MS11-L/M MS11-L/M DIAGNOSTIC
CZMSDDO

AH-F295D-MC
FICHE 01 OF 02

JAN 1984
digital
Made In USA

MS11-L/M MS11-L/M DIAGNOSTIC
CZMSDDO

AH-FZ95D-MC
FICHE 02 OF 02

JAN 1984
digital
Made In USA

```
                        .TITLE  CZMSDO0 MS11 L/M DIAGNOSTIC
                        .ENABLE LC
                        .REM


                                        IDENTIFICATION


                        PRODUCT CODE:       AC F294D MC

                        PRODUCT NAME:       CZMSDO0 MS11 L/M Diag

                        PRODUCT DATE:       January, 1982

                        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)  1979,1982 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	Changes to Sizing Routine
CZMSDC	01-APR-81	Michael D Bibeault	1) Compatible with 11/24 2) Sizng routine will accept all legal memory configurations 3) All Field Service Commands operative
CZMSDD	14-DEC 81	Michael D Bibeault	Changes made to Field Service Command 5

```
                        OPERATIONAL SWITCH SETTINGS
                        SWITCH REGISTER DEFINITIONS
                        *
                        *       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 (UNRELOCATED & 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
```

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 Diagnost cs (where appl'cable)

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 030, and 027 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 first 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 024 and 033 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.

509 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	MXCSRT
Words 2 3	MXPAT
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 'f BIT7 is set n 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's one).

6. Halt if under APT or ACT branch sel.

Page 1

SW7 • DETAILED ERROR REPORTS
(200)

After any normal error report is typed this opt on 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 recurs on 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 setting 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
```

802

Page 18

```
804
805
806
807
808
809                     ;RELOCATE & TEST PROGRAM SPACE (BANK 0  & 1)
810
811                     PATTERN  1.     BANK 0
812                     PATTERN  2.     BANK 0
813                     PATTERN  3.     BANK 0
814                     PATTERN  4.     BANK 0
815                     PATTERN  5.     BANK 0
816                     PATTERN 26.     BANK 0
817                     PATTERN  1.     BANK 1
818                     PATTERN  2.     BANK 1
819                     PATTERN  3.     BANK 1
820                     PATTERN  4.     BANK 1
821                     PATTERN  5.     BANK 1
822                     PATTERN 26.     BANK 1
823
824
825                                     NOTE
826
827                             This is an example & not an actual
828                             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 relocations.

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 relocations.

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 relocations, then presumed good banks are tested from 0 to 200 except those requiring relocations.

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 relocations, then presumed good banks are tested from 0 to 200 except those requiring relocations.

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 relocations.

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 'CACHE' 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 no. the program.
2. If Switch 8 is not set, unrelocate if program was relocated.
3. Flush out any DRE'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 happen ng)

This command will print out the information encoded n the d sp.y, register. This is mainly intended for CPU s thout a hardware display register.

Example:

RELOCATED BANK= 23 PAT= 26

By use of Field Service Command 17 "Trace" can be set so that t m ll automatically type out the bank and pattern numbers as each pattern s 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" (Quint nue)

This command will continue typing that has been stopped by Control "S". If there has been no Control "S" typed then th s command s 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 sect on 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 Configurat on 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 s 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 console 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	0
022132	35	060002	03500002	000000	000100	000100	0	M	1	0
022132	35	060006	03500006	000000	000100	000100	0	M	1	0

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.
CSR NO	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       0
END PASS        0       1
END PASS        0V      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.

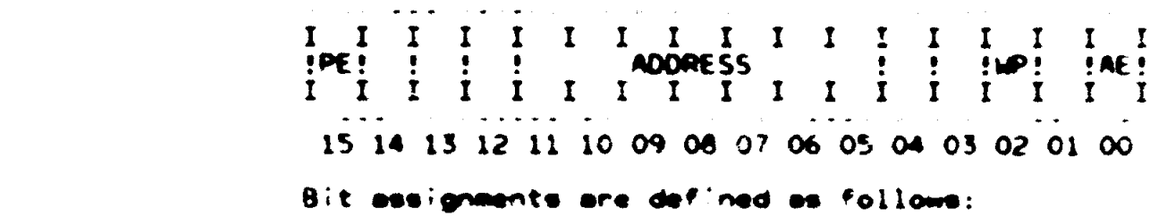
5.0 CSR INFORMATION TABLES

The following is a p cture v ew of the current control status registers which can be tested by this program. It shows b t assignments and definitions to provide a handy reference, and shows the similarities and differences between each one.

NOTE

All unused b ts in each CSR are equal to zero.

5.1 CORE/MOS PARITY REGISTER



Bit assignments are defined as follows:

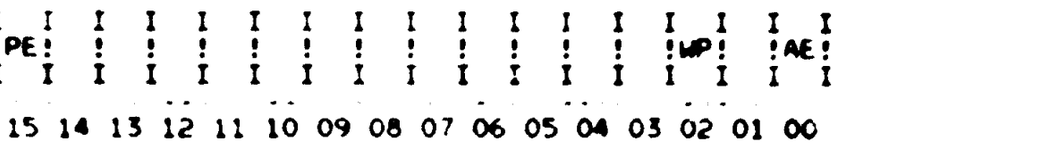
BIT15 PARITY ERROR

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)



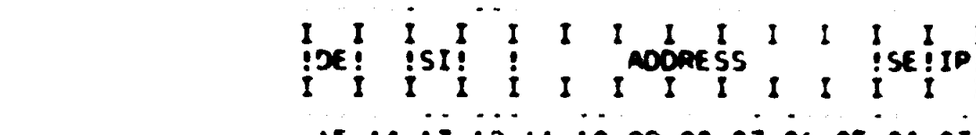
Bit assignments are defined as follows:

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



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 b'ts read will be stored 'n 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

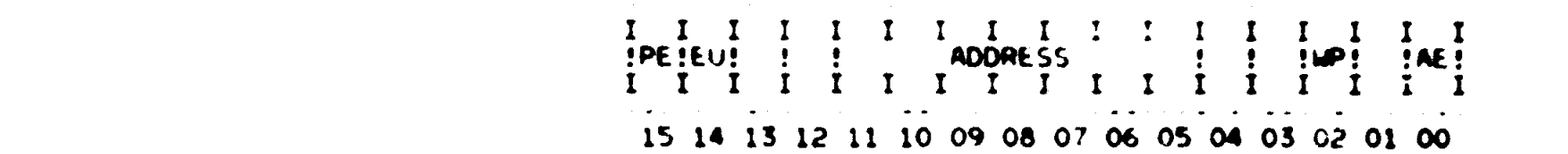
BIT03 INHIBIT MODE POINTER The Inhibit Mode Pointer works in conjunction with the Set Inhibit Mode bit. When BIT13 is set to a 1, a 16K portion of memory is inhibited from operating in the ECC Disable mode or Diagnostic Check mode. the Inhibit Mode Pointer indicates which 16K is being inhibited; e.g.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



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).

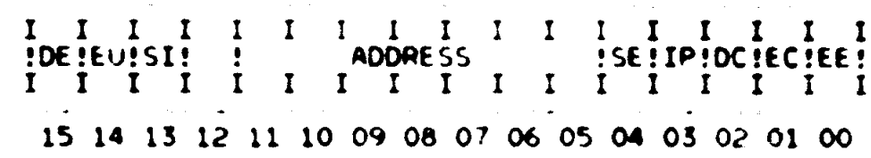
BIT02 WRITE WRONG PARITY Normal parity (odd) when clear; other parity (even) when set.

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

1979

when set

5.5 MS11 M CSR



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

```
2038
2039
2040
```

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.

Page 47

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

BITOO 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 location "FSPAT" is a table of patterns that can be run in Field Service 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.
;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
;                                                        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 's to force a tre parity b'ts to become involved.

It can execute out of the USER Data PDR's, the User Instruct'on 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 's 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.

```
2677
2678                    This checks the memory for the capab'l t, of hold'ng 0 s & 1 s.
```

6.2.2.1' Pattern 20 March ng 0 s & 1 s In Check B'ts Test

1. Write Double Words of 000000,,000000 wh ch causes check b'ts to equal 077 while addressing 'ncrements.
   (Write Up/077 > check b'ts)

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

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

4. Read Double Words & check while writing Zeros wh'le 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 wr'ting Zeros wh'le addressing decrements.
   (Down/100 -> 077)

7. Read Double Words & check while Address'ng 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(TM) 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

```
2910
2911                                        NOTE
2912
2913                                This test is run for ECC memory only.
2914
2915
2916
2917
2918
2919
2920                    6.2.2.23  Pattern 26  Random Data Test
2921
2922                    Write Random Data in a 16K Bank while incrementing the Addresses.
2923
2924                    Read check Random Data.
2925
2926                    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 n 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).

Page 6-

6.2.2.26 Pattern 31 SOB-A-LONG Test

Rationalization

In order to concentrate the memory cycles of a test into a part cular address, we must cut the overhead cycles to a min'mum. Frequently, the instruction itself may provide adequate data or set up a background in which any complemented bit may find 't hard to survive

The SOB instruction is the only PDP-11 'nstruction 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", "$DDM2-5". 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 pass 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 req. patterns when the APT 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).

Page

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
    01234567012345670123456701234567012345670123456701234567012345670123
ERRORS              XX
CPU MAP  11111111111111111111111111111111
INTRLV   ----------------33333333333333333
MEMTYPE  LLLLLLLLMMMMMMMMMMMMMMMMMMMMMMMM
CSP      00000000111111112222222222222222
PROTECT  PP       I         I         P
              0         1         2         3         4         5         6
    45670123456701234567012345670123456701234567012345670123456701234567
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

```
3351                                    Memory  Type  M  (MS11 M);  while Banks 40 167 do not exist.
3352                                    Memory Type K would indicate MF11S K, Memory  Type  P  would
3353                                    indicate  UNIBUS  Parity,  and  Memory Type B would indicate
3354                                    11/45 type Bipolar memory.
3355
3356                                CSR:
3357
3358                                    Banks 0 7 are assigned to CSR 172100, 10 17 to  CSR  172102,
3359                                    and 20-37 to interleaved CSR's 172104 and 172106.
3360
3361                                PROTECT:
3362
3363                                    Banks 0 and 1 are protected because they are program  space.
3364                                    Bank  0  and 1 can also be protected because they are in the
3365                                    bottom 16K of an MS11-M CSR.  The protection is hierarchical
3366                                    and  program  space  overshadows MS11-M protection.  Banks 0
3367                                    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 rules, conditions, etc., that govern MACRO-11 also govern SUPER-MAC. Spacing on a source line 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 blocks. 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 legal in SUPER MAC.

```
                        7.4.1  Sample Source File
                               .ENABL ABS
                               .ENABL AMA
                               .MCALL .SUPER
                               .SUPER
                               ;LST$$ =0
                               BITS_=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  BITS SET.IN A
                          MOV   B,C
                        END ;OF IF  BITS
                        IF  BITS OFF.IN A
                          MOV   C,D
                        END ;OF IF  BITS
                    ;ON.ERROR IS LIKE AN IF STATEMENT ON THE C-BIT
                    ;ON.ERROR EXAMPLES
                        ON.ERROR
                          MOV   A,B
                        ELSE
                          MOV   C,B
                        END ;OF ON.ERROR
                        ON.NOERROR
```

```
3486
3487
3488
3489                                    MOV   C,B
3490                                  ELSE
3491                                    MOV   A,B
3492                                  END ;OF ON.NOERROR
3493                                  ON.ERROR THEN LET A :B_* B
3494                          ;FOR EXAMPLES
3495                                  FOR I : *  -5 TO  23
3496                                    INC   A
3497                                  END ;OF FOR I
3498                                  FOR RO :_*  0 TO  140 BY  4
3499                                    DEC   A(RO)
3500                                  END ;OF FOR RO
3501                                  FOR I :_*  133 DOWNTO  3 BY  2
3502                                    ADD   A,B
3503                                  END ;OF FOR I
3504                          ;BEGIN EXAMPLES
3505                                  BEGIN ALPHA
3506                                    FOR RO :_*  0 TO  167
3507                                      MOVB        A(RO),B
3508                                      IF B LT  0 THEN LEAVE ALPHA
3509                                    END ;OF FOR RO
3510                                    FOR RO :_*  400 TO  567
3511                                      IF B GE  0 THEN LEAVE ALPHA
3512                                    END ;OF FOR RO
3513                                  END ALPHA
3514                          ;$RETURN EXAMPLES
3515                                  $RETURN
3516                                  $RETURN ERROR
3517                                  $RETURN NOERROR
3518                          ;CASE EXAMPLES
3519                                  MOV     A,RO
3520                                  CASE RO
3521                                    A
3522                                    B
3523                                    C
3524                                    D
3525                                    E
3526                                    F
3527                                  END ;OF CASE RO
3528
3529                                  .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                       BITS =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 16:41  PAGE 3

18                                          ;LET EXAMPLES
19 000024                                           LET R0 : = A
20 000030                                           LET B : = C + D
21 000044                                           LET E : = F + 1
22 000056                                           LET G : = H + 2
23 000072                                           LET J : = J + 01
24 000100                                           LET A :B = B
25                                          ;IF EXAMPLES
26 000106                                           IF A IS TRUE
27 000114  012737  000023  000006                     MOV    23,D
28 000122                                           END ;OF IF A
29 000122                                           IF B IS FALSE
30 000130  012737  000034  000010                     MOV    34,E
31 000136                                           END ;OF IF B
32 000136                                           IF A EQ B THEN LET C : = D
33 000154                                           IF A LT B
34 000164  013737  000004  000006                     MOV   C,D
35 000172                                           ELSE
36 000174  013737  000010  000006                     MOV   E,D
37 000202                                           END ;OF IF A
38 000202                                           IF A EQ B AND C NE D
39 000222  013737  000012  000014                     MOV   F,G
40 000230                                           END ;OF IF A
41 000230                                           IF A EQ B OR C NE D
42 000250  013737  000012  000014                     MOV   F,G
43 000256                                           END ;OF IF A
44 000256                                           IFB A EQ B AND C EQ  1
45 000276  013737  000016  000022                     MOV   H,J
46 000304                                           ELSE
47 000306  013737  000010  000022                     MOV   E,J
48 000314                                           END ;OF IFB A
49 000314                                           IFB A EQ B ANDB C EQ  1
50 000334  013737  000016  000022                     MOV   H,J
51 000342                                           ELSE
52 000344  013737  000010  000022                     MOV   E,J
53 000352                                           END ;OF IFB A
54 000352                                           IF RESULT IS EQ
55 000354  013737  000000  000002                     MOV   A,B
56 000362                                           END ;OF IF RESULT
57 000362                                           IF  BITS SET.IN A
58 000372  013737  000002  000004                     MOV   B,C
59 000400                                           END ;OF IF  BITS
60 000400                                           IF  BITS OFF.IN A
61 000410  013737  000004  000006                     MOV   C,D
62 000416                                           END ;OF IF  BITS
63                                          ;ON.ERROR IS LIKE AN IF STATEMENT ON THE C BIT
64                                          ;ON.ERROR EXAMPLES
65 000416                                           ON.ERROR
66 000420  013737  000000  000002                     MOV   A,B
67 000426                                           ELSE
68 000430  013737  000004  000002                     MOV   C,B
```

```
3612                    69 000456                       END ;OF ON.ERROR
3613                    70 000436                       ON.NOERROR
3614                    71 000440  013737  000004  000002     MOV   C,B
3615                    72 000446                       ELSE
3616                    73 000450  013737  000000  000002     MOV   A,B
3617                    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
000474  005237  000000    78 000474                                         INC    A
                        79 000500                                       END ;OF FOR I
                        80 000514                                       FOR R0 :_  0 TO  140 BY  4
000516  005360  000000    81 000516                                         DEC    A(R0)
                        82 000522                                       END ;OF FOR R0
                        83 000534                                       FOR I :_  133 DOWNTO  3 BY  2
000542  063737  000000  000002  84 000542                                  ADD    A,B
                        85 000550                                       END ;OF FOR I
                        86                                          ;BEGIN EXAMPLES
                        87 000566                                       BEGIN ALPHA
                        88 000566                                         FOR R0 :_  0 TO  167
000570  116037  000000  000002  89 000570                                    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
000650  013700  000000   101 000650                                       MOV     A,R0
                       102 000654                                       CASE R0
000664  000000           103 000664                                         A
000666  000002           104 000666                                         B
000670  000004           105 000672                                         C
000672  000006           106 000674                                         D
000674  000010           107 000676                                         E
000676  000012           108 000700                                         F
                       109 000700                                       END ;OF CASE R0
                       110
        000001         111                                              .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                      BITS =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
 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
```

```
3744                    000230                                  15:
3745                 41 000230                                      IF A EQ B OR C NE D
3746                    000230  023737  000000  000002              CMP A,B
3747                    000236  001404                              BEQ L6
3748                    000240  023737  000004  000006              CMP C,D
3749                    000246  001403                              BEQ L7
```

```
                    .MAIN.  MACRO M1111  01 APR 79 16:10  PAGE 3 1

      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  BITS SET.IN A
      000362  032737  000040  000000                    BIT  BITS,A
      000370  001403                                    BEQ L15
   58 000372  013737  000002  000004                      MOV   B,C
   59 000400                                            END ;OF IF  BITS
      000400                                    L15:
   60 000400                                            IF  BITS OFF.IN A

      000400  032737  000040  000000                    BIT  BITS,A
      000406  001003                                    BNE L16
   61 000410  013737  000004  000006                      MOV   C,D

   62 000416                                            END ;OF IF  BITS
      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
```

```
3808                            66 000420  013737  000000  000002          MOV   A,B
3809                            67 000426                                 ELSE
3810                               000426  000403                         BR L20
3811                               000430                          L17:
3812
3813                            68 000430  013737  000004  000002          MOV   C,B
3814
3815                            69 000436                                 END ;OF ON.ERROR
3816                               000436                          L20:
3817                            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                          E0:
             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
```

```
3876                                    000604  005200                  INC R0
3877                                    000606  020027  000167          CMP R0, 167
3878                                    000612  003766                  BLE B4
3879                                    000614                  E4:
3880                                 92 000614                    FOR R0 := 400 TO 567
3881                                    000614  012700  000400          MOV  400,R0
```

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

```
                    000620
                 93 000620                          B5:
                    000620  005737  000002                  IF B GE  0 THEN LEAVE ALPHA
                                                            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

```
3976 000000                             .ENABL  ABS
3977                                    .ENABL  AMA
3978                                    .DSABL  GBL
3979                                    ;NOTE:  CZMSDO.SML IS THE SUPER.MAC SOURCE AND IS RELEASED WITH
3980                                    ;THIS PROGRAM.  ALL THESE .MCALL STATEMENTS REFERENCE THAT FILE.
3981                                    .MCALL  SMACIT,..PUSH,..POP,..TAG,..BRAN,.EMIT,.EMITN,.EMITL,.EMITR
3982                                    .MCALL  .IFOPR,.IS,.GENBR,.OPADD,.OPSUB,CLEAR,SET,CLEARB,SETB
3983                                    .MCALL  RNE,REQ,RLT,RGE,RGT,RLE,RPL,RMI,RHI,RLOS,RHIS,RLO,RCS,RCC
3984                                    .MCALL  IF,..OR,..IFARI,..LEAVE,..GOTO,OR,AND,THEN,ELSE,WHILE,CASE
3985                                    .MCALL  FOR,TO,DOWNTO,REPEAT,UNTIL,THRU,END,BEGIN
3986                                    .MCALL  $$END,LEAVE,JUMPTO,GOTO,PUSH,POP,LET
3987                                    .MCALL  .SIMPLE,..ARITH,ORB,ANDB,IFB,UNTILB,WHILEB,ON.ERROR,ON.NOERROR
3988                                    .MCALL  $CALL,$RETURN
3989
3990                                    .NLIST  TTM                     ;I WANT FAT PAPER!
3991                                    .LIST   MC,SYM                  ;LIST MACRO CALLS, SYMBOL TABLE
3992                                    .NLIST  MD,CND,ME               ;DON'T LIST MACRO DEFS & CONDITIONALS & EXPANSIONS
3993                                    ;LST$$= 0                       ;DEFINED TO LIST SUPERMAC EXPANSIONS
3994          163000                    $SWR=   163000                  ;USE THESE SYSMAC SWITCHES
3995          000001                    $TN=    1                       ;FIRST TEST NUMBER TO ONE(1)
3996 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
```

DEFINE TRAPS

```
4056
4057          104425          LOADCSR=104425          ;LOAD CORRECT CSR
4058          104426          READCSR=104426          ;READ CORRECT CSR
4059
4060          104427          PERR01= 104427          ;PROGRAM DETECTED ERROR
4061          104430          PERR02= 104430          ;PROGRAM DETECTED ERROR
4062          104431          PERR03= 104431          ;PROGRAM DETECTED ERROR
4063          104432          PERR04= 104432          ;PROGRAM DETECTED ERROR
4064          104433          PERR07= 104433          ;PROGRAM DETECTED ERROR
4065          104434          PERR10= 104434          ;PROGRAM DETECTED ERROR
4066          104435          PERR11= 104435          ;PROGRAM DETECTED ERROR
4067          104436          PERR12= 104436          ;PROGRAM DETECTED ERROR
4068          104437          PERR13= 104437          ;PROGRAM DETECTED ERROR
4069          104440          PERR14= 104440          ;PROGRAM DETECTED ERROR
4070          104441          PERR15= 104441          ;PROGRAM DETECTED ERROR
4071          104442          PERR16= 104442          ;PROGRAM DETECTED ERROR
4072          104443          PERR17= 104443          ;PROGRAM DETECTED ERROR
4073          104444          PERR20= 104444          ;PROGRAM DETECTED ERROR
4074          104445          PERR21= 104445          ;PROGRAM DETECTED ERROR
4075          104446          PERR22= 104446          ;PROGRAM DETECTED ERROR
4076          104447          PERR23= 104447          ;PROGRAM DETECTED ERROR
4077          104450          PERR24= 104450          ;PROGRAM DETECTED ERROR
4078          104451          PERR25= 104451          ;PROGRAM DETECTED ERROR
4079          104452          PERR26= 104452          ;PROGRAM DETECTED ERROR
4080          104453          PERR27= 104453          ;PROGRAM DETECTED ERROR
4081          104454          PERR30= 104454          ;PROGRAM DETECTED ERROR
4082          104455          PERR31= 104455          ;PROGRAM DETECTED ERROR
4083          104456          PERR32= 104456          ;PROGRAM DETECTED ERROR
4084          104457          PERR33= 104457          ;PROGRAM DETECTED ERROR
4085          104460          PERR34= 104460          ;PROGRAM DETECTED ERROR
4086          104461          PERR35= 104461          ;PROGRAM DETECTED ERROR
4087          104462          PERR36= 104462          ;PROGRAM DETECTED ERROR
4088          104463          PERR37= 104463          ;PROGRAM DETECTED ERROR
4089          104464          PERR40= 104464          ;PROGRAM DETECTED ERROR
4090          104465          PERR41= 104465          ;PROGRAM DETECTED ERROR
4091          104466          PERR42= 104466          ;PROGRAM DETECTED ERROR
4092          104467          PERR43= 104467          ;PROGRAM DETECTED ERROR
4093
4094          104470          ECCDIS= 104470          ;DISABLE ECC ON ALL CSR'S
4095          104471          ECC1DIS=104471          ;DISABLE ECC ON 1 SELECTED CSR
4096          104472          ECCINIT=104472          ;INITIALIZE ALL ECC CSR'S
4097          104473          ECC1INIT=104473         ;INITIALIZE 1 SELECTED ECC CSR
4098          104474          CBCSR=  104474          ;WRITE GENERATED CHECKBITS IN ALL CSR'S
4099          104475          CB1CSR= 104475          ;WRITE GENERATED CHECKBITS IN 1 SELECTED CSR
4100          104476          WASSBE= 104476          ;WAS THERE A SBE ON ANY CSR?
4101          104477          WAS1SBE=104477          ;WAS THERE A SBE ON 1 SELECTED CSR?
4102          104500          WASDBE= 104500          ;WAS THERE A DBE ON ANY CSR?
4103          104501          WAS1DBE=104501          ;WAS THERE A DBE ON 1 SELECTED CSR?
4104          104502          CLRCSR= 104502          ;CLEAR ALL CSR'S
4105          104503          CLR1CSR=104503          ;CLEAR 1 SELECTED CSR
4106          104504          CHKDIS= 104504          ;DISABLE ECC & WRITE CHECKBITS FROM ALL CSR'S
4107          104505          CHK1DIS=104505          ;DISABLE ECC & WRITE CHECKBITS FROM 1 SELECTED CSR
4108          104506          ENASBE= 104506          ;ENABLE TRAPS ON SBE'S FROM ALL CSR'S
4109          104507          ENA1SBE=104507          ;ENABLE TRAPS ON SBE'S FROM 1 SELECTED CSR
4110          104510          TSTREAD=104510          ;TEST LOC (R1) & TST FOR SBE (WITHOUT FETCHES)
4111          104511          INVALID=104511          ;INVALIDATE BACKGROUND PATTERN ON "BANK"
4112          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
```

```
4285                                    ;*SUPERVISOR  D" PAGE ADDRESS REGISTERS
4286            172260          SDPAR0* 172260
4287                            ;SDPAR1*        172262
4288                            ;SDPAR2*        172264
4289                            ;SDPAR3*        172266
4290                            ;SDPAR4*        172270
4291            172272          SDPAR5* 172272
4292            172274          SDPAR6* 172274
4293            172276          SDPAR7* 172276
4294
4295                                    ;*KERNEL "I" PAGE DESCRIPTOR REGISTERS
4296            172300          KIPDR0* 172300
4297                            ;KIPDR1*        172302
4298                            ;KIPDR2*        172304
4299                            ;KIPDR3*        172306
4300                            ;KIPDR4*        172310
4301                            ;KIPDR5*        172312
4302                            ;KIPDR6*        172314
4303                            ;KIPDR7*        172316
4304
4305                                    ;*KERNEL "D" PAGE DESCRIPTOR REGISTERS
4306                            ;KDPDR0*        172320
4307                            ;KDPDR1*        172322
4308                            ;KDPDR2*        172324
4309                            ;KDPDR3*        172326
4310                            ;KDPDR4*        172330
4311                            ;KDPDR5*        172332
4312                            ;KDPDR6*        172334
4313                            ;KDPDR7*        172336
4314
4315                                    ;*KERNEL "I" PAGE ADDRESS REGISTERS
4316            172340          KIPAR0* 172340
4317                            ;KIPAR1*        172342
4318                            ;KIPAR2*        172344
4319                            ;KIPAR3*        172346
4320            172350          KIPAR4* 172350
4321            172352          KIPAR5* 172352
4322            172354          KIPAR6* 172354
4323                            ;KIPAR7*        172356
4324
4325                                    ;*KERNEL "D" PAGE ADDRESS REGISTERS
4326            172360          KDPAR0* 172360
4327                            ;KDPAR1*        172362
4328                            ;KDPAR2*        172364
4329                            ;KDPAR3*        172366
4330                            ;KDPAR4*        172370
4331                            ;KDPAR5*        172372
4332            172374          KDPAR6* 172374
4333            172376          KDPAR7* 172376
4334
```

```
                                .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
                                    .ENCM   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
```

```
4851                                    .SBTTL  MACRO   TESTAREA
4852                                ;***************** TESTAREA **********************
4853                                ;
4854                                ; THIS MACRO SWITCHES TO THE SPECIFIED TEST MODE.
4855                                ;
4856                                ;ARGUEMENTS: NONE.
4857                                ;
4858                                ;************************************************
4859
4860                                        .MACRO  TESTAREA
4861                                        .NLIST
4862                                        .DSABL  CRF
4863                                        .IIF DF LST$$ .LIST  ME
4864                                        .ENABL  CRF
4865                                        .LIST
4866                                        BIS     TESTMODE,PSW                    ;GO TO SYSTEM TEST MODE
4867                                        .DSABL  CRF
4868                                        .IIF DF LST$$ .NLIST  ME
4869                                        .ENABL  CRF
4870                                        .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 DUM
                ;*************************************************

                        .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
```

```
4918                                    .SBTTL  MACRO   DLEFT
4919                                ;****************** DLEFT ************************
4920                                ;
4921                                ; THIS MACRO DOES A DOUBLE WORD LEFT SHIFT
4922                                ;
4923                                ;ARGUEMENTS:    LOC     ;THE LOCATION TO BE SHIFTED LEFT (CARRY TO LOC+2)
4924                                ;
4925                                ;************************************************
4926
4927                                        .MACRO  DLEFT   ARG
4928                                        .NLIST
4929                                        .DSABL  CRF
4930                                        .IIF DF LST$$ .LIST  ME
4931                                        .ENABL  CRF
4932                                        .LIST
4933                                        ROL     ARG
4934                                        ROL     ARG+2
4935                                        .DSABL  CRF
4936                                        .IIF DF LST$$ .NLIST  ME
4937                                        .ENABL  CRF
4938                                        .ENDM   DLEFT
4939                                        .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  014472                  $ENDAD                          ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
                000052                  .=52
000052  000000                  .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  063176                  $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  002570          START1: CLR     RESTART
000304  000137  003632                  JMP     START
000310                          START2: SET     RESTART
000316  000137  003632                  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
```

VARIABLES INITIALIZED TO ZERO

```
5051 002146  000000                 CSRNO:  0                          ;CSR ADDRESS NUMBER (4 LSB'S)
5052 002150  000000                 SavCsr: .Word   0                  ;Location to Save CsrNo during FS Command 5 ;D
5053 002152  000000                 OLDCSR: .WORD   0                  ;OLD CSR NUMBER(USED IN INH PTR TEST)
5054                                ;THESE LOCATIONS STORE GPR'S DURING SUPERVISOR TESTS
5055 002154  000000                 SUPDR0: 0
5056 002156  000000                 SUPDR1: 0
5057 002160  000000                 SUPDR2: 0
5058 002162  000000                 SUPDR3: 0
5059 002164  000000                 SUPDR4: 0
5060 002166  000000                 SUPDR5: 0
5061 002170  000000                 SUPDR6: 0
5062 002172  000000                 DUMMY:  0                          ;DUMMY LOCATION FOR ADDRESS PASSING
5063                                ;THESE LOCATIONS STORE GPR'S & PSW DURING DETAILED ERROR PRINTOUTS
5064 002174  000000                 DETR0:  0
5065 002176  000000                 DETR1:  0
5066 002200  000000                 DETR2:  0
5067 002202  000000                 DETR3:  0
5068 002204  000000                 DETR4:  0
5069 002206  000000                 DETR5:  0
5070 002210  000000                 DETSP:  0
5071 002212  000000                 DETPSW: 0
5072 002214  000000                 DETFLAG:0                          ;DETAILED REPORT FLAG
5073 002216  000000                 CONTFLAG:0                         ;CSR'S HAVE BEEN TESTED FLAG
5074 002220  000000                 TOTCSRS:.WORD   0                  ;1 BIT PER EXISTING CSR, EG-
5075                                                                   ;CSR 0 REPRESENTED BY BIT 15, ETC.
5076 002222  000000                 CSRFIRST:.WORD  0                  ;FIRST ADDRESS UNDER CONTROL OF THIS CSR
5077 002224  000000                 CSRLAST:  .WORD  0                 ;
5078 002226  000000                 CSRFBANK:.WORD  0                  ;
5079 002230  000000                 CSRLBANK:.WORD  0                  ;
5080 002232  000000                 CSRINT:   .WORD  0                 ;
5081 002234  000000                 SPLTCSR:  .WORD  0                 ;
5082 002236  000000  000000         DATBUF:  .WORD   0,0               ;TWO WORD DATA BUFFER
5083 002242  000000  000000         TSTDAT:  .WORD   0,0               ;TWO WORD TEST DATA
5084 002246  000000  000000         SBEMSK:  .WORD   0,0               ;TWO WORD SINGLE BIT ERROR MASK
5085 002252  000000  000000         DBEMSK:  .WORD   0,0               ;TWO WORD DOUBLE BIT ERROR MASK
5086 002256  000000                 SUPDOADD:.WORD  0                  ;ADDRESS OF SUBROUTINE TO EXECUTE IN SUPERVISOR MODE
5087 002260     000                 PASFLG:  .BYTE  0                  ;LOCAL LOOP PASS CONTROL
5088 002261     000                 UPPFLG:  .BYTE  0                  ;LOCAL LOOP PASS CONTROL
5089 002262  000000                 REALPAT:.WORD   0                  ;REAL PATTERN UNDER TEST
5090 002264  000000                 OLDCACHE:.WORD  0                  ;BACKED UP VALUE OF CACHE CONTROL REGISTER
5091 002266  000000                 PARTHERE:.WORD  0                  ;PARITY TRAPS SOMETIMES GO TO ADDRESS STORED HERE
5092 002270  000000                 FSSTACK:.WORD   0                  ;STACK SAVED HERE IF IN FIELD SERVICE MODE
5093 002272  000000                 NEWBANK:.WORD   0                  ;USED FOR RELOCATION TO A NEW BANK
5094 002274  000000                 SOURCE:  .WORD  0                  ;SOURCE OF DATA WORDS FOR CHECKBIT GENERATION SUBROUTINE
5095 002276  000000                 CHECK:   .WORD  0                  ;CHECKBITS TO BE LOADED INTO CSR
5096 002300  000000                 PCBUMP:  .WORD  0                  ;VALUE TO BUMP THE PC BY TO RECOVER AFTER A PARITY TRAP
5097 002302  000000                 CSRINC:  .WORD  0                  ;VALUE TO INCREMENT ADDRESS BY TO REMAIN IN THE SAME CSR
5098 002304  000000                 CSRLOOP:.WORD   0                  ;LOOP CONTROL FOR CSR TESTING
5099 002306  000000                 SUCCESS:.WORD   0                  ;FLAG SET BY SUCCESSFULL TASK OR SUBROUTINE
5100 002310  000000                 ZEROS:   .WORD  0                  ;FOR AID IN "MOV" INSTRUCTIONS
5101 002312  000000                 TIME:    .WORD  0                  ;SECONDS THAT BATTERIES SHOULD LAST
5102 002314  000000                 SKIPMK:  .WORD  0                  ;FLAG TO SKIP MKCONTROL SUBROUTINE
5103 002316  000000                 NULLFLAG:.WORD  0                  ;SET WHEN RUNNING NULL PATTERNS
5104 002320  000000                 QVFLAG: 0                          ;FLAGS QUICK VERIFY PASS UNDER APT, ACT, OR XXDP CHAIN MODE
5105 002322  000000                 ACTFLAG:0                          ;FLAGS ACT AUTOMATIC MODE PROGRAMMING RULES
5106 002324  000000                 APTFLAG:0                          ;FLAGS APT AUTOMATIC MODE PROGRAMMING RULES
5107 002326  000000                 XXDPCHAIN:0                        ;FLAGS XXDP CHAIN MODE PROGRAMMING RULES
```

