS1D  041214
$QUES   002617          $STN   = 000001         $TPS     002606         $TYPON  057762          $WAS1S  041050
$R     = 177777         $SVLAD  055542          $TRAP    062602         $TYPOS  057722          $XTSTR  055434
$RAND   062252          $SV$   = 000000         $TRAP2   062624         $T1    = 000000         $Y$    = 000000
$RDCHR  061114          $SWR   = 163000         $TRPAD   062644         $T2    = 000455         $ZAP42  014410
$RDDEC  061770          $SWREG  062510          $TSIM    062572         $UNIT   062500          $Z$    = 000000
$RDLIN  061234          $T     = 000456         $TSTRD   040364         $UNITM  062576          $$S    = 000000
$RDOCT  061620          $TESTN  062472          $TTYIN   061532         $USWR   062512          $$T    = 000441
$READC  040344          $TKB    002604          $TYPDS   060150         $VECT1  062536          $$TT   = 000447
$RESRE  062214          $TKS    002602          $TYPE    053320         $VECT2  062540          $OFILL  060145

. ABS.  075264     000
        000000     001
ERRORS DETECTED:  0


VIRTUAL MEMORY USED:  26074 WORDS  ( 102 PAGES)
DYNAMIC MEMORY:  20346 WORDS  ( 78 PAGES)
ELAPSED TIME:  01:11:02
MC8E,MC8E/-SP=CZMSDB.SML,MC8E.P11
```