VARIABLES INITIALIZED TO ZERO

```
                                        ;NOTE: THESE TWO BYTES MUST STAY TOGETHER
002330  000                             $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER FOR FILLS
002331  000                             $FILLS: .BYTE   0               ;CONTAINS # OF FILL CHARACTERS
002332  000                             $TPFLG: .BYTE   0               ;;"TERMINAL NOT AVAILABLE  FLAG
                                        .EVEN
002334  000000                          $ESCAPE:0                       ;;ESCAPE ON ERROR ADDRESS
002336  000000                          EVEN:   0                       ;USED FOR ALTERNATE DATA PATTERNS
002340  000000                          STRIPES:0                       ;COUNTS DIAGONAL STRIPES
002342  000000                          COUNT:  0                       ;BACKED UP COPY OF STRIPES
002344  000000                          NOTAB:  0                       ;NO TABLE BEING PRINTED   NOW
002346  000000                          BSIZE:  0                       ;SIZE OF 11/45 MOS MEMORY IN K WORDS
002350  000000                          KSIZE:  0                       ;SIZE OF MF11S-K MEMORY IN K WORDS
002352  000000                          LSIZE:  0                       ;SIZE OF MS11-L MEMORY IN K WORDS
002354  000000                          MSIZE:  0                       ;SIZE OF MS11-M MEMORY IN K WORDS
002356  000000                          PSIZE:  0                       ;SIZE OF UNIBUS PARITY MEMORY IN K WORDS
002360  000000                          TOOMANY:0                       ;FLAGS WHEN TOO MANY ERRORS HAVE BEEN PRINTED FOR A BANK
002362  000000                          READONLY:0                      ;FLAG TO PATTERNS TO READ ONLY
002364  000000  000000                 TESTADD:0,0                     ;THE ADDRESS TO RUN CSR TESTS ON
002370  000000                          UNITOP: 0                       ;HIGHEST ACCESSABLE BANK OF MEMORY THRU UNIBUS MAP
002372  000000                          STOPOK: 0                       ;FLAG TO ALLOW STOPPING WITH SWITCH REGISTER
002374  000000                          APTPAR: .WORD   0               ;AMOUNT OF PARITY MEMORY ACCORDING TO APT
002376  000000                          APTECC: .WORD   0               ;AMOUNT OF ECC MEMORY ACCORDING TO APT
002400  000000                          NOFSMODE:0                      ;FLAG TO DISABLE FIELD SERVICE MODE
002402  000000                          NOERROR:0                       ;"THIS IS NOT AN ERROR" FLAG
002404  000000                          LOADBANK:0                      ;BANK LOADERS ARE RELOCATED TO
002406  000000                          TEMP:   0                       ;USED FOR JUNK
002410  000000                          QUICK:  0                       ;QUICK STOP FLAG FOR APT POWER FAIL
002412  000000                          NOSCOPE:0                       ;'NO SCOPE LOOP ALLOWED" FLAG
002414  000000                          FSINFLAG:0                      ;"FIELD SERVICE - NO INTERNAL INTERLEAVE" FLAG
002416  000000                          APTSIZE:0                       ;APT SIZING INFO FLAG
002420  000000                          FS7FLAG:0                       ;TRUE WHEN IN FIELD SERVICE COMMAND 7
002422  000000                          CONFGERROR:0                    ;CONFIGURATION ERROR FLAG
002424  000000                          I:      0                       ;USED FOR GENERAL PURPOSE INDEXING
002426  000000                          NO22BIT:0                       ;NO 22-BIT MODE FLAG
002430  000000                          NOSUPER:0                       ;NO SUPERVISOR MODE FLAG
002432  000000                          ERRADD: .WORD   0               ;HOLDS THE CSR'S ERROR ADDRESS
002434  000000  000000  000000  CSRINFO:0,0,0,0,0,0,0,0                 ;USED TO STORE INFORMATION ABOUT THE 16
002442  000000  000000  000000
002450  000000  000000
002454  000000  000000  000000          0,0,0,0,0,0,0,0                 ;POSSIBLE CSR'S
002462  000000  000000  000000
002470  000000  000000
002474  000000                          LINK1:  0                       ;USED TO HOLD LINKS TO PATTERNS WHICH
002476  000000                          LINK2:  0                       ;CAN EXECUTE IN THE PAR/PDR'S OR NOT
002500  000000                          CSRHOLD:0                       ;USED TO STORE CSR VALUES FOR CSR TESTS
002502  000000                          KFLAG:  0                       ;USED TO FLAG MF11S-K MEMORY TO TESTS
002504  000000  000000                 PGMCSR: .WORD   0,0             ;POINTS TO PROGRAM CSR
002510  000000                          INHECC: .WORD   0               ;FLAGS INHIBIT ECC TESTS ON RELOCATION
002512  000000                          INHBANK:.WORD   0               ;
002514  000000                          FULLREL:.WORD   0               ;
002516                                  $CMTGE: ;*END OF COMMON TAGS
```

```
                                        .SBTTL  VARIABLES       INITIALIZED TO NON ZERO
002516  000001  000000          CACHKN: 1,0                     ;CACHE CONSTANT (MOVED TO CONTRL TO TURN ON CACHE)
002522  001415                  CACHKF: 1415                    ;CACHE CONSTANT (MOVED TO CONTRL TO TURN OFF CACHE)
002524  040000                  TESTMODE:40000                  ;USED TO SELECT THE PROPER TEST MODE FOR A PATTERN RUN
002526  000012                  ERRMAX: 10.                     ;MAX # OF ERRORS PER BANK WITH SW11
002530  000167                  LASTBANK:167                    ;HIGHEST BANK OF MEMORY
002532  170000                  LASTBLOCK:170000                ;HIGHEST BANK OF MEMORY+1 (IN PAR FORMAT)
002534  000031                  SOBK:   25.                     ;SOB CONSTANT
002536  002000                  KSTACK: STACK                   ;STACK BEGINNING
002540  000001                  LOADHOME:1                      ;HOME BANK OF LOADERS
002542  177777                  WORST:  177777                  ;SET IF TESTING BANKS IN WORST FIRST MODE(1ST PASS)
002544  176543                  SEEDHI: 176543                  ;WORKING SEED HI (USED FOR RANDOM NUMBER GENERATOR)
002546  123456                  SEEDLO: 123456                  ;WORKING SEED LO (USED FOR RANDOM NUMBER GENERATOR)
002550  176543                  MSEEDH: 176543                  ;MASTER SEED HI (USED FOR RANDOM NUMBER GENERATOR)
002552  123456                  MSEEDL: 123456                  ;MASTER SEED LO (USED FOR RANDOM NUMBER GENERATOR)
002554  177777                  HEADER: 177777                  ;USED TO PRINT HEADINGS ONLY ONCE
002556  177777                  ONES:   177777                  ;FOR AID IN "MOV" INSTRUCTIONS
002560  000003                  FLIPLOC:3                       ;COUNTER FOR FLIPING DATA ON WORST CASE NOISE TEST
002562  052525                  SOFTPAT:52525                   ;PATTERN FOR SOFT ERROR BACKGROUND TESTS
002564  000000                  $LPADR: .WORD   0               ;;CONTAINS SCOPE LOOP ADDRESS
002566  000000                  $LPERR: .WORD   0               ;;CONTAINS SCOPE RETURN FOR ERRORS
002570  000000                  RESTART:0                       ;RESTART (START ADD 202) FLAG
002572  000000                  $ERTTL: .WORD   0               ;;CONTAINS TOTAL ERRORS

                                ;********************** NOTE THESE TWO LOCATIONS MUST STAY TOGETHER **********
002574  000377                  BAKPAT: .WORD   377             ;BACKGROUND PATTERN                            *
002576  177400                  SWAPAT: .WORD   177400          ;SWAPPED BAKPAT                                *
                                ;*************************************************************************

002600  177570                  SWR:    .WORD   DSWR            ;;ADDRESS OF SWITCH REGISTER
002602  177570                  DISPLAY: .WORD  DDISP           ;;ADDRESS OF DISPLAY REGISTER
002604  177560                  $TKS:   177560                  ;;TTY KBD STATUS
002606  177562                  $TKB:   177562                  ;;TTY KBD BUFFER
002610  177564                  $TPS:   177564                  ;;TTY PRINTER STATUS REG. ADDRESS
002612  177566                  $TPB:   177566                  ;;TTY PRINTER BUFFER REG. ADDRESS
002614     012                  $FILLC: .BYTE   12              ;;INSERT FILL CHARS. AFTER A "LINE FEED"
002615     207  377      377    $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
002620     000
002621     077                  $QUES:  .ASCII  /?/             ;;QUESTION MARK
002622     015                  $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
002623     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
002626  000201          CONFIG: .REPT   201
003632                  CONFIEND:
```

```
5271                                            .SBTTL  ********** MAIN **********
5272 003632                                START:  SUBTST  <<INITIALIZE VARIABLES TO ZERO>>
                                           ;*****************************************************************************
                                           ;*SUBTEST       INITIALIZE VARIABLES TO ZERO
                                           ;*****************************************************************************
5276 003632  105737  063116                        TSTB    $ENV
5277 003636  001001                                BNE     NORES
5278 003640  000005                                RESET
5279 003642  013706  002536                NORES:  MOV     KSTACK,SP       ;;SETUP THE STACK POINTER
5285 003646  012700  002000                        MOV     #$CMTAG,R0      ;;FIRST LOCATION TO BE CLEARED
5286 003652  005020                        1$:     CLR     (R0)+           ;;CLEAR MEMORY LOCATION
5287 003654  022700  002516                        CMP     #$CMTGE,R0      ;;DONE?
5288 003660  001374                                BNE     1$              ;LOOP BACK IF NO
5289 003662  012737  000167  002530                MOV     #167,LASTBANK   ;RESTORE LASTBANK (THIS MUST BE DONE PRIOR TO SYSTEM SIZING)
5290 003670                                        SUBTST  <<CLEAR NON-PROGRAM SPACE>>
                                           ;*****************************************************************************
                                           ;*SUBTEST       CLEAR NON-PROGRAM SPACE
                                           ;*****************************************************************************
5291                                               ;THIS ATTEMPS TO GET RID OF ANY PARITY ERRORS BY WRITING INTO
5292                                               ;EVERY LOCATION THAT IS NOT LOADED INTO BY THE PROGRAM OR ALLOCATED
5293                                               ;TO THE XXDP LOADERS
5294 003670  012737  000001  002074                MOV     #1,NOPAR                ;PARITY ACTION = COUNT & IGNORE
5295 003676  005000                                CLR     R0
5296 003700  000241                        2$:     CLC
5297 003702  005520                                ADC     (R0)+
5298 003704  020027  160000                        CMP     R0,#160000
5299 003710  103773                                BLO     2$
5300 003712  005037  002074                        CLR     NOPAR                   ;RESTORE DEFAULT PARITY ACTION
5301
```

```
5310 003716                                     SUBTST  <<TYPE OF SYSTEM SIZER>>
                                        ;*********************************************************************
                                        ;*SUBTEST       TYPE OF SYSTEM SIZER
                                        ;*********************************************************************
5311 003716  000401                             BR      SYSSIZ                  ;SKIP OVER VARIABLE LOCATION
5312 003720  000000                     PROTYP: .WORD   0
5313 003722                             SYSSIZ: SET4    #4$
5314 003730  005737  177746                        TST     CONTRL                  ;SEE IF CACHE REGISTER RESPONDS
5315 003734                                     SET4    #9$                     ;YES - DO WE HAVE 11/44 TYPE CACHE
5316 003742  005737  177750                        TST     MAINT                   ;OR 11/60 TYPE CACHE?
5317 003746  000411                             BR      5$                      ;BRANCH IF 11/44 TYPE CACHE
5318 003750  012737  000014  002522  9$:         MOV     #14,CACHKF              ;TURN OFF CONSTANT FOR 11/60 CACHE
5319 003756  000405                             BR      5$
5320 003760  005037  002516          4$:         CLR     CACHKN                  ;NO CACHE ON SYSTEM
5321 003764  012737  002310  064412             MOV     #ZEROS,DT14             ;DO NOT PRINT CONTRL ERROR MESSAGES
5322 003772                             5$:     SET4    #6$
5323 004000  005737  172516                        TST     MMR3                    ;DO WE HAVE AN MMR3?
5324 004004  005037  172516                        CLR     MMR3                    ;YES WE DO
5325 004010  052737  000020  172516             BIS     #BIT4,MMR3              ;SEE IF THERE IS 22-BIT MODE
5326 004016  032737  000020  172516             BIT     #BIT4,MMR3
5327 004024  001024                             BNE     10$                     ;BRANCH IF 22-BIT RELOCATION
5328 004026  000411                             BR      7$                      ;BRANCH IF MMR3 BUT NO 22-BIT RELOC.
5329                                    ;* 11/34 TYPE MACHINES ENTER HERE
5330 004030  012737  140000  002524  6$:         MOV     #140000,TESTMODE        ;MAKE TEST MODE USER
5331 004036  005237  002430                        INC     NOSUPER                 ;NO SUPERVISOR MODE
5332 004042  005037  064262                        CLR     DT5+10
5333 004046  005037  064422                        CLR     DT14+10
5334                                    ;* 11/45 TYPE MACHINES ENTER HERE
5335 004052  005237  002426             7$:         INC     NO22BIT                 ;NO 22 BIT MODE
5336 004056  012737  000007  002530             MOV     #7,LASTBANK             ;124K MEMORY MAX. MEMORY SIZE
5337 004064  005037  064264                        CLR     DT5+12                  ;DO NOT TRY TO PRINT ERROR REGISTER
5338 004070  005037  064424                        CLR     DT14+12                 ;ERROR MESSAGES, BECAUSE THERE IS
5339                                                                            ;IS NO ERROR REGISTER!
5340 004074  000417                             BR      8$
5341 004076                             10$:    SET4    #8$
5342 004104  000007                             MFPT                            ;TYPE OF PROCESSOR TEST: THIS INSTRUCTION
5343                                                                            ; (AVAILABLE ON NEWER PROCESSORS ONLY) PLACES
5344                                                                            ; A CODE IN THE LOWER BYTE OF R0 THAT
5345                                                                            ; INDICATES THE  PROCESSOR TYPE. 1=11/44
5346                                                                            ; 3=11/24
5347 004106  110037  003720                        MOVB    R0,PROTYP               ;MOV THE CODE TO PROTYP
5348 004112  022737  000003  003720             CMP     #3,PROTYP               ;IS THIS AN 11/24?
5349 004120  001005                             BNE     8$                      ;BRANCH IF NOT - WE HAVE AN 11/44
5350 004122  005237  002430                        INC     NOSUPER                 ;NO SUPERVISOR MODE
5351 004126  012737  140000  002524             MOV     #140000,TESTMODE        ;MAKE TEST MODE USER
5352
5353 004134                             8$:     SET4    #11$                    ;TRAPS GO TO 11$                    ;R C
5354 004142  005037  056560                        CLR     CPERRF                  ;CLEAR THE FLAG                     ;R C
5355 004146  005737  177766                        TST     @#177766                ;IS THERE A CPU ERROR REGISTER?     ;R C
5356 004152  012737  177777  056560             MOV     #-1,CPERRF              ;YES   TRAPPED                      ;R C
5357 004160                             11$:    RES4                                                                ;R C
```

```
5360 004202                                             SUBTST  <<INITIALIZE VARIABLES TO NON ZERO>>
                                                ;*****************************************************************
                                                ;*SUBTEST        INITIALIZE VARIABLES TO NON ZERO
                                                ;*****************************************************************
5361 004202                                             SET     WORST
5362 004210  012737  000003  002560                       MOV     #3,FLIPLOC
5363 004216                                             SET     HEADER
5364 004224  012737  176543  002550                       MOV     #176543,MSEEDH
5365 004232  012737  123456  002552                       MOV     #123456,MSEEDL
5366 004240  013737  002550  002544                       MOV     MSEEDH,SEEDHI   ;PRIME THE RANDOM NUMBER GENERATOR
5367 004246  013737  002552  002546                       MOV     MSEEDL,SEEDLO   ;BOTH HIGH AND LOW WORDS
5368 004254  012737  000377  002574                       MOV     #377,BAKPAT
5369 004262  012737  177400  002576                       MOV     #177400,SWAPAT
5374 004270                                             SUBTST  <<INITIALIZE VECTORS>>
                                                ;*****************************************************************
                                                ;*SUBTEST        INITIALIZE VECTORS
                                                ;*****************************************************************
5375 004270  012737  055444  000020                       MOV     #$SCOPE,IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
5376 004276  012737  000340  000022                       MOV     #340,IOTVEC+2   ;;LEVEL 7
5377 004304  012737  056000  000030                       MOV     #$ERROR,EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
5378 004312  012737  000340  000032                       MOV     #340,EMTVEC+2   ;;LEVEL 7
5379 004320  012737  063212  000034                       MOV     #$TRAP,TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
5380 004326  012737  000340  000036                       MOV     #340,TRAPVEC+2;LEVEL 7
5381 004334  012737  051634  000024                       MOV     #$PWRON,PWRVEC ;;POWER FAILURE VECTOR
5382 004342  012737  000340  000026                       MOV     #340,PWRVEC+2   ;;LEVEL 7
5383 004350  012737  037762  000114                       MOV     #PARITY,PARVEC;GET READY FOR PARITY ERRORS
5384 004356  012737  000340  000116                       MOV     #340,PARVEC+2
5385 004364  012737  040156  000010                       MOV     #PDP1105,RESVEC;RESERVED INSTRUCTION TRAP
5386 004372  012737  000340  000012                       MOV     #340,RESVEC+2
5387 004400  012737  040132  000004                       MOV     #TIMEOUT,ERRVEC;SETUP TIMEOUT ERRORS
5388 004406  012737  000340  000006                       MOV     #340,ERRVEC+2   ;SET PRIORITY OF ERROR TRAPS
5389 004414  012737  040144  000250                       MOV     #MMTRAP,MMVEC   ;VECTOR FOR MEMORY MANAGEMENT
5390 004422  012737  000340  000252                       MOV     #340,MMVEC+2
5395 004430  104423                                     CACHON                  ;TURN CACHE ON
```

```
5398 004432                            SUBTST  <<INITIALIZE PATTERNS>>
                                ;*******************************************************************************
                                ;*SUBTEST        INITIALIZE PATTERNS
                                ;*******************************************************************************
5399                                    ;THE APT E-TABLE DETERMINES WHICH PATTERNS ARE GOING TO BE RUN.
5400                                    ;EACH BIT SET REPRESENTS A PATTERN TABLE ENTRY THAT IS TO BE LEFT
5401                                    ;ALONE (TO BE RUN).  EACH BIT CLEARED REPRESENTS A PATTERN TABLE ENTRY
5402                                    ;THAT IS TO BE OVERLAYED WITH THE ADDRESS OF A NULL PATTERN.
5403 004432  012700  063162              MOV     #IDDWO,R0
5404 004436  012001                      MOV     (R0)+,R1
5405 004440  012703  017430              MOV     #MKCSRT,R3
5406 004444  012702  000020              MOV     #16.,R2
5407 004450  004737  004550              CALL    PATPLUG
5408 004454  012001                      MOV     (R0)+,R1
5409 004456  012702  000010              MOV     #8.,R2
5410 004462  004737  004550              CALL    PATPLUG
5411 004466  012001                      MOV     (R0)+,R1
5412 004470  012703  017660              MOV     #MKPAT,R3
5413 004474  012702  000020              MOV     #16.,R2
5414 004500  004737  004550              CALL    PATPLUG
5415 004504  012001                      MOV     (R0)+,R1
5416 004506  012702  000010              MOV     #8.,R2
5417 004512  004737  004550              CALL    PATPLUG
5418 004516  012001                      MOV     (R0)+,R1
5419 004520  012703  020044              MOV     #MJPAT,R3
5420 004524  012702  000020              MOV     #16.,R2
5421 004530  004737  004550              CALL    PATPLUG
5422 004534  012001                      MOV     (R0)+,R1
5423 004536  012702  000010              MOV     #8.,R2
5424 004542  004737  004550              CALL    PATPLUG
5425 004546  000417                      BR      SUBAAA
5426
5427 004550                     PATPLUG:SUBTST  <<SUBR  PLUG IN NULL PATTERNS>>
                                ;*******************************************************************************
                                ;*SUBTEST        SUBR    PLUG IN NULL PATTERNS
                                ;*******************************************************************************
5428 004550                             FOR I := #1 TO R2
5429 004556  006001                       ROR   R1
5430 004560                               ON.NOERROR              ;IF CARRY CLEAR
5431 004562  012713  026416                 MOV #MT0999,(R3)
5432 004566                               END ;OF ON.ERROR
5433 004566  062703  000002               ADD #2,R3
5434 004572                             END ;OF FOR
5435 004604  000207                     RETURN
```

```
5438 004606                                    SUBAAA: SUBTST  <<CLEAR THE CONFIGURATION TABLE>>
                                               ;*****************************************************************************
                                               ;*SUBTEST       CLEAR THE CONFIGURATION TABLE
                                               ;*****************************************************************************
5439                                                   ;THIS ZEROS (UNLESS WE STARTED AT ADDRESS 202) THE CONFIG TABLE
5440                                                   ;WHICH IS FULLY DISCRIBED AT LOCATION "CONFIG".
5441                                                   .ENABLE LSB
5442 004606                                           IF RESTART IS FALSE
5443 004614  012700  002626                             MOV    #CONFIG,R0
5444 004620  005020                              1$:     CLR    (R0)+
5445 004622  022700  003632                             CMP    #CONFIEND,R0
5446 004626  001374                                     BNE    1$
5447 004630                                           END ;OF IF RESTART
5448                                                   .DSABL  LSB
5449 004630  012737  000002  002104                    MOV     #BIT1,CPUBIT            ;SET ID BIT
5450 004636                                           SUBTST  <<SIZE FOR A HARDWARE SWITCH REGISTER>>
                                               ;*****************************************************************************
                                               ;*SUBTEST       SIZE FOR A HARDWARE SWITCH REGISTER
                                               ;*****************************************************************************
5451                                                   ;;IF NOT FOUND OR IT IS
5452                                                   ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
5453                                                   .ENABL  LSB
5454 004636                                           SET4    #3$                     ;TRAPS TO 4 GOTO 3$
5455 004644  012737  177570  002600                    MOV     #DSWR,SWR               ;;SETUP FOR A HARDWARE SWITCH REGISTER
5456 004652  012737  177570  002602                    MOV     #DDISP,DISPLAY          ;;AND A HARDWARE DISPLAY REGISTER
5457 004660                                           IF #-1 EQ @SWR                  ;IF NO TRAP FROM REFERENCE TO @SWR AND @SWR = # 1
5458 004670  000403                                     BR     2$                      ;;BRANCH IF NO TIMEOUT
5459 004672  012716  004700                      3$:    MOV    #2$,(SP)                ;;SET UP FOR TRAP RETURN
5460 004676  000002                                     RTI
5461 004700                                      2$:    RES4                           ;RESET TRAPS TO 4 TO DEFAULT
5462 004722  012737  000176  002600                     MOV    #SWREG,SWR              ;;POINT TO SOFTWARE SWR
5463 004730  012737  000174  002602                     MOV    #DISPREG,DISPLAY
5464 004736                                           END ;OF IF # 1
5465                                                   .DSABL  LSB
```

```
5468 004736                          SUBAAB: SUBTST  <<SETUP ACT, APT, & XXDP>>
                                     ;*****************************************************************************
                                     ;*SUBTEST       SETUP ACT, APT, & XXDP
                                     ;*****************************************************************************
5469                                         ;THIS SETS UP A BUNCH OF FLAGS TO TELL THE PROGRAM EVERYTHING
5470                                         ;IT CARES TO KNOW ABOUT APT, ACT, & XXDP.
5471 004736  005037  063104                 CLR     $PASS           ;CLEAR PASS COUNT
5472 004742                                  IFB #BIT5 SET.IN $ENVM
5473 004752                                    SET   $TPFLG          ;INDICATE NO TERMINAL
5474 004760                                  END ;OF IFB #BIT5
5475 004760                                  IFB #BIT7 SET.IN $ENVM
5476 004770                                    SET   APTSIZE
5477 004776                                  END ;OF IFB #BIT7
5478 004776                                  IFB $ENV EQ #1
5479 005006                                    SET   APTFLAG,QVFLAG,$AUTO,QUICK
5480 005036  012737  045252  000024            MOV   #APTDOWN,PWRVEC
5481 005044  012737  063120  002600            MOV   #$SWREG,SWR     ;USE APT SWR
5482 005052                                  ELSE
5483 005054                                    IF 42 NE #STACK AND 42 NE #0
5484 005072                                      SET QVFLAG,$AUTO
5485 005106                                      IF 42 EQ #$ENDAD
5486 005116                                        SET       ACTFLAG
5487 005124                                      ELSE
5488 005126                                        SET       XXDPCHAIN
5489 005134                                      END ;OF IF 42
5490 005134                                    END ;OF IF 42
5491 005134                                  END ;OF IFB $ENV
```

```
                                        SUBTST  <<PROTECT PROGRAM & LOADERS>>
                                ;*******************************************************************************
                                ;*SUBTEST        PROTECT PROGRAM & LOADERS
                                ;*******************************************************************************
005134  052737  000200  002626          BIS     @#BIT7,CONFIG           ;PROTECT PROGRAM SPACE (BANK 0)
005142  052737  000200  002632          BIS     @#BIT7,CONFIG+4         ;PROTECT LOADER SPACE (BANK 1)
005150                                  IF @#ENDAD NE 42                ;NOT ACT-11?
005160                                    TYPE  MSG000                  ;TYPE PROGRAM TITLE
005164                                  END ;OF IF @#ENDAD

005164                                  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.
005164                                  SET4    #3$
005172  005737  177746                  TST     CONTRL                  ;IS THERE A CONTROL REGISTER?
005176                                  SET4    #2$
005204  005737  177750                  TST     MAINT                   ;IS THERE A MAINTENANCE REGISTER?
005210                                  SET4    #1$
005216  005737  177754                  TST     DATARG                  ;IS THERE A DATA REGISTER?
005222                                  TYPE    MSG117                  ; 11/44
005226  000405                          BR      4$
005230                          1$:     TYPE    MSG116                  ; 11/34
005234  000402                          BR      4$
005236                          2$:     TYPE    MSG118                  ; 11/60
005242  052737  000014  177746  4$:     BIS     @#BIT2!BIT3,CONTRL      ;SET CACHE DISABLE BITS
005250  042737  000014  177746          BIC     @#BIT2!BIT3,CONTRL      ;CLEAR CACHE DISABLE BITS
005256  032737  000004  177746          BIT     @#BIT2,CONTRL           ;IS THE BIT SET?
005264  001004                          BNE     7$                      ;BRANCH IF THE BIT IS SET
005266  032737  000010  177746          BIT     @#BIT3,CONTRL           ;IS THE BIT SET?
005274  001413                          BEQ     6$                      ;BRANCH IF THE BIT IS SET
005276                          7$:     TYPE    MSG121                  ; CACHE BYPASSED
005302  104424                          CACHOFF
005304  013737  002516  002520          MOV     CACHKN,CACHKN+2         ;SAVE INFO ABOUT CACHE
005312  005037  002516                  CLR     CACHKN                  ;CACHE CANNOT BE USED - IT'S BYPASSED
005316  000404                          BR      8$
005320                          3$:     TYPE    MSG119                  ; NO
005324                          6$:     TYPE    MSG120                  ; CACHE AVAILABLE
```

```
5575 005330                                    SUBTST  <<SETUP USER & SUPERVISOR STACK>>
                                       ;*****************************************************************
                                       ;*SUBTEST       SETUP USER & SUPERVISOR STACK
                                       ;*****************************************************************
5576 005330  104421                    8$:     DEENERGIZE              ;TURN OFF MEMORY MANAGEMENT
5577 005332  005737  002430                     TST     NOSUPER         ;IS THERE A SUPERVISOR MODE?
5578 005336  001011                             BNE     5$              ;NO SKIP SUPERVISOR SETUP.
5579
5580                                            ;SET PREVIOUS MODE TO SUPERVISOR
5581 005340  042737  030000  177776             BIC     #BIT13!BIT12,PSW
5582 005346  052737  010000  177776             BIS     #BIT12,PSW
5583
5584 005354                                    PUSH    #SUPSTK
5585 005360  006606                            MTPI    SSP
5586
5587                                            ;SET PREVIOUS MODE TO USER
5588 005362  052737  030000  177776  5$:       BIS     #BIT13!BIT12,PSW
5589
5590 005370                                    PUSH    #USESTK
5591 005374  006606                            MTPI    USP
5592
5593 005376                                    SUBTST  <<GET SOFTWARE SWITCH REGISTER IF NECESSARY>>
                                       ;*****************************************************************
                                       ;*SUBTEST       GET SOFTWARE SWITCH REGISTER IF NECESSARY
                                       ;*****************************************************************
5597 005376                                    IF #AUTO IS FALSE                       ;IF NOT(APT OR ACT)
5598 005404                                      IF SWR EQ #SWREG              ;IF SOFTWARE SWITCH REG SELECTED
5599 005414  104407                                GTSWR                               ;;GET SOFT-SWR SETTINGS
5600 005416                                      END ;OF IF SWR
5601 005416                                    END ;OF IF #AUTO
5605
5606 005416                                    SUBTST  <<GET MEMORY MANAGEMENT READY>>
                                       ;*****************************************************************
                                       ;*SUBTEST       GET MEMORY MANAGEMENT READY
                                       ;*****************************************************************
5610 005416  104422                            KMAP                    ;MAP KERNEL SPACE 1 TO 1
5614 005420                                    MAP                     ;MAP SUPERVISOR SPACE (TEST AREA) 1 TO 1
5615 005434  104420                            ENERGIZE                ;TURN ON MEMORY MANAGEMENT
```

```
5618 005436                                     NEWTST  <<BIT TEST OF ALL CSR'S>>
                                        ;*****************************************************************************
                                        ;*TEST 1        BIT TEST OF ALL CSR'S
                                        ;*****************************************************************************
     005436  000004                     TST1:   SCOPE
5619                                    ;* THE FIRST PART OF THE CONFIGURATION ANALYSIS DOES THE FOLLOWING:
5620                                    ;*      1) FINDS WHICH CSR'S RESPOND, AND PUTS THEM INTO THE CSR INFORMATION
5621                                    ;*              TABLE, AND STORES ANOTHER BIT FOR "TOTCSRS".
5622                                    ;*      2) TESTS THE CSR BITS COMMON TO ALL CSR'S.
5623                                    ;*      3) FIGURES OUT IF THERE IS AN EUB BIT, AN ECC BIT, AND THE EXISTANCE
5624                                    ;*              OF THE ERROR ADDRESS BITS, AND MARKS THIS IN THE CSR
5625                                    ;*              INFORMATION TABLE.
5626                                    ;*      4) TESTS THE BITS PARTICULAR TO THAT TYPE OF CSR.
5627                                    ;*      5) IF ANY BITS TEST BAD IN THE CSR UNDER TEST, THE CSR OK BIT IN THE
5628                                    ;*              CSR INFORMATION TABLE IS CLEARED.
5629                                    ;* THE INFORMATION BITS ONE THROUGH THREE FORM A CODE WHICH GIVES THE TYPE
5630                                    ;* OF CSR:
5631                                    ;*                      ERR. ADDR.   PARITY  NOT EUB
5632                                    ;*      TYPE              BIT2        BIT1    BIT0     CODE TOTALS
5633                                    ;* UNIBUS PARITY          1           1       1           7
5634                                    ;*     MS11-L             1           1       0           6
5635                                    ;*     MF11S-K            1           0       1           5
5636                                    ;*     MS11-M             1           0       0           4
5637                                    ;* 11/45 BIPOLAR         0           1       1           3
5638                                    ;*
5639                                    ;* THIS MEMORY CODE WILL BE USED IN THE SECOND PART OF THIS ANALYSIS
5640                                    ;*
5641 005440  005005                             CLR     R5              ;R5 IS THE TOTAL CSR NUMBER
5642 005442  005000                             CLR     R0              ;R0 IS A TABLE INDEX
5643 005444  012703  172100                     MOV     #CSRADD,R3      ;R3 HAS THE CSR ADDRESS
5644 005450  012737  000001  002074             MOV     #1,NOPAR        ;IGNORE PARITY ERRORS
5645 005456                                     SET4    #2$
5646 005464  005713                     1$:     TST     (R3)    ;DOES THE CSR RESPOND?
5647 005466  052705  000001                     BIS     #1,R5
5648 005472  005004                             CLR     R4              ;CLEAR THE LAST CSR INDICATOR
5649 005474  052760  000007  002434             BIS     #7,CSRINFO(R0)  ;SET ALL THE MEMORY INFO BITS
5650 005502  052760  000030  002434             BIS     #BIT4!BIT3,CSRINFO(R0)  ;YES-MARK IT IN CSR INFORMATION TABLE
5651 005510  004537  006044                     JSR     R5,TEST         ;TEST BIT 0 AND 15
5652 005514  100001                             .WORD   BIT15!BIT0
5653 005516  012713  040000                     MOV     #BIT14,(R3)     ;IS THERE A BIT 14 RESPONDING
5654 005522  032713  040000                     BIT     #BIT14,(R3)     ;(IT'S THE EUB BIT)
5655 005526  001403                             BEQ     3$              ;BRANCH IF NO EUB BIT
5656 005530  042760  000001  002434             BIC     #BIT0,CSRINFO(R0) ;CLEAR EUB INFO IN THE CSR TABLE
5657 005536  005013                     3$:     CLR     (R3)            ;CLEAR THE CSR UNDER TEST
5658 005540  012713  020000                     MOV     #BIT13,(R3)     ;DOES BIT 13 RESPOND
5659 005544  032713  020000                     BIT     #BIT13,(R3)     ;(TO TEST FOR ECC CSR)
5660 005550  001417                             BEQ     4$              ;BRANCH IF NOT ECC CSR
5661 005552  042760  000002  002434             BIC     #BIT1,CSRINFO(R0) ;CLEAR PARITY INFO IN THE CSR TABLE
5662 005560  012713  020000                     MOV     #BIT13,(R3)     ;SET THE INHIBIT MODE POINTER TO 1ST 16K
5663 005564  004537  006044                     JSR     R5,TEST         ;TEST BIT 3
5664 005570  000010                             .WORD   BIT3
5665 005572  012713  020000                     MOV     #BIT13,(R3)
5666 005576  004537  006044                     JSR     R5,TEST         ;TEST BIT 1 AND 4
5667 005602  000022                             .WORD   BIT4!BIT1
5668 005604  012713  020000                     MOV     #BIT13,(R3)
```

```
5670 005610 004537 006044          4$.     JSR     R5,TEST         ;TEST BIT 2
5671 005614 000004                         .WORD   BIT2
5672 005616 005013                         CLR     (R3)
5673 005620 052713 007740                  BIS     #7740,(R3)      ;ARE THERE ERROR ADDRESS BITS?
5674 005624 032713 007740                  BIT     #7740,(R3)
5675 005630 001404                         BEQ     5$              ;BRANCH IF NO ERROR ADDR. BITS.
5676 005632 004537 006044                  JSR     R5,TEST         ;TEST BITS 5->11
5677 005636 007740                         .WORD   7740
5678 005640 000403                         BR      6$              ;SKIP OVER THE INFORMATION REPORTING
5679 005642 042760 000004 002434   5$:     BIC     #BIT2,CSRINFO(R0) ;REPORT THAT THERE ARE NO ERROR ADDRESS BITS.
5680 005650 032760 000002 002434   6$:     BIT     #BIT1,CSRINFO(R0) ;IS THIS CSR AN ECC CSR?
5681 005656 001014                         BNE     7$              ;BRANCH IF NOT
5682 005660 032760 000001 002434           BIT     #BIT0,CSRINFO(R0) ;IS THE EUB BIT SET?
5683 005666 001410                         BEQ     7$              ;BRANCH IF IT IS
5684                                       ;WE MUST NOW TEST FOR MS11-M ON THE UNIBUS
5685 005670 012713 007760                  MOV     #7760,(R3)      ;PUT PATTERN & SBE BIT INTO BITS 4 >11
5686 005674 022713 007760                  CMP     #7760,(R3)      ;ARE THEY STILL THERE?
5687 005700 001403                         BEQ     7$              ;YES - BRANCH FOR MF11S-K CSR
5688 005702 042760 000001 002434           BIC     #BIT0,CSRINFO(R0) ;NO - SET EUB BIT FOR MS11 M
5689 005710 005013                 7$:     CLR     (R3)            ;CLEAR CSR
5690 005712 022760 000040 002434           CMP     #40,CSRINFO(R0) ;IS THIS A LEGAL CONFIGURATION?
5691 005720 100004                         BPL     10$             ;BRANCH IF IT'S LEGAL
5692 005722 016037 002434 002050           MOV     CSRINFO(R0),BAD
5693 005730 104021                         ERROR   .21             ;ILLEGAL TYPE ERROR
5694 005732 032760 000004 002434   10$:    BIT     #BIT2,CSRINFO(R0) ;DOES THIS CSR HAVE ERROR BITS
5695 005740 001016                         BNE     2$              ;BRANCH IF TRUE
5696 005742 032760 000002 002434           BIT     #BIT1,CSRINFO(R0) ;ARE THE OTHER 2 BITS SET?
5697 005750 001404                         BEQ     11$             ;BRANCH IF NOT
5698 005752 032760 000001 002434           BIT     #BIT0,CSRINFO(R0)
5699 005760 001006                         BNE     2$
5700 005762 016037 002434 002050   11$:    MOV     CSRINFO(R0),BAD
5701 005770 104021                         ERROR   .21             ;ILLEGAL TYPE ERROR
5702 005772 005060 002434                  CLR     CSRINFO(R0)     ;CLEAR THE CSR INFO-IT WILL NOT EXIST IN THE PROGRAM
5703 005776 062700 000002          2$:     ADD     #2,R0           ;INCREMENT TO NEXT CSR TABLE
5704 006002 062703 000002                  ADD     #2,R3           ;INCREMENT TO NEXT CSR
5705 006006 006305                         ASL     R5
5706 006010 103001                         BCC     8$              ;IS THERE A CSR 0?
5707 006012 005204                         INC     R4              ;YES - SET CSR PRESENT FLAG
5708 006014 022700 000040          8$:     CMP     #40,R0          ;ARE WE DONE?
5709 006020 001221                         BNE     1$              ;BRANCH IF MORE TO DO
5710 006022 000241                         CLC
5711 006024 006005                         ROR     R5              ;RESYNC R5
5712 006026 005704                         TST     R4              ;WAS THERE A CSR 0?
5713 006030 001402                         BEQ     9$              ;BRANCH IF NOT
5714 006032 052705 100000                  BIS     #BIT15,R5       ;YES - SET IN THE CSR TABLE
5715 006036 010537 002220          9$:     MOV     R5,TOTCSRS      ;STORE R5 IN TOTCSRS
5716 006042 000437                         BR      CTEST           ;JUMP OVER SUBROUTINE
5717                                       ;THIS SUBROUTINE TESTS THE CSR BITS
5718 006044 012501                 TEST:   MOV     (R5)+,R1        ;GET THE BIT TO TEST
5719 006046 050113                         BIS     R1,(R3)         ;SET THAT IN THE CSR UNDER TEST
5720 006050 030113                         BIT     R1,(R3)         ;IS THE BIT STILL THERE?
5721 006052 001013                         BNE     1$              ;BRANCH IF STILL THERE
5722 006054 011337 002144                  MOV     (R3),CSR        ;READ CSR
5723 006060 010137 002042                  MOV     R1,GOOD
5724 006064 032713 100020                  BIT     #BIT15!BIT4,(R3);IS IT BECAUSE OF A SBE OR DBE?
5725 006070 001004                         BNE     1$              ;BRANCH IF IT IS
5726 006072 104035                         ERROR   .35             ;BIT SET ERROR
```

```
 5727 006074  042760  000010  002434          BIC     @BIT3,CSRINFO(R0) ;CLEAR CSR OK BIT
 5728 006102  040113                  1$:     BIC     R1,(R3)           ;CLEAR THE SELECTED BIT
 5729 006104  030113                          BIT     R1,(R3)           ;IS IT CLEARED?
 5730 006106  001413                          BEQ     2$                ;BRANCH IF IT IS CLEARED
 5731 006110  011337  002144                  MOV     (R3),CSR          ;READ CSR
 5732 006114  010137  002042                  MOV     R1,GOOD
 5733 006120  032713  100020                  BIT     @BIT15!BIT4,(R3);IS IT BECAUSE OF A SBE OR DBE?
 5734 006124  001004                          BNE     2$                ;BRANCH IF TRUE
 5735 006126  104010                          ERROR   .10               ;BIT CLEAR ERROR
 5736 006130  042760  000010  002434          BIC     @BIT3,CSRINFO(R0) ;CLEAR CSR OK BIT
 5737 006136  000205                  2$:     RTS     R5
 5738 006140  000000                  TRACE:  .WORD   0
```

```
                                        ;THE FOLLOWING ROUTINE DETERMINES WHICH CSR CONTROLS PROGRAM SPACE
                                        ;
006142  104424                  CTEST:  CACHOFF
006144  012737  177777  002504          MOV     #177777,PGMCSR
006152  012737  002000  172350          MOV     #2000,KIPAR4            ;SET UP MAP REGISTER
006160  012701  002364                  MOV     #TESTADD,R1
006164  012737  100000  002364          MOV     #100000,TESTADD
006172  012737  100002  002366          MOV     #100002,TESTADD+2
006200  005000                          CLR     R0                      ;CLEAR CSR COUNTER
006202  005037  002146                  CLR     CSRNO
006206  013703  002220                  MOV     TOTCSRS,R3              ;OBTAIN CSR MAP
006212  000240                          NOP                             ;DEBUG AID
006214  006303                  4$:     ASL     R3                      ;PUT HIGH ORDER BIT INTO C BIT
006216  103407                          BCS     2$                      ;BRANCH IF CSR EXISTS
006220  062700  000002          1$:     ADD     #2,R0                   ;UPDATE CSR COUNTER
006224  010037  002146                  MOV     R0,CSRNO
006230  005703                          TST     R3                      ;IS MAP EMPTY?
006232  001464                          BEQ     3$                      ;BRANCH IF SO
006234  000767                          BR      4$
006236  000240                  2$:     NOP                             ;DEBUG AID
006240  000241                          CLC                             ;CLEAR CARRY
006242  032760  000002  002434          BIT     #BIT1,CSRINFO(R0)       ;IS THIS PARITY MEMORY?
006250  001414                          BEQ     5$                      ;BRACH IF NOT
006252  052760  000004  172100          BIS     #BIT2,CSRADD(R0)        ;SET WRITE WRONG PARITY
006260  012771  123456  000000          MOV     #123456,@(R1)           ;WRITE DATA
006266  012771  123456  000002          MOV     #123456,@2(R1)
006274  005060  172100                  CLR     CSRADD(R0)              ;RESTORE CSR
006300  000414                          BR      6$
006302  012760  000000  172100  5$:     MOV     #0,CSRADD(R0)           ;CLEAR THE CSR UNDER TEST
006310  012771  123456  000000          MOV     #123456,@(R1)           ;WRITE DATA
006316  012771  123456  000002          MOV     #123456,@2(R1)
006324  012760  020006  172100          MOV     #20006,CSRADD(R0)       ;SET DIAG CHECK MODE
006332  005771  000000          6$:     TST     @(R1)                   ;WRITE CHECKBITS TO CSR
006336  016004  172100                  MOV     CSRADD(R0),R4           ;WRITE CSR TO R4
006342  032760  000002  002434          BIT     #BIT1,CSRINFO(R0)       ;PARITY MEMORY?
006350  001403                          BEQ     7$                      ;BRACH IF NOT
006352  005704                          TST     R4                      ;PARITY ERROR?
006354  100411                          BMI     8$                      ;BRACH IF SO
006356  000720                          BR      1$                      ;TRY NEXT CSR
006360  000240                  7$:     NOP                             ;DEBUG AID
006362  072427  177773                  ASH     #-5,R4
006366  042704  177600                  BIC     #^C177,R4
006372  022704  000157                  CMP     #157,R4                 ;CORRECT CHECKBITS?
006376  001310                          BNE     1$                      ;BRANCH IF NOT
006400  010037  002504          8$:     MOV     R0,PGMCSR
006404  000240                  3$:     NOP                             ;DEBUG AID
006406  104502                          CLRCSR                          ;CLEAR ALL CSR'S
006410  012771  000000  000000          MOV     #0,@(R1)                ;RESTORE TEST LOCATIONS
006416  012771  000000  000002          MOV     #0,@2(R1)
006424  023727  002504  177777          CMP     PGMCSR,#177777
006432  001402                          BEQ     FINT                    ;IF PROGRAM CSR NOT FOUND GO TO FINT
006434  000137  007040                  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
                                        ;
006440                          FINT:   SET4    #2$                     ;NE MEMORY TRAPS GO TO 2$
006446  012771  123456  000000  1$:     MOV     #123456,@(R1)           ;WRITE DATA AT FIRST LOCATION OF BANK 2 IN BOARD
006454  012771  123456  000002          MOV     #123456,@2(R1)          ;WRITE DATA AT SECOND LOCATION OF BANK 2 IN BOARD
006462  062737  010000  172350          ADD     #10000,KIPAR4           ;UPDATE PAR4 TO POINT TO UPPER BOARDS
006470  000766                          BR      1$                      ;KEEP GOING TILL NO MORE MEMORY
006472  012700  177776          2$:     MOV     #-2,R0
006476  013703  002220                  MOV     TOTCSRS,R3              ;PUT CSR MAP IN R3
006502  062700  000002          3$:     ADD     #2,R0                   ;UPDATE CSR COUNTER
006506  010037  002146                  MOV     R0,CSRNO                ;UPDATE CSRNO
006512  006303                          ASL     R3
006514  103403                          BCS     4$                      ;BRANCH IF CSR EXISTS
006516  005703                          TST     R3                      ;ANY CSR'S LEFT?
006520  001405                          BEQ     5$                      ;BRANCH IF NOT
006522  000767                          BR      3$                      ;LOOK FOR NEXT CSR
006524  012760  020006  172100  4$:     MOV     #20006,CSRADD(R0)       ;SET DIAGNOSTIC CHECK MODE IN CSR
006532  000763                          BR      3$                      ;LOOK FOR NEXT CSR
006534                          5$:     SET4    #6$                     ;NE MEMORY TRAPS NOW GO TO 6$
006542  012700  177776                  MOV     #-2,R0                  ;RESET CSR POINTER
006546  012737  002000  172350          MOV     #2000,KIPAR4            ;REMAP PAR4 TO POINT TO BANK 2
006554  005771  000000                  TST     @(R1)                   ;TEST NONASSERTED LOCATIONS
006560  062700  000002          6$:     ADD     #2,R0                   ;UPDTAE CSR POINTER
006564  010037  002146                  MOV     R0,CSRNO
006570  022700  000040                  CMP     #40,R0                  ;NOT FOUND?
006574  001515                          BEQ     10$                     ;BRANCH IF NOT
006576  016004  172100                  MOV     CSRADD(R0),R4           ;GET CSR CONTENTS
006602  072427  177773                  ASH     #-5,R4
006606  042704  177600                  BIC     #^C177,R4               ;CLEAR ALL BUT CHECKBITS
006612  022704  000157                  CMP     #157,R4                 ;PROPER CHECKBITS?
006616  001401                          BEQ     7$                      ;BRANCH IF SO
006620  000757                          BR      6$                      ;TRY NEXT CSR IF NOT
006622  110037  002504          7$:     MOVB    R0,PGMCSR               ;WRITE NON-ASSERTED CSR # IN PGMCSR
006626                                  SET4    #8$                     ;NE TRAPS GO TO 8$
006634  012700  177776                  MOV     #-2,R0
006640  013703  002220                  MOV     TOTCSRS,R3              ;PUT CSR MAP IN R3
006644  062700  000002          23$:    ADD     #2,R0                   ;UPDATE CSR COUNTER
006650  010037  002146                  MOV     R0,CSRNO                ;UPDATE CSRNO
006654  006303                          ASL     R3
006656  103403                          BCS     24$                     ;BRANCH IF CSR EXISTS
006660  005703                          TST     R3                      ;ANY CSR'S LEFT?
006662  001405                          BEQ     25$                     ;BRANCH IF NOT
006664  000767                          BR      23$                     ;LOOK FOR NEXT CSR
006666  012760  020006  172100  24$:     MOV     #20006,CSRADD(R0)       ;SET DIAGNOSTIC CHECK MODE IN CSR
006674  000763                          BR      23$                     ;LOOK FOR NEXT CSR
006676  012700  177776          25$:    MOV     #-2,R0
006702  005771  000002                  TST     @2(R1)                  ;TEST ASSERTED LOCATIONS
006706  062700  000002          8$:     ADD     #2,R0
006712  010037  002146                  MOV     R0,CSRNO
006716  022700  000040                  CMP     #40,R0
006722  001442                          BEQ     10$
006724  016004  172100                  MOV     CSRADD(R0),R4
006730  072427  177773                  ASH     #-5,R4
006734  042704  177600                  BIC     #^C177,R4
006740  022704  000157                  CMP     #157,R4                 ;PROPER CHECKBITS?
```

```
5850 006744  001401                        BEQ     9$                      ;BRANCH IF SO
5851 006746  000757                        BR      8$                      ;TRY NEXT CSR IF NOT
5852 006750  110037  002505          9$:   MOVB    R0,PGMCSR+1             ;WRITE ASSERTED CSR # IN PGMCSR
5853 006754  052737  100000  002504        BIS     #BIT15,PGMCSR           ;SET INTERLEAVED INDICATOR IN PGMCSR
5854 006762  104502                        CLRCSR
5855 006764  012737  002000  172350        MOV     #2000,KIPAR4
5856 006772                                SET4    #12$                    ;NE MEMORY TRAPS GO TO 12$
5857 007000  012771  000000  000000  11$:  MOV     #0,@(R1)                ;WRITE DATA AT FIRST LOCATION OF BANK 2 IN BOARD
5858 007006  012771  000000  000002        MOV     #0,@2(R1)               ;WRITE DATA AT SECOND LOCATION OF BANK 2 IN BOARD
5859 007014  062737  010000  172350        ADD     #10000,KIPAR4           ;UPDATE PAR4 TO POINT TO UPPER BOARDS
5860 007022  000766                        BR      11$
5861 007024  104423                  12$:  CACHON
5862 007026  000404                        BR      CLRMEM
5863 007030                          ' ':  TYPE    MSG126                  ;ERROR   PROGRAM CSR NOT FOUND!
5864 007034  005037  002504                CLR     PGMCSR                  ;SET TO DEFAULT OF 0
```

```
                                        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.
                                ;
007040                  CLRMEM: SET4    #CLREX          ;NONEM TRAPS GO TO CLREX
007046  005037  006140          CLR     TRACE
007052  012737  000001  002074  MOV     #1,NOPAR        ;IGNORE PARITY ERRORS
007060  012737  002000  172350  MOV     #2000,KIPAR4    ;SET UP MAP TO START AT BANK 2
007066  012701  100000          MOV     #100000,R1      ;R1 MAPS TO KIPAR4
007072  020127  117776  1$:     CMP     R1,#117776      ;WHOLE 16K BANK DONE?   ;R C
007076  001003                  BNE     2$              ;KEEP GOING IF NOT
007100  012737  177777  006140  MOV     #-1,TRACE       ;USE TRACE FLAG TO FLAG END OF BANK
007106  005021          2$:     CLR     (R1)+           ;CLEAR CONTENTS & INCREMENT
007110  005737  006140          TST     TRACE           ;EOB FLAG SET?
007114  001001                  BNE     3$              ;GO TO NEXT BANK IF SO
007116  000765                  BR      1$
007120  062737  000200  172350  3$:     ADD     #200,KIPAR4     ;SET MAP FOR NEXT BANK ;R-C
007126  005037  006140          CLR     TRACE           ;RESET FLAG
007132  012701  100000          MOV     #100000,R1      ;RESET R1
007136  000755                  BR      1$              ;CLEAR NEXT BANK
007140  000240          CLREX:  NOP
007142  005037  006140          CLR     TRACE
007146                          RES4
```

```
5894 007170                     ANA2:   SUBTST  <<MATCH ALL CSR S WITH MEMORY>>
                                ;*****************************************************************************
                                ;*SUBTEST        MATCH ALL CSR'S WITH MEMORY
                                ;*****************************************************************************
5895                            ;* THE SECOND PART OF THE ANALYSIS MATCHES UP THE CSR S WITH THE MEMORY, AND
5896                            ;* INSTALLS ALL THE INFORMATION FOUND IN THE CONFIGURATION TABLE.  FOR ECC,
5897                            ;* THIS IS DONE BY TAKING EACH CSR FOUND IN THE PREVIOUS SECTION SEQUENTIALLY
5898                            ;* AND CHECKING THROUGH ALL OF MEMORY, ONE BANK AT AT TIME, TO SEE WHICH BANKS
5899                            ;* RESPOND TO THE CSR IN QUESTION. THE FIRST DOUBLE WORD PAIR IN EACH BANK ARE
5900                            ;* WRITTEN WITH DATA AND DIAGNOSTIC CHECK MODE SET IN THE CSR IN ORDER TO AC
5901                            ;* COMPLISH THIS. ALL POSSIBLE CONFIGURATIONS OF DOUBLE WORD PAIRS (NON INTER
5902                            ;* LEAVED, 64K INTERLEAVED, OR 128K INTERLEAVED) ARE CHECKED FOR EACH BANK
5903                            ;* THROUGH USE OF TESTADD AND KERNEL INSTRUCTION PAGE ADDRESS REGISTERS 4 AND
5904                            ;* 5. IF WE GET THE PROPER CHECKBITS BACK, WE HAVE A MATCH. IF NOT, THE ROUT
5905                            ;* INE CHECKS FOR SINGLE OR DOUBLE BIT ERRORS.
5906                            ;*              IF ONE OR THE OTHER IS FOUND, THE ERROR ADDRESS IS CHECKED
5907                            ;* TO SEE IF IT IS THAT BANK.  IF IT IS, WE HAVE A MATCH.  AT THE END OF EACH
5908                            ;* BANK PASS, FOR EACH CSR PASS, THE PROGRAM COMES UP WITH A NUMBER, STORED IN
5909                            ;* 'I', WHICH DENOTES THE FOLLOWING:
5910                            ;*
5911                            ;*      I       MEMORY DESCRIPTION
5912                            ;*      .       ...... .. ... . ...
5913                            ;*      0       NON-EXISTANT MEMORY
5914                            ;*      1       NON-INTERLEAVED MEMORY
5915                            ;*      2       64K INTERLEAVED, A1 NOT ASSERTED MEMORY
5916                            ;*      3       128K INTERLEAVED, A1 NOT ASSERTED MEMORY
5917                            ;*      4       64K INTERLEAVED, A1 ASSERTED MEMORY
5918                            ;*      5       128K INTERLEAVED, A1 ASSERTED MEMORY
5919                            ;*
5920                            ;*  NOTE - I=2 THROUGH I=5 CAN ONLY OCCUR WITH MS11 M MEMORY.
5921                            ;*
5922                            ;* NOTE THAT PARITY MEMORY WRITES WRONG PARITY TO THE DOUBLE WORDS, THEN LOOKS
5923                            ;* FOR THE PARITY ERROR BIT TO BE SET.  IF THE BIT IS SET, WE HAVE A MATCH
5924                            ;*
5925 007170                            SET4    #1008                   ;NE MEMORY TRAPS GO TO 1008
5926 007176  005037  002276            CLR     CHECK                   ;CLEAR CHECK
5927 007202  012701  002364            MOV     #TESTADD,R1             ;SET UP THE VIRTUAL ADDR. POINTER
5928 007206  013703  002220            MOV     TOTCSRS,R3              ;MOVE CSR MAP INTO R3
5929 007212  005000                    CLR     R0                      ;CLEAR THE CSR POINTER
5930 007214  005005                    CLR     R5                      ;CLEAR THE PROGRAM CSR STATUS POINTER
5931 007216  005737  002426            TST     NO22BIT                 ;IS THIS AN 11/44 OR 11/24?
5932 007222  001403                    BEQ     7$                      ;BRANCH IF IT IS
5933 007224  005037  002532            CLR     LASTBLOCK               ;ADJUST LASTBLOCK INDICATOR FOR 124K MACHINE
5934 007230  000413                    BR      1$                      ;BRANCH OVER NEXT PIECE OF CODE
5935 007232  022737  000167  002530  7$:       CMP     #167,LASTBANK           ;IS THERE UNIBUS MEMORY ABOVE 17000000?
5936 007240  001407                    BEQ     1$                      ;BRANCH IF NOT
5937 007242  013702  002530            MOV     LASTBANK,R2             ;SET UP A NEW LAST BLOCK INDICATOR
5938 007246  005202                    INC     R2
5939 007250  072227  000011            ASH     #9.,R2
5940 007254  010237  002532            MOV     R2,LASTBLOCK
5941 007260  012702  000004        1$:     MOV     #4,R2                   ;R2 IS INDEX FOR CONFIG TABLE
5942 007264  012737  001000  172350        MOV     #1000,KIPAR4            ;SET KIPAR4 FOR BANK 1
5943 007272  012737  001000  172352        MOV     #1000,KIPAR5            ;SET KIPAR5 FOR BANK 1
5944 007300  006303                2$:     ASL     R3                      ;DOES THIS CSR EXIST?
5945 007302  103420                    BCS     3$                      ;BRANCH IF IT DOES EXIST
5946 007304  062700  000002            ADD     #2,R0                   ;INCREMENT THE CSR POINTER
5947 007310  010077  002146            MOV     R0,CSRNO                ;STORE IT IN CSRNO ALSO
```

```
5948 007314 005703                      TST     R3                      ;ARE THERE ANY MORE CSR'S TO DO?
5949 007316 001370                      BNE     2$                      ;BRANCH IF ALL CSRS NOT DONE
5950 007320 012737 001000 172350        MOV     #1000,KIPAR4            ;RESTORE KIPAR4
5951 007326 012737 001200 172352        MOV     #1200,KIPAR5            ;RESTORE KIPAR5
5952 007334 013706 002536               MOV     KSTACK,SP               ;RESTORE STACK
5953 007340 000137 010540               JMP     SUBAAS                  ;JUMP TO SUBAAS IF ALL CSR'S ARE DONE
5954 007344 010037 002146       3$:     MOV     R0,CSRNO                ;MAKE SURE CSRNO IS UPDATED
5955 007350 104424              13$:    CACHOFF                         ;TURN THE CACHE OFF
5956 007352 000240                      NOP
5957 007354 012737 100000 002364 45$:   MOV     #100000,TESTADD         ;SET UP VIRTUAL ADDRESS TO KIPAR4
5958 007362 012737 120002 002366        MOV     #120002,TESTADD+2       ;SET UP VIRTUAL ADDRESS TO KIPAR5
5959 007370 032762 000040 002626        BIT     #BIT5,CONFIG(R2)        ;IS THIS A BANK TO SKIP?
5960 007376 001402                      BEQ     43$                     ;NO   BRANCH AROUND NEXT INSTRUCTION
5961 007400 000137 010454               JMP     6$                      ;YES - GO TO END OF BANK
5962 007404 005037 002424       43$:    CLR     I                       ;CLEAR THE MEMORY CONFIGURATION COUNTER
5963 007410 005771 000000       4$:     TST     @(R1)                   ;TEST TO SEE THAT THERE IS MEMORY PRESENT
5964 007414 005237 002424               INC     I
5965 007420                             PUSH    @(R1),@2(R1)            ;SAVE THE LOCATIONS UNDER TEST
5966 007430 032760 000002 002434        BIT     #BIT1,CSRINFO(R0)       ;IS THIS PARITY MEMORY?
5967 007436 001414                      BEQ     34$                     ;NO - BRANCH
5968 007440 052760 000004 172100        BIS     #BIT2,CSRADD(R0)        ;SET WRITE WRONG PARITY
5969 007446 012771 123456 000000        MOV     #123456,@(R1)           ;SET THE FIRST LOCATION UNDER TEST
5970 007454 012771 123456 000002        MOV     #123456,@2(R1)          ;SET THE SECOND LUT
5971 007462 005060 172100               CLR     CSRADD(R0)              ;CLEAR THE CSR
5972 007466 000411                      BR      41$                     ;TEST LOCATIONS
5973 007470 012771 123456 000000 34$:   MOV     #123456,@(R1)           ;SET THE FIRST LOCATION UNDER TEST
5974 007476 012771 123456 000002        MOV     #123456,@2(R1)          ;SET THE SECOND LUT
5975 007504 104503                      CLR1CSR                         ;RESET CSR
5976 007506 104475                      CB1CSR                          ;SET DIAG. CHECK MODE IN CSR UNDER TEST
5977 007510 000240                      NOP                             ;DEBUG AID
5978 007512 005771 000000       41$:    TST     @(R1)                   ;READ THE FIRST LUT TO WRITE CKBITS. INTO CSR
5979 007516 104426                      READCSR                         ;READ THE CSR UNDER TEST
5980 007520 000240                      NOP                             ;DEBUG AID
5981 007522 013704 002144               MOV     CSR,R4                  ;GET THE CHECKBITS FROM THE CSR
5982 007526 000240                      NOP                             ;DEBUG AID
5983 007530 010437 002406               MOV     R4,TEMP                 ;SAVE IN TEMP FOR LATER
5984 007534 104503                      CLR1CSR                         ;RESET CSR
5985 007536                             POP     @2(R1),@(R1)            ;RESTORE LOCATIONS UNDER TEST
5986 007546 032760 000002 002434        BIT     #BIT1,CSRINFO(R0)       ;IS THIS PARITY MEMORY?
5987 007554 001404                      BEQ     42$                     ;NO - BRANCH
5988 007556 005704                      TST     R4                      ;DID WE GET A PARITY ERROR?
5989 007560 100414                      BMI     25$                     ;YES - FILL IN CONFIG TABLE
5990 007562 000137 010454               JMP     6$                      ;NO - JUMP TO END OF BANK
5991 007566 072427 177773       42$:    ASH     #-5,R4                  ;MANIPULATE THE CSR BITS
5992 007572 042704 177600               BIC     #^C177,R4               ;INTO A USABLE FORM.
5993 007576 000240                      NOP                             ;DEBUG AID
5994 007600 022704 000157               CMP     #157,R4                 ;DO THE CHECKBITS COMPARE TO WHAT WAS WRITTEN?
5995 007604 001402                      BEQ     25$                     ;BRANCH IF THERE IS A MATCH
5996 007606 000137 010134               JMP     22$                     ;ELSE BRANCH IF NOT THE SAME
5997                                    ;*
5998                                    ;* WE COME HERE IF THERE IS A MATCH
5999                                    ;*
6000 007612 010004              25$:    MOV     R0,R4                   ;GET THE CSR NUMBER
6001 007614 006204                      ASR     R4                      ;SET IT UP FOR USE IN THE
6002 007616 000304                      SWAB    R4                      ;CONFIGURATION TABLE.
6003 007620 042704 170377               BIC     #170377,R4              ;CLEAR OFF EXTRANEOUS BITS
6004 007624 032737 000004 002424        BIT     #BIT2,I                 ;INTERLEAVED A1 ASSERTED MEMORY FOUND?
```

```
6005 007632  001402                        BEQ     15$                     ;BRANCH IF NOT
6006 007634  072427  000004                ASH     #4,R4                   ;PUT CSR NUMBER IN INTERLEAVED CSR SLOT
6007 007640  050462  002626        15$:    BIS     R4,CONFIG(R2)           ;PUT CSR NUMBER IN CONFIG. TABLE
6008 007644  016004  002434                MOV     CSRINFO(R0),R4          ;GET MEMORY TYPE
6009 007650  042704  177770                BIC     #^C7,R4                 ;CLEAR OFF THE EXTRANEOUS BITS
6010 007654  000304                        SWAB    R4                      ;MOVE INTO PROPER POSITION
6011 007656  050462  002630                BIS     R4,CONFIG+2(R2)         ;SET IT INTO THE CONFIG TABLE
6012 007662  022737  000001  002424        CMP     #1,I                    ;WAS THIS NON INTERLEAVED MEMORY?
6013 007670  001431                        BEQ     24$                     ;BRANCH IF IT WAS
6014 007672  052762  010000  002630        BIS     #BIT12,CONFIG+2(R2)     ;SET THE INTERLEAVED BIT
6015 007700  010204                        MOV     R2,R4                   ;SAVE THE CURRENT BANK INDEX
6016 007702  032737  000001  002424        BIT     #BIT0,I                 ;WAS THIS 128K INTERLEAVED?
6017 007710  001006                        BNE     5$                      ;BRANCH IF TRUE
6018 007712  052762  004000  002630        BIS     #BIT11,CONFIG+2(R2)     ;SET 64K INTERLEAVED FLAG IN CONFIG
6019 007720  062704  000020                ADD     #20,R4                  ;SET NEW BANK POINTER TO 4 BANKS AHEAD
6020 007724  000402                        BR      16$                     ;JUMP OVER NEXT INSTRUCTION
6021 007726  062704  000040        5$:     ADD     #40,R4                  ;SET NEW BANK POINTER 8 BANKS AHEAD
6022 007732  052764  000040  002626  16$:    BIS     #BIT5,CONFIG(R4)        ;SET SKIP ECC LOGIC TESTS FLAG
6023 007740  056264  002626  002626        BIS     CONFIG(R2),CONFIG(R4)   ;SET OTHER INFO INTO THAT BANK
6024 007746  056264  002630  002630        BIS     CONFIG+2(R2),CONFIG+2(R4)
6025                                        ;+
6026                                        ;+ THIS SECTION IS EXECUTED ONLY WHEN THE BANK=1
6027                                        ;+
6028 007754  022737  001000  172350  24$:    CMP     #1000,KIPAR4            ;IS THIS BANK 1 ?
6029 007762  001402                        BEQ     30$                     ;BRANCH IF TRUE
6030 007764  000137  010314                JMP     33$                     ;ELSE JUMP TO END OF THIS BANK
6031 007770  032737  100020  002406  30$:    BIT     #BIT15!BIT4,TEMP        ;WAS THERE A SBE OR DBE?
6032 007776  001417                        BEQ     10$                     ;BRANCH IF NOT
6033 010000  013704  002406                MOV     TEMP,R4                 ;GET CSR CONTENTS
6034 010004  072427  177767                ASH     #-9.,R4                 ;MAKE ERROR ADDRESS INTO BANK #
6035 010010  022704  000001                CMP     #1,R4                   ;ERROR IN BANKS 0 OR 1?
6036 010014  003010                        BGT     10$                     ;BRANCH IF NOT
6037 010016  052762  000001  002626        BIS     #BIT0,CONFIG(R2)        ;SET ERROR FLAG IN CONFIG TABLE
6038 010024  105262  002630                INCB    CONFIG+2(R2)            ;ADD ONE TO BANK ERROR COUNT
6039 010030                                SET     CONFGERROR              ;PRINT CONFIG TABLE
6040 010036  053737  002632  002626  10$:    BIS     CONFIG+4,CONFIG         ;SET UP INFORMATION IN BANK ZERO
6041 010044  053737  002634  002630        BIS     CONFIG+6,CONFIG+2
6042 010052  000240                        NOP                             ;DEBUG AID
6043 010054  022737  000001  002424        CMP     #1,I                    ;WAS THIS NON-INTERLEAVED MEMORY
6044 010062  001002                        BNE     46$                     ;NO - BRANCH OVER NEXT STMT.
6045 010064  000137  010454                JMP     6$                      ;YES - JUMP TO END OF THIS BANK
6046 010070  012704  000020        46$:    MOV     #20,R4                  ;SET UP COUNTER FOR 64K INTERLEAVED
6047 010074  032737  000001  002424        BIT     #BIT0,I                 ;WAS IT 128K INTERLEAVED?
6048 010102  001402                        BEQ     26$                     ;BRANCH IF NOT
6049 010104  062704  000020                ADD     #20,R4                  ;SET UP COUNTER FOR 128K INTERLEAVED
6050 010110  053764  002626  002626  26$:    BIS     CONFIG,CONFIG(R4)       ;SET OTHER BANK WITH SAME INFORMATION
6051 010116  053764  002630  002630        BIS     CONFIG+2,CONFIG+2(R4)   ;AS IN BANK 0
6052 010124  052764  000040  002626        BIS     #BIT5,CONFIG(R4)        ;SET SKIP ECC LOGIC TESTS FLAG
6053 010132  000470                        BR      33$                     ;BRANCH
6054                                        ;+
6055                                        ;+ IF CHECKBITS DID NOT MATCH, WE COME HERE
6056                                        ;+
6057 010134  032737  100020  002144  22$:    BIT     #BIT15!BIT4,CSR         ;SBE OR DBE FLAGS SET?
6058 010142  001001                        BNE     8$                      ;BRANCH IF TRUE
6059 010144  000463                        BR      33$                     ;CHECK TO SEE IF IT IS MS11-M
6060 010146  013704  002146        8$:     MOV     CSRNO,R4                ;GET CSRNO
6061 010152  042764  000006  172100        BIC     #6,CSRADD(R4)           ;TURN OFF DIAG CHECK & ECC DISABLE
```

```
6062 010160                                          PUSH    R0,R1                   ;SAVE R0 & R1
6063 010164  016401  172100                          MOV     CSRADD(R4),R1           ;GET CSR INFORMATION
6064 010170  072127  177773                          ASH     #-5,R1                  ;SET UP ERROR ADDRESS
6065 010174  042701  177600                          BIC     #^C177,R1
6066 010200  005737  002426                          TST     NO22BIT                 ;IS THIS AN 11/44 OR 11/24?
6067 010204  001015                                  BNE     27$                     ;BRANCH IF NOT
6068 010206  052764  040000  172100                  BIS     #BIT14,CSRADD(R4)       ;GET EXTENDED ERROR ADDRESS BITS
6069 010214  016400  172100                          MOV     CSRADD(R4),R0           ;READ FROM CSR
6070 010220  042764  040000  172100                  BIC     #BIT14,CSRADD(R4)       ;TURN OFF EUB BIT
6071 010226  042700  177037                          BIC     #^C740,R0               ;SET UP EXTENDED BITS
6072 010232  006300                                  ASL     R0
6073 010234  006300                                  ASL     R0
6074 010236  060001                                  ADD     R0,R1                   ;SET UP TOTAL ERROR ADDRESS
6075 010240  010104                          27$:    MOV     R1,R4                   ;SAVE IN R4
6076 010242                                          POP     R1,R0                   ;RESTORE R0 & R1
6077 010246  072427  000005                          ASH     #5,R4                   ;SET ERROR ADDRESS UP IN PAR NOTATION
6078 010252  020437  172350                          CMP     R4,KIPAR4               ;DOES IT EQUAL KIPAR4?
6079 010256  001001                                  BNE     28$                     ;BRANCH IF FALSE
6080 010260  000403                                  BR      35$                     ;YES - MARK INFO IN CONFIG TABLE
6081 010262  020437  172352                  28$:    CMP     R4,KIPAR5               ;DOES IT EQUAL KIPAR5?
6082 010266  001012                                  BNE     33$                     ;BRANCH IF FALSE
6083 010270  052762  000001  002626          35$:    BIS     #BIT0,CONFIG(R2)        ;SET BANK ERROR FLAG
6084 010276  105262  002630                          INCB    CONFIG+2(R2)            ;INCREMENT BANK ERROR COUNTER
6085 010302                                          SET     CONFGERROR              ;PRINT CONFIG TABLE
6086 010310  000137  007612                          JMP     25$                     ;YES - MARK INFO IN CONFIG TABLE
6087                                                 ;*
6088                                                 ;* THIS SECTION SETS UP ALL THE POSSIBLE CONFIGURATIONS OF
6089                                                 ;* MS11-M MEMORY.
6090                                                 ;*
6091 010314  032760  000003  002434  33$:    BIT     #BIT0!BIT1,CSRINFO(R0)  ;IS THIS MS11-M MEMORY?
6092 010322  001054                                  BNE     6$                      ;NO   GO TO END OF BANK
6093 010324  032760  000004  002434                  BIT     #BIT2,CSRINFO(R0)
6094 010332  001450                                  BEQ     6$
6095 010334  022737  000001  002424                  CMP     #1,I                    ;IS THIS 1ST TIME THROUGH?
6096 010342  103410                                  BLO     18$                     ;BRANCH IF NOT
6097 010344  162737  000002  002366                  SUB     #2,TESTADD+2            ;TRY AS 64K INTERLEAVED
6098 010352  062737  004000  172352                  ADD     #4000,KIPAR5            ;A1 NON-ASSERTED MEMORY
6099 010360  000137  007410                          JMP     4$                      ;TRY TO MATCH AGAIN
6100 010364  022737  000004  002424  18$:    CMP     #4,I                    ;4TH TIME THROUGH?
6101 010372  001404                                  BEQ     20$                     ;YES - BRANCH
6102 010374  022737  000002  002424                  CMP     #2,I                    ;2ND TIME THROUGH
6103 010402  103405                                  BLO     12$                     ;NO - BRANCH
6104 010404  062737  004000  172352  20$:    ADD     #4000,KIPAR5            ;TRY AS 128K INTERLEAVED
6105 010412  000137  007410                          JMP     4$                      ;TRY TO MATCH AGAIN
6106 010416  022737  000003  002424  12$:    CMP     #3,I                    ;THIRD TIME THROUGH?
6107 010424  103413                                  BLO     6$                      ;NO - BRANCH
6108 010426  062737  000002  002364                  ADD     #2,TESTADD              ;TRY TESTING THE BANK
6109 010434  062737  000002  002366                  ADD     #2,TESTADD+2            ;AS A1 ASSERTED
6110 010442  162737  004000  172352                  SUB     #4000,KIPAR5            ;64K INTERLEAVED MEMORY
6111 010450  000137  007410                          JMP     4$                      ;TRY TO MATCH AGAIN
6112                                                 ;*
6113                                                 ;*END OF BANK ROUTINE
6114                                                 ;*
6115 010454  104503                          6$:     CLR1CSR                         ;CLEAR THE CSR UNDER TEST
6116 010456  062702  000004                          ADD     #4,R2                   ;UPDATE CONFIGURATION POINTER
6117 010462  062737  001000  172350                  ADD     #1000,KIPAR4            ;UPDATE KIPAR4 TO NEXT BANK
6118 010470  013737  172350  172352                  MOV     KIPAR4,KIPAR5           ;AND UPDATE KIPAR5
```

```
6119 010476  000240                        NOP                    ;DEBUG AID
6120 010500  023737  002532  1?2350        CMP     LASTBLOCK,KIPAR4  ;HAVE WE DONE THE WHOLE MEMORY SPACE?
6121 010506  101402                        BLOS    19$            ;BRANCH IF DONE                    ;R C
6122 010510  000137  007354                JMP     45$            ;JUMP IF NOT DONE
6123 010514  062700  000002        19$:    ADD     #2,R0          ;INCREMENT CSR POINTER
6124 010520  000240                        NOP                    ;DEBUG AID
6125 010522  104423                        CACHON                 ;TURN ON THE CACHE
6126 010524  000137  007260                JMP     1$             ;JUMP TO TRY NEXT CSR
6127                              ;
6128 010530  062706  000004        100$:   ADD     #4,SP          ;RESTORE STACK                     ;R C
6129 010534  000137  010454                JMP     6$             ;GO TO END OF BANK ROUTINE         ;R C
```

```
6131 010540  104423                SUBAAS: CACHON          ;MAKE SURE THE CACHE IS ON
6132 010542  104472                        ECCINIT         ;TRAP ON DOUBLE BIT ERRORS (NORMAL)
6133 010544                                NEWTST  <<TEST BANK 0 ACCESSES>>
                                   ;*********************************************************************
                                   ;*TEST 2        TEST BANK 0 ACCESSES
                                   ;*********************************************************************
     010544  000004                TST2:   SCOPE
6134                               ;THIS DOES A "TST" INSTRUCTION ON EVERY LOCATION IN BANK #0 TO SEE
6135                               ;IF IT GETS ANY PARITY TRAPS.
6136                               ;SINCE EVERY LOCATION IS EITHER LOADED OR WRITTEN INTO BY THE PROGRAM
6137                               ;PRIOR TO THIS POINT  THEN A PARITY ERROR IMPLIES THAT THERE IS A
6138                               ;HARDWARE FAILURE IN THE MEMORY.
6139                               ;THESE ERRORS ARE COUNTED AND A FATAL ACTION IS TAKEN
6140 010546  005037  002070                CLR     PARCNT                  ;CLEAR PARITY ERROR COUNTER
6141 010552  012737  000001  002074        MOV     #1,NOPAR                ;SET THE NO PARITY ERROR FLAG
6142 010560  005037  002066                CLR     NEMCNT                  ;CLEAR NON-EXISTANT MEMORY ERROR COUNTER
6143 010564  012737  000001  002076        MOV     #1,NONEM                ;SET THE NON-EXISTANT MEMORY ERROR MODE TO COUNT
6144 010572                                SET4    #NONEXIST               ;TRAPS TO 4 GOTO NONEXIST
6145 010600  005000                        CLR     R0
6146 010602  012701  040000                MOV     #SIZE,R1
6147 010606  104424                        CACHOFF                         ;TURN CACHE OFF
6148 010610  005720                1$:     TST     (R0)+                   ;SEE IF I CAN DO A READ ACCESS WITHOUT A PARITY TRAP
6149 010612  077102                        SOB     R1,1$
6150 010614  104423                        CACHON                          ;TURN CACHE ON
6151                                       ;SEE IF ANY FAILURES
6152 010616  005737  002070                TST     PARCNT                  ;ANY PARITY ERRORS?
6153 010622  001403                        BEQ     2$                      ;NO   SKIP
6154 010624                                FATAL   3
6155 010632  005737  002066        2$:     TST     NEMCNT                  ;ANY NON-EXISTANT MEMORY (HOLES)?
6156 010636  001406                        BEQ     3$                      ;SKIP IF EQUAL
6157 010640  162737  000002  002032        SUB     #2,ADDRESS              ;UPDATE 1ST ADDRESS FAILURE FROM AUTO INCREMENT #
6158 010646                                FATAL   4
6159 010654  053737  002104  002626  3$:     BIS     CPUBIT,CONFIG           ;SET CORRECT ACCESSED BIT ON BANK C
6160 010662                                RES4                            ;RESET TRAPS TO 4 TO DEFAULT
6161
6162 010704                                SUBTST  <<ENABLE ECC FOR CORRECT TRAPS>>
                                   ;*********************************************************************
                                   ;*SUBTEST       ENABLE ECC FOR CORRECT TRAPS
                                   ;*********************************************************************
6163 010704                                IF #SW0 SET,IN @SWR OR ACTFLAG IS TRUE
6164 010722  104506                        ENASBE                          ;TRAP ON SINGLE BIT ERRORS
6165 010724                                ELSE
6166 010726  104472                        ECCINIT                         ;TRAP ON DOUBLE BIT ERRORS (NORMAL)
6167 010730                                END ;OF IF #SW0
```

```
6170 010730                                    NEWTST  <<TEST BANKS 1 200 (OCTAL) FOR ZEROS & ONES>>
                                       ;*****************************************************************
                                       ;*TEST 3        TEST BANKS 1-200 (OCTAL) FOR ZEROS & ONES
                                       ;*****************************************************************
     010730  000004                    TST3:   SCOPE
6171                                           ;EACH BANK IS TESTED FOR EXISTANCE AND IF IT EXISTS
6172                                           ;THEN IS IS TESTED FOR ZEROS & ONES.
6173                                           ;EXCEPT -
6174                                           ;       PROTECTED BANKS (WHERE THE PROGRAM IS) ARE ONLY TESTED BY
6175                                           ;       "TST" INSTRUCTIONS LIKE BANK #0
6176                                           ;ANY BAD BANKS ARE LOGGED IN THE CONFIGURATION TABLE.
6177                                           ;THIS ROUTINE IS ONLY DOING A SMART SIZE   NOT ACTUAL TESTING!
6178 010732  005037  002100                   CLR     BANK
6179 010736  012737  000001  002074           MOV     #1,NOPAR                ;SET NO PARITY ERROR FLAG
6180 010744  012737  000002  002076           MOV     #2,NONEM                ;SET NON-EXISTANT MEMORY MODE TO EXIT TEST LOOP
6181 010752                                    SET4    #NONEXIST               ;TRAPS TO 4 GOTO NONEXIST
6182 010760  022737  000001  003720           CMP     #1,PROTYP               ;IS THIS AN 11/44?
6183 010766  001407                           BEQ     1$                      ;BRANCH IF TRUE
6184 010770  012737  011566  002474           MOV     #MTST3+4,LINK1          ;SET UP LINKS
6185 010776  012737  011570  002476           MOV     #MTST3+6,LINK2
6186 011004  000411                           BR      TAG9$
6187 011006                            1$:     BMOV    MTST3                   ;PUT IN FAST MEMORY
6188 011014  012737  177644  002474           MOV     #UIPAR2,LINK1           ;SET UP LINKS
6189 011022  012737  177646  002476           MOV     #UIPAR3,LINK2
6190 011030  005237  002100            TAG9$:  INC     BANK
6191 011034  023737  002530  002100           CMP     LASTBANK,BANK           ;DONE?
6192 011042  103457                           BLO     TAG2$                   ;YES   SKIP TO NEXT TEST
6193 011044  013701  002100                   MOV     BANK,R1
6194 011050  006301                           ASL     R1
6195 011052  006301                           ASL     R1                      ;BANK * 4
6196 011054  010137  002102                   MOV     R1,BANKINDEX
6197 011060  005037  002072                   CLR     PATERR                  ;CLEAR PATTERN ERROR COUNTER
6198 011064  005037  002070                   CLR     PARCNT                  ;CLEAR PARITY ERROR COUNTER
6199 011070  005037  002066                   CLR     NEMCNT                  ;CLEAR NON-EXISTANT MEMORY COUNTER (HOLES)
6200 011074                                    MAP     BANK                    ;MAP SUPERVISOR SPACE (TEST AREA) TO BANK
6201 011110  105761  002626                   TSTB    CONFIG(R1)              ;IS THIS BANK PROTECTED?
6202 011114  100555                           BMI     TSTBANK                 ;YES - GO TEST BANK SPECIAL
6203 011116  012777  000207  171350  WARN1:   MOV     #207,@LINK1             ;PUT "RETURN" INSTRUCTION AFTER WRITE ROUTINE
6204 011124  012700  060000                   MOV     #FIRST,R0
6205 011130  010004                           MOV     R0,R4
6206 011132  012701  040000                   MOV     #SIZE,R1
6207 011136  010103                           MOV     R1,R3
6208 011140  005002                           CLR     R2                      ;DATA IS ZEROS
6209 011142  104424                           CACHOFF                         ;TURN CACHE OFF
6210 011144                                    TESTAREA                        ;ENTER SUPERVISOR MODE
6211 011152  022737  000001  003720           CMP     #1,PROTYP               ;IS THIS AN 11/44?
6212 011160  001403                           BEQ     1$                      ;BRANCH IF TRUE
6213 011162  004737  011562                   CALL    MTST3
6214 011166  000402                           BR      2$
6215 011170  004737  177640            1$:    CALL    FASTCITY                ;CALL TO THE USER INSTRUCTION PAR'S
6216 011174  104417                    2$:    KERNEL                          ;ENTER KERNEL MODE
6217 011176  104423                           CACHON                          ;TURN CACHE ON
6218 011200  000416                           BR      TAG3$                   ;SKIP NEXT INSTRUCTION
6219 011202  005037  002100            TAG2$:  CLR     BANK
6220 011206                                    RES4                            ;RESET TRAPS TO 4 TO DEFAULT
6221 011230  005037  002074                   CLR     NOPAR                   ;INDICATE DEFAULT PARITY ACTION
6222 011234  000564                           BR      SUBAAI
```

```
6223 011236  005737  002066          TAG3$:  TST     NEMCNT          ;ANY TRAPS?
6224 011242  001401                          BEQ     1$              ;NO   SKIP
6225 011244  000671                          BR      TAG9$           ;NOW   TRY NEXT BANK
6226 011246  104424                  1$:     CACHOFF                 ;TURN CACHE OFF
6227 011250                                  TESTAREA                ;ENTER SUPERVISOR MODE
6228 011256  004777  171214                  CALL    @LINK2          ;FINISH PATTERN
6229 011262  104417                          KERNEL                  ;ENTER KERNEL MODE
6230 011264  104423                          CACHON                  ;TURN CACHE ON
6231 011266  005737  002072                  TST     PATERR          ;ANY PATTERN ERRORS
6232 011272  001040                          BNE     2$              ;YES   SKIP
6233 011274  005737  002070                  TST     PARCNT          ;ANY PARITY ERRORS
6234 011300  001035                          BNE     2$              ;YES - SKIP
6235 011302  005737  002066                  TST     NEMCNT          ;ANY NON EXISTANT MEMORY
6236 011306  001032                          BNE     2$              ;YES - SKIP
6237 011310  012700  060000                  MOV     @FIRST,R0
6238 011314  010004                          MOV     R0,R4
6239 011316  012701  040000                  MOV     @SIZE,R1
6240 011322  010103                          MOV     R1,R3
6241 011324  013702  002556                  MOV     ONES,R2         ;DATA IS ONES
6242 011330  012777  000240  171136          MOV     @000240,@LINK1  ;PUT "NOP" INSTRUCTION BACK IN SUBROUTINE
6243 011336  104424                          CACHOFF                 ;TURN CACHE OFF
6244 011340                                  TESTAREA                ;ENTER TEST MODE
6245 011346  022737  000001  003720          CMP     @1,PROTYP       ;IS THIS AN 11/44?
6246 011354  001403                          BEQ     5$              ;BRANCH IF IT IS
6247 011356  004737  011562                  CALL    MTST3           ;DO IN MEMORY IF NOT
6248 011362  000402                          BR      6$              ;JUMP OVER NEXT INSTRUCTION
6249 011364  004737  177640          5$:     CALL    FASTCITY        ;CALL TO THE USER INSTRUCTION PAR S
6250 011370  104417                  6$:     KERNEL                  ;ENTER KERNEL MODE
6251 011372  104423                          CACHON                  ;TURN CACHE ON
6252 011374  013700  002102          2$:     MOV     BANKINDEX,R0
6253 011400  005737  002072                  TST     PATERR          ;ANY PATTERN ERRORS?
6254 011404  001006                          BNE     3$              ;YES - SKIP
6255 011406  005737  002070                  TST     PARCNT          ;ANY PARITY ERRORS?
6256 011412  001003                          BNE     3$              ;YES - SKIP
6257 011414  005737  002066                  TST     NEMCNT          ;ANY HOLES?
6258 011420  001406                          BEQ     4$              ;NONE - SKIP
6259 011422  052760  000001  002626  3$:     BIS     @BIT0,CONFIG(R0)  ;SET ERROR BIT IN THIS BANK
6260 011430                                  SET     CONFGERROR      ;FORCE PRINTING OF CONFIGURATION TABLE
6261 011436  053760  002104  002626  4$:     BIS     CPUBIT,CONFIG(R0)  ;SET ACCESSED BIT
6262 011444  000137  011030                  JMP     TAG9$
6263
6264                                         ;TEST A PROTECTED BANK
6265 011450                          TSTBANK:PUSH    R1
6266 011452  012737  000001  002076          MOV     @1,NONEM        ;SET NON-EXISTANT MEMORY TO COUNT
6267 011460  012700  060000                  MOV     @FIRST,R0
6268 011464  012701  020000                  MOV     @20000,R1
6269 011470  104424                          CACHOFF                 ;TURN CACHE OFF
6270 011472                                  TESTAREA                ;ENTER TEST MODE
6271 011500  005720                  4$:     TST     (R0)+
6272 011502  077102                          SOB     R1,4$
6273 011504  104417                          KERNEL                  ;ENTER KERNEL MODE
6274 011506  104423                          CACHON                  ;TURN CACHE ON
6275 011510  012737  000002  002076          MOV     @2,NONEM        ;RESET NON EXISTANT MEMORY TO EXIT TEST LOOP
6276 011516                                  POP     R1
6277 011520                                  IF PARCNT NE @0
6278 011526  052761  000001  002626            BIS   @BIT0,CONFIG(R1)  ;ERROR BANK
6279 011534                                    SET   CONFGERROR
```

```
6280 011542                                          END ;OF IF PARCNT
6281 011542                                          IF NEMCNT EQ #0
6282 011550  053761  002104  002626                    BIS   CPUBIT,CONFIG(R1)      ;ACCESSED BANK
6283 011556                                          END ;OF IF NEMCNT
6284 011556  000137  011030                          JMP   TAG9$
6285 011562  010220                        MTST3:    MOV   R2,(R0)+               ;V177640
6286 011564  077102                                  SOB   R1,MTST3               ;V177642
6287 011566  000240                                  NOP                          ;V177644
6288 011570  012401                        2$:       MOV   (R4)+,R1               ;V177646
6289 011572  020102                                  CMP   R1,R2                  ;V177650
6290 011574  001402                                  BEQ   3$                     ;V177652
6291 011576  005237  002072                          INC   PATERR                 ;V177654
6292 011602  077306                        3$:       SOB   R3,2$                  ;V177660
6293 011604  000207                                  RETURN                       ;V177662
```

```
6295 011606                          SUBAAI: SUBTST  <<FIND SHADOW INHIBIT MODE POINTERS>>
                                     ;*****************************************************************************
                                     ;*SUBTEST       FIND SHADOW INHIBIT MODE POINTERS
                                     ;*****************************************************************************
6296                                 ;* THIS SECTION LOOKS FOR INTERLEAVED MS11-M MEMORIES AND FIGURES OUT
6297                                 ;* WHERE THE SHADOW INHIBIT MODE POINTERS ARE LOCATED.  THESE AREAS
6298                                 ;* ARE THEN MARKED AS PROGRAM SPACE.
6299 011606  005037  002100                CLR     BANK                    ;RESET BANK TO ZERO
6300 011612  004737  044302          SHADL1: CALL    EXBANK                  ;SET BANK PARAMETERS
6301 011616  013700  002102                MOV     BANKINDEX,R0
6302 011622                                IF ACFLAG IS TRUE AND INTFLAG IS TRUE
6303 011636                                  IF INT64K IS TRUE
6304 011644  062700  000020                    ADD #20,R0                  ;POINT TO BANKINDEX * 4
6305 011650  062737  000010  002100            ADD #10,BANK                ;POINT TO BANK * 8
6306 011656                                  ELSE
6307 011660  062702  000040                    ADD #40,R2                  ;POINT TO BANKINDEX * 8
6308 011664  062737  000020  002100            ADD #20,BANK                ;POINT TO BANK * 16
6309 011672                                  END; OF IF INT64K
6310 011672  052760  000200  002626          BIS   #BIT7,CONFIG(R0)        ;MAKE NEW BANK PROGRAM SPACE
6311 011700                                ELSE
6312 011702  005237  002100                  INC   BANK                    ;GO TO NEXT BANK
6313 011706                                END; OF IF ACFLAG
6314 011706  023737  002530  002100          CMP     LASTBANK,BANK         ;HAVE WE DONE ALL THE BANKS?
6315 011714  002336                        BGE     SHADL1                  ;BRANCH IF NOT
```

```
6318 011716                                     NEWTST  <<ECC INHIBIT MODE POINTER TEST>>
                                        ;*****************************************************************
                                        ;*TEST 4        ECC INHIBIT MODE POINTER TEST
                                        ;*****************************************************************
     011716  000004                     TST4:   SCOPE
6319                                            ;THE MS11-M OR MF11S K INHIBIT ECC DISABLE AND DIAGNOSTIC CHECK MODE
6320                                            ;ON THE BOTTOM  FIRST OR SECOND 16K WORDS CONTROLLED BY A CSR.  THIS
6321                                            ;IS CONSIDERED  TO BE A  PROTECTED BANK  BY THE  PROGRAM.  IT MAY BE
6322                                            ;QUITE  COMPLEX TO DETERMINE ON A  GIVEN SYSTEM  CONFIGURATION WHICH
6323                                            ;BANKS CAN BE PROTECTED;
6324                                            ;SO
6325                                            ;THIS ROUTINE ATTEMPS TO  CREATE A DOUBLE BIT ERROR IN ADDRESS 0 & 2
6326                                            ;OF  EVERY ECC BANK.  ECC HARDWARE WILL PREVENT THIS  FROM HAPPENING
6327                                            ;IN  PROTECTED BANKS  WHICH SHOULD ALWAYS  INCLUDE BANK ZERO   WHERE
6328                                            ;THE PROGRAM IS.
6329                                            ;
6330                                            ;
6331                                            ;WARNING:!!!!!!!!!!
6332                                            ;  IN CASE OF HARDWARE FAILURE IT IS COMMON  THAT A DOUBLE BIT ERROR
6333                                            ;  WILL BE  CREATED ON  THE KERNEL  STACK & "CRASH"  THE  DIAGNOSTIC
6334                                            ;DURING  THIS ROUTINE.  YOUR ONLY CLUE IS THAT YOU CAN GET AS FAR AS
6335                                            ;THIS ROUTINE BUT NOT PAST IT!
6336 011720  104424                            CACHOFF                                 ;TURN CACHE OFF
6337 011722  012737  177777  002152            MOV     #-1,OLDCSR
6338 011730                                     FOR BANK := #0 TO LASTBANK
6339 011734  012701  060000                      MOV    #FIRST,R1                       ;SET UP VIRT ADDR POINTER
6340 011740  004737  044302                      CALL   EXBANK
6341 011744  013700  002102                      MOV    BANKINDEX,R0
6342 011750                                      IF ACFLAG IS TRUE
6343 011756                                        IF MKFLAG IS TRUE
6344 011764                                        IF SKIPMK IS FALSE
6345 011772                                     IF INTFLAG IS TRUE
6346 012000  012703  040000                      MOV     #40000,R3                      ;SET INDEX COUNTER
6347 012004  012737  000001  002234              MOV     #1,SPLTCSR                     ;MAP AS INTERLEAVED BANK
6348 012012                                     ELSE
6349 012014  012703  000002                      MOV     #2,R3                          ;SET INDEX COUNTER
6350 012020                                     END; OF IF INTFLAG
6351 012020  116002  002627                     MOVB    CONFIG+1(R0),R2
6352 012024  006302                             ASL     R2
6353 012026  042702  177741                     BIC     #^C36,R2
6354 012032  010237  002146                     MOV     R2,CSRNO
6355 012036                                     IF CSRNO NE OLDCSR
6356 012046  013737  002146  002152              MOV     CSRNO,OLDCSR
6357 012054                                      IF PFLAG IS FALSE
6358 012062  052760  000100  002626                BIS  #BIT6,CONFIG(R0)
6359 012070                                      END; OF IF PFLAG
6360 012070  004737  012224                      CALL    IMPTEST
```

```
6362 012074                                          IF INTFLAG IS TRUE
6363 012102  116002  002627                            MOVB  CONFIG+1(R0),R2
6364 012106  072227  177775                            ASH   #-3,R2
6365 012112  042702  177741                            BIC   #^C36,R2
6366 012116  010237  002146                            MOV   R2,CSRNO
6367 012122  062701  000002                            ADD   #2,R1            ;FIX POINTER FOR A1 ASSERTED HALF
6368 012126  004737  012224                            CALL  IMPTEST
6369 012132  005037  002234                            CLR   SPLTCSR
6370 012136                                          END; OF IF INTFLAG
6371 012136                                        END; OF IF CSRNO
6372 012136                                      END; OF IF SKIPMK
6373 012136                                      END; OF IF MKFLAG
6374 012136                                    END; OF IF ACFLAG
6375 012136                                  END; OF FOR BANK
6376 012152                                  MAP                       ;MAP TEST SPACE TO BANK 0
6377 012166  005037  002100                  CLR     BANK
6378 012172                                  IF @SW0 SET.IN @SWR OR ACTFLAG IS TRUE
6379 012210  104506                            ENASBE                  ;TRAP ON SINGLE BIT ERRORS
6380 012212                                  ELSE
6381 012214  104472                            ECCINIT                 ;TRAP ON DOUBLE BIT ERRORS (NORMAL)
6382 012216                                  END; OF IF @SW0
6383 012216  104423                          CACHON                    ;TURN THE CACHE BACK ON
6384 012220  000137  012466                  JMP     SUBAAR            ;JUMP OVER THE SUBROUTINE
```

N12


```
6386 012224  005004                  IMPTEST:CLR     R4
6387 012226                                  MAP BANK                                ;MAP SUPERVISOR SPACE (TEST AREA) TO BANK
6388 012242  005005                          CLR     R5
6389 012244  012737  020000  002144          MOV     @BIT13,CSR
6390 012252                                  TESTAREA                                ;ENTER TEST MODE
6391 012260                                  PUSH    (R1)                            ;SAVE TEST LOCATION
6392 012262  060301                          ADD     R3,R1                   ;INDEX TO NEXT LOCATION
6393 012264                                  PUSH    (R1)                    ;SAVE TEST LOCATION
6394 012266  104505                          CHK1DIS                         ;DISABLE ECC & WRITE CHECKBITS FOR 1 CSR
6395 012270  010411                          MOV     R4,(R1)                 ;WRITE CHECKBITS (ALL ZEROS)
6396 012272  160301                          SUB     R3,R1
6397 012274  010411                          MOV     R4,(R1)
6398 012276  104503                          CLR1CSR                                 ;CLEAR CSR
6399 012300  005711                          TST     (R1)                            ;READ CHECKBITS INTO REAL CSR
6400 012302  104501                          WAS1DBE                                 ;WAS THERE A DOUBLE BIT ERROR
6401
6402                                         ;THIS MAKES SURE THAT SBE'S DON T LOOK LIKE PROTECTED AREAS
6403
6404 012304                                  ON.NOERROR ;1
6405 012306  012737  020000  002144            MOV   @BIT13,CSR
6406 012314  104505                            CHK1DIS                               ;DISABLE ECC & WRITE CHECKBITS FOR 1 CSR
6407 012316  013711  002556                    MOV   ONES,(R1)
6408 012322  060301                            ADD   R3,R1
6409 012324  013711  002556                    MOV   ONES,(R1)
6410 012330  160301                            SUB   R3,R1
6411 012332  104503                            CLR1CSR                               ;CLEAR CSR
6412 012334  005711                            TST   (R1)
6413 012336  104501                            WAS1DBE                               ;WAS THERE A DOUBLE BIT ERROR
6414 012340                                    ON.NOERROR ;2
6415 012342  012737  027400  002144              MOV #27400,CSR
6416 012350  104505                              CHK1DIS                                     ;DISABLE ECC & WRITE CHECKBITS FOR 1 CSR
6417 012352  010411                              MOV R4,(R1)
6418 012354  060301                              ADD R3,R1                           ;ADD INDEX TO GET TO SECOND WORD
6419 012356  010411                              MOV R4,(R1)
6420 012360  160301                              SUB R3,R1                           ;SUBTRACT INDEX TO FIRST WORD
6421 012362  104503                              CLR1CSR                                     ;CLEAR CSR
6422 012364  005711                              TST (R1)
6423 012366  104501                              WAS1DBE                                     ;WAS THERE A DOUBLE BIT ERROR
6424 012370                                      ON.NOERROR ;3
6425 012372  012737  074000  002144                MOV       @74000,CSR
6426 012400  104505                                CHK1DIS                                   ;DISABLE ECC & WRITE CHECKBITS FOR 1 CSR
6427 012402  010411                                MOV       R4,(R1)
6428 012404  060301                                ADD       R3,R1               ;INDEX TO SECOND WORD
6429 012406  010411                                MOV       R4,(R1)
6430 012410  104503                                CLR1CSR                               ;CLEAR CSR
6431 012412  160301                                SUB       R3,R1               ;GO BACK TO FIRST WORD
6432 012414  005711                                TST       (R1)
6433 012416  104501                                WAS1DBE                               ;WAS THERE A DOUBLE BIT ERROR
6434 012420                                      END ;OF ON.NOERROR ;3
6435 012420                                    END ;OF ON.NOERROR ;2
6436 012420                                  END ;OF ON.NOERROR ;1
6437 012420                                  ON.ERROR
6438 012422  005205                            INC   R5                                  ;IDENTIFY AS BAD BANK
6439 012424                                  END ;OF ON.ERROR
6440 012424  104471                          ECC1DIS                                 ;DISABLE ERROR CORRECTION
6441 012426  010411                          MOV     R4,(R1)                 ;CLEAR OUT DOUBLE BIT ERROR!
6442 012430  060301                          ADD     R3,R1                   ,INDEX TO SECOND WORD
```

```
6443 012432 010411                      MOV     R4,(R1)             ;CLEAR OUT DOUBLE BIT ERROR!
6444 012434 104503                      CLR1CSR
6445 012436 005705                      TST     R5
6446 012440 001405                      BEQ     1$
6447 012442 050560 002626               BIS     R5,CONFIG(R0)
6448 012446 105260 002630               INCB    CONFIG+2(R0)
6449 012452 104036                      ERROR   +36
6450 012454                         1$: POP     (R1)                ;RESTORE TEST LOCATION (2ND WORD)
6451 012456 160301                      SUB     R3,R1               ;GO BACK TO FIRST WORD
6452 012460                             POP     (R1)                ;RESTORE TEST LOCATION (1ST WORD)
6453 012462 104417                      KERNEL
6454 012464 000207                      RETURN
```

```
6798
6799 012466                    SUBAAR: SET     STOPOK          ;PROGRAM CAN NOW BE HA.TED
```

```
6802 012474                                                     SUBTST  <<LEGAL CONFIGURATION CHECK>>
                                                        ;*****************************************************************************
                                                        ;*SUBTEST       LEGAL CONFIGURATION CHECK
                                                        ;*****************************************************************************
6803 012474  012700  000020                             MOV     #16.,R0
6804 012500  012701  002434                             MOV     #CSRINFO,R1
6805 012504  005021                                     1$:     CLR     (R1)+
6806 012506  077002                                             SOB     R0,1$
6810 012510                                                       FOR BANK := #0 TO LASTBANK
6811 012514  004737  044302                                        CALL        EXBANK
6812 012520  013700  002102                                        MOV BANKINDEX,R0
6834
6835 012524                                                        IF ACFLAG IS TRUE
6836 012532  116003  002627                                         MOVB       CONFIG+1(R0),R3
6837 012536  042703  177760                                         BIC        #^C17,R3
6838 012542  006303                                                 ASL        R3
6839 012544  005263  002434                                         INC        CSRINFO(R3)
6840 012550                                                         IF MKFLAG IS TRUE
6841                                                                ;MAKE SURE THAT EACH BANK HAS NO MORE THAN 2 CSRS
6842 012556                                                           BEGIN   LEGALCSR
6843 012556                                                             IF INTFLAG IS TRUE
6844 012564  116003  002627                                               MOVB        CONFIG+1(R0),R3
6845 012570  010304                                                     MOV R3,R4
6846 012572  042703  177760                                           BIC #^C17,R3
6847 012576  072427  177774                                           ASH #-4,R4
6848 012602  042704  177760                                           BIC #^C17,R4
6849 012606                                                           IF R3 EQ R4
6850 012612  042760  014000  002630                                     BIC        #BIT11!BIT12,CONFIG+2(R0)
6851 012620  042760  170000  002626                                     BIC        #170000,CONFIG(R0)
6852 012626                                                             LEAVE LEGALCSR
6853 012630                                                           END; OF IF R3
6854 012630                                                           IF KFLAG IS FALSE
6855 012636                                                             LEAVE     LEGALCSR
6856 012640                                                           END; OF IF KFLAG
6857 012640  006303                                                   ASL R3
6858 012642  006304                                                   ASL R4
6859 012644  005263  002434                                           INC CSRINFO(R3)
6860 012650  005264  002434                                           INC CSRINFO(R4)
6861 012654                                                         ELSE
6862 012656                                                           LEAVE       LEGALCSR
6863 012660                                                         END; OF IF INTFLAG
6864 01266                                                        TYPE    MSG124                          ;# OF CSR'S IS WRONG
6865 012664                                                       TYPOCS  BANK,<TYPE BANK #>,3
6866 012674                                                       SET     CONFGERROR
6867 012702                                                       END     LEGALCSR
6868 012702                                                     END ;OF IF MKFLAG
6869 012702                                                   END ;OF IF ACFLAG
6870 012702                                                 END; OF FOR BANK
6871 012716                                             PUSH    R5,R0                                   ;SAVE CONTENTS OF R5, R0
6872 012722  005000                                     CLR     R0                                      ;CLEAR REGISTERS
6873 012724  005001                                     CLR     R1
6874 012726  005005                                     CLR     R5
6875 012730  005037  013136                             CLR     @#BERR                                  ;CLEAR ERROR INDICATOR
6876 012734  022761  000010  002434  2$:                CMP     #10.,CSRINFO(R1                         ;IS CURRENT CSR <= 10
6877 012742  002043                                     BGE     5$                                      ;BRANCH IF SO
6878 012744  022761  000020  002434                     CMP     #20.,CSRINFO(R1)                        ;IS CURRENT CSR < 20
6879 012752  002003                                     BGE     3$                                      ;BRANCH IF SO
```

```
6880 012754 004737 013260          CALL    ILLCSR          ;CALL ERROR ROUTINE
6881 012760 000434                 BR      5$              ;TRY NEXT CSR
6882 012762 016005 002626    3$:   MOV     CONFIG(R0),R5   ;MOVE LOW WORD TO R5
6883 012766 032705 000002          BIT     #BIT1,R5        ;DOES MEMORY EXIST HERE?
6884 012772 001415                 BEQ     4$              ;BRANCH IF NOT
6885 012774 042705 170377          BIC     #^C7400,R5      ;ISOLATE CSR NUMBER IN
6886 013000 072527 177771          ASH     #-7,R5          ;REGISTER 5
6887 013004 020501                 CMP     R5,R1           ;IS IT THE CURRENT CSR?
6888 013006 001007                 BNE     4$              ;TRY NEXT WORD OF CONFIG IF NOT
6889 013010 032760 010000 002630   BIT     #BIT12,CONFIG+2(R0) ;IS IT INTERLEAVED?
6890 013016 001003                 BNE     4$              ;BRANCH IF SO
6891 013020 012737 000001 013136   MOV     #1,MBERR        ;SET ERROR INDICATOR
6892 013026 062700 000004    4$:   ADD     #4,R0           ;UPDATE CONFIG COUNTER
6893 013032 022700 000340          CMP     #340,R0         ;CONFIG TABLE ALL DONE?
6894 013036 001351                 BNE     3$              ;BRANCH IF NOT
6895 013040 005737 013136          TST     MBERR           ;ERRORS FOUND?
6896 013044 001402                 BEQ     5$              ;TRY NEXT CSR IF NOT
6897 013046 004737 013260          CALL    ILLCSR          ;CALL ERROR ROUTINE
6898 013052 005000           5$:   CLR     R0              ;REINITIALIZE CONFIG COUNTER
6899 013054 005037 013136          CLR     MBERR           ;CLEAR ERROR INDICATOR
6900 013060 062701 000002          ADD     #2,R1           ;UPDATE CSR COUNTER
6901 013064 022701 000040          CMP     #40,R1          ;ALL CSR'S DONE?
6902 013070 001321                 BNE     2$              ;BRANCH IF NOT
6903 013072                        POP     R0,R5           ;RESTORE REGISTERS
6904 013076 005037 013136          CLR     MBERR           ;RESET ERROR INDICATOR
6908 013102 012700 000734          MOV     #734,R0       ;INDEX TO TOP OF CONFIG TABLE   ;R C
6909 013106 032760 000002 002626 6$:   BIT     #BIT1,CONFIG(R0)  ;MEMORY PRESENT?          ;R C
6910 013114 001003                 BNE     7$              ;BRANCH IF SO                   ;R C
6911 013116 162700 000004          SUB     #4,R0           ;TRY NEXT LOWER ENTRY IN CONFIG TABLE   ;R C
6912 013122 000771                 BR      6$                                              ;R C
6913 013124 006200           7$:   ASR     R0                                              ;R C
6914 013126 006200                 ASR     R0              ;DIVIDE INDEX BY 4 TO GET BANK #;R C
6915 013130 010037 002530          MOV     R0,LASTBANK     ;STORE IN LASTBANK              ;R C
6916 013134 000402                 BR      SKUJ
6917 013136 000000           MBERR: .WORD 0                 ;SAVE SPACE FOR ERROR INDICATOR
6918 013140 000000           PHEBE: .WORD 0                 ;SAVE SPACE FOR ODD BOUNDARY INTERLEAVED INDICATOR
6919 013142 005000           SKUJ:  CLR     R0              ;CLEAR CONFIG COUNTER
6920 013144 005037 013140          CLR     PHEBE           ;CLEAR COUNTER
6921 013150 032760 000002 002626 1$:   BIT     #BIT1,CONFIG(R0)  ;IS THERE MEMORY PRESENT?
6922 013156 001431                 BEQ     3$              ;BRANCH IF NOT
6923 013160 032760 010000 002630   BIT     #BIT12,CONFIG+2(R0) ;IS IT INTERLEAVED?
6924 013166 001005                 BNE     2$              ;BRANCH IF SO
6925 013170 005237 013140          INC     PHEBE           ;INCREMENT COUNTER
6926 013174 062700 000004          ADD     #4,R0           ;INCREMENT CONFIG COUNTER
6927 013200 000763                 BR      1$              ;TRY NEXT BANK
6928 013202 023727 013140 000010 2$:   CMP     PHEBE,#10       ;IS THE COUNTER EQUAL TO...
6929 013210 001417                 BEQ     4$              ;ONE OF THE SPECIAL VALUES.
6930 013212 023727 013140 000030   CMP     PHEBE,#30       ;IF IT IS...
6931 013220 001413                 BEQ     4$              ;BRANCH TO 4$
6932 013222 023727 013140 000050   CMP     PHEBE,#50
6933 013230 001407                 BEQ     4$
6934 013232 023727 013140 000070   CMP     PHEBE,#70
6935 013240 001403                 BEQ     4$
6936 013242 005037 013140    3$:   CLR     PHEBE           ;CLEAR INDICATOR
6937 013246 000403                 BR      5$
6938 013250 012737 000001 013140 4$:   MOV     #1,PHEBE        ;SET INDICATOR
6939 013256 000421           5$:   BR      SUBAAP          ;BRANCH TO NEXT SUBTEST
```

```
6940 013260  010102          I.LCSR: MOV     R1,R2           ;R2 HAS CSR NUMBER
6941 013262  006202                  ASR     R2              ;MAKE ACCEPTABLE FOR PRINTING
6942 013264  022702  000012          CMP     #10.,R2
6943 013270  100002                  BPL     1$
6944 013272  062702  000007          ADD     #7,R2
6945 013276  062702  000060  1$:     ADD     #60,R2
6946 013302  110237  075377          MOVB    R2,MSGA122      ;PUT NUMBER INTO ERROR MESSAGE
6947 013306                          TYPE    MSG122
6948 013312                          SET     CONFGERROR
6949 013320  000207                  RETURN
```

```
6952 013322                                SUBAAP: SUBTST  <<PRINT CONFIGURATION DETAILS>>
                                           ;*****************************************************************
                                           ;*SUBTEST       PRINT CONFIGURATION DETAILS
                                           ;*****************************************************************
6953 013322                                        CLEAR   BSIZE,KSIZE,LSIZE,MSIZE,PSIZE
6954 013346  013702  002530                        MOV     LASTBANK,R2
6955 013352  006302                                ASL     R2
6956 013354  006302                                ASL     R2
6957 013356                                        FOR R1 := #0 TO R2 BY #4
6958 013360                                         IF CPUBIT SET.IN CONFIG(R1)
6959 013370                                           IF #BIT10 SET.IN CONFIG+2(R1)
6960 013400                                             IF #BIT8 SET.IN CONFIG+2(R1)
6961 013410                                               IF #BIT9 SET.IN CONFIG+2(R1)
6962 013420                                                 LET PSIZE := PSIZE + #1
6963 013424                                               ELSE
6964 013426                                                 LET KSIZE := KSIZE + #1
6965 013432                                               END;IF BIT9
6966 013432                                             ELSE
6967 013434                                               IF #BIT9 SET.IN CONFIG+2(R1)
6968 013444                                                 LET LSIZE := LSIZE + #1
6969 013450                                               ELSE
6970 013452                                                 LET MSIZE := MSIZE + #1
6971 013456                                               END; IF BIT9
6972 013456                                             END;IF BIT8
6973 013456                                           ELSE
6974 013460                                             IF #BIT9 SET.IN CONFIG+2(R1)
6975 013470                                               IF #BIT8 SET.IN CONFIG+2(R1)
6976 013500                                                 LET BSIZE := BSIZE + #1
6977 013504                                               END; OF IF #BIT8
6978 013504                                             END; OF IF #BIT9
6979 013504                                           END;IF BIT10
6980 013504                                         END; OF IF CPUBIT
6981 013504                                        END ;OF FOR ALL BANKS IN TABLE
6982
6983 013514  005037  002424                        CLR     I
6984 013520                                        FOR R1 := #0 TO #10 BY #2
6985 013522  006361  002346                          ASL   BSIZE(R1)
6986 013526  006361  002346                          ASL   BSIZE(R1)
6987 013532  006361  002346                          ASL   BSIZE(R1)
6988 013536  006361  002346                          ASL   BSIZE(R1)                ;BSIZE(R1) := BSIZE(R1) * 16.
6989 013542  066137  002346  002424                  ADD   BSIZE(R1),I              ;I <  I + BSIZE(R1)
6990 013550                                        END; FOR R1
6991 013562                                        FOR R1 := #0 TO #200 BY #4
6992 013564                                           IF CPUBIT SET.IN CONFIG(R1)
6993 013574                                             LET UNITOP := UNITOP   #1
6994 013600                                           END; OF IF CPUBIT
6995 013600                                        END; OF FOR R1
6996 013612  006337  002370                        ASL     UNITOP
6997 013616  006337  002370                        ASL     UNITOP
6998 013622  006337  002370                        ASL     UNITOP
6999 013626  006337  002370                        ASL     UNITOP                 ;UNITOP := UNITOP * 16.
7000 013632                                        IF I LT UNITOP THEN LET I := UNITOP
7001 013650                                        TYPE    $CRLF
7002 013654  005737  002346                        TST     BSIZE
7003 013660  001405                                BEQ     1$
7004 013662                                        TYPDEC  BSIZE
7005 013670                                        TYPE    MSG071
```

```
7006 013674  005737  002350          1$.     TST     KSIZE
7007 013700  001405                          BEQ     2$
7008 013702                                  TYPDEC  KSIZE
7009 013710                                  TYPE    MSG072
7010 013714  005737  002352          2$:     TST     LSIZE
7011 013720  001405                          BEQ     3$
7012 013722                                  TYPDEC  LSIZE
7013 013730                                  TYPE    MSG112
7014 013734  005737  002354          3$:     TST     MSIZE
7015 013740  001405                          BEQ     4$
7016 013742                                  TYPDEC  MSIZE
7017 013750                                  TYPE    MSG113
7018 013754  005737  002356          4$:     TST     PSIZE
7019 013760  001405                          BEQ     5$
7020 013762                                  TYPDEC  PSIZE
7021 013770                                  TYPE    MSG114
7022 013774                          5$:     TYPDEC  I
7023 014002                                  TYPE    MSG070
7024 014006                                  IF @SW6 OFF.IN @SWR
7025 014016  004737  036632                    CALL  PCONFIG
7026 014022                                  END; OF IF @SW6
```

```
7029 014022                                     SUBTST  <<CHECK APT SIZING>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       CHECK APT SIZING
                                        ;*****************************************************************************
7030 014022                                     IF APTFLAG IS TRUE AND APTSIZE IS TRUE
7031 014036  005037  002406                       CLR    TEMP
7032 014042  012700  063126                       MOV    @#MAMS1,R0
7033 014046                                       FOR R2 := #0 TO #4
7034 014050                                         IFB 1(R0) NE #0
7035 014056  111001                                   MOVB       (R0),R1
7036 014060  042701  177400                           BIC        #177400,R1
7037 014064                                           IF 2(R0) LT #0
7038 014072  000261                                     SEC
7039 014074                                           ELSE
7040 014076  000241                                     CLC
7041 014100                                           END ;OF IF 2(R0)
7042 014100  006101                                   ROL        R1
7043 014102  005201                                   INC        R1                ;TO COMPENSATE FOR 4 BANKS BEING (0 3,
7044 014104  006301                                   ASL        R1
7045 014106  006301                                   ASL        R1
7046 014110  006301                                   ASL        R1
7047 014112  006301                                   ASL        R1
7048 014114  163701  002406                           SUB        TEMP,R1
7049 014120  010137  002406                           MOV        R1,TEMP
7050 014124                                           IFB 1(R0) EQ #3
7051 014134  060137  002374                             ADD      R1,APTPAR
7052 014140                                           END ;OF IFB 1(R0)
7053 014140                                           IFB 1(R0) EQ #4
7054 014150  060137  002376                             ADD      R1,APTECC
7055 014154                                           END ;OF IFB 1(R0)
7056 014154  062700  000004                           ADD        #4,R0
7057 014160                                         END ;OF IFB 1(R0)
7058 014160                                       END ;OF FOR R2
7059 014170                                       IF APTPAR NE LSIZE OR APTECC NE MSIZE
7060 014210  104046                                 ERROR       .46
7061 014212                                       END ;OF IF APTPAR
7073 014212                                     END ;OF IF APTFLAG
```

```
7075 014212                          LOOP:   NEWTST  <<DIAGNOSTIC MODE DISPATCH ROUTINE>>
                                     ;*****************************************************************************
                                     ;*TEST 5          DIAGNOSTIC MODE DISPATCH ROUTINE
                                     ;*****************************************************************************
     014212  000004                  TST5:   SCOPE
7076 014214  005037  002216                  CLR     CONTFLAG
7077 014220  017700  166354                  MOV     @SWR,R0          ;GET SWITCHES
7078 014224  042700  177761                  BIC     #^C16,R0         ;MASK TO ONLY MODE BITS
7079 014230  004770  014240                  CALL    @DISPTBL(R0)     ;DISPATCH TO ROUTINE THROUGH NEXT TABLE
7080 014234  000137  014260                  JMP     MEMDONE          ;GO TO NEXT TEST
7081 014240  014712                  DISPTBL:BAFPAF  ;MODE 0;BANKS FORWARD, PATTERNS FORWARD
7082 014242  015020                          BAFPAR  ;MODE 1;BANKS FORWARD, PATTERNS REVERSE
7083 014244  015126                          BAWPAF  ;MODE 2;BANKS WORST FIRST, PATTERNS FORWARD
7084 014246  015256                          BAWPAR  ;MODE 3;BANKS WORST FIRST, PATTERNS REVERSE
7085 014250  015406                          PAFBAF  ;MODE 4;PATTERNS FORWARD, BANKS FORWARD
7086 014252  015536                          PAFBAW  ;MODE 5;PATTERNS FORWARD, BANKS WORST FIRST
7087 014254  015710                          PARBAF  ;MODE 6;PATTERNS REVERSE, BANKS FORWARD
7088 014256  016040                          PARBAW  ;MODE 7;PATTERNS REVERSE, BANKS WORST FIRST
7089
7090 014260  004737  014612          MEMDONE:CALL    DOBACK                          ;CHECK BACKGROUND PATTERN
7091
7092 014264                                  NEWTST<<UNIQUE BANK TEST>>
                                     ;*****************************************************************************
                                     ;*TEST 6          UNIQUE BANK TEST
                                     ;*****************************************************************************
     014264  000004                  TST6:   SCOPE
7093                                         ;MAKE SURE THAT EACH BANK CAN HAVE UNIQUE DATA
7094                                         ;WRITE AND READ THE BANK NUMBER IN EACH BANK (EXCEPT WHERE THE PROGRAM IS)
7095 014266                                  IF SELONLY IS FALSE
7096 014274                                    SET   HEADER,MUT
7097 014310  004737  024234                    CALL  MT0027
7098 014314                                    SET   HEADER
7099 014322  005037  002106                    CLR   MUT
7100 014326                                  END ;OF IF SELONLY
7101 014326  004737  014612                  CALL    DOBACK           ;RESTORE BACKROUND PATTERN
7105
7106 014332                          FLUSH:  SUBTST  <<FLUSH OUT DBE'S>>
                                     ;*****************************************************************************
                                     ;*SUBTEST         FLUSH OUT DBE S
                                     ;*****************************************************************************
7107 014332  004737  024720                  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
014336  005037  002414          $EOP:   CLR     FSINFLAG
014342  012700  002630                  MOV     #CONFIG+2,R0    ;MOVE 2ND WORD OF CONFIG TO R0
014346  042710  020000          1$:     BIC     #BIT13,(R0)     ;CLEAR BACKGROUND VALID BIT
014352  062700  000004                  ADD     #4,R0           ;INCREMENT TO NEXT BANK
014356  020027  003620                  CMP     R0,#3620        ;DONE?
014362  003771                          BLE     1$              ;NO   BRANCH
014364  013737  002572  002014          MOV     $ERTTL,LASTERROR
014372  005237  063104                  INC     $PASS           ;;INCREMENT THE PASS NUMBER
014376  042737  100000  063104          BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
014404                                  TYPE    MSG077          ;;TYPE "END PASS #'
014410                                  IF #SW11 SET.IN @SWR OR QVFLAG IS TRUE
014426                                    TYPE  MSG035          ;QV
014432  005037  002320                    CLR   QVFLAG
014436                                  END ;OF IF SW11
014436                                  TYPDEC  $PASS
014444  013700  000042                  MOV     42,R0           ;;GET MONITOR ADDRESS
014450  001456                          BEQ     $DOAGAIN        ;;BRANCH IF NO MONITOR
014452  022700  002000          $ZAP42: CMP     #STACK,R0       ;ARE WE UNDER RT11
014456  001453                          BEQ     $DOAGAIN        ;YES - BRANCH
                                        ;WE ARE UNDER (HEAVEN HELP US) XXDP!
014460                                  PUSH    R0
014462  004737  045170                  CALL    SHUTUP
014466                                  POP     R0
014470  000005                          RESET                   ;;CLEAR THE WORLD
014472  004710                  $ENDAD: CALL    (R0)            ;;GO TO MONITOR
014474  000240                          NOP                     ;;SAVE ROOM
014476  000240                          NOP                     ;;FOR
014500  000240                          NOP                     ;;ACT11
014502                          $DOAGN: ;UNDO SHUTUP STUFF
                                        ;       RESTORE STACK
                                        ;       ENERGIZE UNIBUS MAP & 22 BIT ADDRESSING
                                        ;       ENERGIZE MEMORY MANAGEMENT
                                        ;       PUT LOADERS BACK HOME
014502  013706  002536                  MOV     KSTACK,SP
014506  005737  002426                  TST     NO22BIT                 ;IS THIS AN 11/44 OR 11/24?
014512  001003                          BNE     1$
014514  052737  000060  172516          BIS     #BIT5!BIT4,MMR3
014522  104420                  1$:     ENERGIZE                        ;TURN ON MEMORY MANAGEMENT
014524  013700  002540                  MOV     LOADHOME,R0             ;DESTINATION BANK
014530  012701  000001                  MOV     #1,R1                   ;SOURCE BANK
014534  004737  043752                  CALL    BANKMOV
014540                                  IF APTFLAG IS TRUE
014546                                    IF $USWR EQ $PASS
014556  012701  000050          APTHANG:    MOV #50,R1
014562  077001                  2$:         SOB R0,2$
014564  062737  000001  063106            ADD #1,$DEVCT
014572  005537  063110                    ADC $UNIT
014576  077107                            SOB R1,2$
014600  005237  063104                    INC $PASS
014604  000764                            BR  APTHANG
```

END OF PASS ROUTINE

```
 7167 014606                                    END ;OF IF $USWR
 7168 014606                                  END ;OF IF APTFLAG
 7169 014606  000137  014212          $DOAGAIN: JMP   LOOP              ;RETURN
```

```
END OF PASS ROUTINE

 7172 014612                        DOBACK: SUBTST  <<WRITE BACKGROUND PATTERNS>>
                                    ;*****************************************************************************
                                    ;*SUBTEST       WRITE BACKGROUND PATTERNS
                                    ;*****************************************************************************
 7173 014612  005037  002110                CLR     PATTERN
 7174 014616                                FOR BANK := #0 TO LASTBANK
 7175 014622  004737  044302                  CALL  EXBANK
 7176 014626                                  IF ACFLAG IS TRUE AND RRFLAG IS FALSE
 7177 014642                                    SET         HEADER,MUT
 7178 014656  004737  017520                   CALL        MKTEST          ;CALL   MJTEST WOULD ALSO WORK
 7179 014662  005037  002106                   CLR         MUT
 7180 014666                                    SET         HEADER
 7181 014674                                  END ;OF IF ACFLAG
 7182 014674                                END ;OF FOR BANK
 7183 014710  000207                        RETURN
```

```
                                .SBTTL  MTEST MODES

                        BAFPAF: SUBTST  <<BANKS FORWARD,PATTERNS FORWARD      **RECURSIVE**>>
                        ;*****************************************************************************
                        ;*SUBTEST       BANKS FORWARD,PATTERNS FORWARD      **RECURSIVE**
                        ;*****************************************************************************
014712  005037  002100              CLR     BANK            ;SET BANK TO 0
                                    ;START OF BANK LOOP
014716  004737  044302      1$:     CALL    EXBANK          ;EXAMINE BANK
014722  005737  002114              TST     ACFLAG          ;CAN WE ACCESS THIS BANK?
014726  001412                      BEQ     4$              ;NO - GO TO BANK LOOP TERMINATION
014730  005737  002122              TST     RRFLAG          ;RELOCATION REQUIRED?
014734  001007                      BNE     4$              ;YES - GO TO BANK LOOP TERMINATION
014736  005037  002110              CLR     PATTERN         ;SET PATTERN TO 0
                                    ;START OF PATTERN LOOP
014742  004737  016212      2$:     CALL    MTEST           ;GO TEST CORRECT MEMORY
                                    ;TERMINATION OF PATTERN LOOP
014746  004737  044770              CALL    INCPAT          ;GO SEE IF THIS IS THE LAST PATTERN
014752  001373                      BNE     2$              ;NO   LOOP ON THIS PATTERN
                                    ;TERMINATION OF BANK LOOP
014754  005037  002216      4$:     CLR     CONTFLAG
014760  004737  045014              CALL    INCBNK          ;NEXT HIGHER BANK
014764  002354                      BGE     1$              ;IF NOT DONE - LOOP ON THIS BANK
                                    ;END OF LOOPS
014766  005737  002124              TST     RLFLAG          ;HAVE WE BEEN RELOCATED?
014772  001401                      BEQ     5$              ;NO - SKIP
014774  000207                      RETURN                  ;YES - RETURN
014776  004737  042530      5$:     CALL    RELOCATE        ;MOVE & MAP PROGRAM
015002                              ON.ERROR THEN $RETURN
                                    ;**NOTE** RECURSIVE CALL
015006  004737  014712              CALL    BAFPAF          ;CALL SELF
015012  004737  043420              CALL    UNRELOCATE      ;UNMOVE & UNMAP PROGRAM
015016  000207                      RETURN
```

```
7218 015020                             BAFPAR: SUBTST  ••BANKS FORWARD,PATTERNS REVERSE      ••RECURSIVE•••
                                        ;•••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••
                                        ;•SUBTEST       BANKS FORWARD,PATTERNS REVERSE      ••RECURSIVE••
                                        ;•••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••
7219 015020  005037  002100                     CLR     BANK                ;SET BANK TO 0
7220                                            ;START OF BANK LOOP
7221 015024  004737  044302             1$:     CALL    EXBANK              ;EXAMINE BANK
7222 015030  005737  002114                     TST     ACFLAG              ;CAN WE ACCESS THIS BANK?
7223 015034  001412                             BEQ     4$                  ;NO - GO TO BANK LOOP TERMINATION
7224 015036  005737  002122                     TST     RRFLAG              ;RELOCATION REQUIRED?
7225 015042  001007                             BNE     4$                  ;YES - GO TO BANK LOOP TERMINATION
7226 015044  004737  045004                     CALL    SETPAT              ;SET HIGH PATTERN FOR CORRECT MEMORY
7227                                            ;START OF PATTERN LOOP
7228 015050  004737  016212             2$:     CALL    MTEST               ;GO TEST CORRECT MEMORY
7229                                            ;TERMINATION OF PATTERN LOOP
7230 015054  005337  002110                     DEC     PATTERN             ;IS THIS THE LAST PATTERN?
7231 015060  100373                             BPL     2$                  ;NO   LOOP ON THIS PATTERN
7232                                            ;TERMINATION OF BANK LOOP
7233 015062  005037  002216             4$:     CLR     CONTFLAG
7234 015066  004737  045014                     CALL    INCBNK              ;NEXT HIGHER BANK
7235 015072  002354                             BGE     1$                  ;IF NOT DONE   LOOP ON THIS BANK
7236                                            ;END OF LOOPS
7237 015074  005737  002124                     TST     RLFLAG              ;HAVE WE BEEN RELOCATED?
7238 015100  001401                             BEQ     5$                  ;NO - SKIP
7239 015102  000207                             RETURN                      ;YES   RETURN
7240 015104  004737  042550             5$:     CALL    RELOCATE            ;MOVE & MAP PROGRAM
7241 015110                                     ON.ERROR THEN $RETURN
7242                                            ;••NOTE•• RECURSIVE CALL
7243 015114  004737  015020                     CALL    BAFPAR              ;CALL SELF
7244 015120  004737  043420                     CALL    UNRELOCATE          ;UNMOVE & UNMAP PROGRAM
7245 015124  000207                             RETURN
```

```
*248 015126                                BAWPAF: SUBTST  <<BANKS WORST FIRST,PATTERNS FORWARD  ..RECURSIVE..>
                                           ;*****************************************************************
                                           ;*SUBTEST       BANKS WORST FIRST,PATTERNS FORWARD  ..RECURSIVE..
                                           ;*****************************************************************
7249 015126  005037  002100                        CLR     BANK            ;SET BANK TO 0
7250                                               ;START OF BANK LOOP
7251 015132  004737  044302                1$:     CALL    EXBANK          ;EXAMINE BANK
7252 015136  005737  002114                        TST     ACFLAG          ;CAN WE ACCESS THIS BANK?
7253 015142  001415                                BEQ     4$              ;NO   GO TO BANK LOOP TERMINATION
7254 015144  005737  002126                        TST     BMFLAG          ;IS THIS BAD MEMORY (WORST FIRST)?
7255 015150  001412                                BEQ     4$              ;NO - GO TO BANK LOOP TERMINATION
7256 015152  005737  002122                        TST     RRFLAG          ;RELOCATION REQUIRED?
7257 015156  001007                                BNE     4$              ;YES   GO TO BANK LOOP TERMINATION
7258 015160  005037  002110                        CLR     PATTERN         ;SET PATTERN TO 0
7259                                               ;START OF PATTERN LOOP
7260 015164  004737  016212                2$:     CALL    MTEST           ;GO TEST CORRECT MEMORY
7261                                               ;TERMINATION OF PATTERN LOOP
7262 015170  004737  044770                        CALL    INCPAT          ;GO SEE IF THIS IS THE LAST PATTERN
7263 015174  001373                                BNE     2$              ;NO   LOOP ON THIS PATTERN
7264                                               ;TERMINATION OF BANK LOOP
7265 015176  005037  002216                4$:     CLR     CONTFLAG
7266 015202  004737  045014                        CALL    INCBNK          ;NEXT HIGHER BANK
7267 015206  002351                                BGE     1$              ;IF NOT DONE - LOOP ON THIS BANK
7268                                               ;END OF LOOPS
7269 015210  005137  002542                        COM     WORST           ;IS THIS AN EVEN NUMBERED PASS?
7270 015214  001003                                BNE     5$              ;YES - SKIP
7271                                               ;..NOTE.. RECURSIVE CALL
7272 015216  004737  015126                        CALL    BAWPAF          ;CALL SELF
7273 015222  000207                                RETURN
7274 015224  005737  002124                5$:     TST     RLFLAG          ;HAVE WE BEEN RELOCATED?
7275 015230  001401                                BEQ     6$              ;NO - SKIP
7276 015232  000207                                RETURN                  ;YES   RETURN
7277 015234  004737  042530                6$:     CALL    RELOCATE        ;MOVE & MAP PROGRAM
7278 015240                                        ON.ERROR THEN $RETURN
7279                                               ;..NOTE.. RECURSIVE CALL
7280 015244  004737  015126                        CALL    BAWPAF          ;CALL SELF
7281 015250  004737  043420                        CALL    UNRELOCATE      ;UNMOVE & UNMAP PROGRAM
7282 015254  000207                                RETURN
```

```
7285 015256                        BAWPAR: SUBTST  <<BANKS WORST FIRST,PATTERNS REVERSE  ••RECURSIVE••>>
                                   ;*****************************************************************
                                   ;*SUBTEST       BANKS WORST FIRST,PATTERNS REVERSE  ••RECURSIVE••
                                   ;*****************************************************************
7286 015256  005037  002100               CLR     BANK            ;SET BANK TO 0
7287                               ;START OF BANK LOOP
7288 015262  004737  044302        1$:     CALL    EXBANK          ;EXAMINE BANK
7289 015266  005737  002114               TST     ACFLAG          ;CAN WE ACCESS THIS BANK?
7290 015272  001415                       BEQ     4$              ;NO - GO TO BANK LOOP TERMINATION
7291 015274  005737  002126               TST     BMFLAG          ;IS THIS BAD MEMORY (WORST FIRST)
7292 015300  001412                       BEQ     4$              ;NO - GO TO BANK LOOP TERMINATION
7293 015302  005737  002122               TST     RRFLAG          ;RELOCATION REQUIRED?
7294 015306  001007                       BNE     4$              ;YES   GO TO BANK LOOP TERMINATION
7295 015310  004737  045004               CALL    SETPAT          ;SET HIGH PATTERN FOR CORRECT MEMORY
7296                               ;START OF PATTERN LOOP
7297 015314  004737  016212        2$:     CALL    MTEST           ;GO TEST CORRECT MEMORY
7298                               ;TERMINATION OF PATTERN LOOP
7299 015320  005337  002110               DEC     PATTERN         ;IS THIS THE LAST PATTERN?
7300 015324  100373                       BPL     2$              ;NO - LOOP ON THIS PATTERN
7301                               ;TERMINATION OF BANK LOOP
7302 015326  005037  002216        4$:     CLR     CONTFLAG
7303 015332  004737  045014               CALL    INCBNK          ;NEXT HIGHER BANK
7304 015336  002351                       BGE     1$              ;IF NOT DONE - LOOP ON THIS BANK
7305                               ;END OF LOOPS
7306 015340  005137  002542               COM     WORST           ;IS THIS AN EVEN NUMBERED PASS?
7307 015344  001003                       BNE     5$              ;YES - SKIP
7308                               ;••NOTE•• RECURSIVE CALL
7309 015346  004737  015256               CALL    BAWPAR          ;CALL SELF
7310 015352  000207                       RETURN
7311 015354  005737  002124        5$:     TST     RLFLAG          ;HAVE WE BEEN RELOCATED?
7312 015360  001401                       BEQ     6$              ;NO - SKIP
7313 015362  000207                       RETURN                  ;YES - RETURN
7314 015364  004737  042530        6$:     CALL    RELOCATE        ;MOVE & MAP PROGRAM
7315 015370                               ON.ERROR THEN $RETURN
7316                               ;••NOTE•• RECURSIVE CALL
7317 015374  004737  015256               CALL    BAWPAR          ;CALL SELF
7318 015400  004737  043420               CALL    UNRELOCATE      ;UNMOVE & UNMAP PROGRAM
7319 015404  000207                       RETURN
```

```
                        PAFBAF: SUBTST  <<PATTERNS FORWARD,BANKS FORWARD      ..RECURSIVE..>>
                        ;*****************************************************************
                        ;*SUBTEST       PATTERNS FORWARD,BANKS FORWARD      ..RECURSIVE..
                        ;*****************************************************************
7323 015406  005037  002110          CLR     PATTERN             ;SET PATTERN TO 0
7324                                 ;START OF PATTERN LOOP
7325 015412  005037  002100  1$:     CLR     BANK                ;SET BANK TO 0
7326                                 ;START OF BANK LOOP
7327 015416  004737  044302  2$:     CALL    EXBANK              ;EXAMINE BANK
7328 015422  004737  044752          CALL    BANKOK              ;CORRECT MEMORY FOR THIS BANK?
7329 015426  001010                  BNE     4$                  ;NO - GO TO BANK LOOP TERMINATOR
7330 015430  005737  002114          TST     ACFLAG              ;CAN WE ACCESS THIS BANK?
7331 015434  001405                  BEQ     4$                  ;NO   GO TO BANK LOOP TERMINATION
7332 015436  005737  002122          TST     RRFLAG              ;RELOCATION REQUIRED?
7333 015442  001002                  BNE     4$                  ;YES - GO TO BANK LOOP TERMINATION
7334 015444  004737  016212          CALL    MTEST               ;GO TEST CORRECT MEMORY
7335                                 ;TERMINATION OF BANK LOOP
7336 015450  005037  002216  4$:     CLR     CONTFLAG
7337 015454  004737  045014          CALL    INCBNK              ;NEXT HIGHER BANK
7338 015460  002356                  BGE     2$                  ;IF NOT DONE   LOOP ON THIS BANK
7339                                 ;TERMINATION OF PATTERN LOOP
7340 015462  004737  044770          CALL    INCPAT              ;NEXT HIGHER PATTERN
7341 015466  001351                  BNE     1$                  ;OK   LOOP; ELSE CONTINUE
7342                                 ;END OF LOOPS
7343 015470  005137  002132          COM     TMFLAG              ;COMPLEMENT TYPE OF MEMORY
7344                                                             ;IS THIS AN EVEN NUMBER PASS?
7345 015474  001403                  BEQ     5$                  ;YES - SKIP
7346                                 ;..NOTE.. RECURSIVE CALL
7347 015476  004737  015406          CALL    PAFBAF              ;CALL SELF
7348 015502  000207                  RETURN
7349 015504  005737  002124  5$:     TST     RLFLAG              ;HAVE WE BEEN RELOCATED?
7350 015510  001401                  BEQ     6$                  ;NO - SKIP
7351 015512  000207                  RETURN                      ;YES   RETURN
7352 015514  004737  042530  6$:     CALL    RELOCATE            ;MOVE & MAP PROGRAM
7353 015520                          ON.ERROR THEN $RETURN
7354                                 ;..NOTE.. RECURSIVE CALL
7355 015524  004737  015406          CALL    PAFBAF              ;CALL SELF
7356 015530  004737  043420          CALL    UNRELOCATE          ;UNMOVE & UNMAP PROGRAM
7357 015534  000207                  RETURN
```

```
7360 015536                        PAFBAW: SUBTST  <<PATTERNS FORWARD,BANKS WORST FIRST  ..RECURSIVE..>
                                   ;*****************************************************************************
                                   ;.SUBTEST       PATTERNS FORWARD,BANKS WORST FIRST  ..RECURSIVE..
                                   ;*****************************************************************************
7361 015536  005037  002110                CLR     PATTERN             ;SET PATTERN TO 0
7362                               ;START OF PATTERN LOOP
7363 015542  005037  002100        1$:     CLR     BANK                ;SET BANK TO 0
7364                               ;START OF BANK LOOP
7365 015546  004737  044302        2$:     CALL    EXBANK              ;EXAMINE BANK
7366 015552  004737  044752                CALL    BANKOK              ;CORRECT MEMORY FOR THIS BANK?
7367 015556  001013                        BNE     4$                  ;NO - GO TO BANK LOOP TERMINATOR
7368 015560  005737  002114                TST     ACFLAG              ;CAN WE ACCESS THIS BANK?
7369 015564  001410                        BEQ     4$                  ;NO - GO TO BANK LOOP TERMINATION
7370 015566  005737  002126                TST     BMFLAG              ;IS THIS BAD MEMORY (WORST FIRST)
7371 015572  001405                        BEQ     4$                  ;NO - GO TO BANK LOOP TERMINATION
7372 015574  005737  002122                TST     RRFLAG              ;RELOCATION REQUIRED?
7373 015600  001002                        BNE     4$                  ;YES   GO TO BANK LOOP TERMINATION
7374 015602  004737  016212                CALL    MTEST               ;GO TEST CORRECT MEMORY
7375                               ;TERMINATION OF BANK LOOP
7376 015606  005037  002216        4$:     CLR     CONTFLAG
7377 015612  004737  045014                CALL    INCBNK              ;NEXT HIGHER BANK
7378 015616  002353                        BGE     2$                  ;IF NOT DONE - LOOP ON THIS BANK
7379                               ;TERMINATION OF PATTERN LOOP
7380 015620  004737  044770                CALL    INCRPT              ;NEXT HIGHER PATTERN
7381 015624  001346                        BNE     1$                  ;OK - LOOP; ELSE CONTINUE
7382                               ;END OF LOOPS
7383 015626  005137  002132                COM     TMFLAG              ;COMPLEMENT TYPE OF MEMORY
7384                                                                   ;IS THIS AN EVEN NUMBER PASS?
7385 015632  001403                        BEQ     5$                  ;YES - SKIP
7386                               ;..NOTE.. RECURSIVE CALL
7387 015634  004737  015536                CALL    PAFBAW              ;CALL SELF
7388 015640  000207                        RETURN
7389 015642  005137  002542        5$:     COM     WORST               ;4TH PASS?
7390 015646  001003                        BNE     6$                  ;YES - SKIP
7391                               ;..NOTE.. RECURSIVE CALL
7392 015650  004737  015536                CALL    PAFBAW              ;CALL SELF
7393 015654  000207                        RETURN
7394 015656  005737  002124        6$:     TST     RLFLAG              ;HAVE WE BEEN RELOCATED?
7395 015662  001401                        BEQ     7$                  ;NO    SKIP
7396 015664  000207                        RETURN                      ;YES - RETURN
7397 015666  004737  042530        7$:     CALL    RELOCATE            ;MOVE & MAP PROGRAM
7398 015672                        ON.ERROR THEN $RETURN
7399                               ;..NOTE.. RECURSIVE CALL
7400 015676  004737  015536                CALL    PAFBAW              ;CALL SELF
7401 015702  004737  043420                CALL    UNRELOCATE          ;UNMOVE & UNMAP PROGRAM
7402 015706  000207                        RETURN
```

```
7405 015710                          PARBAF: SUBTST  <<PATTERNS REVERSE,BANKS FORWARD      ••RECURSIVE••>>
                                     ;*****************************************************************************
                                     ;•SUBTEST       PATTERNS REVERSE,BANKS FORWARD      ••RECURSIVE••
                                     ;*****************************************************************************
7406 015710  004737  045004                  CALL    HIPAT           ;SET HIGHTEST PATTERNS
7407                                         ;START OF PATTERN LOOP
7408 015714  005037  002100          1$:     CLR     BANK            ;SET BANK TO 0
7409                                         ;START OF BANK LOOP
7410 015720  004737  044302          2$:     CALL    EXBANK          ;EXAMINE BANK
7411 015724  004737  044752                  CALL    BANKOK          ;CORRECT MEMORY FOR THIS BANK?
7412 015730  001010                          BNE     4$              ;NO - GO TO BANK LOOP TERMINATOR
7413 015732  005737  002114                  TST     ACFLAG          ;CAN WE ACCESS THIS BANK?
7414 015736  001405                          BEQ     4$              ;NO - GO TO BANK LOOP TERMINATION
7415 015740  005737  002122                  TST     RRFLAG          ;RELOCATION REQUIRED?
7416 015744  001002                          BNE     4$              ;YES - GO TO BANK LOOP TERMINATION
7417 015746  004737  016212                  CALL    MTEST           ;GO TEST CORRECT MEMORY
7418                                         ;TERMINATION OF BANK LOOP
7419 015752  005037  002216          4$:     CLR     CONTFLAG
7420 015756  004737  045014                  CALL    INCBNK          ;NEXT HIGHER BANK
7421 015762  002356                          BGE     2$              ;IF NOT DONE - LOOP ON THIS BANK
7422                                         ;TERMINATION OF PATTERN LOOP
7423 015764  005337  002110                  DEC     PATTERN         ;NEXT LOWER PATTERN
7424 015770  100351                          BPL     1$              ;OK - LOOP; ELSE CONTINUE
7425                                         ;END OF LOOPS
7426 015772  005137  002132                  COM     TMFLAG          ;COMPLEMENT TYPE OF MEMORY
7427                                                                 ;IS THIS AN EVEN NUMBER PASS?
7428 015776  001403                          BEQ     5$              ;YES - SKIP
7429                                         ;••NOTE•• RECURSIVE CALL
7430 016000  004737  015710                  CALL    PARBAF          ;CALL SELF
7431 016004  000207                          RETURN
7432 016006  005737  002124          5$:     TST     RLFLAG          ;HAVE WE BEEN RELOCATED?
7433 016012  001401                          BEQ     6$              ;NO - SKIP
7434 016014  000207                          RETURN                  ;YES - RETURN
7435 016016  004737  042530          6$:     CALL    RELOCATE        ;MOVE & MAP PROGRAM
7436 016022                                  ON.ERROR THEN $RETURN
7437                                         ;••NOTE•• RECURSIVE CALL
7438 016026  004737  015710                  CALL    PARBAF          ;CALL SELF
7439 016032  004737  043420                  CALL    UNRELOCATE      ;UNMOVE & UNMAP PROGRAM
7440 016036  000207                          RETURN
```

```
7443 016040                        PARBAW: SUBTST  <<PATTERNS REVERSE,BANKS WORST FIRST  **RECURSIVE**>>
                                   ;*****************************************************************************
                                   ;*SUBTEST       PATTERNS REVERSE,BANKS WORST FIRST  **RECURSIVE**
                                   ;*****************************************************************************
7444 016040  004737  045004               CALL    HIPAT           ;SET HIGHTEST PATTERN
7445                               ;START OF PATTERN LOOP
7446 016044  005037  002100        1$:     CLR     BANK            ;SET BANK TO 0
7447                               ;START OF BANK LOOP
7448 016050  004737  044302        2$:     CALL    EXBANK          ;EXAMINE BANK
7449 016054  004737  044752               CALL    BANKOK          ;CORRECT MEMORY FOR THIS BANK?
7450 016060  001013                       BNE     4$              ;NO - GO TO BANK LOOP TERMINATOR
7451 016062  005737  002114               TST     ACFLAG          ;CAN WE ACCESS THIS BANK?
7452 016066  001410                       BEQ     4$              ;NO - GO TO BANK LOOP TERMINATION
7453 016070  005737  002126               TST     BMFLAG          ;IS THIS BAD MEMORY (WORST FIRST)
7454 016074  001405                       BEQ     4$              ;NO - GO TO BANK LOOP TERMINATION
7455 016076  005737  002122               TST     RRFLAG          ;RELOCATION REQUIRED?
7456 016102  001002                       BNE     4$              ;YES - GO TO BANK LOOP TERMINATION
7457 016104  004737  016212               CALL    MTEST           ;GO TEST CORRECT MEMORY
7458                               ;TERMINATION OF BANK LOOP
7459 016110  005037  002216        4$:     CLR CONTFLAG
7460 016114  004737  045014               CALL    INCBNK          ;NEXT HIGHER BANK
7461 016120  002353                       BGE     2$              ;IF NOT DONE - LOOP ON THIS BANK
7462                               ;TERMINATION OF PATTERN LOOP
7463 016122  005337  002110               DEC     PATTERN         ;NEXT LOWER PATTERN
7464 016126  100346                       BPL     1$              ;OK - LOOP; ELSE CONTINUE
7465                               ;END OF LOOPS
7466 016130  005137  002132               COM     TMFLAG          ;COMPLEMENT TYPE OF MEMORY
7467                                                               ;IS THIS AN EVEN NUMBER PASS?
7468 016134  001403                       BEQ     5$              ;YES - SKIP
7469                               ;**NOTE** RECURSIVE CALL
7470 016136  004737  016040               CALL    PARBAW          ;CALL SELF
7471 016142  000207                       RETURN
7472 016144  005137  002542        5$:     COM     WORST           ;4TH PASS?
7473 016150  001003                       BNE     6$              ;YES - SKIP
7474                               ;**NOTE** RECURSIVE CALL
7475 016152  004737  016040               CALL    PARBAW          ;CALL SELF
7476 016156  000207                       RETURN
7477 016160  005737  002124        6$:     TST     RLFLAG          ;HAVE WE BEEN RELOCATED?
7478 016164  001401                       BEQ     7$              ;NO - SKIP
7479 016166  000207                       RETURN                  ;YES   RETURN
7480 016170  004737  042530        7$:     CALL    RELOCATE        ;MOVE & MAP PROGRAM
7481 016174                               ON.ERROR THEN $RETURN
7482                               ;**NOTE** RECURSIVE CALL
7483 016200  004737  016040               CALL    PARBAW          ;CALL SELF
7484 016204  004737  043420               CALL    UNRELOCATE      ;UNMOVE & UNMAP PROGRAM
7485 016210  000207                       RETURN
```

```
7488 016212                          MTEST:  SUBTST  <<SUBR  SETUP MEMORY TEST>>
                                     ;*****************************************************************************
                                     ;*SUBTEST        SUBR    SETUP MEMORY TEST
                                     ;*****************************************************************************
7489 016212                                  SET     HEADER                  ;INITIALIZE HEADER MESSAGE TYPEOUT
7490 016220                                  SET     MUT                     ;INDICATE THERE IS A MEMORY UNDER TEST
7491 016226  005037  002260                  CLR     PASFLG
7492 016232  005737  002116                  TST     MKFLAG                  ;ECC?
7493 016236  001413                          BEQ     MT1                     ;NO   SKIP
7494 016240                                  BEGIN   HOLDLOOP
7495 016240                                    IF CONTFLAG IS TRUE THEN LEAVE HOLDLOOP
7496 016246                                    IF SKIPMK IS FALSE
7497 016254  004737  016306                      CALL        MKCONTROL
7498 016260                                    END; OF IF SKIPMK
7499 016260                                  END     HOLDLOOP
7500 016260  004737  017520                  CALL    MKTEST                  ;YES   DO ECC TESTS
7501 016264  000402                          BR      MT2
7502 016266  004737  017740          MT1:    CALL    MUTEST                  ;DO PARITY TESTS
7503 016272  005037  002106          MT2:    CLR     MUT                     ;NOW   NO MEMORY UNDER TEST
7504 016276                                  SET     HEADER                  ;ALLOW HEADERS NORMAL
7505 016304  000207                          RETURN
```

```
7508 016306                                  MKCONTROL:SUBTST        <<SUBR  TEST ECC CSR LOGIC DISPATCH>>
                                             ;*****************************************************************
                                             ;*SUBTEST        SUBR    TEST ECC CSR LOGIC DISPATCH
                                             ;*****************************************************************
7509                                                 ;THE NEXT TWO MODULES SOLVE THE PROBLEM OF
7510                                                 ;HOW TO RUN THE CSR TESTS ON EACH ECC MEMORY
7511                                                 ;
7512 016306                                          IF SELONLY IS TRUE THEN $RETURN
7513 016316                                          IF INHECC IS TRUE THEN $RETURN
7514 016326                                          PUSH    BANK,R0,R1,R2,R3
7515 016342  012737  060000  002226                 MOV     #FIRST,CSRFBANK         ;SET FIRST TEST ADDRESS TO FIRST ADDR.
7516 016350  012737  157776  002230                 MOV     #LAST,CSRLBANK
7517 016356  005037  002232                         CLR     CSRINT
7518 016362  005037  002234                         CLR     SPLTCSR
7519 016366  005037  002304                         CLR     CSRLOOP                 ; AND ZERO THE LOOP COUNTER
7520 016372  013700  002102                         MOV     BANKINDEX,R0            ;GET THE BANK INDEX
7521 016376  016001  002626                         MOV     CONFIG(R0),R1           ;GET CSR NUMBER
7522 016402  000301                                 SWAB    R1
7523 016404  042701  177760                         BIC     #^C17,R1
7524 016410  006301                                 ASL     R1
7525 016412  010137  002500                         MOV     R1,CSRHOLD              ;STORE IN THE LOW BYTE
7526 016416  005737  002134                         TST     INTFLAG                 ;IS THIS BANK INTERLEAVED?
7527 016422  001421                                 BEQ     1$                      ;BRANCH IF NOT INTERLEAVED
7528 016424  005237  002234                         INC     SPLTCSR
7529 016430  012737  120000  002230                 MOV     #120000,CSRLBANK
7530 016436  005237  002304                         INC     CSRLOOP                 ;WE MUST LOOP TWICE FOR AN INTERLEAVED BANK
7531 016442  005237  002232                         INC     CSRINT
7532 016446  016001  002626                         MOV     CONFIG(R0),R1           ;GET THE INTERLEAVE CSR NUMBER
7533 016452  072127  177775                         ASH     #-3,R1
7534 016456  042701  160777                         BIC     #^C17000,R1
7535 016462  050137  002500                         BIS     R1,CSRHOLD              ;STORE IT IN CSRHOLD'S UPPER BYTE
7536 016466  005003                         1$:     CLR     R3
7537 016470  116337  002500  002146     MKLOOP:     MOVB    CSRHOLD(R3),CSRNO
7538 016476  042737  177741  002146                 BIC     #^C36,CSRNO             ;CLEAR ANY UNNECESSARY BITS
7539 016504                                            FOR R2 := #0 TO CSRINT
7540 016506                                              FOR CSRFIRST := CSRFBANK TO CSRLBANK BY #4000
7541 016514                                              MAP BANK                    ;MAP TEST SPACE TO BANK
7542 016530  104511                                      INVALIDATE                  ;INVALIDATE BACKROUND PATTERN
7543 016532                                                BEGIN CSRSTUFF
7544 016532  005037  002306                               CLR       SUCCESS
7545 016536                                                IF ACFLAG IS TRUE AND RRFLAG IS FALSE
7546 016552  013737  002222  002224                         MOV   CSRFIRST,CSRLAST
7547 016560  062737  004000  002224                         ADD   #4000,CSRLAST
7548 016566                                                  FOR TESTADD := CSRFIRST TO CSRLAST BY #4
7549 016574  013737  002364  002366                           MOV       TESTADD,TESTADD+2
7550 016602  005737  002234                                 TST       SPLTCSR
7551 016606  001404                                         BEQ       1$
7552 016610  062737  040000  002366                         ADD       #40000,TESTADD+2
7553 016616  000403                                         BR        2$
7554 016620  062737  000002  002366                   1$:   ADD       #2,TESTADD+2
7555 016626  004737  017114                           2$:   CALL      SBETEST
7556 016632                                                  ON.NOERROR
7557 016634  104424                                           CACHOFF                 ;TURN CACHE OFF
7558 016636  005037  002074                                   CLR     NOPAR           ;INDICATE PARITY ACTION
7559 016642                                                    FOR I := #0 TO #27
7560 016646                                                      SET   HEADER
7561 016654  005037  002260                                     CLR   PASFLG
```

```
7562 016660                                                  LET R0 := I
7563 016664                                                  PUSH  R3                ;SAVE LOOP COUNTER
7564 016666  010637  002142                                  MOV   SP,CTLKVEC        ;SAVE VECTOR IN CSR OF TK
7565 016672  162737  000002  002142                          SUB   #2,CTLKVEC
7566 016700  004737  017420                                  CALL  CSRCASE
7567 016704                                                  POP   R3                ;RESTORE LOOP COUNTER
7568 016706                                                END ;OF FOR I
7569 016722  104423                                        CACHON                    ;TURN CACHE ON
7570 016724                                                SET     SUCCESS
7571 016732                                                LEAVE CSRSTUFF
7572 016734                                              END ;OF ON.NOERROR
7573 016734                                            END ;OF FOR TESTADD
7574 016752                                          END ;OF IF
7575 016752                                        END CSRSTUFF
7576 016752                                        IF SUCCESS IS FALSE
7577 016760                                          TYPE    MSGA34
7578 016764                                          TYPOCS  BANK,<TYPES BANK NUMBER>,3
7579 016774                                          TYPE    MSGB34
7580 017000                                        END ;OF IF SUCCESS
7581 017000                                      END; OF FOR CSRFIRST
7582 017016  005237  002234                       INC   SPLTCSR
7583 017022                                    END; OF FOR R2
7584 017032  062737  000002  002226          ADD     #2,CSRFBANK
7585 017040  012737  000001  002234          MOV     #1,SPLTCSR
7586 017046  005203                          INC     R3
7587 017050  020337  002304                  CMP     R3,CSRLOOP
7588 017054  003002                          BGT     1$
7589 017056  000137  016470                  JMP     MKLOOP
7590 017062  104472                  1$:     ECCINIT                 ;TRAP ON DOUBLE BIT ERRORS (NORMAL)
7591 017064                                  SET     CONTFLAG
7592 017072  005037  002234                  CLR     SPLTCSR
7593 017076                                  POP     R3,R2,R1,R0,BANK
7594 017112  000207                          RETURN
```

```
 7597 017114                            SBETEST:SUBTST  <<CHECK FOR SBE FREE LOCATIONS>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       CHECK FOR SBE FREE LOCATIONS
                                        ;*****************************************************************************
 7598                                           ;IN ORDER TO DETERMINE IF A LOCATION IS SBE FREE I DO THIS
 7599                                           ;
 7600                                           ;WRITE ZEROS WITH ECC DISABLE
 7601                                           ;READ ZEROS BACK
 7602                                           ;IF NOT ZEROS THEN RETURN ERROR
 7603                                           ;
 7604                                           ;WRITE ZEROS WITH ECC ENABLED BUT TRAPS DISABLED
 7605                                           ;READ ZEROS BACK
 7606                                           ;IF NOT ZEROS THEN RETURN ERROR
 7607                                           ;
 7608                                           ;TEST THE LOCATION FROM THE PAR'S (WITH NO PROGRAM FETCHES)
 7609                                           ;IF THERE WERE ANY SBE'S OR DBE'S THEN RETURN ERROR
 7610                                           ;
 7611                                           ;COMPLIMENT ZEROS TO ONES WITH ECC DISABLE
 7612                                           ;READ ONES BACK
 7613                                           ;IF NOT ONES THEN RETURN ERROR
 7614                                           ;
 7615                                           ;WRITE 100,,100000,00000 (CHECKBITS COMPLIMENT OF BEFORE)
 7616                                           ;  WITH ECC ENABLED BUT TRAPS DISABLED
 7617                                           ;TEST THE LOCATION FROM THE PAR'S (WITH NO PROGRAM FETCHES)
 7618                                           ;IF THERE WERE ANY SBE'S OR DBE'S THEN RETURN ERROR
 7619                                           ;
 7620                                           ;IF NONE OF THE ABOVE FORCES A RETURN ERROR THEN RETURN NO.ERROR
 7621                                           .ENABL  LSB
 7622 017114                            SBENT:  PUSH    R0,R1,R4                ;PUSH R0,R1,R4 ONTO STACK
 7623 017122  013701  002364                    MOV     TESTADD,R1
 7624 017126  013704  002366                    MOV     TESTADD+2,R4
 7625 017132                                    TESTAREA                        ;ENTER TEST MODE
 7626 017140  104424                            CACHOFF                         ;TURN CACHE OFF
 7627 017142  104471                            ECC1DIS                         ;DISABLE ECC ON 1 SELECTED CSR
 7628 017144                                    CLEAR   (R1),(R4)
 7629 017150  005711                            TST     (R1)
 7630 017152  001107                            BNE     SBENT
 7631 017154  005714                            TST     (R4)
 7632 017156  001105                            BNE     SBENT
 7633
 7634 017160  104503                            CLR1CSR                         ;CLEAR 1 SELECTED CSR
 7635 017162                                    CLEAR   (R1),(R4)
 7636 017166  005711                            TST     (R1)
 7637 017170  001100                            BNE     SBENT
 7638 017172  005714                            TST     (R4)
 7639 017174  001076                            BNE     SBENT
 7640
 7641 017176  104510                            TSTREAD                 ;TEST LOC (R1) & TST FOR SBE (WITHOUT FETCHES)
 7642 017200                                    IF @BIT15!BIT4 SET.IN CSR
 7643 017210                                      SET SKPERR                    ;DISABLE ERRGEN'S ERROR PRINTOUT
 7644 017216  104512                              ERRGEN
 7645 017220  013700  002432                      MOV   ERRADD,R0
 7646 017224  072027  177774                      ASH   #-4,R0
 7647 017230  042700  177600                      BIC   #^C177,R0
 7648 017234                                      IF BANK EQ R0 THEN GOTO SBENT
 7649 017242                                    END; OF IF @BIT15
 7650 017242  104471                            ECC1DIS                         ;DISABLE ECC ON 1 SELECTED CSR
```

```
7651 017244  005111                    COM     (R1)
7652 017246  005114                    COM     (R4)
7653 017250  023711  002556            CMP     ONES,(R1)
7654 017254  001046                    BNE     SBENT
7655 017256  023714  002556            CMP     ONES,(R4)
7656 017262  001043                    BNE     SBENT
7657
7658 017264  104503                    CLR1CSR                         ;CLEAR 1 SELECTED CSR
7659 017266  005011                    CLR     (R1)
7660 017270  012714  100000            MOV     #BIT15,(R4)
7661 017274  005711                    TST     (R1)
7662 017276  001035                    BNE     SBENT
7663 017300  022714  100000            CMP     #BIT15,(R4)
7664 017304  001032                    BNE     SBENT
7665
7666 017306  104510                    TSTREAD                 ;TEST LOC (R1) & TST FOR SBE (WITHOUT FETCHES)
7667 017310                            IF #BIT15!BIT4 SET.IN CSR
7668 017320                              SET SKPERR                    ;DISABLE ERRGEN'S ERROR PRINTOUT
7669 017326  104512                      ERRGEN
7670 017330  013700  002432              MOV   ERRADD,R0
7671 017334  072027  177774              ASH   #-4,R0
7672 017340  042700  177600              BIC   #^C177,R0
7673 017344                              IF BANK EQ R0 THEN GOTO SBENT
7674 017352                            END; OF IF #BIT15
7675
7676 017352  104417                    KERNEL                          ;ENTER KERNEL MODE
7677 017354  104473                    ECC1INIT                        ;INITIALIZE 1 SELECTED CSR
7678 017356  104423                    CACHON                          ;TURN CACHE ON
7679 017360                            POP     R4,R1,R0                ;POP R0,R1 & R4 FROM STACK
7680 017366                            $RETURN NOERROR
7681
7682 017372  104503            SBENT:  CLR1CSR                         ;CLEAR 1 SELECTED CSR
7683 017374                            CLEAR   (R1),(R4)
7684 017400  104417                    KERNEL                          ;ENTER KERNEL MODE
7685 017402  104473                    ECC1INIT                        ;INITIALIZE 1 SELECTED CSR
7686 017404  104423                    CACHON                          ;TURN CACHE ON
7687 017406                            POP     R4,R1,R0                ;POP R0,R1 & R4 FROM STACK
7688 017414                            $RETURN ERROR
7689                                   .DSABL  LSB
```

```
7692 017420                     CSRCASE:SUBTST  <<CSR PATTERN CASE STATEMENT>>
                                ;*****************************************************************************
                                ;*SUBTEST       CSR PATTERN CASE STATEMENT
                                ;*****************************************************************************
7693 017420                             CASE R0
7694                                      ;WARNING IF YOU CHANGE THIS TABLE ALSO
7695                                      ;CHANGE "$DDW0"    "$DDW5" (THE PATTERN BIT MAP)
7696
7697                                      ;PAT  TIME              DISCRIPTION
7698 017430  021174                MKCSRT:  MT0006          ;<1 SEC          INITIAL DATA TEST
7699 017432  021272                         MT0010          ;<1 SEC          BYTE ADDRESSING TEST
7700 017434  023664                         MT0025          ;<1 SEC          INTERRUPT ENABLE TEST
7701 017436  021326                         MT0011          ;<2 SEC          CREATE SINGLE BIT ERROR TEST
7702 017440  021374                         MT0012          ;<1 SEC          WRITE BYTE CLEARS SBE TEST
7703 017442  021470                         MT0013          ; 1 SEC          CREATE DOUBLE BIT ERROR TEST
7704 017444  021544                         MT0014          ; 1 SEC          WRITE INHIBIT DURING DATIP WITH DBE
7705 017446  021622                         MT0015          ; 1 SEC          WRITE INHIBIT OF BYTE WITH DBE
7706 017450  021670                         MT0016          ;<1 SEC          WRITE INHIBIT OF WORD WITH DBE
7707 017452  026416                         MT0999          ; 0 SEC          NULL TEST
7708 017454  026416                         MT0999          ; 0 SEC          NULL TEST
7709 017456  026416                         MT0999          ; 0 SEC          NULL TEST
7710 017460  026416                         MT0999          ; 0 SEC          NULL TEST
7711 017462  026416                         MT0999          ; 0 SEC          NULL TEST
7712 017464  026416                         MT0999          ; 0 SEC          NULL TEST
7713 017466  026416                         MT0999          ; 0 SEC          NULL TEST
7714 017470  026416                         MT0999          ; 0 SEC          NULL TEST
7715 017472  026416                         MT0999          ; 0 SEC          NULL TEST
7716 017474  026416                         MT0999          ; 0 SEC          NULL TEST
7717 017476  026416                         MT0999          ; 0 SEC          NULL TEST
7718 017500  026416                         MT0999          ; 0 SEC          NULL TEST
7719 017502  026416                         MT0999          ; 0 SEC          NULL TEST
7720 017504  026416                         MT0999          ; 0 SEC          NULL TEST
7721 017506  026416                         MT0999          ; 0 SEC          NULL TEST
7722 017510                     END ;OF CASE R0
7723 017516  000207            RETURN
```

```
**26 017520                             MKTEST: SUBTST  <<SUB=  ECC TEST DISPATCH>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       SUBR    ECC TEST DISPATCH
                                        ;*****************************************************************************
**30 017520                                     IF @SW0 SET.IN @SWR OR ACTFLAG IS TRUE
**31 017536  104470                               ECCDIS                        ;DISABLE ERROR CORRECTION
**32 017540                                     ELSE
**33 017542  104502                               CLRCSR                        ;CLEAR ALL CSR'S
**34 017544                                     END ;OF IF
**35 017544  012737  000002  002074             MOV     #2,NOPAR              ;INDICATE PARITY ACTION
**36 017552  012737  000002  002300             MOV     #2,PCBUMP             ;TRAPS ADD 2 TO PC
**37 017560  013700  002110                     MOV     PATTERN,R0            ;GET PATTERN NUMBER
**38 017564  006300                             ASL     R0                    ;MAKE IT A WORD ADDRESS
**39 017566                                     IF MKPAT(R0) NE @MT0034 AND MKPAT(R0) NE @MT0999
**40 017606  104511                               INVALIDATE                  ;INVALIDATE BACKGROUND PATTERN ON 'BAN
**41 017610                                     END ;OF IF MKPAT(R0)
**42 017610  010637  002142                     MOV     SP,CTLKVEC            ;SAVE VECTOR IN CASE OF **
**43 017614  162737  000002  002142             SUB     #2,CTLKVEC
**44 017622  004770  017660                     CALL    @MKPAT(R0)            ;INDEX OFF TABLE
**45 017626                                     IF @SW0 SET.IN @SWR OR ACTFLAG IS TRUE
**46 017644  104506                               ENASBE                        ;TRAP ON SINGLE BIT ERRORS
**47 017646                                     ELSE
**48 017650  104472                               ECCINIT                       ;TRAP ON DOUBLE BIT ERRORS (NORMAL)
**49 017652                                     END ;OF IF @SW0
**50 017652  005037  002074                     CLR     NOPAR                 ;INDICATE PARITY ACTION
**51 017656  000207                             RETURN
**52
**53                                     ;WARNING IF YOU CHANGE THIS TABLE ALSO
**54                                     ;CHANGE "@DDW0"   "@DDW5" (THE PATTERN BIT MAP)
**55                                     ;PAT    TIME            DISCRIPTION
**56 017660                             MKPAT:  ;NOTE MT0034 MUST BE FIRST & LAST
**57 017660  026132                             MT0034  ;<1 SEC         ;SOFT ERROR   BACKGROUND PATTERN TEST
**58 017662  021736                             MT0017  ;<1 SEC         ;HOLDING 1'S & 0'S TEST
**59 017664  021230                             MT0007  ;<1 SEC         ;ADDRESS BIT TEST
**60 017666  020204                             MT0001  ;<1 SEC         ;ADDRESS TEST
**61 017670  020324                             MT0002  ;<1 SEC         ;COMPLEMENT ADDRESS TEST
**62 017672  020716                             MT0004  ; 1 SEC         ;ROTATING ZEROS TEST
**63 017674  021040                             MT0005  ; 1 SEC         ;ROTATING ONES TEST
**64 017676  023050                             MT0021  ; 1 SEC         ;MARCHING 0'S & 1'S TEST
**65 017700  021760                             MT0020  ;<1 SEC         ;MARCHING 1'S & 0'S IN CHECK BITS
**66 017702  023322                             MT0022  ;10 SEC         ;REFRESH & SHIFTING DIAGONAL  TEST
**67 017704  023732                             MT0026  ;<1 SEC         ;RANDOM DATA TEST
**68 017706  023420                             MT0024  ;20 SEC         ;FAST GALLOPING PATTERN TEST
**69 017710  025222                             MT0031  ; 3 SEC         ;SOB-A-LONG TEST
**70 017712  025412                             MT0032  ;<1 SEC         ;WRITE RECOVERY TEST
**71 017714  025744                             MT0033  ;35 SEC         ;BRANCH GOBBLE TEST
**72 017716  026132                             MT0034  ;<1 SEC         ;SOFT ERROR   BACKGROUND PATTERN TES
**73                                    ;NOTE MT0034 MUST BE FIRST & LAST
**74 017720  026416                             MT0999  ; 0 SEC         ;NULL TEST
**75 017722  026416                             MT0999  ; 0 SEC         ;NULL TEST
**76 017724  026416                             MT0999  ; 0 SEC         ;NULL TEST
**77 017726  026416                             MT0999  ; 0 SEC         ;NULL TEST
**78 017730  026416                             MT0999  ; 0 SEC         ;NULL TEST
**79 017732  026416                             MT0999  ; 0 SEC         ;NULL TEST
**80 017734  026416                             MT0999  ; 0 SEC         ;NULL TEST
**81 017736  026416                             MT0999  ; 0 SEC         ;NULL TEST
```

```
                                        MJTEST: SUBTST  <<SUBR  PARITY TEST DISPATCH>>
                                        ;*****************************************************************************
                                        ;*SUBTEST        SUBR    PARITY TEST DISPATCH
                                        ;*****************************************************************************
7788 017740  012737  000002  002074              MOV     #2,NOPAR                ;INDICATE PARITY ACTION
7789 017746  012737  000002  002300              MOV     #2,PCBUMP               ;TRAPS ADD 2 TO PC
7790 017754  012737  060000  002364              MOV     #FIRST,TESTADD
7791 017762  012737  060002  002366              MOV     #FIRST+2,TESTADD+2
7792 017770  013700  002110                      MOV     PATTERN,R0              ;GET PATTERN NUMBER
7793 017774  006300                              ASL     R0                      ;MAKE IT A WORD ADDRESS
7794 017776                                      IF MJPAT(R0) NE #MT0034 AND MJPAT(R0) NE #MT0999
7795 020016  104511                                INVALIDATE                    ;INVALIDATE BACKGROUND PATTERN ON 'BAN
7796 020020                                      END ;OF IF MJPAT(R0)
7797 020020  010637  002142                      MOV     SP,CTLKVEC              ;SAVE VECTOR IN CASE OF **
7798 020024  162737  000002  002142              SUB     #2,CTLKVEC
7799 020032  004770  020044                      CALL    @MJPAT(R0)              ;INDEX OFF TABLE
7800 020036  005037  002074                      CLR     NOPAR                   ;INDICATE PARITY ACTION
7801 020042  000207                              RETURN
7802
7803                                             ;WARNING IF YOU CHANGE THIS TABLE ALSO
7804                                             ;CHANGE "$DDW0" - "$DDW5" (THE PATTERN BIT MAP)
7805
7806                                             ;PAT     TIME              DISCRIPTION
7807 020044                             MJPAT:  ;NOTE MT0034 MUST BE FIRST & LAST
7808 020044  026132                             MT0034  ;<1 SEC           ;SOFT ERROR - BACKGROUND PATTERN TEST
7809 020046  021174                             MT0006  ;<1 SEC           ;INITIAL DATA TEST
7810 020050  021736                             MT0017  ;<1 SEC           ;HOLDING 1'S & 0'S TEST
7811 020052  021230                             MT0007  ;<1 SEC           ;ADDRESS BIT TEST
7812 020054  020204                             MT0001  ;<1 SEC           ;ADDRESS TEST
7813 020056  020324                             MT0002  ;<1 SEC           ;COMPLEMENT ADDRESS TEST
7814 020060  020464                             MT0003  ; 1 SEC           ;3 XOR 9 WORST CASE NOISE TEST
7815 020062  020716                             MT0004  ; 1 SEC           ;ROTATING ZEROS TEST
7816 020064  021040                             MT0005  ; 1 SEC           ;ROTATING ONES TEST
7817 020066  023050                             MT0021  ; 1 SEC           ;MARCHING 0'S & 1'S TEST
7818 020070  026304                             MT0035  ;<1 SEC           ;WORSE CASE NOISE PARITY TEST
7819 020072  023322                             MT0022  ;10 SEC           ;REFRESH TEST
7820 020074  023354                             MT0023  ;10 SEC           ;SHIFTING DIAGONAL  TEST
7821 020076  023732                             MT0026  ;<1 SEC           ;RANDOM DATA TEST
7822 020100  023420                             MT0024  ;20 SEC           ;FAST GALLOPING PATTERN TEST
7823 020102  025222                             MT0031  ; 3 SEC           ;SOB-A-LONG TEST
7824 020104  025412                             MT0032  ;<1 SEC           ;WRITE RECOVERY TEST
7825 020106  025744                             MT0033  ;35 SEC           ;BRANCH GOBBLE TEST
7826 020110  026132                             MT0034  ;<1 SEC           ;SOFT ERROR   BACKGROUND PATTERN TEST
7827                                            ;NOTE MT0034 MUST BE FIRST & LAST
7828 020112  026416                             MT0999  ; 0 SEC           ;NULL TEST
7829 020114  026416                             MT0999  ; 0 SEC           ;NULL TEST
7830 020116  026416                             MT0999  ; 0 SEC           ;NULL TEST
7831 020120  026416                             MT0999  ; 0 SEC           ;NULL TEST
7832 020122  026416                             MT0999  ; 0 SEC           ;NULL TEST
```

```
                                        .SBTTL  PATTERNS
                                        .SBTTL  MEMORY TEST SETUP ROUTINES
020124                          MT0000: SUBTST  <<MT0000        SETUP DATA PATTERN TEST>>
                                ;*****************************************************************
                                ;*SUBTEST       MT0000  SETUP DATA PATTERN TEST
                                ;*****************************************************************
020124  005037  002262                CLR     REALPAT                 ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
020130  012700  060000                MOV     #FIRST,R0
020134  012701  040000                MOV     #SIZE,R1
020140  004737  036372                CALL    REGCOPY
020144  022737  000001  003720        CMP     #1,PROTYP               ;ARE WE ON AN 11/44?
020152  001406                        BEQ     1$                      ;BRANCH IF YES
020154  012737  027036  002256        MOV     #MTP000,SUPDOADD        ;ELSE DO PATTERN IN MAIN MEMORY
020162  004737  026644                CALL    SUPD03
020166  000207                        RETURN
020170                          1$:     BMOV    MTP000
020176  004737  026466                CALL    SUPD01                  ;DO IT IN SUPERVISOR MODE
020202  000207                        RETURN
020204                          MT0001: SUBTST  <<MT0001        SETUP ADDRESS TEST>>
                                ;*****************************************************************
                                ;*SUBTEST       MT0001  SETUP ADDRESS TEST
                                ;*****************************************************************
020204  012737  000001  002262        MOV     #1,REALPAT              ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
020212  012700  060000                MOV     #FIRST,R0
020216  012701  040000                MOV     #SIZE,R1
020222  005737  002430                TST     NOSUPER
020226  001005                        BNE     2$
020230  023737  172252  172254        CMP     SIPAR5,SIPAR6
020236  001007                        BNE     4$
020240  000404                        BR      3$
020242  023737  177652  177654  2$:   CMP     UIPAR5,UIPAR6
020250  001002                        BNE     4$
020252  012701  030000          3$:   MOV     #30000,R1
020256  005002                  4$:   CLR     R2
020260  004737  036372                CALL    REGCOPY
020264  022737  000001  003720        CMP     #1,PROTYP               ;IS THIS AN 11/44?
020272  001406                        BEQ     1$                      ;BRANCH IF IT IS
020274  012737  027062  002256        MOV     #MTP001,SUPDOADD        ;SET UP CALLING ADDRESS
020302  004737  026644                CALL    SUPD03
020306  000207                        RETURN
020310                          1$:     BMOV    MTP001
020316  004737  026466                CALL    SUPD01                  ;DO IT IN SUPERVISOR MODE
020322  000207                        RETURN
020324                          MT0002: SUBTST  <<MT0002        SETUP COMPLEMENT ADDRESS TEST>>
                                ;*****************************************************************
                                ;*SUBTEST       MT0002  SETUP COMPLEMENT ADDRESS TEST
                                ;*****************************************************************
020324  012737  000002  002262        MOV     #2,REALPAT              ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
020332  012700  160000                MOV     #LAST+2,R0
020336  012701  040000                MOV     #SIZE,R1
020342  012704  060000                MOV     #FIRST,R4
020346  012705  100001                MOV     #100001,R5
020352  005737  002430                TST     NOSUPER
020356  001005                        BNE     2$
020360  023737  172252  172254        CMP     SIPAR5,SIPAR6
020366  001013                        BNE     4$
```

```
7882 020370 000404                          BR      3$
7883 020372 023737 177652 177654 2$:        CMP     UIPAR5,UIPAR6
7884 020400 001006                          BNE     4$
7885 020402 012701 030000            3$:    MOV     #30000,R1
7886 020406 012700 140000                   MOV     #140000,R0
7887 020412 012705 120001                   MOV     #120001,R5
7888 020416 012702 000001            4$:    MOV     #1,R2
7889 020422 010103                          MOV     R1,R3
7890 020424 022737 000001 003720            CMP     #1,PROTYP       ;IS THIS AN 11/44?
7891 020432 001406                          BEQ     1$              ;BRANCH IF TRUE
7892 020434 012737 027114 002256            MOV     #MTP002,SUPDOADD ;SET UP CALLING ADDRESS
7893 020442 004737 026644                   CALL    SUPD03
7894 020446 000207                          RETURN
7895 020450                          1$:    BMOV    MTP002
7896 020456 004737 026466                   CALL    SUPD01
7897 020462 000207                          RETURN
```

```
 7900 020464                                MT0003: SUBTST  <<MT0003        SETUP 3 XOR 9 WORST CASE NOISE TEST>>
                                            ;*****************************************************************************
                                            ;*SUBTEST       MT0003  SETUP 3 XOR 9 WORST CASE NOISE TEST
                                            ;*****************************************************************************
 7901 020464                                        IF EUFLAG IS TRUE THEN $RETURN
 7902 020474  012737  000003  002262                MOV     #3,REALPAT              ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
 7903 020502  005037  002300                        CLR     PCBUMP                  ;TRAPS DO NOT ADD TO PC
 7904 020506  004737  036402                1$:     CALL    FLIPWARN                ;SETUP WARNING CONSTANTS & R2
 7905 020512  012701  060000                2$:     MOV     #FIRST,R1               ;R1 <-  STARTING ADDRESS
 7906 020516  012703  020000                        MOV     #20000,R3
 7907 020522  072327  177770                        ASH     #-8.,R3                 ;R3 < - R3 / 256.
 7908 020526  012702  000004                        MOV     #4,R2                   ;SMALL LOOP SIZE
 7909 020532  012705  000100                        MOV     #64.,R5                 ;MEDIUM LOOP SIZE
 7910 020536  022737  000001  003720                CMP     #1,PROTYP               ;IS THIS AN 11/44?
 7911 020544  001415                                BEQ     3$                      ;BRANCH IF IT IS
 7912 020546  104415                                SAVREG
 7913 020550  012737  027146  002256                MOV     #MTPA03,SUPDOADD
 7914 020556  004737  026644                        CALL    SUPD03                  ;DO IT IN MAIN MEMORY
 7915 020562  104416                                RESREG
 7916 020564  012737  027206  002256                MOV     #MTPB03,SUPDOADD
 7917 020572  004737  026660                        CALL    SUPD04
 7918 020576  000442                                BR      4$
 7919 020600                                3$:     BMOV    MTPA03
 7920 020606  104415                                SAVREG
 7921 020610  004737  026466                        CALL    SUPD01
 7922 020614                                        BMOV    MTPB03
 7923 020622                                        BMOV    MTPC03,KDPAR0,8.
 7924 020634                                        BMOV    MTPD03,SDPAR0,8.
 7925 020646  012737  172360  177642                MOV     #KDPAR0,UIPAR1          ;SET UP PAR LINKS
 7926 020654  012737  172260  172374                MOV     #SDPAR0,KDPAR6
 7927 020662  012737  177644  172276                MOV     #UIPAR2,SDPAR7
 7928 020670  012737  001032  172272                MOV     #1032,SDPAR5            ;CHANGE INST TO BR .+66 (BR TO KDPAR1)
 7929 020676  104416                                RESREG
 7930 020700  004737  026502                        CALL    SUPD02
 7931 020704  022737  000003  002560  4$:     CMP     #3,FLIPLOC              ;DONE WITH 4 PATTERNS
 7932                                                                        ;[(0,177777);(177777,0);(401,177777);(177777,401)]?
 7933 020712  001275                                BNE     1$                      ;NO - LOOP
 7934 020714  000207                                RETURN
 7935
 7936 020716                                MT0004: SUBTST  <<MT0004        SETUP ROTATING ZEROS TEST>>
                                            ;*****************************************************************************
                                            ;*SUBTEST       MT0004  SETUP ROTATING ZEROS TEST
                                            ;*****************************************************************************
 7937 020716  012737  000004  002262                MOV     #4,REALPAT              ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
 7938 020724  012737  000004  002300                MOV     #4,PCBUMP               ;TRAPS ADD 4 TO PC
 7939 020732  013702  002556                        MOV     ONES,R2
 7940 020736  004737  036532                        CALL    BACKGND                 ;WRITE BACKGROUND OF ONES
 7941 020742  012700  060000                        MOV     #FIRST,R0
 7942 020746  012701  040000                        MOV     #SIZE,R1
 7943 020752  022737  000001  003720                CMP     #1,PROTYP               ;IS THIS AN 11/44?
 7944 020760  001406                                BEQ     1$                      ;BRANCH IF IT IS
 7945 020762  012737  027304  002256                MOV     #MTPA04,SUPDOADD        ;SET UP LINKS
 7946 020770  004737  026660                        CALL    SUPD04
 7947 020774  000207                                RETURN
 7948 020776                                1$:     BMOV    MTPA04
 7949 021004                                        BMOV    MTPB04,KDPAR0,8.
 7950 021016  012737  172360  17765?                MOV     #KDPAR0,UIPAR5
```

MT0004 SETUP ROTATING ZEROS TEST

```
 7951 021024  012737  177654  172376          MOV     @UIPAR6,KDPAR7
 7952 021032  004737  026502                  CALL    SUPD02
 7953 021036  000207                          RETURN
 7954 021040                          MT0005: SUBTST  <<MT0005         SETUP ROTATING ONES TEST>>
                                      ;*****************************************************************************
                                      ;*SUBTEST        MT0005  SETUP ROTATING ONES TEST
                                      ;*****************************************************************************
 7955 021040  012737  000005  002262          MOV     #5,REALPAT              ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
 7956 021046  012737  000004  002300          MOV     #4,PCBUMP               ;TRAPS ADD 4 TO PC
 7957 021054  005002                          CLR     R2
 7958 021056  004737  036532                  CALL    BACKGND                 ;WRITE BACKGROUND OF ZEROS
 7959 021062  012700  060000                  MOV     #FIRST,R0
 7960 021066  012701  040000                  MOV     #SIZE,R1
 7961 021072  022737  000001  003720          CMP     #1,PROTYP               ;IS THIS AN 11/44?
 7962 021100  001414                          BEQ     1$                      ;BRANCH IF IT IS
 7963 021102  012737  027360  002256          MOV     #MTP005,SUPDOADD        ;SET UP LINKS
 7964 021110  012737  027374  027356          MOV     #MTP005+14,MTPB04+16
 7965 021116  004737  026660                  CALL    SUPD04
 7966 021122  012737  027320  027356          MOV     #MTPA04+14,MTPB04+16    ;RESET TEST'S ORIGINAL VALUE
 7967 021130  000207                          RETURN
 7968 021132                          1$:     BMOV    MTP005
 7969 021140                                  BMOV    MTPB04,KDPAR0,8.
 7970 021152  012737  172360  177652          MOV     #KDPAR0,UIPAR5
 7971 021160  012737  177654  172376          MOV     #UIPAR6,KDPAR7
 7972 021166  004737  026502                  CALL    SUPD02
 7973 021172  000207                          RETURN
```

```
 7976 021174                                MT0006: SUBTST  <<MT0006        SETUP INITIAL DATA TEST>>
                                            ;*****************************************************************************
                                            ;*SUBTEST       MT0006  SETUP INITIAL DATA TEST
                                            ;*****************************************************************************
 7977 021174  012737  000006  002262               MOV     #6,REALPAT              ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
 7978 021202  012737  000004  002300               MOV     #4,PCBUMP               ;TRAPS ADD 4 TO PC
 7979 021210  012701  002364                       MOV     #TESTADD,R1
 7980 021214  012737  027414  002256               MOV     #MTP006,SUPDOADD
 7981 021222  004737  026644                       CALL    SUPD03                  ;DO IT IN SUPERVISOR MODE
 7982 021226  000207                               RETURN
 7983 021230                                MT0007: SUBTST  <<MT0007        SETUP ADDRESS BIT TEST>>
                                            ;*****************************************************************************
                                            ;*SUBTEST       MT0007  SETUP ADDRESS BIT TEST
                                            ;*****************************************************************************
 7984 021230  012737  000007  002262               MOV     #7,REALPAT              ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
 7985 021236  005002                               CLR     R2
 7986 021240  004737  036532                       CALL    BACKGND                 ;OF ZEROS
 7987 021244  012701  060000                       MOV     #FIRST,R1
 7988 021250  012702  000001                       MOV     #1,R2
 7989 021254  050201                               BIS     R2,R1
 7990 021256  012737  027614  002256               MOV     #MTP007,SUPDOADD
 7991 021264  004737  026644                       CALL    SUPD03                  ;DO IT IN SUPERVISOR MODE
 7992 021270  000207                               RETURN
 7993 021272                                MT0010: SUBTST  <<MT0010        SETUP BYTE ADDRESSING TEST>>
                                            ;*****************************************************************************
                                            ;*SUBTEST       MT0010  SETUP BYTE ADDRESSING TEST
                                            ;*****************************************************************************
 7994 021272  012737  000010  002262               MOV     #10,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
 7995 021300  012737  000004  002300               MOV     #4,PCBUMP               ;TRAPS ADD 4 TO PC
 7996 021306  013704  002364                       MOV     TESTADD,R4
 7997 021312  012737  027714  002256               MOV     #MTP010,SUPDOADD
 7998 021320  004737  026644                       CALL    SUPD03                  ;DO IT IN SUPERVISOR MODE
 7999 021324  000207                               RETURN
```

```
8002 021326                                MT0011: SUBTST  <<MT0011        SETUP CREATE SINGLE BIT ERROR TEST>>
                                           ;*****************************************************************************
                                           ;*SUBTEST       MT0011  SETUP CREATE SINGLE BIT ERROR TEST
                                           ;*****************************************************************************
8003 021326                                        IF ACTFLAG IS TRUE OR APTFLAG IS TRUE
8004 021342                                          IF $PASS NE #0 THEN $RETURN
8005 021352                                        END; OF IF ACTFLAG
8006 021352  012737  000011  002262                MOV     #11,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
8007 021360  012737  030022  002256                MOV     #MTP011,SUPDOADD
8008 021366  004737  026644                        CALL    SUPD03                  ;DO IT IN SUPERVISOR MODE
8009 021372  000207                                RETURN
8010 021374                                MT0012: SUBTST  <<MT0012        SETUP WRITE BYTE CLEARS SBE TEST>>
                                           ;*****************************************************************************
                                           ;*SUBTEST       MT0012  SETUP WRITE BYTE CLEARS SBE TEST
                                           ;*****************************************************************************
8011 021374                                        IF ACTFLAG IS TRUE OR APTFLAG IS TRUE
8012 021410                                          IF $PASS NE #0 THEN $RETURN
8013 021420                                        END; OF IF ACTFLAG
8014 021420  012737  000012  002262                MOV     #12,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
8015 021426  013700  002102                        MOV     BANKINDEX,R0
8016 021432                                        IF #BIT12 SET.IN CONFIG+2(R0)
8017 021442  012705  040000                          MOV   #40000,R5
8018 021446                                        ELSE
8019 021450  012705  000002                          MOV   #2,R5
8020 021454                                        END; OF IF #BIT12
8021 021454  012737  030620  002256                MOV     #MTP012,SUPDOADD
8022 021462  004737  026644                        CALL    SUPD03                  ;DO IT IN SUPERVISOR MODE
8023 021466  000207                                RETURN
8024 021470                                MT0013: SUBTST  <<MT0013        SETUP CREATE DOUBLE BIT ERROR TEST>>
                                           ;*****************************************************************************
                                           ;*SUBTEST       MT0013  SETUP CREATE DOUBLE BIT ERROR TEST
                                           ;*****************************************************************************
8025 021470                                        IF ACTFLAG IS TRUE OR APTFLAG IS TRUE
8026 021504                                          IF $PASS NE #0 THEN $RETURN
8027 021514                                        END; OF IF ACTFLAG
8028 021514  012737  000013  002262                MOV     #13,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
8029 021522  012737  031206  002256                MOV     #MTP013,SUPDOADD
8030 021530  012737  000003  002074                MOV     #3,NOPAR                ;INDICATE PARITY ACTION
8031 021536  004737  026644                        CALL    SUPD03                  ;DO IT IN SUPERVISOR MODE
8032 021542  000207                                RETURN
8033 021544                                MT0014: SUBTST  <<MT0014        SETUP WRITE INHIBIT DURING DATIP WITH DBE>>
                                           ;*****************************************************************************
                                           ;*SUBTEST       MT0014  SETUP WRITE INHIBIT DURING DATIP WITH DBE
                                           ;*****************************************************************************
8034 021544                                        IF ACTFLAG IS TRUE OR APTFLAG IS TRUE
8035 021560                                          IF $PASS NE #0 THEN $RETURN
8036 021570                                        END; OF IF ACTFLAG
8037 021570                                        IF KFLAG IS FALSE THEN $RETURN
8038 021600  012737  000014  002262                MOV     #14,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
8039 021606  012737  031722  002256                MOV     #MTP014,SUPDOADD
8040 021614  004737  026644                        CALL    SUPD03                  ;DO IT IN SUPERVISOR MODE
8041 021620  000207                                RETURN
```

```
 8044 021622                                MT0015: SUBTST  <<MT0015        SETUP WRITE INHIBIT OF BYTE WITH DBE>>
                                            ;*****************************************************************************
                                            ;*SUBTEST       MT0015  SETUP WRITE INHIBIT OF BYTE WITH DBE
                                            ;*****************************************************************************
 8045 021622                                        IF ACTFLAG IS TRUE OR APTFLAG IS TRUE
 8046 021636                                          IF $PASS NE #0 THEN $RETURN
 8047 021646                                        END ;OF IF ACTFLAG
 8048 021646  012737  000015  002262                MOV     #15,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
 8049 021654  012737  032504  002256                MOV     #MTP015,SUPDOADD
 8050 021662  004737  026644                        CALL    SUPD03                  ;DO IT IN SUPERVISOR MODE
 8051 021666  000207                                RETURN
 8052 021670                                MT0016: SUBTST  <<MT0016        SETUP WRITE INHIBIT OF WORD WITH DBE>>
                                            ;*****************************************************************************
                                            ;*SUBTEST       MT0016  SETUP WRITE INHIBIT OF WORD WITH DBE
                                            ;*****************************************************************************
 8053 021670                                        IF ACTFLAG IS TRUE OR APTFLAG IS TRUE
 8054 021704                                          IF $PASS NE #0 THEN $RETURN
 8055 021714                                        END ;OF IF ACTFLAG
 8056 021714  012737  000016  002262                MOV     #16,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
 8057 021722  012737  033250  002256                MOV     #MTP016,SUPDOADD
 8058 021730  004737  026644                        CALL    SUPD03                  ;DO IT IN SUPERVISOR MODE
 8059 021734  000207                                RETURN
 8060 021736                                MT0017: SUBTST  <<MT0017        SETUP HOLDING 1'S & 0'S>>
                                            ;*****************************************************************************
                                            ;*SUBTEST       MT0017  SETUP HOLDING 1'S & 0'S
                                            ;*****************************************************************************
 8061 021736  012737  000017  002262                MOV     #17,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
 8062 021744  012737  034032  002256                MOV     #MTP017,SUPDOADD
 8063 021752  004737  026644                        CALL    SUPD03                  ;DO IT IN SUPERVISOR MODE
 8064 021756  000207                                RETURN
```

```
8067 021760                                  MT0020: SUBTST  <<MT0020        SETUP MARCHING 0'S & 1'S IN CHECKBITS TEST>>
                                             ;*****************************************************************************
                                             ;*SUBTEST       MT0020  SETUP MARCHING 0'S & 1'S IN CHECKBITS TEST
                                             ;*****************************************************************************
8068 021760                                          IF ACTFLAG IS TRUE OR ACTFLAG IS TRUE
8069 021774                                            IF $PASS NE #0 THEN $RETURN
8070 022004                                          END ;OF IF ACTFLAG
8071 022004  012737  000020  002262                  MOV     #20,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
8072 022012  012737  000003  002074                  MOV     #3,NOPAR                ;INDICATE PARITY ACTION
8073 022020  005001                                  CLR     R1                      ;CLEAR LOOP COUNTER
8074 022022  005004                                  CLR     R4                      ;CLEAR INTERLEAVE ODD/EVEN FLAG
8075 022024  012700  060000                        MOV     #FIRST,R0
8076 022030  013702  002102                        MOV     BANKINDEX,R2            ;SET BANK INDEX
8077 022034                                  MTL020: IF INTFLAG IS FALSE
8078 022042                                            BEGIN MTB020
8079 022042                                            IF NO22BIT IS TRUE
8080 022050                                              IF BANK EQ #7
8081 022060  012737  140000  002364                        MOV       #140000,TESTADD ;SET UP 12K NON INTERLEAVED VIRT ADDR
8082 022066  012705  140002                              MOV       #140002,R5
8083 022072                                              LEAVE     MTB020
8084 022074                                            END; OF IF BANK
8085 022074                                            END; OF IF NO22BIT
8086 022074                                            IF NO22BIT IS FALSE
8087 022102                                              IF BANK EQ #177
8088 022112  012737  140000  002364                        MOV       #140000,TESTADD
8089 022120  012705  140000                              MOV       #140000,R5
8090 022124                                              LEAVE     MTB020
8091 022126                                            END; OF IF BANK
8092 022126                                            END; OF IF NO22BIT
8093 022126  012737  160000  002364                      MOV   #LAST+2,TESTADD
8094 022134  012705  160002                            MOV   #LAST+4,R5
8095 022140                                            END   MTB020
8096 022140  010537  002366                            MOV   R5,TESTADD+2              ;SET UP NON-INTERLEAVED VIRT. ADDR.
8097 022144                                          ELSE
8098 022146  005737  002314                            TST   SKIPMK                    ;IS THIS BANK IN SKIP RANGE?
8099 022152  001401                                    BEQ   1$                        ;BANK IS OUT OF RANGE   DO TEST
8100 022154  000207                                    RETURN                          ;LEAVE TEST-BANK'S ALREADY TESTED
8101 022156  012737  120000  002364  1$:               MOV   #120000,TESTADD           ;SET UP 1ST INTERLEAVED VIRT. ADDR.
8102 022164  012705  160000                            MOV   #LAST+2,R5                ;SET UP END OF BANK FLAG
8103 022170  010537  002366                            MOV   R5,TESTADD+2              ;SET UP 2ND INT'L. VIRT. ADDR.
8104 022174  005237  002234                            INC   SPLTCSR                   ;FLAG THE MAPPING ROUTINE FOR INTERLEAVING
8105 022200  005201                                    INC   R1                        ;SET LOOP COUNTER FOR INTERLEAVING
8106 022202  005204                                    INC   R4                        ;SET ODD/EVEN FLAG
8107 022204                                          END; OF IF INTFLAG
8108 022204  016203  002626                          MOV     CONFIG(R2),R3           ;SET UP CSR NUMBER
8109 022210                                          IF R4 EQ #2                     ;IF THE SECOND TIME AROUND
8110 022216  060437  002364                            ADD   R4,TESTADD
8111 022222  060437  002366                            ADD   R4,TESTADD+2            ;TEST THE A1 ASSERTED ADDRESSES
8112 022226  060400                                    ADD   R4,R0
8113 022230  060405                                    ADD   R4,R5
8114 022232  072327  177775                            ASH   #-3,R3                  ;MOVE INTERLEAVED CSR NUMBER
8115 022236                                          ELSE
8116 022240  006303                                    ASL   R3                      ;MOVE CSR NUMBER
8117 022242                                          END; IF R4
8118 022242  000303                                  SWAB    R3
8119 022244  042703  177741                          BIC     #^C36,R3
8120 022250  010337  002146                          MOV     R3,CSRNO                ;MOVE R3 INTO CSR NUMBER
```

```
 8121 022254                                IF @SW0 SET.IN @SWR
 8122 022264  104506                          ENASBE                          ;TRAP ON SINGLE BIT ERRORS
 8123 022266                                ELSE
 8124 022270  104472                          ECCINIT                         ;TRAP ON UNCORRECTABLE ERRORS
 8125 022272                                END; OF IF @SW0
 8126 022272                                PUSH    R2,R4
 8127 022276                                FOR MTV020 := @0 TO R1
 8128 022302                                  PUSH  R1
 8129 022304  005002                          CLR   R2                        ;PATTERN TO WRITE INTO BANK
 8130 022306  004737  036532                  CALL  BACKGND                   ;SET UP ZEROS IN BANK
 8131 022312                                  IF NO22BIT IS TRUE AND MTV020 EQ @1 AND BANK EQ @3
 8132 022340  162737  020000  002364            SUB @20000,TESTADD            ;SET UP 12K INTERLEAVED BANK
 8133 022346  162705  020000                    SUB @20000,R5
 8134 022352  010537  002366                    MOV R5,TESTADD+2
 8135 022356                                  END; OF IF NO22BIT
 8136 022356  004737  022432                  CALL  MT020Z                    ;START TEST
 8137 022362  005237  002234                  INC   SPLTCSR                   ;UPDATE INTERLEAVED MAPPING FLAG
 8138 022366                                  POP   R1
 8139 022370                                END; OF FOR MTV020
 8140 022402                                POP     R4,R2
 8141 022406  005001                        CLR     R1                        ;RESET LOOP FLAG
 8142 022410  005037  002234                CLR     SPLTCSR                   ;RESET INTERLEAVED MAP FLAG
 8143 022414  022704  000001                CMP     @1,R4                     ;ODD/EVEN FLAG SET?
 8144 022420  001605                        BEQ     MTL020                    ;BRANCH IF TRUE
 8145 022422  005037  002074                CLR     NOPAR                     ;INDICATE PARITY ACTION
 8146 022426  000207                        RETURN
 8147 022430  000000              MTV020:   0                                 ;VARIABLE FOR PAT 20
```

```
8149 022432  012702  000004          MT020Z: MOV     #4,R2                   ;SET UP WORD INCR/DECR AMOUNT
8150 022436  013701  002364                  MOV     TESTADD,R1
8151 022442  013704  002366                  MOV     TESTADD+2,R4
8152 022446  012703  100000                  MOV     #BIT15,R3
8153 022452                                  IF #SW11 SET.IN @SWR OR QVFLAG IS TRUE
8154 022470                                    GOTO  MT020Y
8155 022472                                  END ;OF IF #SW11
8156 022472  022737  000001  003720          CMP     #1,PROTYP               ;IS THIS AN 11/44?
8157 022500  001411                          BEQ     1$                      ;BRANCH IF IT IS
8158 022502  012737  034110  002256          MOV     #MTPA20,SUPDOADD
8159 022510  012737  034124  002266          MOV     #MTPA20+14,PARTHERE     ;VECTOR FOR TRAPS
8160 022516  004737  026644                  CALL    SUPD03
8161 022522  000410                          BR      2$
8162 022524                          1$:     BMOV    MTPA20
8163 022532  012737  177654  002266          MOV     #UIPAR6,PARTHERE        ;VECTOR FOR TRAPS
8164 022540  004737  026466                  CALL    SUPD01
8165 022544  022737  000001  003720  2$:     CMP     #1,PROTYP               ;IS THIS AN 11/44?
8166 022552  001411                          BEQ     4$                      ;BRANCH IF IT IS
8167 022554  012737  034140  002256          MOV     #MTPB20,SUPDOADD
8168 022562  012737  034150  002266          MOV     #MTPB20+10,PARTHERE     ;VECTOR FOR TRAPS
8169 022570  004737  026660                  CALL    SUPD04
8170 022574  000410                          BR      MT020Y
8171 022576                          4$:     BMOV    MTPB20
8172 022604  012737  177650  002266          MOV     #UIPAR4,PARTHERE        ;VECTOR FOR TRAPS
8173 022612  004737  026502                  CALL    SUPD02
8174 022616  005737  002134          MT020Y: TST     INTFLAG                 ;ARE WE INTERLEAVED?
8175 022622  001405                          BEQ     7$                      ;BRANCH IF NOT INTERLEAVED
8176 022624  162701  040000                  SUB     #40000,R1               ;RESET FIRST WORD TO BEGINNING OF BANK
8177 022630  162704  040000                  SUB     #40000,R4               ;RESET SECOND WORD TO BEGINNING OF BANK
8178 022634  000404                          BR      8$
8179 022636  012701  060000          7$:     MOV     #FIRST,R1               ;RESET FIRST WORD TO BEGINNING OF BANK
8180 022642  012704  060002                  MOV     #FIRST+2,R4             ;RESET SECOND WORD TO BEGINNING OF BANK
8181 022646  022737  000001  003720  8$:     CMP     #1,PROTYP               ;IS THIS AN 11/44?
8182 022654  001411                          BEQ     1$                      ;BRANCH IF IT IS
8183 022656  012737  034170  002256          MOV     #MTPC20,SUPDOADD
8184 022664  012737  034200  002266          MOV     #MTPC20+10,PARTHERE     ;VECTOR FOR TRAPS
8185 022672  004737  026644                  CALL    SUPD03
8186 022676  000410                          BR      2$
8187 022700                          1$:     BMOV    MTPC20
8188 022706  012737  177650  002266          MOV     #UIPAR4,PARTHERE        ;VECTOR FOR TRAPS
8189 022714  004737  026466                  CALL    SUPD01
8190 022720  022737  000001  003720  2$:     CMP     #1,PROTYP               ;IS THIS AN 11/44?
8191 022726  001411                          BEQ     3$                      ;BRANCH IF IT IS
8192 022730  012737  034220  002256          MOV     #MTPD20,SUPDOADD
8193 022736  012737  034234  002266          MOV     #MTPD20+14,PARTHERE     ;VECTOR FOR TRAPS
8194 022744  004737  026660                  CALL    SUPD04
8195 022750  000410                          BR      4$
8196 022752                          3$:     BMOV    MTPD20
8197 022760  012737  177654  002266          MOV     #UIPAR6,PARTHERE        ;VECTOR FOR TRAPS
8198 022766  004737  026502                  CALL    SUPD02
8199 022772  022737  000001  003720  4$:     CMP     #1,PROTYP               ;IS THIS AN 11/44?
8200 023000  001411                          BEQ     5$                      ;BRANCH IF IT IS
8201 023002  012737  034250  002256          MOV     #MTPE20,SUPDOADD
8202 023010  012737  034260  002266          MOV     #MTPE20+10,PARTHERE     ;VECTOR FOR TRAPS
8203 023016  004737  026660                  CALL    SUPD04
8204 023022  000410                          BR      6$
8205 023024                          5$:     BMOV    MTPE20
```

```
 8206 023032  012737  177650  002266          MOV     @UIPAR4,PARTHERE        ;VECTOR FOR TRAPS
 8207 023040  004737  026502                  CALL    SUPD02
 8208 023044  104503                  6$:    CLR1CSR                         ;CLEAR 1 SELECTED CSR
 8209 023046  000207                         RETURN
```

```
8212 023050                                MT0021: SUBTST  <<MT0021        SETUP MARCHING 0'S & 1 S TEST>>
                                           ;*****************************************************************
                                           ;*SUBTEST        MT0021  SETUP MARCHING 0'S & 1'S TEST
                                           ;*****************************************************************
8213 023050                                        SET NOSCOPE
8214 023056  012737  000021  002262                MOV     #21,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLA
8215 023064  013702  002574                        MOV     BAKPAT,R2
8216 023070  004737  036532                        CALL    BACKGND
8217 023074  010203                                MOV     R2,R3
8218 023076  000303                                SWAB    R3
8219 023100  012701  160000                        MOV     #LAST-2,R1
8220 023104  010105                                MOV     R1,R5
8221 023106  012704  060000                        MOV     #FIRST,R4
8222 023112  022737  000001  003720                CMP     #1,PROTYP       ;IS THIS AN 11/44?
8223 023120  001441                                BEQ     1$              ;BRANCH IF IT IS
8224 023122  022737  000003  003720                CMP     #3,PROTYP       ;IS THIS AN 11/24?
8225 023130  001407                                BEQ     3$              ;BRANCH IF SO
8226 023132  022737  000007  002100                CMP     #7,BANK
8227 023140  001003                                BNE     3$
8228 023142  012701  140000                        MOV     #140000,R1
8229 023146  010105                                MOV     R1,R5
8230 023150  012737  034274  002256  3$:           MOV     #MTPA21,SUPDOADD
8231 023156  004737  026644                        CALL    SUPD03
8232 023162  012737  034324  002256                MOV     #MTPB21,SUPDOADD
8233 023170  004737  026660                        CALL    SUPD04
8234 023174  010401                                MOV     R4,R1
8235 023176  012737  034360  002256                MOV     #MTPC21,SUPDOADD
8236 023204  004737  026660                        CALL    SUPD04
8237 023210  012737  034414  002256                MOV     #MTPD21,SUPDOADD
8238 023216  004737  026660                        CALL    SUPD04
8239 023222  000434                                BR      2$
8240 023224  022737  000177  002100  1$:           CMP     #177,BANK
8241 023232  001003                                BNE     4$
8242 023234  012701  140000                        MOV     #140000,R1
8243 023240  010105                                MOV     R1,R5
8244 023242                              4$:       BMOV    MTPA21
8245 023250  004737  026466                        CALL    SUPD01
8246
8247 023254                                        BMOV    MTPB21
8248 023262  004737  026502                        CALL    SUPD02
8249
8250 023266  010401                                MOV     R4,R1
8251 023270                                        BMOV    MTPC21
8252 023276  004737  026502                        CALL    SUPD02
8253
8254 023302                                        BMOV    MTPD21
8255 023310  004737  026502                        CALL    SUPD02
8256 023314  005037  002412                2$:     CLR     NOSCOPE
8257 023320  000207                                RETURN
```

```
8259 023322                                MT0022: SUBTST  <<MT0022        SETUP REFRESH & SHIFTING DIAGONAL TEST>>
                                           ;*****************************************************************************
                                           ;*SUBTEST       MT0022  SETUP REFRESH & SHIFTING DIAGONAL TEST
                                           ;*****************************************************************************
8260 023322  004737  026432                        CALL    KAMITEST                ;CHECK FOR KAMIKAZE MODE
8261 023326                                        ON.ERROR THEN $RETURN           ;IF NOT IN KAMIKAZE MODE RETURN
8262 023332  012737  000022  002262                MOV     #22,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
8263 023340  012737  034444  002256                MOV     #MTP022,SUPDOADD
8264 023346  004737  026644                        CALL    SUPDO3                  ;DO IT IN SUPERVISOR MODE
8265 023352  000207                              RETURN
8266
8267 023354                                MT0023: SUBTST  <<MT0023        SHIFTING DIAGONAL TEST>>
                                           ;*****************************************************************************
                                           ;*SUBTEST       MT0023  SHIFTING DIAGONAL TEST
                                           ;*****************************************************************************
8268 023354  004737  026432                        CALL    KAMITEST                ;CHECK FOR KAMIKAZE MODE
8269 023360                                        ON.ERROR THEN $RETURN           ;IF NOT IN KAMIKAZE MODE RETURN
8270 023364  012737  000023  002262                MOV     #23,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
8271 023372  012737  034444  002256                MOV     #MTP022,SUPDOADD
8272 023400                                        SET     DIAGFLAG                ;IDENTIFY DIAGONAL TEST TO MTP022
8273 023406  004737  026644                        CALL    SUPDO3                  ;DO IT IN SUPERVISOR MODE
8274 023412  005037  002002                        CLR     DIAGFLAG
8275 023416  000207                              RETURN
```

```
 8277 023420                                MT0024: SUBTST  <<MT0024        SETUP FAST GALLOPING PATTERN TEST>>
                                            ;*****************************************************************************
                                            ;*SUBTEST       MT0024  SETUP FAST GALLOPING PATTERN TEST
                                            ;*****************************************************************************
 8278 023420  004737  026432                        CALL    KAMITEST                ;CHECK FOR KAMIKAZE MODE
 8279 023424                                        ON.ERROR THEN $RETURN           ;IF NOT IN KAMIKAZE MODE RETURN
 8280 023430                                        SET     NOSCOPE
 8281 023436  012737  000024  002262                MOV     #24,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
 8282 023444  013702  002574                        MOV     BAKPAT,R2
 8283 023450  004737  036532                        CALL    BACKGND
 8284 023454  010203                                MOV     R2,R3
 8285 023456  010304                                MOV     R3,R4
 8286 023460  000304                                SWAB    R4
 8287 023462  012701  060000                        MOV     #FIRST,R1
 8288 023466  012705  157776                        MOV     #LAST,R5
 8289 023472  022737  000001  003720                CMP     #1,PROTYP
 8290 023500  001417                                BEQ     1$
 8291 023502  022737  000003  003720                CMP     #3,PROTYP
 8292 023510  001406                                BEQ     3$
 8293 023512  022737  000007  002100                CMP     #7,BANK
 8294 023520  001002                                BNE     3$
 8295 023522  012705  137776                        MOV     #137776,R5
 8296 023526  104415                        3$:     SAVREG
 8297 023530  012737  035160  002256                MOV     #MTPB24,SUPDOADD
 8298 023536  000440                                BR      2$
 8299 023540  022737  000177  002530        1$:     CMP     #177,LASTBANK
 8300 023546  001002                                BNE     4$
 8301 023550  012705  137776                        MOV     #137776,R5
 8302 023554  104415                        4$:     SAVREG
 8303 023556                                        BMOV    MTPA24
 8304 023564                                        BMOV    MTPB24,SDPAR0,8.
 8305 023576                                        BMOV    MTPC24,KDPAR0,8.
 8306 023610  012737  172260  002256                MOV     #SDPAR0,SUPDOADD
 8307 023616  012737  172260  177676                MOV     #SDPAR0,UDPAR7          ;SET UP PAR LINKS
 8308 023624  012737  172360  172272                MOV     #KDPAR0,SDPAR5
 8309 023632  012737  177660  172374                MOV     #UDPAR0,KDPAR6
 8310 023640  004737  026660                2$:     CALL    SUPD04
 8311
 8312                                               ;DO IT AGAIN FOR COMPLEMENT DATA
 8313 023644  104416                                RESREG
 8314 023646  000302                                SWAB    R2
 8315 023650  000303                                SWAB    R3
 8316 023652  004737  026660                        CALL    SUPD04
 8317 023656  005037  002412                        CLR     NOSCOPE
 8318 023662  000207                                RETURN
 8319 023664                                MT0025: SUBTST  <<MT0025        SETUP INTERRUPT ENABLE TEST>>
                                            ;*****************************************************************************
                                            ;*SUBTEST       MT0025  SETUP INTERRUPT ENABLE TEST
                                            ;*****************************************************************************
 8320 023664                                        IF ACTFLAG IS TRUE OR APTFLAG IS TRUE
 8321 023700                                          IF $PASS NE #0 THEN $RETURN
 8322 023710                                        END ;OF IF ACTFLAG
 8323 023710  012737  000025  002262                MOV     #25,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
 8324 023716  012737  035212  002256                MOV     #MTP025,SUPDOADD
 8325 023724  004737  026644                        CALL    SUPD03                  ;DO IT IN SUPERVISOR MODE
 8326 023730  000207                                RETURN
```

```
 8329 023732                                MT0026: SUBTST  <<MT0026        SETUP RANDOM DATA TEST>>
                                            ;*****************************************************************
                                            ;*SUBTEST       MT0026  SETUP RANDOM DATA TEST
                                            ;*****************************************************************
 8330 023732  012737  000026  002262               MOV     #26,REALPAT
 8331 023740  005037  002300                        CLR     PCBUMP                  ;TRAPS DO NOT ADD TO THE PC
 8332 023744  013703  002546                        MOV     SEEDLO,R3               ;INITIALIZE RANDOM NUMBERS
 8333 023750  013702  002544                        MOV     SEEDHI,R2
 8334 023754  010305                               MOV     R3,R5
 8335 023756  010204                               MOV     R2,R4
 8336 023760  012701  060000                        MOV     #FIRST,R1
 8337 023764  012700  020000                        MOV     #SIZE/2,R0
 8338 023770  022737  000001  003720               CMP     #1,PROTYP               ;DO WE HAVE AN 11/44?
 8339 023776  001437                               BEQ     1$                      ;BRANCH IF WE DO
 8340 024000  022737  000003  003720               CMP     #3,PROTYP               ;11/24?
 8341 024006  001406                               BEQ     3$                      ;BRANCH IF SO
 8342 024010  022737  000007  002100               CMP     #7,BANK
 8343 024016  001002                               BNE     3$
 8344 024020  012700  014000                        MOV     #14000,R0
 8345 024024  104415                      3$:      SAVREG
 8346 024026  012737  035664  035764               MOV     #MTPA26+4,MTPD26+14
 8347 024034  012737  035660  002256               MOV     #MTPA26,SUPDOADD
 8348 024042  004737  026644                        CALL    SUPDO3
 8349 024046  005037  035710                        CLR     RANDDO                  ;FOR ERROR REPORTING
 8350 024052  012737  035700  035764               MOV     #MTPB26+4,MTPD26+14     ;SET UP NEXT LINK
 8351 024060  012737  035674  002256               MOV     #MTPB26,SUPDOADD
 8352 024066  104416                               RESREG
 8353 024070  004737  026644                        CALL    SUPDO3
 8354 024074  000452                               BR      2$
 8355 024076  022737  000177  002100  1$:      CMP     #177,BANK
 8356 024104  001002                               BNE     4$
 8357 024106  012700  014000                        MOV     #14000,R0
 8358 024112  104415                      4$:      SAVREG
 8359 024114                                        BMOV    MTPA26                  ;WRITE ROUTINE TO FAST MEMORY
 8360 024122                                        BMOV    MTPC26,KDPAR0,8.        ;RANDOM SUBPROGRAM TO FAST MEMORY
 8361 024134  012737  000730  172376               MOV     #730,KDPAR7             ;WRITES "BR .-116" IN (BR SDPAR0)
 8362 024142                                        BMOV    MTPD26,SDPAR0,8.        ;RANDOM SUBSUBPROGRAM TO FAST MEMORY
 8363 024154  012737  172360  177642               MOV     #KDPAR0,UIPAR1
 8364 024162  012737  177644  172274               MOV     #UIPAR2,SDPAR6
 8365 024170  004737  026466                        CALL    SUPDO1                  ;WRITE RANDOM DATA
 8366 024174  005037  035710                        CLR     RANDDO                  ;FOR ERROR REPORTING
 8367 024200                                        BMOV    MTPB26                  ;READ ROUTINE TO FAST MEMORY
 8368 024206  012737  172360  177642               MOV     #KDPAR0,UIPAR1          ;SET UP PAR LINK
 8369 024214  104416                               RESREG
 8370 024216  004737  026466                        CALL    SUPDO1                  ;READ RANDOM DATA
 8371 024222  010337  002546              2$:      MOV     R3,SEEDLO               ;UPDATE FOR NEW RANDOM NUMBERS
 8372 024226  010237  002544                        MOV     R2,SEEDHI
 8373 024232  000207                               RETURN
```

```
8376 024234                                MT0027: SUBTST  <<MT0027        UNIQUE BANK TEST>>
                                           ;*****************************************************************************
                                           ;*SUBTEST       MT0027  UNIQUE BANK TEST
                                           ;*****************************************************************************
8377                                               ;MAKE SURE THAT EACH BANK CAN HAVE UNIQUE DATA
8378                                               ;WRITE AND READ THE BANK NUMBER IN EACH BANK (EXCEPT WHERE THE PROGRAM IS)
8379 024234  012737  000027  002262          MOV     #27,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
8380 024242  104502                        CLRCSR                  ;CLEAR CSRS
8381 024244  022737  000001  003720          CMP     #1,PROTYP               ;IS THIS AN 11/44?
8382 024252  001404                          BEQ     1$                      ;BRANCH IF TRUE
8383 024254  012737  026644  002474          MOV     #SUPD03,LINK1           ;SET UP LINK
8384 024262  000414                          BR      STAR27                  ;BRANCH TO RUN
8385 024264                                1$:     BMOV    MTP034
8386 024272  012737  177646  002256    WARN7:  MOV     #UIPAR3,SUPDOADD
8387 024300  012737  026466  002474          MOV     #SUPD01,LINK1           ;SET UP LINK
8388 024306                                      SET NOFSMODE
8389 024314                                STAR27: FOR I := #1 TO #2
8390 024322                                          FOR BANK := #0 TO LASTBANK
8391 024326  004737  044302                             CALL EXBANK
8392 024332                                             IF ACFLAG IS TRUE AND RRFLAG IS FALSE
8393 024346  104511                                       INVALIDATE            ;INVALIDATE BACKGROUND PATTERN ON  BANK
8394 024350                                               LET R2 := BANK
8395 024354  012700  060000                               MOV     #FIRST,R0
8396 024360  010004                                       MOV     R0,R4
8397 024362  012701  040000                               MOV     #SIZE,R1
8398 024366  010103                                       MOV     R1,R3
8399 024370                                               IF I EQ #1
8400 024400  022737  000001  003720                         CMP     #1,PROTYP
8401 024406  001403                                         BEQ     2$
8402 024410  012737  036204  002256                         MOV     #MTP034,SUPDOADD
8403 024416  004777  156052                            2$: CALL    @LINK1
8404 024422                                               END ;OF IF
8405 024422                                               IF I EQ #2
8406 024432  022737  000001  003720                         CMP     #1,PROTYP
8407 024440  001403                                         BEQ     3$
8408 024442  012737  036212  002256                         MOV     #MTP034+6,SUPDOADD
8409 024450  004737  026644                            3$: CALL    SUPD03
8410 024454                                               END ;OF IF
8411 024454                                             END ;OF IF
8412 024454                                          END ;OF FOR BANK
8413 024470                                     END ;OF FOR I
8414 024504                                     IF FS7FLAG IS TRUE
8415 024512  005037  002400                       CLR   NOFSMODE
8416 024516  000207                              RETURN
8417 024520                                     END ;OF IF FS7FLAG
8418 024520                                     FOR I := #1 TO #2
8419 024526                                       FOR BANK := LASTBANK DOWNTO #0
8420 024534  004737  044302                         CALL EXBANK
8421 024540                                         IF ACFLAG IS TRUE AND RRFLAG IS FALSE
8422 024554                                           LET R2 := BANK
8423 024560  005102                                   COM     R2
8424 024562  012700  060000                           MOV     #FIRST,R0
8425 024566  010004                                   MOV     R0,R4
8426 024570  012701  040000                           MOV     #SIZE,R1
8427 024574  010103                                   MOV     R1,R3
8428 024576                                           IF I EQ #1
8429 024606  022737  000001  003720                     CMP     #1,PROTYP
```

```
8430 024614  001403                          BEQ     4$
8431 024616  012737  036204  002256          MOV     #MTP034,SUPDOADD
8432 024624  004777  155644              4$: CALL    @LINK1
8433 024630                                  END ;OF IF
8434 024630                                  IF I EQ #2
8435 024640  022737  000001  003720          CMP     #1,PROTYP
8436 024646  001403                          BEQ     5$
8437 024650  012737  036212  002256          MOV     #MTP034+6,SUPDOADD
8438 024656  004737  026644              5$: CALL    SUPD03
8439 024662                                  END ;OF IF
8440 024662                              END ;OF IF
8441 024662                          END ;OF FOR BANK
8442 024676                      END ;OF FOR I
8443 024712  005037  002400      CLR     NOFSMODE
8444 024716  000207              RETURN
```

```
8447 024720                                MT0030: SUBTST  <<MT0030        SETUP FLUSH OUT DBE'S TEST>>
                                           ;*****************************************************************************
                                           ;*SUBTEST       MT0030  SETUP FLUSH OUT DBE'S TEST
                                           ;*****************************************************************************
8448 024720  005037  002260                        CLR     PASFLG
8449 024724                                        SET     FULLREL
8450 024732  012737  000030  002262  MTA030: MOV     #30,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
8451 024740  012737  000001  002074        MOV     #1,NOPAR                ;INDICATE COUNT PARITY ERRORS
8452 024746  022737  000001  003720        CMP     #1,PROTYP
8453 024754  001007                        BNE     4$
8454 024756                                BMOV    MTP030
8455 024764  012737  026466  002474        MOV     #SUPD01,LINK1
8456 024772  000406                        BR      1$
8457 024774  012737  026644  002474  4$:   MOV     #SUPD03,LINK1
8458 025002  012737  035766  002256        MOV     #MTP030,SUPDOADD
8459 025010  104470                  1$:   ECCDIS                          ;DISABLE ERROR CORRECTION
8460 025012                                SET     NOFSMODE,NOSCOPE
8461 025026                                FOR BANK := #0 TO LASTBANK
8462 025032  004737  044302                  CALL  EXBANK
8463 025036                                  IF MKFLAG IS TRUE
8464 025044                                    IF ACFLAG IS TRUE AND RRFLAG IS FALSE
8465 025060  012701  040000                      MOV       #SIZE,R1
8466 025064  012700  060000                      MOV       #FIRST,R0
8467 025070  004777  155400                      CALL      @LINK1
8468 025074                                    END ;OF IF ACFLAG
8469 025074                                  END ;OF IF MKFLAG
8470 025074                                END ;OF FOR
8471 025110                                IF PASFLG IS FALSE
8472 025116                                  SET PASFLG
8473 025124  104502                          CLRCSR                          ;CLEAR CSRS
8474 025126  004737  042530                  CALL  RELOCATE
8475 025132                                  ON.ERROR
8476 025134  104472                            ECCINIT                 ;TRAP ON DOUBLE BIT ERRORS (NORMAL)
8477 025136                                    CLEAR       NOFSMODE,NOSCOPE,FULLREL
8478 025152  000207                          RETURN
8479 025154                                  END ;OF ON.ERROR
8480 025154  013737  002272  002100          MOV   NEWBANK,BANK
8481 025162  004737  044302                  CALL  EXBANK
8482 025166  004737  024732                  CALL  MTA030
8483 025172  104472                          ECCINIT                         ;TRAP ON DOUBLE BIT ERRORS (NORMAL)
8484 025174  004737  043420                  CALL  UNRELOCATE
8485 025200  000207                          RETURN
8486 025202                                END ;OF IF PASFLG
8487 025202  104472                       ECCINIT                  ;TRAP ON DOUBLE BIT ERRORS (NORMAL)
8488 025204                                CLEAR   NOFSMODE,NOSCOPE,FULLREL
8489 025220  000207                       RETURN
```

```
 8492 025222                                MT0031: SUBTST  <<MT0031        SETUP SOB-A-LONG TEST>>
                                            ;*****************************************************************************
                                            ;*SUBTEST       MT0031  SETUP SOB-A-LONG TEST
                                            ;*****************************************************************************
 8493 025222  004737  026432                        CALL    KAMITEST                ;CHECK FOR KAMIKAZE MODE
 8494 025226                                        ON.ERROR THEN $RETURN           ;IF NOT IN KAMIKAZE MODE RETURN
 8495 025232                                        SET     NOSCOPE
 8496 025240  012737  000031  002262               MOV     #31,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
 8497 025246  005037  002074                        CLR     NOPAR                   ;SETUP PARITY ACTION
 8498 025252                                        MAP     BANK                    ;MAP FIRST SO BLOCK MOVE WORKS
 8499 025266                                        TESTAREA                        ;ENTER TEST MODE
 8500 025274                                        BMOV    MTP031,FIRST,SOBLENGTH/2
 8501 025306  104417                                KERNEL                          ;ENTER KERNEL MODE
 8502 025310  013702  002534                        MOV     SOBK,R2
 8503 025314  010200                                MOV     R2,R0
 8504 025316  012701  100776                        MOV     #100776,R1              ;COMPLEMENT OF INSTRUCTION 'SOB R0,.0T"
 8505 025322  012705  060056                        MOV     #FIRST+SOBLENGTH,R5
 8506 025326  012737  060002  002256               MOV     #FIRST+2,SUPDOADD
 8507 025334  012737  160000  002474               MOV     #LAST+2,LINK1
 8508 025342  005737  002430                        TST     NOSUPER
 8509 025346  001005                                BNE     1$
 8510 025350  023737  172252  172254               CMP     SIPAR5,SIPAR6
 8511 025356  001405                                BEQ     2$
 8512 025360  000407                                BR      3$
 8513 025362  023737  177652  177654  1$:          CMP     UIPAR5,UIPAR6
 8514 025370  001003                                BNE     3$
 8515 025372  012737  140000  002474  2$:          MOV     #140000,LINK1
 8516 025400  004737  026660            3$:         CALL    SUPD04
 8517 025404  005037  002412                        CLR     NOSCOPE
 8518 025410  000207                                RETURN
```

```
8521 025412                                MT0032: SUBTST  <<MT0032        SETUP WRITE RECOVERY TEST>>
                                           ;*****************************************************************************
                                           ;*SUBTEST       MT0032  SETUP WRITE RECOVERY TEST
                                           ;*****************************************************************************
8522 025412  004737  026432                        CALL    KAMITEST                ;CHECK FOR KAMIKAZE MODE
8523 025416                                        ON.ERROR THEN $RETURN           ;IF NOT IN KAMIKAZE MODE RETURN
8524 025422                                        SET     NOSCOPE
8525 025430  012737  000032  002262                MOV     #32,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
8526 025436  005037  002074                        CLR     NOPAR                   ;SETUP PARITY ACTION
8527 025442                                        MAP     BANK                    ;MAP FIRST SO THAT THE BLOCK MOVE WORKS
8528 025456  012700  010247                        MOV     #10247,R0               ;OP CODE OF INSTRUCTION "MOV    R2,-(PC)"
8529 025462  012701  177667                        MOV     #177667,R1              ;OP CODE OF COMPLEMENT OF INSTRUCTION "JMP (R0)"
8530 025466  012702  020000                        MOV     #SIZE/2,R2              ;USED FOR  /2 BANK LOOP
8531 025472  010237  002474                        MOV     R2,LINK1
8532 025476  012703  060000                        MOV     #FIRST,R3
8533 025502  012704  160000                        MOV     #LAST+2,R4
8534 025506  005037  002476                        CLR     LINK2
8535 025512  005737  002430                        TST     NOSUPER
8536 025516  001005                               BNE     1$
8537 025520  023737  172252  172254                CMP     SIPAR5,SIPAR6
8538 025526  001405                               BEQ     2$
8539 025530  000415                               BR      3$
8540 025532  023737  177652  177654  1$:          CMP     UIPAR5,UIPAR6
8541 025540  001011                               BNE     3$
8542 025542  012704  140000          2$:          MOV     #140000,R4
8543 025546  012702  014000                        MOV     #14000,R2
8544 025552  010237  002474                        MOV     R2,LINK1
8545 025556  012737  000001  002476                MOV     #1,LINK2
8546
8547 025564                             3$:        TESTAREA                        ;ENTER TEST MODE
8548                                               ;MOVE TEST TO MEMORY UNDER TEST
8549 025572  010023                     4$:        MOV     R0,(R3)+
8550 025574  010144                                MOV     R1,-(R4)
8551 025576  077203                                SOB     R2,4$
8552
8553 025600  022737  000001  003720                CMP     #1,PROTYP
8554 025606  001003                               BNE     5$
8555                                               ;MOVE LAST PART OF TEST TO FASTCITY
8556 025610                                        BMOV    MTP032
8557 025616  104417                     5$:        KERNEL                          ;ENTER KERNEL MODE
8558
8559 025620  012702  005141                        MOV     #5141,R2                ;OP CODE OF INSTRUCTION "COM     (R1)"
8560 025624  012700  025742                        MOV     #10$,R0                 ;ADDRESS TO RETURN TO IN R0
8561 025630  012701  160000                        MOV     #LAST+2,R1              ;TOP OF BANK
8562 025634  012737  060000  002256                MOV     #FIRST,SUPDOADD
8563 025642  005737  002476                        TST     LINK2
8564 025646  001402                               BEQ     6$
8565 025650  012701  140000                        MOV     #140000,R1
8566 025654  004737  026660          6$:          CALL    SUPDO4
8567 025660  012703  020000                        MOV     #SIZE/2,R3
8568 025664  012705  000110                        MOV     #110,R5
8569 025670  012704  060000                        MOV     #FIRST,R4
8570 025674  005737  002476                        TST     LINK2
8571 025700  001402                               BEQ     7$
8572 025702  012703  014000                        MOV     #14000,R3
8573 025706  022737  000001  003720  7$:          CMP     #1,PROTYP
8574 025714  001406                               BEQ     8$
```

```
 8575 025716  012737  036054  002256          MOV     @MTP032,SUPDOADD
 8576 025724  004737  026660                  CALL    SUPD04
 8577 025730  000402                          BR      9$
 8578 025732  004737  026502          8$:     CALL    SUPD02
 8579 025736  005037  002412          9$:     CLR     NOSCOPE
 8580 025742  000207                  10$:    RETURN                  ;THIS RETURN ACTS AS A NORMAL RETURN FROM MT0032
 8581                                                                 ;ALSO A RETURN FROM THE "CALL   SUPD04" ABOVE
 8582
```

```
 8585 025744                          MT0033: SUBTST  <<MT0033        SETUP BRANCH GOBBLE TEST>>
                                      ;*****************************************************************************
                                      ;*SUBTEST       MT0033  SETUP BRANCH GOBBLE TEST
                                      ;*****************************************************************************
 8586 025744  004737  026432                  CALL    KAMITEST                ;CHECK FOR KAMIKAZE MODE
 8587 025750                                  ON.ERROR THEN $RETURN           ;IF NOT IN KAMIKAZE MODE RETURN
 8588 025754                                  SET     NOSCOPE
 8589 025762  012737  000033  002262          MOV     #33,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
 8590 025770  005037  002074                  CLR     NOPAR                   ;SETUP PARITY ACTION
 8591 025774                                  MAP     BANK                    ;MAP FIRST SO THAT BLOCK MOVE WORKS
 8592
 8593 026010                                  TESTAREA                ;ENTER TEST MODE
 8594 026016                                  BMOV    MTP033,FIRST,GBLENGTH/2
 8595 026030  104417                          KERNEL                  ;ENTER KERNEL MODE
 8596
 8597 026032  012705  060076                  MOV     #FIRST+GBLENGTH,R5
 8598 026036  012737  060004  002256          MOV     #FIRST+4,SUPDOADD
 8599 026044  012701  060002                  MOV     #FIRST+2,R1
 8600 026050  012702  060003                  MOV     #FIRST+3,R2
 8601 026054  012737  160000  002474          MOV     #LAST+2,LINK1
 8602 026062  005737  002430                  TST     NOSUPER
 8603 026066  001005                          BNE     1$
 8604 026070  023737  172252  172254          CMP     SIPAR5,SIPAR6
 8605 026076  001405                          BEQ     2$
 8606 026100  000407                          BR      3$
 8607 026102  023737  177652  177654  1$:     CMP     UIPAR5,UIPAR6
 8608 026110  C01003                          BNE     3$
 8609 026112  012737  140000  002474  2$:     MOV     #140000,LINK1
 8610
 8611 026120  004737  026660          3$:     CALL    SUPD04
 8612 026124  005037  002412                  CLR     NOSCOPE
 8613 026130  000207                          RETURN
 8614
 8615 026132                          MT0034: SUBTST  <<MT0034        SOFT ERROR - BACKGROUND PATTERN TEST>>
                                      ;*****************************************************************************
                                      ;*SUBTEST       MT0034  SOFT ERROR - BACKGROUND PATTERN TEST
                                      ;*****************************************************************************
 8616 026132  012737  000034  002262          MOV     #34,REALPAT
 8617 026140  012700  060000                  MOV     #FIRST,R0
 8618 026144  012701  040000                  MOV     #SIZE,R1
 8619 026150  013702  002562                  MOV     SOFTPAT,R2
 8620 026154  010103                          MOV     R1,R3
 8621 026156  013705  002102                  MOV     BANKINDEX,R5
 8622 026162  010004                          MOV     R0,R4
 8623 026164  022737  000001  003720          CMP     #1,PROTYP               ;IS THIS AN 11/44?
 8624 026172  001006                          BNE     1$                      ;BRANCH IF NOT
 8625 026174                                  BMOV    MTP034
 8626 026202  012737  177646  002256          MOV     #UIPAR3,SUPDOADD
 8627 026210                          1$:     IF #BIT13 SET.IN CONFIG+2(R5)
 8628                                            ;BACKGROUND PATTERN IS VALID
 8629 026220  022737  000001  003720              CMP   #1,PROTYP
 8630 026226  001403                              BEQ   2$
 8631 026230  012737  036212  002256              MOV   #MTP034+6,SUPDOADD
 8632 026236  004737  026644          2$:         CALL  SUPD03                  ;READ IT
 8633 026242                                  ELSE
 8634                                            ;BACKGROUND PATTERN HAS BEEN INVALIDATED
 8635 026244  022737  000001  003720              CMP   #1,PROTYP
```

```
 8636 026252  001406                          BEQ   3$
 8637 026254  012737  036204  002256          MOV   @MTP034,SUPDOADR
 8638 026262  004737  026644                  CALL  SUPD03
 8639 026266  000402                          BR    4$
 8640 026270  004737  026466          3$:     CALL  SUPD01                    ;WRITE IT
 8641 026274  052765  020000  002630  4$:     BIS   @BIT13,CONFIG+2(R5)       ;VALIDATE IT
 8642 026302                                 END ;OF IF @BIT13
 8643 026302  000207                         RETURN
 8644
 8645 026304                                 MT0035: SUBTST  <<MT0035        SETUP WORST CASE NOISE PARITY TEST>>
                                             ;*****************************************************************************
                                             ;*SUBTEST       MT0035  SETUP WORST CASE NOISE PARITY TEST
                                             ;*****************************************************************************
 8646 026304  012737  000035  002262          MOV     #35,REALPAT             ;SET UP TEST NUMBER FOR DISPLAY
 8647 026312  013703  002102                  MOV     BANKINDEX,R3
 8648 026316  016301  002626                  MOV     CONFIG(R3),R1
 8649 026322  000301                          SWAB    R1
 8650 026324  042701  177760                  BIC     #^C17,R1
 8651 026330  006301                          ASL     R1
 8652 026332  010137  002146                  MOV     R1,CSRNO
 8653 026336  023737  002146  002504          CMP     CSRNO,PGMCSR
 8654 026344  001001                          BNE     1$
 8655 026346  000207                          RETURN
 8656 026350  012702  052524          1$:     MOV     #52524,R2
 8657 026354  004737  036532                  CALL    BACKGND                 ;WRITE BACKROUND OF ALMOST ALT. 1'S AND 0'S
 8658 026360  012737  036230  002256          MOV     @MTP035,SUPDOADD
 8659 026366  004737  026644                  CALL    SUPD03
 8660 026372                                  IF QVFLAG IS TRUE THEN $RETURN
 8661 026402  005102                          COM     R2
 8662 026404  004737  036532                  CALL    BACKGND                 ;WRITE COMPLEMENT PATTERN INTO MUT
 8663 026410  004737  026660                  CALL    SUPD04
 8664 026414  000207                          RETURN
```

```
8667 026416                          MT0999: SUBTST  <<MT0999        SETUP NULL TEST>>
                                     ;*****************************************************************************
                                     ;*SUBTEST        MT0999  SETUP NULL TEST
                                     ;*****************************************************************************
8668 026416  005037  002262                 CLR     REALPAT
8669 026422                                 SET     NULLFLAG
8670 026430  000207                         RETURN
8671
8672 026432                          KAMITEST:SUBTST <<CHECK FOR KAMIKAZE MODE>>
                                     ;*****************************************************************************
                                     .*SUBTEST        CHECK FOR KAMIKAZE MODE
                                     ;*****************************************************************************
8673 026432                                 IF KAMIKAZE IS TRUE OR ACTFLAG IS TRUE OR APTFLAG IS TRUE
8674 026454                                   $RETURN NOERROR                ;RUN THE TEST
8675 026460                                 ELSE
8676 026462                                   $RETURN ERROR                  ;DON T RUN THE TEST
8677 026464                                 END ;OF IF KAMIKAZE
```

```
8680 026466                                    SUPD01: SUBTST  ;;SUBR  EXECUTE PATTERN IN SUPERVISOR..
                                                       ;*****************************************************************
                                                       ;*SUBTEST        SUBR    EXECUTE PATTERN IN SUPERVISOR
                                                       ;*****************************************************************
8681 026466                                                    MAP     BANK                            ;MAP SUPERVISOR SPACE (TEST AREA) TO BANK
8682 026502  004737  055726                    SUPD02: CALL    GETDIS
8683 026506                                            PUSH    $LPERR,$LPADR
8684 026516  010037  002154                            MOV     R0,SUPDR0
8685 026522  012700  002156                            MOV     #SUPDR1,R0
8686 026526  010120                                    MOV     R1,(R0)+
8687 026530  010220                                    MOV     R2,(R0)+
8688 026532  010320                                    MOV     R3,(R0)+
8689 026534  010420                                    MOV     R4,(R0)+
8690 026536  010520                                    MOV     R5,(R0)+
8691 026540  010620                                    MOV     SP,(R0)+
8692 026542  013700  002154                            MOV     SUPDR0,R0
8693 026546  012737  026562  002564                    MOV     #TAG4$,$LPADR
8694 026554  013737  002564  002566                    MOV     $LPADR,$LPERR
8695 026562  012700  002172                TAG4$:      MOV     #SUPDR6+2,R0
8696 026566  014006                                    MOV     -(R0),SP
8697 026570  014005                                    MOV     -(R0),R5
8698 026572  014004                                    MOV     -(R0),R4
8699 026574  014003                                    MOV     -(R0),R3
8700 026576  014002                                    MOV     -(R0),R2
8701 026600  014001                                    MOV     -(R0),R1
8702 026602  014000                                    MOV     -(R0),R0
8703 026604                                            SUPERVISOR                      ;ENTER SUPERVISOR MODE
8704 026612  012706  000740                            MOV     #SUPSTK,SSP
8705 026616  104424                                    CACHOFF                         ;TURN CACHE OFF
8706 026620  004737  177640                            CALL    FASTCITY                        ;CALL TO THE USER INSTRUCTION PAR S
8707 026624  104423                                    CACHON                          ;TURN CACHE ON
8708 026626  104417                                    KERNEL                          ;ENTER KERNEL MODE
8709 026630  000004                                    SCOPE
8710 026632                                            POP     $LPADR,$LPERR
8711 026642  000207                                    RETURN
```

```
8714 026644                                SUPD03: MAP     BANK                    ;MAP SUPERVISOR SPACE, TEST AREA, BANK
8715 026660  004737  055726                SUPD04: CALL    GETDIS
8716 026664                                        PUSH    $LPERR,$LPADR
8717 026674  010037  002154                        MOV     R0,SUPDR0
8718 026700  012700  002156                        MOV     #SUPDR1,R0
8719 026704  010120                        MOV:    R1,(R0)+
8720 026706  010220                        MOV     R2,(R0)+
8721 026710  010320                        MOV     R3,(R0)+
8722 026712  010420                        MOV     R4,(R0)+
8723 026714  010520                        MOV     R5,(R0)+
8724 026716  010620                        MOV     SP,(R0)+
8725 026720  013700  002154                MOV     SUPDR0,R0
8726 026724  012737  026740  002564        MOV     #TBG4$,$LPADR
8727 026732  013737  002564  002566        MOV     $LPADR,$LPERR
8728 026740  012700  002172        TBG4$:  MOV     #SUPDR6+2,R0
8729 026744  014006                        MOV     (R0),SP
8730 026746  014005                        MOV     (R0),R5
8731 026750  014004                        MOV     -(R0),R4
8732 026752  014003                        MOV     -(R0),R3
8733 026754  014002                        MOV     -(R0),R2
8734 026756  014001                        MOV     -(R0),R1
8735 026760  014000                        MOV     -(R0),R0
8736 026762                                TESTAREA                                ;ENTER SUPERVISOR MODE
8737 026770  005737  002430                TST     NOSUPER
8738 026774  001403                        BEQ     1$
8739 026776  012706  000700                MOV     #USESTK,USP
8740 027002  000402                        BR      2$
8741 027004  012706  000740        1$:     MOV     #SUPSTK,SSP
8742 027010  104424                2$:     CACHOFF                                 ;TURN CACHE OFF
8743 027012  004777  153240                CALL    @SUPDOADD
8744 027016  104423                        CACHON                                  ;TURN CACHE ON
8745 027020  104417                        KERNEL                                  ;ENTER KERNEL MODE
8746 027022  000004                        SCOPE
8747 027024                                POP     $LPADR,$LPERR
8748 027034  000207                        RETURN
```

```
8751                                            .SBTTL  MEMORY TEST PATTERN ROUTINES
8752                                    ;*******************************************************************************
8753                                    ; PATTERN REGISTER CONVENTIONS
8754                                    ;       R0      FIRST ADDRESS OF PATTERN (FIRST,LAST+2,ETC)
8755                                    ;       R1      NUMBER OF ADDRESSES IN PATTERN (SIZE)
8756                                    ;       R2      DATA FOR PATTERN (ONES,52525,ETC)
8757                                    ;       R3      COPY OF R1 (IF NECESSARY)
8758                                    ;       R4      COPY OF R0 (IF NECESSARY)
8759                                    ;       R5      COPY OF R2 (IF NECESSARY)
8760                                    ;*******************************************************************************
8761 027036                            MTP000: SUBTST  <<MTP000        BASIC DATA TEST>>
                                        ;*******************************************************************************
                                        ;+SUBTEST       MTP000  BASIC DATA TEST
                                        ;*******************************************************************************
8762 027036  010220                     1$:     MOV     R2,(R0)+        ;V177640
8763 027040  077102                             SOB     R1,MTP000       ;V177642
8764 027042  000240                             NOP                     ;V177644
8765 027044  012401                     2$:     MOV     (R4)+,R1        ;V177646
8766 027046  020102                             CMP     R1,R2           ;V177650
8767 027050  001402                             BEQ     3$              ;V177652
8768 027052  104430                             PERR02                  ;V177654
8769 027054  000240                             NOP                     ;V177656
8770 027056  077306                     3$:     SOB     R3,2$           ;V177660
8771 027060  000207                             RETURN                  ;V177662
8772 027062                            MTP001: SUBTST  <<MTP001        ADDRESS TEST>>
                                        ;*******************************************************************************
                                        ;+SUBTEST       MTP001  ADDRESS TEST
                                        ;*******************************************************************************
8773 027062  010220                     3$:     MOV     R2,(R0)+        ;V177640
8774 027064  062702  000002                     ADD     #2,R2           ;V177642
8775 027070  077104                             SOB     R1,3$           ;V177646
8776 027072  000240                             NOP                     ;V177650
8777 027074  012400                     1$:     MOV     (R4)+,R0        ;V177652
8778 027076  020005                             CMP     R0,R5           ;V177654
8779 027100  001401                             BEQ     2$              ;V177656
8780 027102  104427                             PERR01                  ;V177660
8781 027104  062705  000002             2$:     ADD     #2,R5           ;V177662
8782 027110  077307                             SOB     R3,1$           ;V177666
8783 027112  000207                             RETURN                  ;V177672
8784 027114                            MTP002: SUBTST  <<MTP002        COMPLEMENT ADDRESS TEST (WRITE DOWN, READ UP)>>
                                        ;*******************************************************************************
                                        ;+SUBTEST       MTP002  COMPLEMENT ADDRESS TEST (WRITE DOWN, READ UP)
                                        ;*******************************************************************************
8785 027114  010540                     3$:     MOV     R5,-(R0)        ;V177640
8786 027116  062705  000002                     ADD     #2,R5           ;V177642
8787 027122  077104                             SOB     R1,3$           ;V177646
8788 027124  000240                             NOP                     ;V177650
8789 027126  162702  000002             1$:     SUB     #2,R2           ;V177652
8790 027132  012401                             MOV     (R4)+,R1        ;V177656
8791 027134  020102                             CMP     R1,R2           ;V177660
8792 027136  001401                             BEQ     2$              ;V177662
8793 027140  104430                             PERR02                  ;V177664
8794 027142  077307                     2$:     SOB     R3,1$           ;V177666
8795 027144  000207                             RETURN                  ;V177670
```

```
8798 027146                            MTPA03: SUBTST  <<MTPA03        3 XOR 9 WORST CASE NOISE TEST (WRITE)>>
                                       ;*****************************************************************************
                                       ;*SUBTEST       MTPA03  3 XOR 9 WORST CASE NOISE TEST (WRITE)
                                       ;*****************************************************************************
8799                                           ;R1 - ADDRESS
8800                                           ;R2 - SMALL LOOP CONSTANT
8801                                           ;R3 - NUM OF ADD TO TEST (LARGE LOOP)
8802                                           ;R4 - GOOD DATA
8803                                           ;R5 - MEDIUM LOOP CONSTANT
8804                                           .ENABL  LSB
8805 027146  010421                    1$:     MOV     R4,(R1)+        ;V177640
8806 027150  010421                            MOV     R4,(R1)+        ;V177642
8807 027152  077203                            SOB     R2,1$           ;V177644
8808 027154  005104                            COM     R4              ;V177646
8809 027156  052704                            BIS     (PC)+,R4        ;V177650
8810 027160  000401                    WARN2:  401                     ;V177652 WARNING LOCATION IS MODIFIED BEFORE LOADING
8811 027162  012702  000004                    MOV     #4,R2           ;V177654
8812 027166  077511                            SOB     R5,1$           ;V177660
8813 027170  005104                            COM     R4              ;V177662
8814 027172  052704                            BIS     (PC)+,R4        ;V177664
8815 027174  000401                    WARN3:  401                     ;V177666 WARNING LOCATION IS MODIFIED BEFORE LOADING
8816 027176  012705  000100                    MOV     #64.,R5         ;V177670
8817 027202  077317                            SOB     R3,1$           ;V177674
8818 027204  000207                            RETURN                  ;V177676
8819                                           .DSABL  LSB
8820
8821 027206                            MTPB03: SUBTST  <<MTPB03        3 XOR 9 WORST CASE NOISE TEST (READ)>>
                                       ;*****************************************************************************
                                       ;*SUBTEST       MTPB03  3 XOR 9 WORST CASE NOISE TEST (READ)
                                       ;*****************************************************************************
8822                                           .ENABL  LSB
8823 027206  000137  027246            1$:     JMP     @#MTPC03        ;V177640        GO TO V172360
8824 027212  077203                            SOB     R2,1$           ;V177644
8825 027214  005104                            COM     R4              ;V177646
8826 027216  052704                            BIS     (PC)+,R4        ;V177650
8827 027220  000401                    WARN4:  401                     ;V177652 WARNING LOCATION IS MODIFIED BEFORE LOADING
8828 027222  012702  000004                    MOV     #4,R2           ;V177654
8829 027226  077511                            SOB     R5,1$           ;V177660
8830 027230  005104                            COM     R4              ;V177662
8831 027232  052704                            BIS     (PC)+,R4        ;V177664
8832 027234  000401                    WARN5:  401                     ;V177666 WARNING LOCATION IS MODIFIED BEFORE LOADING
8833 027236  012705  000100                    MOV     #64.,R5         ;V177670
8834 027242  077317                            SOB     R3,1$           ;V177674
8835 027244  000207                            RETURN                  ;V177676
8836                                           .DSABL  LSB
```

```
 8839 027246                            MTPC03: SUBTST  <<MTPC03         TEST DATA SUBPROGRAM>>
                                        ;*****************************************************************************
                                        ;*SUBTEST        MTPC03  TEST DATA SUBPROGRAM
                                        ;*****************************************************************************
 8840 027246  020421                            CMP     R4,(R1)+        ;V172360
 8841 027250  001401                            BEQ     1$              ;V172362
 8842 027252  104431                            PERR03                  ;V172364
 8843 027254  005141                    1$:     COM     -(R1)           ;V172366
 8844 027256  005111                            COM     (R1)            ;V172370
 8845 027260  000137  027264                    JMP     @#MTPD03        ;V172372        GO TO V172260
 8846
 8847 027264                            MTPD03: SUBTST  <<MTPD03         TEST DATA SUBSUBPROGRAM>>
                                        ;*****************************************************************************
                                        ;*SUBTEST        MTPD03  TEST DATA SUBSUBPROGRAM
                                        ;*****************************************************************************
 8848 027264  020421                            CMP     R4,(R1)+        ;V172260
 8849 027266  001401                            BEQ     1$              ;V172262
 8850 027270  104431                            PERR03                  ;V172264
 8851 027272  005127                    1$:     COM     (PC)+           ;V172266
 8852 027274  000000                            0                       ;V172270
 8853 027276  001363                            BNE     MTPC03          ;V172272        GO TO V172360
 8854 027300  000137  027212                    JMP     @#MTPB03+4      ;V172274        GO TO V177644
```

```
8857 027304                                MTPA04: SUBTST  <<MTPA04         ROTATING ZEROS TEST>>
                                           ;*****************************************************************************
                                           ;*SUBTEST        MTPA04  ROTATING ZEROS TEST
                                           ;*****************************************************************************
8858 027304  012705  000010                1$:     MOV     #8.,R5           ;V177640
8859 027310  010504                                MOV     R5,R4            ;V177644
8860 027312  000241                                CLC                      ;V177646
8861 027314  000137  027340                        JMP     @#MTPB04         ;V177650
8862 027320  016004  177776                        MOV     -2(R0),R4        ;V177654
8863 027324  103402                                BCS     2$               ;V177660
8864 027326  020204                                CMP     R2,R4            ;V177662
8865 027330  001401                                BEQ     3$               ;V177664
8866 027332  104432                2$:     PERR04                           ;V177666
8867 027334  077115                3$:     SOB     R1,1$                    ;V177670
8868 027336  000207                        RETURN                           ;V177672
8869
8870 027340                                MTPB04: SUBTST  <<MTPB04         SUBR    ROTATING BIT>>
                                           ;*****************************************************************************
                                           ;*SUBTEST        MTPB04  SUBR    ROTATING BIT
                                           ;*****************************************************************************
8871 027340  106110                1$:     ROLB    (R0)                     ;V172360
8872 027342  077502                        SOB     R5,1$                    ;V172362
8873 027344  106120                        ROLB    (R0)+                    ;V172364
8874 027346  106110                2$:     ROLB    (R0)                     ;V172366
8875 027350  077402                        SOB     R4,2$                    ;V172370
8876 027352  106120                        ROLB    (R0)+                    ;V172372
8877 027354  000137  027320                JMP     @#MTPA04+14              ;V172374
8878
8879 027360                                MTP005: SUBTST  <<MTP005         ROTATION ONES TEST>>
                                           ;*****************************************************************************
                                           ;*SUBTEST        MTP005  ROTATION ONES TEST
                                           ;*****************************************************************************
8880 027360  012705  000010                1$:     MOV     #8.,R5           ;V177640
8881 027364  010504                                MOV     R5,R4            ;V177644
8882 027366  000261                                SEC                      ;V177646
8883 027370  000137  027340                        JMP     @#MTPB04         ;V177650
8884 027374  016004  177776                        MOV     -2(R0),R4        ;V177654
8885 027400  103002                                BCC     2$               ;V177660 IF THIS HAPPENS THE GOOD & BAD MATCH
8886 027402  020204                                CMP     R2,R4            ;V177662
8887 027404  001401                                BEQ     3$               ;V177664
8888 027406  104432                2$:     PERR04                           ;V177666
8889 027410  077115                3$:     SOB     R1,1$                    ;V177670
8890 027412  000207                        RETURN                           ;V177672
```

```
8893 027414                                MTP006: SUBTST  <<MTP006        INITIAL DATA TEST>>
                                           ;*****************************************************************
                                           ;*SUBTEST       MTP006  INITIAL DATA TEST
                                           ;*****************************************************************
8894                                                ;       THIS TEST CHECKS THE DI/DO LINES BY
8895                                                ;       SHIFTING A 1 THROUGH THE WORD.
8896 027414  012737  000001  002236          MOV     #1,DATBUF       ;SET THE FIRST TEST BIT
8897 027422  005037  002240                  CLR     DATBUF+2        ;CLEAR 2ND WORD
8898 027426  013771  002236  000000  1$:     MOV     DATBUF,@(R1)    ;WRITE TEST WORD 1
8899 027434  013771  002240  000002          MOV     DATBUF+2,@2(R1) ;AND TEST WORD 2
8900 027442  017102  000000                  MOV     @(R1),R2
8901 027446  023702  002236                  CMP     DATBUF,R2       ;NOW READ THEM
8902 027452  001401                          BEQ     2$              ;BR IF FIRST 16 OK
8903 027454  104433                          PERR07                  ;ERROR TRAP
8904
8905 027456  017102  000002          2$:     MOV     @2(R1),R2
8906 027462  023702  002240                  CMP     DATBUF+2,R2     ;NOW READ SECOND WORD
8907 027466  001401                          BEQ     3$              ;BR IF OK
8908 027470  104434                          PERR10                  ;ERROR TRAP
8909
8910 027472  005737  002240          3$:     TST     DATBUF+2        ;HAS LAST BIT BEEN TESTED ?
8911 027476  100405                          BMI     4$              ;MINUS MEANS BIT 31
8912 027500                                  DLEFT   DATBUF          ;NO, SHIFT TEST BIT LEFT
8913 027510  000746                          BR      1$              ;GO WRITE NEW TEST DATA
8914                                                                 ;NOW GOING TO SHIFT A 0 IN DATA DIRECTION
8915 027512  012737  177776  002236  4$:     MOV     #177776,DATBUF  ;PUT A 0 IN BIT 0
8916 027520  012737  177777  002240          MOV     #-1, DATBUF+2   ;AND 1'S IN ALL OTHERS
8917 027526  013771  002236  000000  5$:     MOV     DATBUF,@(R1)    ;WRITE THE DATA
8918 027534  013771  002240  000002          MOV     DATBUF+2,@2(R1) ;2 WORDS WORTH
8919 027542  017102  000000                  MOV     @(R1),R2
8920 027546  023702  002236                  CMP     DATBUF,R2       ;NOW READ FIRST WORD
8921 027552  001401                          BEQ     6$              ;BR IF OK
8922 027554  104433                          PERR07
8923
8924 027556  017102  000002          6$:     MOV     @2(R1),R2
8925 027562  023702  002240                  CMP     DATBUF+2,R2     ;NOW, READ SECOND WORD
8926 027566  001401                          BEQ     7$              ;BR IF OK
8927 027570  104434                          PERR10
8928
8929 027572  005737  002240          7$:     TST     DATBUF+2        ;TESTED BIT 31 YET?
8930 027576  100005                          BPL     8$              ;BR IF YES, WE'RE DONE
8931 027600                                  DLEFT   DATBUF
8932 027610  000746                          BR      5$              ;KEEP GOING
8933 027612  000207                  8$:     RETURN
```

```
 8936 027614                    MTP007: SUBTST  <<MTP007        ADDRESS BIT TEST>>
                                ;*****************************************************************
                                ;*SUBTEST        MTP007  ADDRESS BIT TEST
                                ;*****************************************************************
 8937                                   ;       THIS TEST CHECKS TO SEE THAT EACH ADDRESS
 8938                                   ;       BIT IN EACH 16K BANK CAN BE ASSERTED UNIQUELY.
 8939                                   ;       IT CHECKS FOR ADDRESS BITS THAT MAY BE STUCK
 8940                                   ;       HIGH, STUCK LOW OR STUCK TOGETHER.
 8941 027614  111100                       MOVB    (R1),R0
 8942 027616  105700                       TSTB    R0              ;READ AND COMPARE FOR ZEROS
 8943 027620  001401                       BEQ     1$              ;BR IF OK
 8944 027622  104435                       PERR11
 8945
 8946 027624  105111                1$:    COMB    (R1)            ;COMPLEMENT THE BYTE
 8947 027626  111100                       MOVB    (R1),R0
 8948 027630  105700                       TSTB    R0              ;READ FOR NON ZEROS
 8949 027632  001001                       BNE     2$              ;BR IF OK
 8950 027634  104436                       PERR12
 8951
 8952 027636  040201                2$:    BIC     R2,R1           ;MASK OFF THE ASSERTED BIT
 8953 027640  006302                       ASL     R2              ;SHIFT R2 FOR NEXT BIT
 8954 027642  050201                       BIS     R2,R1           ;SET THE NEW BIT INTO R1
 8955 027644  011100                       MOV     (R1),R0
 8956 027646  005700                       TST     R0              ;READ THE NEW ADDRESS
 8957 027650  001401                       BEQ     3$              ;READ FOR ZEROS
 8958 027652  104437                       PERR13
 8959
 8960 027654  005111                3$:    COM     (R1)            ;COMPL THE WORD
 8961 027656  011100                       MOV     (R1),R0
 8962 027660  005700                       TST     R0              ;READ IT AGAIN
 8963 027662  001001                       BNE     4$
 8964 027664  104440                       PERR14
 8965
 8966 027666  022702  100000        4$:    CMP     #100000,R2
 8967 027672  001407                       BEQ     5$
 8968 027674  022702  010000              CMP     #10000,R2       ;CHECK FOR MSB IN 4K BANK
 8969 027700  001356                       BNE     2$              ;NOT LAST BIT, BRANCH
 8970 027702  006302                       ASL     R2
 8971 027704  012701  160000              MOV     #160000,R1
 8972 027710  000752                       BR      2$
 8973 027712  000207                5$:    RETURN
```

```
8976 027714                                MTP010: SUBTST  <<MTP010        BYTE ADDRESSING TEST>>
                                           ;*****************************************************************************
                                           ;*SUBTEST       MTP010  BYTE ADDRESSING TEST
                                           ;*****************************************************************************
8977                                                   ;TEST 3 THIS TEST CHECKS FOR PROPER
8978                                                   ;       BYTE ADDRESSING WITH ECC DISABLED
8979 027714  010402                              MOV     R4,R2           ;R4 HAS LOWEST ADDRESS
8980 027716  010403                              MOV     R4,R3           ;PUT IT IN R3 ALSO
8981 027720  062702  000004                      ADD     #4,R2           ;POINT R2 TO LAST BYTE +1
8982 027724  012713  177777                      MOV     #-1,(R3)        ;WRITE ALL ONES IN
8983 027730  012763  177777  000002              MOV     #-1,2(R3)       ;THE 4 TEST BYTES
8984 027736  105013                    1$:       CLRB    (R3)            ;CLEAR A BYTE
8985 027740  010401                              MOV     R4,R1           ;INITIALIZE R1 FOR EACH PASS
8986 027742  020201                    2$:       CMP     R2,R1           ;IF EQUAL, JUST READ LAST BYTE
8987 027744  001420                              BEQ     6$              ;BR IF EQUAL
8988 027746  020301                              CMP     R3,R1           ;IS THIS THE BYTE OF ZEROS
8989 027750  001007                              BNE     4$              ;BR IF NOT
8990 027752  111100                              MOVB    (R1),R0
8991                                             ;WARNING IF YOU OPTOMIZE CHANGE THE PCBUMP FOR THIS ERROR INCASE OF TRAP
8992 027754  022700  000000                      CMP     #0,R0           ;IT IS, COMPARE FOR ZEROS
8993 027760  001401                              BEQ     3$
8994 027762  104435                              PERR11
8995
8996 027764  005201                    3$:       INC     R1              ;NEXT BYTE
8997 027766  000765                              BR      2$              ;RETURN
8998 027770  111100                    4$:       MOVB    (R1),R0
8999 027772  122700  177777                      CMPB    #-1,R0          ;ITS NOT THE BYTE OF 0'S, READ 1 S
9000 027776  001401                              BEQ     5$
9001 030000  104436                              PERR12
9002
9003 030002  005201                    5$:       INC     R1              ;MOVE TO NEXT BYTE
9004 030004  000756                              BR      2$
9005 030006  112713  177777            6$:       MOVB    #-1,(R3)        ;RESTORE 1'S TO BYTE JUST TESTED
9006 030012  005203                              INC     R3              ;INC TO NEXT BYTE
9007 030014  020302                              CMP     R3,R2           ;WAS THAT JUST THE LAST ONE?
9008 030016  001347                              BNE     1$              ;BR IF NO
9009 030020  000207                              RETURN
```

```
                                        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)
030022  104503                                 CLR1CSR                 ;CLEAR 1 SELECTED CSR
030024  005737  013140                          TST     PHEBE           ;TEST SPECIAL CASE INDICATOR
030030  001407                                  BEQ     MTLA11          ;BRANCH IF NOT SET
030032  013702  172246                          MOV     SIPAR3,R2       ;SAVE CONTENTS OF SIPAR #3
030036  013737  172252  172246                  MOV     SIPAR5,@#SIPAR3 ;COPY CONTENTS OF #5 INTO #3
030044  010237  172252                          MOV     R2,@#SIPAR5     ;COPY CONTENTS OF #3 INTO #5
                                                ;BIG LOOP
030050  012737  000001  002236  MTLA11: MOV     #1,DATBUF       ;INITIAL DATA
030056  005037  002240                          CLR     DATBUF+2        ;32 BITS WORTH
                                                ;MEDIUM LOOP
030062  012737  000001  002246  MTLB11: MOV     #1,SBEMSK       ;INITIAL ERROR MASK
030070  005037  002250                          CLR     SBEMSK+2        ;32 BITS WORTH
                                                ;LITTLE LOOP
030074  013737  002236  002242  MTLC11: MOV     DATBUF,TSTDAT   ;
030102  013737  002240  002244                  MOV     DATBUF+2,TSTDAT+2;TO SAVE ORIG DATA
030110  105737  002260                          TSTB    PASFLG  ;COMP DATA ON SECOND PASSONLY
030114  001404                                  BEQ     4$              ;BR IF FIRST PASS
030116  005137  002242                          COM     TSTDAT          ;SECOND PASS, COMP BOTH WORDS
030122  005137  002244                          COM     TSTDAT+2
030126  013702  002242          4$:     MOV     TSTDAT,R2
030132  013703  002244                  MOV     TSTDAT+2,R3
030136  012737  002242  002274                  MOV     #TSTDAT,SOURCE  ;SET UP ADDRESS FOR CHKGEN
030144  004737  041724                          CALL    CHKGEN          ;GEN CHECKBITS ON TSTDAT
                                                ;*******************************
                                                ;** CREATE A SINGLE BIT ERROR **
                                                ;*******************************
030150  013701  002246                          MOV     SBEMSK,R1
030154  074137  002242                          XOR     R1,TSTDAT
030160  013701  002250                          MOV     SBEMSK+2,R1
030164  074137  002244                          XOR     R1,TSTDAT+2
030170  013701  002364          MTLD11: MOV     TESTADD,R1      ;FIRST TEST ADDRESS
030174  013705  002366                  MOV     TESTADD+2,R5    ;SECOND TEST ADDRESS
030200  104471                                  ECC1DIS                 ;DISABLE ECC ON 1 SELECTED CSR
030202  013711  002242                          MOV     TSTDAT,(R1)     ;WRITE FIRST 16 BITS
030206  104475                                  CB1CSR                  ;WRITE GENERATED CHECKBITS IN 1 SELECTED CSR
030210  013715  002244                          MOV     TSTDAT+2,(R5)   ;WRITE SECOND 16 BITS AND
                                                                        ;CHECK BITS. WE NOW HAVE CHECKBITS
```

```
9066                                                    ;GENERATED ON DATBUF AND DATA WITH
9067                                                    ;ONE BIT IN ERROR (AS PER SBEMSK).
9068 030214  104471                       ECC1DIS     ;DISABLE ECC ON 1 SELECTED CSR
9069 030216  011100                       MOV     (R1),R0
9070 030220  020037  002242               CMP     R0,TSTDAT       ;READ THE LOW WORD (UNCORRECTED)
9071 030224  001403                       BEQ     6$              ;BR IF OK
9072 030226  010137  002032               MOV     R1,ADDRESS
9073 030232  104455                       PERR31
9074
9075 030234  011500                 6$:   MOV     (R5),R0
9076 030236  020037  002244               CMP     R0,TSTDAT+2     ;READ THE HIGH WORD (UNCORRECTED)
9077 030242  001403                       BEQ     7$              ;BR IF OK
9078 030244  010537  002032               MOV     R5,ADDRESS
9079 030250  104455                       PERR31
9080
9081 030252                          7$:   IF KFLAG IS FALSE
9082 030260  104426                         READCSR
9083 030262                                 IF @BIT4 OFF.IN CSR OR @BIT15 OFF.IN CSR
9084 030302  104045                           ERROR       +45
9085 030304                                 END; OF IF @BIT4
9086 030304                               END; OF IF KFLAG
9087 030304  005737  013140               TST     PHEBE
9088 030310  001001                       BNE     17$
9089 030312  104512                       ERRGEN
9090 030314  104503                 17$:  CLR1CSR                 ;CLEAR 1 SELECTED CSR
9091 030316  011100                       MOV     (R1),R0
9092 030320  020002                       CMP     R0,R2           ;SEE IF ITS BEEN CORRECTED
9093 030322  001401                       BEQ     8$              ;IT SHOULD HAVE BEEN
9094 030324  104456                       PERR32
9095
9096 030326  104510                 8$:   TSTREAD                 ;TEST LOC (R1) & TST FOR SBE (WITHOUT FETCHES)
9097 030330  103411                       BCS     9$              ;BR IF IT IS SET
9098 030332                               SET     HEADER          ;ENABLE PRINTING OF ERROR HEADER INFO
9099 030340  010137  002032               MOV     R1,ADDRESS
9100 030344  104460                       PERR34
9101 030346                               SET     HEADER          ;ENABLE PRINTING OF ERROR HEADER INFO
9102
9103 030354  104503                 9$:   CLR1CSR                 ;CLEAR 1 SELECTED CSR
9104 030356  011500                       MOV     (R5),R0
9105 030360  020003                       CMP     R0,R3           ;SEE IF ITS BEEN CORRECTED
9106 030362  001401                       BEQ     10$             ;BR IF OK
9107 030364  104456                       PERR32
9108
9109 030366  104510                10$:   TSTREAD                 ;TEST LOC (R1) & TST FOR SBE (WITHOUT FETCHES)
9110 030370  103411                       BCS     11$             ;BR IF YES
9111 030372                               SET     HEADER          ;ENABLE PRINTING OF ERROR HEADER INFO
9112 030400  010137  002032               MOV     R1,ADDRESS
9113 030404  104460                       PERR34
9114 030406                               SET     HEADER          ;ENABLE PRINTING OF ERROR HEADER INFO
9115 030414  104512                11$:   ERRGEN                  ;TEST ERROR ADDRESS
9116 030416  105737  002260               TSTB    PASFLG
9117 030422  100452                       BMI     15$
9118 030424  005737  002250               TST     SBEMSK+2        ;TEST FOR LAST MASK BIT
9119 030430  100405                       BMI     12$             ;MINUS MEANS BIT 31
9120 030432                               DLEFT   SBEMSK
9121 030442  000614                       BR      MTLC11
9122 030444                          12$:   IF @SW11 SET.IN @SWR THEN GOTO 13$
```

```
                                        IF QVFLAG IS TRUE THEN GOTO 13$
9123 030454
9124 030462  005737  002240             TST     DATBUF+2        ;LAST DATA BIT ;
9125 030466  100406                     BMI     13$             ;WHICH IS BIT 31
9126 030470                             DLEFT   DATBUF
9127 030500  000137  030062             JMP     MTLB11
9128 030504  105737  002260     13$:    TSTB    PASFLG  ;FIRST OR SECOND PASS ?
9129 030510  001004                     BNE     14$             ;NON ZERO MEANS WE'RE DONE
9130 030512  105237  002260             INCB    PASFLG  ;NOT DONE, GO DO SECOND PASS
9131 030516  000137  030050             JMP     MTLA11
9132 030522  052737  000200  002260 14$:    BIS     #BIT7,PASFLG
9133 030530  005002                     CLR     R2
9134 030532  005003                     CLR     R3
9135 030534  005037  002242             CLR     TSTDAT
9136 030540  005037  002244             CLR     TSTDAT+2
9137 030544  012704  000040             MOV     #40,R4
9138 030550  012737  003740  002276 15$:    MOV     #3740,CHECK
9139 030556  074437  002276             XOR     R4,CHECK
9140 030562  006304                     ASL     R4
9141 030564  032704  020000             BIT     #BIT13,R4
9142 030570  001002                     BNE     16$
9143 030572  000137  030170             JMP     MTLD11
9144                                    ;CLEAR OUT ANY DBE'S OR SBE'S
9145 030576  104471             16$:    ECC1DIS                 ;DISABLE ECC ON 1 SELECTED CSR
9146 030600  013701  002364             MOV     TESTADD,R1
9147 030604  013705  002366             MOV     TESTADD+2,R5
9148 030610                             CLEAR   (R1),(R5)
9149 030614  104503                     CLR1CSR                 ;CLEAR 1 SELECTED CSR
9150 030616  000207                     RETURN
```

```
9153 030620                                MTP012: SUBTST  <<MTP012        WRITE BYTE CLEARS SBE TEST>>
                                           ;*****************************************************************
                                           ;*SUBTEST       MTP012  WRITE BYTE CLEARS SBE TEST
                                           ;*****************************************************************
9154                                                       ;SINGLE BIT ERROR TEST TO INSURE THAT A WRITE
9155                                                       ;BYTE CLEARS SINGLE BIT ERRORS.
9156 030620  104503                              CLR1CSR                 ;CLEAR 1 SELECTED CSR
9157 030622  012737  000001  002236              MOV     #1,DATBUF       ;INITIAL DATA
9158 030630  005037  002240                      CLR     DATBUF+2        ;32 BITS WORTH
9159 030634  012737  000001  002246  1$:         MOV     #1,SBEMSK       ;INITIAL ERROR MASK
9160 030642  005037  002250                      CLR     SBEMSK+2        ;32 BITS WORTH
9161 030646  013737  002236  002242  2$:         MOV     DATBUF,TSTDAT   ;SAVE ORIGINAL DATA
9162 030654  013737  002240  002244              MOV     DATBUF+2,TSTDAT+2;BOTH WORDS
9163 030662  012737  002242  002274              MOV     #TSTDAT,SOURCE  ;NEED ADDRESS FOR CHKGEN
9164 030670  004737  041724                      CALL    CHKGEN          ;GENERATE CHECK BITS
9165 030674  013701  002246                      MOV     SBEMSK,R1
9166 030700  074137  002242                      XOR     R1,TSTDAT
9167 030704  013701  002250                      MOV     SBEMSK+2,R1
9168 030710  074137  002244                      XOR     R1,TSTDAT+2
9169 030714  013704  002364                      MOV     TESTADD,R4      ;FIRST TEST ADDRESS
9170 030720  010401                              MOV     R4,R1           ;PUT IT IN R1 ALSO
9171 030722  104471                              ECC1DIS                 ;DISABLE ECC ON 1 SELECTED CSR
9172 030724  013711  002242                      MOV     TSTDAT,(R1)     ;WRITE 16 BITS
9173 030730  104475                              CB1CSR                  ;WRITE GENERATED CHECKBITS IN 1 SELECTED CSR
9174 030732  060501                              ADD     R5,R1           ;INDEX UP TO SECOND WORD
9175 030734  013711  002244                      MOV     TSTDAT+2,(R1)   ;WRITE HIGH WORD+CHECKBITS
9176 030740  104503                              CLR1CSR                 ;CLEAR 1 SELECTED CSR
9177                                                                    ;IT'S DANGEROUS IF WE DON'T
9178 030742  012702  002246                      MOV     #SBEMSK,R2      ;ADDRESS OF ERROR MASK
9179 030746  160501                              SUB     R5,R1           ;RETURN TO FIRST WORD
9180 030750  112711  177777              3$:     MOVB    #-1,(R1)        ;WRITE A BYTE OF 1'S
9181 030754  005737  002502                      TST     KFLAG           ;IS THIS MF11S-K
9182 030760  001403                              BEQ     4$              ;BRANCH IF NOT   IT'S MS11-M
9183 030762  132712  177777                      BITB    #-1,(R2)        ;DID THIS BYTE HAVE THE BAD BIT IN IT?
9184 030766  001420                              BEQ     6$              ;NO   BRANCH
9185 030770  104510                  4$:         TSTREAD                 ;TEST LOC (R1) & TST FOR SBE (WITHOUT FETCHES)
9186 030772  103011                              BCC     5$              ;NO   SKIP
9187 030774                                      SET     HEADER          ;ENABLE PRINTING OF ERROR HEADER INFO
9188 031002  010137  002032                      MOV     R1,ADDRESS
9189 031006  104017                              ERROR   17
9190 031010                                      SET     HEADER          ;ENABLE PRINTING OF ERROR HEADER INFO
9191
9192 031016  111100                  5$:         MOVB    (R1),R0
9193 031020  122700  177777                      CMPB    #-1,R0          ;CHECK DATA
9194 031024  001414                              BEQ     7$              ;BR IF OK
9195 031026  104457                              PERR33
9196
9197 031030  104510                  6$:         TSTREAD                 ;TEST LOC (R1) & TST FOR SBE (WITHOUT FETCHES)
9198                                                                    ;READ THE BYTE
9199                                                                    ;SBE ERROR BIT ONLY SET ?
9200 031032  103771                              BCS     5$              ;SHOULD BE SET, BR IF OK
9201 031034                                      SET     HEADER          ;ENABLE PRINTING OF ERROR HEADER INFO
9202 031042  010137  002032                      MOV     R1,ADDRESS
9203 031046  104460                              PERR34
9204 031050                                      SET     HEADER          ;ENABLE PRINTING OF ERROR HEADER INFO
9205
9206 031056  132712  177777              7$:     BITB    #-1,(R2)        ;CHECK FOR LAST BYTE
```

```
 9207 031062  001012                        BNE     8$                      ;
 9208 031064  005202                        INC     R2
 9209 031066  005201                        INC     R1                      ;MOVE TO NEXT BYTE
 9210 031070  013704  002364                MOV     TESTADD,R4              ;FIRST TEST ADDRESS
 9211 031074  032701  000002                BIT     #2,R1                   ;TEST FOR LOWER WORD
 9212 031100  001723                        BEQ     3$                      ;BR IF IT'S LOW 16 BITS
 9213 031102  062704  000002                ADD     #2,R4                   ;ADJUST POINTER FOR ERROR REPT.
 9214 031106  000720                        BR      3$
 9215 031110  005737  002250          8$:   TST     SBEMSK+2                ;LAST ERROR BIT ?
 9216 031114  100405                        BMI     9$                      ;MINUS MEANS BIT 31
 9217 031116                                DLEFT   SBEMSK
 9218 031126  000647                        BR      2$
 9219 031130                          9$:   IF @SW11 SET.IN @SWR THEN GOTO 10$
 9220 031140                                IF QVFLAG IS TRUE THEN GOTO 10$
 9221 031146  005737  002240                TST     DATBUF+2                ;LAST DATA BIT?
 9222 031152  100405                        BMI     10$                     ;MINUS = BIT 31
 9223 031154                                DLEFT   DATBUF
 9224 031164  000623                        BR      1$
 9225                                       ;CLEAR OUT ANY DBE'S OR SBE'S
 9226 031166  104471                  10$:  ECC1DIS                         ;DISABLE ECC ON 1 SELECTED CSR
 9227 031170  013701  002364                MOV     TESTADD,R1
 9228 031174  005011                        CLR     (R1)
 9229 031176  060501                        ADD     R5,R1
 9230 031200  005011                        CLR     (R1)
 9231 031202  104503                        CLR1CSR                         ;CLEAR 1 SELECTED CSR
 9232 031204  000207                        RETURN
```

```
      031206                                 MTP013: SUBTST  <<MTP013        CREATE DOUBLE BIT ERROR TEST
                                             ;**************************************************************************
                                             ;*SUBTEST       MTP013  CREATE DOUBLE BIT ERROR TEST
                                             ;**************************************************************************
                                                             ;DOUBLE BIT ERROR FORCE TO CHECK DOUBLE ERROR LOGIC
      031206  104503                                 CLR1CSR                 ;CLEAR 1 SELECTED CSR
      031210  012701  002364                         MOV     #TESTADD,R1
      031214  005037  002236                 1$:     CLR     DATBUF          ;MAKE INITIAL DATA
      031220  005037  002240                         CLR     DATBUF+2        ;ALL ZEROS
      031224  012737  000001  002246         2$:     MOV     #1,SBEMSK       ;INITIAL SINGLE ERROR MASK
      031232  005037  002250                         CLR     SBEMSK+2        ;SECOND WORD
      031236  012737  000001  002252         3$:     MOV     #1,DBEMSK       ;INITIAL DOUBLE ERROR MASK
      031244  005037  002254                         CLR     DBEMSK+2        ;32 BITS HERE ALSO
      031250  013737  002236  002242         4$:     MOV     DATBUF,TSTDAT   ;
      031256  013737  002240  002244                 MOV     DATBUF+2,TSTDAT+2
      031264  105737  002260                         TSTB    PASFLG  ;NO COMPLEMENTING FIRST PASS
      031270  001404                                 BEQ     5$
      031272  005137  002242                         COM     TSTDAT          ;COMP FIRST WORD
      031276  005137  002244                         COM     TSTDAT+2        ;SECOND WORD
      031302  104503                         5$:     CLR1CSR                 ;CLEAR 1 SELECTED CSR
      031304  023737  002246  002252                 CMP     SBEMSK,DBEMSK   ;CAN'T HAVE THE SAME ERROR BIT SET
      031312  001004                                 BNE     6$              ;IN BOTH MASKS
      031314  023737  002250  002254                 CMP     SBEMSK+2,DBEMSK+2;COULD BE EQUAL IN SECOND WORD
      031322  001460                                 BEQ     13$             ;GO MAKE THEM NOT EQUAL
      031324  012737  002242  002274         6$:     MOV     #TSTDAT,SOURCE  ;SOURCE ADDRESS FOR CHKGEN
      031332  004737  041724                         CALL    CHKGEN          ;GO GENERATE CHECK BITS
      031336  013702  002246                         MOV     SBEMSK,R2
      031342  074237  002242                         XOR     R2,TSTDAT
      031346  013702  002250                         MOV     SBEMSK+2,R2
      031352  074237  002244                         XOR     R2,TSTDAT+2
      031356  013702  002252                         MOV     DBEMSK,R2
      031362  074237  002242                         XOR     R2,TSTDAT
      031366  013702  002254                         MOV     DBEMSK+2,R2
      031372  074237  002244                         XOR     R2,TSTDAT+2
      031376  104471                         16$:    ECC1DIS                 ;DISABLE ECC ON 1 SELECTED CSR
      031400  013731  002242                         MOV     TSTDAT,@(R1)+   ;WRITE 16 BITS
      031404  104^75                                 CB1CSR                  ;WRITE GENERATED CHECKBITS IN 1 SELECTED CSR
      031406  013771  002244  000000                 MOV     TSTDAT+2,@(R1)  ;WRITE HIGH WORD
      031414  104503                                 CLR1CSR                 ;CLEAR 1 SELECTED CSR
      031416  1627^1  000002                         SUB     #2,R1           ;ADJUST TEST ADDRESS
      031422  005771  000000                         TST     @(R1)           ;READ THE LOCATION
      031426  104501                                 WAS1DBE                 ;WAS THERE ANY DOUBLE BIT ERRORS ON 1 SELECTED CSR
      031430  103411                                 BCS     9$              ;IT SHOULD BE SET
      031432                                         SET     HEADER
      031440  011137  002032                         MOV     (R1),ADDRESS
      031444  104030                                 ERROR   +30
      031446                                         SET     HEADER
```

```
 9281 031454  104512                    9$.     ERRGEN
 9282 031456  105737  002260                    TSTB    PASFLG
 9283 031462  100452                    BMI     14$
 9284 031464  005737  002254            13$:    TST     DBEMSK+2        ;CHECK MASK FOR LAST BIT
 9285 031470  100405                    BMI     10$             ;MINUS = BIT31
 9286 031472                            DLEFT   DBEMSK
 9287 031502  000662                    BR      4$
 9288 031504                            10$:    IF @SW11 SET.IN @SWR THEN GOTO 11$
 9289 031514                            IF QVFLAG IS TRUE THEN GOTO 11$
 9290 031522  005737  002250                    TST     SBEMSK+2        ;CHECK SINGLE ERROR MASK TOO
 9291 031526  100405                    BMI     11$             ;BR IF DONE
 9292 031530                            DLEFT   SBEMSK
 9293 031540  000636                    BR      3$
 9294 031542  105737  002260            11$:    TSTB    PASFLG  ;FIRST PASS
 9295 031546  001003                    BNE     12$             ;NON ZERO MEANS WE'RE DONE
 9296 031550  105237  002260                    INCB    PASFLG  ;FIRST PASS, NOT DONE
 9297                                   ;CLEAR OUT ANY DBE'S OR SBE'S
 9298 031554  000617                    BR      1$              ;KEEP GOING
 9299 031556  052737  000200  002260    12$:    BIS     @BIT7,PASFLG    ;SET UP FOR CHECK BIT TEST
 9300 031564  005037  002242                    CLR     TSTDAT
 9301 031570  005037  002244                    CLR     TSTDAT+2
 9302 031574  012737  000040  002246            MOV     @40,SBEMSK
 9303 031602  012737  000100  002252            MOV     @100,DBEMSK
 9304 031610  012737  003740  002276    14$:    MOV     @3740,CHECK
 9305 031616  013702  002246                    MOV     SBEMSK,R2
 9306 031622  074237  002276                    XOR     R2,CHECK
 9307 031626  013702  002252                    MOV     DBEMSK,R2
 9308 031632  074237  002276                    XOR     R2,CHECK
 9309 031636  006337  002252                    ASL     DBEMSK
 9310 031642  032737  020000  002252            BIT     @BIT13,DBEMSK
 9311 031650  001652                    BEQ     16$
 9312 031652  006337  002246                    ASL     SBEMSK
 9313 031656  032737  004000  002246            BIT     @BIT11,SBEMSK
 9314 031664  001006                    BNE     15$
 9315 031666  013737  002246  002252            MOV     SBEMSK,DBEMSK
 9316 031674  006337  002252                    ASL     DBEMSK
 9317 031700  000743                    BR      14$
 9318 031702  104471                    15$:    ECC1DIS                 ;DISABLE ECC ON 1 SELECTED CSR
 9319 031704  012701  002364                    MOV     @TESTADD,R1
 9320 031710                            CLEAR   @(R1)+,@(R1)
 9321 031716  104503                    CLR1CSR                 ;CLEAR 1 SELECTED CSR
 9322 031720  000207                    RETURN
```

```
 9325 031722                                MTP014: SUBTST  <<MTP014        WRITE INHIBIT DURING DATIP WITH DBE TEST>>
                                            ;*****************************************************************************
                                            ;*SUBTEST       MTP014  WRITE INHIBIT DURING DATIP WITH DBE TEST
                                            ;*****************************************************************************
 9326                                                       ;THIS TEST CHECKS THE WRITE INHIBIT ON DOUBLE
 9327                                                       ;BIT ERRORS DURING A DATIP OPERATION BY USE
 9328                                                       ;OF AN 'ASRB' INSTRUCTION.
 9329 031722                                        IF KFLAG IS TRUE THEN $RETURN
 9330
 9336                                                       ;NOTE- THIS TEST WILL ONLY BE RUN FOR MF11S K MEMORY.
 9337 031732  005037  002236            1$:         CLR     DATBUF              ;INITIAL DATA
 9338 031736  005037  002240                        CLR     DATBUF+2            ;2 WORDS WORTH
 9339 031742  012737  000001  002246    2$:         MOV     #1,SBEMSK           ;INITIAL ERROR MASK
 9340 031750  005037  002250                        CLR     SBEMSK+2            ;
 9341 031754  012737  000001  002252    3$:         MOV     #1,DBEMSK           ;DOUBLE ERROR MASK
 9342 031762  005037  002254                        CLR     DBEMSK+2            ;2 WORDS
 9343 031766  013737  002236  002242    4$:         MOV     DATBUF,TSTDAT       ;PRESERVE ORIG DATA
 9344 031774  013737  002240  002244                MOV     DATBUF+2,TSTDAT+2
 9345 032002  105737  002260                        TSTB    PASFLG  ;SECOND PASS YET ?
 9346 032006  001404                              BEQ     5$                  ;BR IF NO
 9347 032010  005137  002242                        COM     TSTDAT              ;COMPL DATA  ON SECOND PASS
 9348 032014  005137  002244                        COM     TSTDAT+2
 9349 032020  104503                    5$:         CLR1CSR                     ;CLEAR 1 SELECTED CSR
 9350 032022  023737  002252  002246              CMP     DBEMSK,SBEMSK       ;CHECK FOR SAME MASKS
 9351 032030  001004                              BNE     6$                  ;BR IF OK
 9352 032032  023737  002254  002250              CMP     DBEMSK+2,SBEMSK+2
 9353 032040  001476                              BEQ     11$                 ;BR IF THEY'RE EQUAL
 9354 032042  012737  002242  002274    6$:         MOV     #TSTDAT,SOURCE      ;SET UP ADDRESS FOR CHKGEN
 9355 032050  004737  041724                        CALL    CHKGEN              ;GENERATE CHECK BITS
 9356 032054  013701  002246                        MOV     SBEMSK,R1
 9357 032060  074137  002242                        XOR     R1,TSTDAT
 9358 032064  013701  002250                        MOV     SBEMSK+2,R1
 9359 032070  074137  002244                        XOR     R1,TSTDAT+2
 9360 032074  013701  002252                        MOV     DBEMSK,R1
 9361 032100  074137  002242                        XOR     R1,TSTDAT
 9362 032104  013701  002254                        MOV     DBEMSK+2,R1
 9363 032110  074137  002244                        XOR     R1,TSTDAT+2
 9364 032114  012701  002364            7$:         MOV     #TESTADD,R1         ;TEST ADDRESS
 9365 032120  104471                              ECC1DIS                     ;DISABLE ECC ON 1 SELECTED CSR
 9366 032122  013731  002242                        MOV     TSTDAT,@(R1)+       ;WRITE FIRST 16 BITS
 9367 032126  104475                              CB1CSR                      ;WRITE GENERATED CHECKBITS IN 1 SELECTED CSR
 9368 032130  013771  002244  000000              MOV     TSTDAT+2,@(R1)      ;SECOND 16 BITS+CHECKBITS
 9369 032136  105037  002261                        CLRB    UPPFLG              ;INDICATE LOWER WORD
 9370 032142  013703  002364                        MOV     TESTADD,R3          ;TEST ADDRESS
 9371 032146  104503                    8$:         CLR1CSR                     ;CLEAR 1 SELECTED CSR
 9372 032150  106223                              ASRB    (R3)+               ;SPECIAL DATIP INSTRUCTION
 9373 032152  015100                              MOV     @-(R1),R0
 9374 032154  023700  002242                        CMP     TSTDAT,R0           ;CHECK FOR UNCHANGED DATA
 9375 032160  001404                              BEQ     9$                  ;SHOULD BE UNCHANGED
 9376 032162  017137  000000  002032              MOV     @(R1),ADDRESS
 9377 032170  104455                              PERR31
 9378
 9379 032172  062701  000002            9$:         ADD     #2,R1               ;POINT TO UPPER WORD
 9380 032176  017100  000000                        MOV     @(R1),R0
 9381 032202  023700  002244                        CMP     TSTDAT+2,R0         ;READ IT
 9382 032206  001404                              BEQ     10$                 ;BR IF UNCHANGED
 9383 032210  017137  000000  002032              MOV     @(R1),ADDRESS
```

```
9384 032216  10445*                          PERR31
9385
9386 032220  122737  000003  002261  10$:    CMPB    #3,UPPFLG               ;LOWER WORD
9387 032226  001403                          BEQ     11$             ;BR IF NO
9388 032230  105237  002261                  INCB    UPPFLG
9389 032234  000744                          BR      8$
9390 032236  105737  002260          11$:    TSTB    PASFLG
9391 032242  100453                          BMI     15$             ;BRANCH IF WE'RE TESTING CHECK BIT
9392 032244  005737  002254                  TST     DBEMSK+2        ;LAST BIT IN MASK ?
9393 032250  100405                          BMI     12$             ;BR IF BIT 31
9394 032252                                  DLEFT   DBEMSK
9395 032262  000641                          BR      4$
9396 032264                          12$:    IF @SW11 SET.IN @SWR THEN GOTO 13$
9397 032274                                  IF QVFLAG IS TRUE THEN GOTO 13$
9398 032302  005737  002250                  TST     SBEMSK+2        ;LAST BIT IN SINGLE ERROR MASK ?
9399 032306  100405                          BMI     13$             ;BR IF YES
9400 032310                                  DLEFT   SBEMSK
9401 032320  000615                          BR      3$
9402 032322  105737  002260          13$:    TSTB    PASFLG  ;WHICH PASS
9403 032326  001004                          BNE     14$             ;BR IF WE'RE DONE
9404 032330  105237  002260                  INCB    PASFLG  ;INDICATE SECOND PASS COMING
9405                                         ;CLEAR OUT ANY DBE'S OR SBE'S
9406 032334  000137  031732                  JMP     1$              ;GO DO IT!
9407 032340  052737  000200  002260  14$:    BIS     #BIT7,PASFLG
9408 032346  005037  002242                  CLR     TSTDAT
9409 032352  005037  002244                  CLR     TSTDAT+2
9410 032356  012737  000040  002246          MOV     #40,SBEMSK
9411 032364  012737  000100  002252          MOV     #100,DBEMSK
9412 032372  012737  003740  002276  15$:    MOV     #3740,CHECK
9413 032400  013702  002246                  MOV     SBEMSK,R2
9414 032404  074237  002276                  XOR     R2,CHECK
9415 032410  013702  002252                  MOV     DBEMSK,R2
9416 032414  074237  002276                  XOR     R2,CHECK
9417 032420  006337  002252                  ASL     DBEMSK
9418 032424  032737  020000  002252          BIT     #BIT13,DBEMSK
9419 032432  001630                          BEQ     7$
9420 032434  006337  002246                  ASL     SBEMSK
9421 032440  032737  004000  002246          BIT     #BIT11,SBEMSK
9422 032446  001006                          BNE     16$
9423 032450  013737  002246  002252          MOV     SBEMSK,DBEMSK
9424 032456  006337  002252                  ASL     DBEMSK
9425 032462  000743                          BR      15$
9426 032464  104471                  16$:    ECC1DIS                 ;DISABLE ECC ON 1 SELECTED CSR
9427 032466  012701  002364                  MOV     #TESTADD,R1
9428 032472                                  CLEAR   @(R1)+,@(R1)
9429 032500  104503                          CLR1CSR                 ;CLEAR 1 SELECTED CSR
9430 032502  000207                          RETURN
```

```
9433 032504                                MTP015: SUBTST  <<MTP015        WRITE INHIBIT OF BYTE WITH DBE..
                                           ;*****************************************************************
                                           ;*SUBTEST        MTP015  WRITE INHIBIT OF BYTE WITH DBE
                                           ;*****************************************************************
9434                                                       ;CHECK FOR WRITE INHIBIT DURING A WRITE BYTE.
9435                                                       ;CHECKS FOR UNCORRECTED DATA.
9436 032504  005037  002236          1$:      CLR      DATBUF              ;INITIAL DATA
9437 032510  005037  002240                   CLR      DATBUF+2            ;32 BITS WORTH
9438 032514  012737  000001  002246  2$:      MOV      #1,SBEMSK           ;SINGLE ERROR MASK
9439 032522  005037  002250                   CLR      SBEMSK+2            ;
9440 032526  012737  000001  002252  3$:      MOV      #1,DBEMSK           ;DOUBLE ERROR MASK
9441 032534  005037  002254                   CLR      DBEMSK+2            ;
9442 032540  013737  002236  002242  4$:      MOV      DATBUF,TSTDAT       ;PRESERVE ORIG DATA
9443 032546  013737  002240  002244           MOV      DATBUF+2,TSTDAT+2
9444 032554  105737  002260                   TSTB     PASFLG  ;WHICH PASS ?
9445 032560  001404                           BEQ      5$                  ;FIRST PASS, NO COMPLEMENTING
9446 032562  005137  002242                   COM      TSTDAT
9447 032566  005137  002244                   COM      TSTDAT+2            ;SECOND PASS, COMPLEMENT TSTDAT
9448 032572  104503                5$:      CLR1CSR                      ;CLEAR 1 SELECTED CSR
9449 032574  023737  002246  002252           CMP      SBEMSK,DBEMSK       ;CHECK FOR SAME MASKS
9450 032602  001004                           BNE      6$                  ;BR IF NOT EQUAL
9451 032604  023737  002250  002254           CMP      SBEMSK+2,DBEMSK+2       ;SECOND WORD ALSO
9452 032612  001474                           BEQ      11$                 ;BR TO MAKE THEM NOT EQUAL
9453 032614  012737  002242  002274  6$:      MOV      #TSTDAT,SOURCE      ;ADDRESS FOR CHKGEN
9454 032622  004737  041724                   CALL     CHKGEN              ;GO GENERATE CHECK BITS
9455 032626  013701  002246                   MOV      SBEMSK,R1
9456 032632  074137  002242                   XOR      R1,TSTDAT
9457 032636  013701  002250                   MOV      SBEMSK+2,R1
9458 032642  074137  002244                   XOR      R1,TSTDAT+2
9459 032646  013701  002252                   MOV      DBEMSK,R1
9460 032652  074137  002242                   XOR      R1,TSTDAT
9461 032656  013701  002254                   MOV      DBEMSK+2,R1
9462 032662  074137  002244                   XOR      R1,TSTDAT+2
9463 032666  012701  002364          7$:      MOV      #TESTADD,R1         ;TEST LOCATION
9464 032672  104471                           ECC1DIS                      ;DISABLE ECC ON 1 SELECTED CSR
9465 032674  013731  002242                   MOV      TSTDAT,@(R1)+       ;WRITE FIRST 16 BITS
9466                                           ;LOAD CSR WITH IMAGE FROM R2
9467 032700  104475                           CB1CSR                       ;WRITE GENERATED CHECKBITS IN 1 SELECTED CSR
9468 032702  013771  002244  000000           MOV      TSTDAT+2,@(R1)      ;WRITE SECOND 16 BITS + CHECKBITS
9469 032710  104503                           CLR1CSR                      ;CLEAR 1 SELECTED CSR
9470 032712  013702  002364                   MOV      TESTADD,R2          ;GET ADDRESS OF TEST LOC
9471 032716  010203                           MOV      R2,R3               ;R2 DESIGNATES FIRST BYTE
9472 032720  062703  000003                   ADD      #3,R3               ;R3 DESIGNATES LAST BYTE
9473 032724  112722  000360          8$:      MOVB     #360,(R2)+          ;TRY WRITING A BYTE
9474 032730  012701  002364                   MOV      #TESTADD,R1
9475 032734  017100  000000                   MOV      @(R1),R0
9476 032740  023700  002242                   CMP      TSTDAT,R0           ;CHECK FOR UNCHANGED DATA
9477 032744  001404                           BEQ      9$                  ;BR IF OK
9478 032746  017137  000000  002032           MOV      @(R1),ADDRESS
9479 032754  104455                           PERR31
9480
9481 032756  017100  000002          9$:      MOV      @2(R1),R0
9482 032762  023700  002244                   CMP      TSTDAT+2,R0         ;READ SECOND WORD
9483 032766  001404                           BEQ      10$                 ;BR IF UNCHANGED
9484 032770  017137  000002  002032           MOV      @2(R1),ADDRESS
9485 032776  104455                           PERR31
9486
```

```
9487 033000  020203                    10$:   CMP     R2,R3            ;TESTED LAST BYTE ?
9488 033002  001350                           BNE     8$               ;BR IF NO
9489 033004  105737  002260          11$:     TSTB    PASFLG
9490 033010  100452                           BMI     15$              ;BRANCH IF TESTING CHECK BITS
9491 033012  005737  002254                   TST     DBEMSK+2         ;CHECKING FOR LAST ERROR BIT
9492 033016  100405                           BMI     12$              ;BR IF DONE HERE
9493 033020                                   DLEFT   DBEMSK
9494 033030  000643                           BR      4$
9495 033032                          12$:     IF @SW11 SET.IN @SWR THEN GOTO 13$
9496 033042                                   IF QVFLAG IS TRUE THEN GOTO 13$
9497 033050  005737  002250                   TST     SBEMSK+2         ;LAST SBE MASK
9498 033054  100405                           BMI     13$              ;BR IF DONE WITH THIS PASS
9499 033056                                   DLEFT   SBEMSK
9500 033066  000617                           BR      3$
9501 033070  105737  002260          13$:     TSTB    PASFLG  ;TEST PASS FLAG
9502 033074  001003                           BNE     14$              ;NON ZERO MEANS WE RE DONE
9503 033076  105237  002260                   INCB    PASFLG  ;NOT DONE
9504 033102  000600                           BR      1$
9505 033104  052737  000200  002260  14$:     BIS     @BIT7,PASFLG
9506 033112  005037  002242                   CLR     TSTDAT
9507 033116  005037  002244                   CLR     TSTDAT+2
9508 033122  012737  000040  002246           MOV     @40,SBEMSK
9509 033130  012737  000100  002252           MOV     @100,DBEMSK
9510 033136  012737  003740  002276  15$:     MOV     @3740,CHECK
9511 033144  013702  002246                   MOV     SBEMSK,R2
9512 033150  074237  002276                   XOR     R2,CHECK
9513 033154  013702  002252                   MOV     DBEMSK,R2
9514 033160  074237  002276                   XOR     R2,CHECK
9515 033164  006337  002252                   ASL     DBEMSK
9516 033170  032737  020000  002252           BIT     @BIT13,DBEMSK
9517 033176  001633                           BEQ     7$
9518 033200  006337  002246                   ASL     SBEMSK
9519 033204  032737  004000  002246           BIT     @BIT11,SBEMSK
9520 033212  001006                           BNE     16$
9521 033214  013737  002246  002252           MOV     SBEMSK,DBEMSK
9522 033222  006337  002252                   ASL     DBEMSK
9523 033226  000743                           BR      15$
9524 033230  104471                  16$:     ECC1DIS                  ;DISABLE ECC ON 1 SELECTED CSR
9525 033232  012701  002364                   MOV     @TESTADD,R1      ;TEST LOCATION
9526 033236                                   CLEAR   @(R1)+,@(R1)     ;TO ERASE ANY DBE'S FROM TESTING
9527                                          ;RESTORE CSR
9528 033244  104503                           CLR1CSR                  ;CLEAR 1 SELECTED CSR
9529 033246  000207                           RETURN
```

```
 9532 033250                            MTP016: SUBTST  <<MTP016        WRITE INHIBIT OF WORD WITH DBE
                                        ;*****************************************************************
                                        ;*SUBTEST       MTP016  WRITE INHIBIT OF WORD WITH DBE
                                        ;*****************************************************************
 9533                                           ;DOUBLE BIT ERROR WRITE CANCEL WITH
 9534                                           ;WORD WRITE.
 9535                                           ;CHECKS WRITE INHIBIT WITH WORD WRITES TO
 9536                                           ;WORD WITH DOUBLE ERROR.
 9537 033250  005037  002236           T12A:   CLR     DATBUF          ;BACKGROUND FOR DOUBLE ERRORS
 9538 033254  005037  002240                   CLR     DATBUF+2        ;2 WORDS WORTH
 9539 033260  012737  000001  002246           MOV     #1,SBEMSK       ;SINGLE ERROR MASK
 9540 033266  005037  002250                   CLR     SBEMSK+2        ;
 9541 033272  012737  000001  002252  T12B:    MOV     #1,DBEMSK       ;DOUBLE ERROR MASK
 9542 033300  005037  002254                   CLR     DBEMSK+2
 9543 033304  013737  002236  002242  1$:     MOV     DATBUF,TSTDAT   ;DATA FOR TEST
 9544 033312  013737  002240  002244          MOV     DATBUF+2,TSTDAT+2;BOTH WORDS
 9545 033320  105737  002260                   TSTB    PASFLG  ;COMP DATA ON SECOND PASS ONLY
 9546 033324  001404                          BEQ     2$              ;BR IF FIRST PASS
 9547 033326  005137  002242                   COM     TSTDAT          ;COMP FIRST WORD
 9548 033332  005137  002244                   COM     TSTDAT+2        ;NOW SECOND WORD
 9549 033336  023737  002246  002252  2$:     CMP     SBEMSK,DBEMSK   ;CHECK FOR IDENTICAL MASKS
 9550 033344  001004                          BNE     3$              ;BR IF DIFFERENT
 9551 033346  023737  002250  002254          CMP     SBEMSK+2,DBEMSK+2;UPPER WORD TOO
 9552 033354  001502                          BEQ     8$              ;BR TO MAKE THEM NOT EQUAL
 9553 033356  012737  002242  002274  3$:     MOV     #TSTDAT,SOURCE  ;NEED ADDR OF DATA FOR CHKGEN
 9554 033364  004737  041724                   CALL    CHKGEN          ;GO GENERATE CHECK BITS
 9555 033370  013701  002246                   MOV     SBEMSK,R1
 9556 033374  074137  002242                   XOR     R1,TSTDAT
 9557 033400  013701  002250                   MOV     SBEMSK+2,R1
 9558 033404  074137  002244                   XOR     R1,TSTDAT+2
 9559 033410  013701  002252                   MOV     DBEMSK,R1
 9560 033414  074137  002242                   XOR     R1,TSTDAT
 9561 033420  013701  002254                   MOV     DBEMSK+2,R1
 9562 033424  074137  002244                   XOR     R1,TSTDAT+2
 9563 033430  012701  002364          4$:     MOV     #TESTADD,R1     ;FIRST TEST ADDRESS
 9564 033434  104471                          ECC1DIS                 ;DISABLE ECC ON 1 SELECTED CSR
 9565 033436  013731  002242                   MOV     TSTDAT,@(R1)+   ;WRITE FIRST 16 BITS
 9566 033442  104475                          CB1CSR                  ;WRITE GENERATED CHECKBITS IN 1 SELECTED CSR
 9567 033444  013771  002244  000000          MOV     TSTDAT+2,@(R1)  ;WRITE SECOND 16 BITS + CHECKBITS
 9568 033452  105037  002261                   CLRB    UPPFLG          ;SET FOR 2 LOOPS
 9569 033456  162701  000002                   SUB     #2,R1           ;POINT TO LOW WORD
 9570 033462  104503                  5$:     CLR1CSR                 ;CLEAR 1 SELECTED CSR
 9571 033464  012771  177400  000000          MOV     #177400,@(R1)   ;TRY WRITING LOCATION
 9572 033472  012701  002364                   MOV     #TESTADD,R1
 9573 033476  017100  000000                   MOV     @(R1),R0
 9574 033502  023700  002242                   CMP     TSTDAT,R0       ;CHECK FOR ORIGINAL DATA
 9575 033506  001404                          BEQ     6$              ;SHOULD BE UNCHANGED
 9576 033510  017137  000000  002032          MOV     @(R1),ADDRESS
 9577 033516  104455                          PERR31
 9578
 9579 033520  062701  000002          6$:     ADD     #2,R1
 9580 033524  017100  000000                   MOV     @(R1),R0
 9581 033530  023700  002244                   CMP     TSTDAT+2,R0     ;THIS SHOULD BE UNCHANGED ALSO
 9582 033534  001404                          BEQ     7$
 9583 033536  017137  000000  002032          MOV     @(R1),ADDRESS
 9584 033544  104455                          PERR31
```

```
9587 033546  105737  002261          7$:     TSTB    UPPFLG          ;WHICH LOOP ?
9588 033552  001003                          BNE     8$              ;SECOND, BR OUT
9589 033554  105237  002261                  INCB    UPPFLG          ;FIRST, KEEP GOING
9590 033560  000740                          BR      5$
9591 033562  105737  002260          8$:     TSTB    PASFLG
9592 033566  100454                          BMI     12$
9593 033570  005737  002254                  TST     DBEMSK+2        ;LAST BIT ?
9594 033574  100405                          BMI     9$              ;MINUS - BIT 31
9595 033576                                  DLEFT   DBEMSK
9596 033606  000636                          BR      1$
9597 033610                          9$:     IF @SW11 SET.IN @SWR THEN GOTO 10$
9598 033620                                  IF QVFLAG IS TRUE THEN GOTO 10$
9599 033626  005737  002250                  TST     SBEMSK+2        ;LAST BIT IN THIS MASK ?
9600 033632  100406                          BMI     10$             ;BR IF LAST BIT
9601 033634                                  DLEFT   SBEMSK
9602 033644  000137  033272                  JMP     T12B
9603 033650  105737  002260          10$:    TSTB    PASFLG  ;FIRST PASS ?
9604 033654  001004                          BNE     11$             ;BR IF SECOND
9605 033656  105237  002260                  INCB    PASFLG  ;INDICATE SECOND PASS COMING
9606 033662  000137  033250                  JMP     T12A
9607 033666  052737  000200  002260  11$:    BIS     @BIT7,PASFLG
9608 033674  005037  002242                  CLR     TSTDAT
9609 033700  005037  002244                  CLR     TSTDAT+2
9610 033704  012737  000040  002246          MOV     @40,SBEMSK
9611 033712  012737  000100  002252          MOV     @100,DBEMSK
9612 033720  012737  003740  002276  12$:    MOV     @3740,CHECK
9613 033726  013702  002246                  MOV     SBEMSK,R2
9614 033732  074237  002276                  XOR     R2,CHECK
9615 033736  013702  002252                  MOV     DBEMSK,R2
9616 033742  074237  002276                  XOR     R2,CHECK
9617 033746  006337  002252                  ASL     DBEMSK
9618 033752  032737  020000  002252          BIT     @BIT13,DBEMSK
9619 033760  001623                          BEQ     4$
9620 033762  006337  002246                  ASL     SBEMSK
9621 033766  032737  004000  002246          BIT     @BIT11,SBEMSK
9622 033774  001006                          BNE     13$
9623 033776  013737  002246  002252          MOV     SBEMSK,DBEMSK
9624 034004  006337  002252                  ASL     DBEMSK
9625 034010  000743                          BR      12$
9626 034012  104471                  13$:    ECC1DIS                 ;DISABLE ECC ON 1 SELECTED CSR
9627 034014  012701  002364                  MOV     @TESTADD,R1     ;RESTORE TEST ADDRESS
9628 034020  005031                          CLR     @(R1)+          ;CLEAR ANY DBE'S FROM TEST
9629 034022  005071  000000                  CLR     @(R1)
9630 034026  104503                          CLR1CSR                 ;CLEAR 1 SELECTED MK11 CSR
9631 034030  000207                          RETURN
```

```
 9634 034032                            MTP017: SUBTST  <<MTP017        HOLDING 1'S & 0'S TEST>>
                                        ;*****************************************************************************
                                        ;*SUBTEST        MTP017  HOLDING 1'S & 0'S TEST
                                        ;*****************************************************************************
 9635                                           ;*(1)    THIS TEST CHECKS THE MEMORY FOR THE CAPABILITY
 9636                                           ;*       OF HOLDING 1'S AND 0'S BY WRITING A BACKGROUND
 9637                                           ;*       OF 000377 AND READING IT
 9638                                           ;*(2)    MEMORY IS WRITTEN USING A BYTE AT A TIME
 9639                                           ;*(3)    STEPS 1 & 2 ARE REPEATED WITH A SWAPPED BACKGROUND PATTERN
 9640                                           ;NOTE:   THIS TEST WRITES BYTES & READS WORDS
 9641 034032  012701  060000                   MOV      #FIRST,R1
 9642 034036  010104                           MOV      R1,R4
 9643 034040  012705  160000                   MOV      #LAST-2,R5
 9644 034044  012700  000377                   MOV      #377,R0         ;GET THE PATTERN INTO R0
 9645 034050  010003                           MOV      R0,R3
 9646 034052  000303                           SWAB     R3
 9647 034054  110021                  1$:      MOVB     R0,(R1)+        ;WRITE A BYTE
 9648 034056  110321                           MOVB     R3,(R1)+        ;WRITE THE MEMORY WITH THE BYTE STORED IN BAKPAT.1
 9649 034060  020105                           CMP      R1,R5           ;COMPARE TEST LOC TO TOP - 2
 9650 034062  103774                           BLO      1$              ;BRANCH IF LOWER
 9651
 9652 034064  014102                  2$:      MOV      (R1),R2
 9653 034066  020002                           CMP      R0,R2           ;TEST THE MEMORY TO SEE IF IT CONTAINS
 9654                                                                   ;THE WORD STORED IN BAKPAT
 9655 034070  001401                           BEQ      3$
 9656 034072  104446                           PERR22
 9657
 9658 034074  020104                  3$:      CMP      R1,R4           ;KEEP ON TESTING THE MEMORY UNTIL
 9659 034076  101372                           BHI      2$              ;R1 EQUALS THE LOWEST ADDRESS
 9660 034100  000303                           SWAB     R3              ;CHANGE THE DATA PATTERN
 9661 034102  000300                           SWAB     R0
 9662 034104  001763                           BEQ      1$              ;IF THE DATA PATTERN DOES NOT HAVE LOW
 9663                                                                   ; BYTE =0 THEN FALL THRU
 9664 034106  000207                           RETURN
```

```
9667 034110                    MTP020: SUBTST  <<MTP020        MARCHING 1'S & 0'S IN CHECK BITS TEST>>
                               ;*****************************************************************************
                               ;*SUBTEST       MTP020  MARCHING 1'S & 0'S IN CHECK BITS TEST
                               ;*****************************************************************************
9668                                   ;*THIS TEST IS CONCERNED ONLY WITH THE INTEGRITY
9669                                   ;*OF THE MOS RAMS THAT STORE THE CHECKBITS.
9670
9671                                   ;077    > 100 DOWN
9672 034110  160201           MTPA20: SUB     R2,R1           ;V177640
9673 034112  160204                   SUB     R2,R4           ;V177642
9674 034114  005711                   TST     (R1)            ;V177644        ;1ST WORD OK?
9675 034116  001002                   BNE     1$              ;V177646        ;NO   SKIP
9676 034120  005714                   TST     (R4)            ;V177650        ;2ND WORD OK?
9677 034122  001401                   BEQ     2$              ;V177652        ;YES   SKIP
9678 034124  104453           1$:     PERR27                  ;V177654        ;GOOD=000000..000000..077
9679 034126  010314           2$:     MOV     R3,(R4)         ;V177656        ;2ND WORD <= 100000
9680 034130  005011                   CLR     (R1)            ;V177660        ;CLEAR 1ST WORD
9681 034132  020100                   CMP     R1,R0           ;V177662        ;ARE WE DONE?
9682 034134  101365                   BHI     MTPA20          ;V177664        ;BRANCH IF NOT
9683 034136  000207                   RETURN                  ;V177666
9684
9685                                   ;100 --> 077 UP
9686 034140  005711           MTPB20: TST     (R1)            ;V177640        ;1ST WORD OK?
9687 034142  001002                   BNE     3$              ;V177642        ;NO - SKIP
9688 034144  020314                   CMP     R3,(R4)         ;V177644        ;2ND WORD OK?
9689 034146  001401                   BEQ     4$              ;V177646        ;YES   SKIP
9690 034150  104452           3$:     PERR26                  ;V177650        ;GOOD=000000..100000..100
9691 034152  005014           4$:     CLR     (R4)            ;V177652        ;CLEAR 2ND WORD
9692 034154  005011                   CLR     (R1)            ;V177654        ;CLEAR 1ST WORD
9693 034156  060201                   ADD     R2,R1           ;V177656
9694 034160  060204                   ADD     R2,R4           ;V177660
9695 034162  020405                   CMP     R4,R5           ;V177662        ;TOP + 2 YET?
9696 034164  001365                   BNE     MTPB20          ;V177664        ;NO   LOOP
9697 034166  000207                   RETURN                  ;V177666
9698
9699                                   ;077 --> 100 UP
9700 034170  005711           MTPC20: TST     (R1)            ;V177640        ;1ST WORD OK?
9701 034172  001002                   BNE     5$              ;V177642        ;NO   SKIP
9702 034174  005714                   TST     (R4)            ;V177644        ;2ND WORD OK?
9703 034176  001401                   BEQ     6$              ;V177646        ;YES   SKIP
9704 034200  104453           5$:     PERR27                  ;V177650        ;GOOD=000000..000000..077
9705 034202  010314           6$:     MOV     R3,(R4)         ;V177652        ;WRITE 1ST WORD
9706 034204  005011                   CLR     (R1)            ;V177654        ;WRITE 2ND WORD
9707 034206  060204                   ADD     R2,R4           ;V177656
9708 034210  060201                   ADD     R2,R1           ;V177660
9709 034212  020405                   CMP     R4,R5           ;V177662        ;TOP + 2 YET?
9710 034214  001365                   BNE     MTPC20          ;V177664        ;NO   LOOP
9711 034216  000207                   RETURN                  ;V177666
```

```
 9714
 9715 034220  160201                    ;100  -> 077 DOWN
 9716 034222  160204  MTPD20: SUB   R2,R1          ;V177640
 9717 034224  020314          SUB   R2,R4          ;V177642
 9718 034226  001002          CMP   R3,(R4)        ;V177644    ;2ND WORD OK?
 9719 034230  005711          BNE   7$             ;V177646    ;NO   SKIP
 9720 034232  001401          TST   (R1)           ;V177650    ;1ST WORD OK?
 9721 034234  104452          BEQ   8$             ;V177652    ;YES   SKIP
 9722 034236  005014  7$:     PERR26               ;V177654    ;GOOD=000000..100000..100
 9723 034240  005011  8$:     CLR   (R4)           ;V177656    ;WRITE 1ST WORD
 9724 034242  020100          CLR   (R1)           ;V177660    ;WRITE 2ND WORD
 9725 034244  101365          CMP   R1,R0          ;V177662
 9726 034246  000207          BHI   MTPD20         ;V177664
 9727                         RETURN               ;V177666
 9728
 9729 034250  005711                    ;077 UP
 9730 034252  001002  MTPE20: TST   (R1)           ;V177640    ;1ST WORD OK?
 9731 034254  005714          BNE   9$             ;V177642    ;NO   SKIP
 9732 034256  001401          TST   (R4)           ;V177644    ;2ND WORD OK?
 9733 034260  104453          BEQ   10$            ;V177646    ;YES   SKIP
 9734 034262  060201  9$:     PERR27               ;V177650    ;GOOD=000000..000000..077
 9735 034264  060204  10$:    ADD   R2,R1          ;V177652
 9736 034266  020405          ADD   R2,R4          ;V177654
 9737 034270  001367          CMP   R4,R5          ;V177656    ;TOP + 2 YET?
 9738 034272  000207          BNE   MTPE20         ;V177660    ;NO   LOOP
                              RETURN               ;V177662
```

```
9741 034274                          MTPA21: SUBTST  <<MTPA21        MARCHING 1'S & 0'S PATTERN TE>'..
                                     ;*****************************************************************************
                                     ;*SUBTEST        MTPA21  MARCHING 1'S & 0'S PATTERN TEST
                                     ;*****************************************************************************
9742                                         ;READ,BYTESWAP MODIFY,READ,DOWN
9743 034274  014100                  1$:     MOV     -(R1),R0;V177640
9744 034276  020200                          CMP     R2,R0   ;V177642
9745 034300  001401                          BEQ     2$      ;V177644
9746 034302  104443                          PERR17          ;V177646
9747
9748 034304  000311                  2$:     SWAB    (R1)    ;V177650
9749 034306  011100                          MOV     (R1),R0 ;V177652
9750 034310  020300                          CMP     R3,R0   ;V177654
9751 034312  001401                          BEQ     3$      ;V177656
9752 034314  104444                          PERR20          ;V177660
9753
9754 034316  020401                  3$:     CMP     R4,R1   ;V177662        ;DONE?
9755 034320  001365                          BNE     1$      ;V177664        ;NO   LOOP
9756 034322  000207                          RETURN          ;V177666        ;YES   RETURN
9757
9758 034324                          MTPB21: ;READ,BYTESWAP MODIFY,READ,UP
9759 034324  011100                  1$:     MOV     (R1),R0 ;V177640
9760 034326  020300                          CMP     R3,R0   ;V177642
9761 034330  001401                          BEQ     2$      ;V177644
9762 034332  104444                          PERR20          ;V177646
9763
9764 034334  000311                  2$:     SWAB    (R1)    ;V177650
9765 034336  011100                          MOV     (R1),R0 ;V177652
9766 034340  020200                          CMP     R2,R0   ;V177654
9767 034342  001401                          BEQ     3$      ;V177656
9768 034344  104443                          PERR17          ;V177660
9769
9770 034346  062701  000002          3$:     ADD     #2,R1   ;V177662
9771 034352  020501                          CMP     R5,R1   ;V177666        ;DONE?
9772 034354  001363                          BNE     1$      ;V177670        ;NO   LOOP
9773 034356  000207                          RETURN          ;V177672        ;YES   RETURN
9774
9775 034360                          MTPC21: ;READ,BYTESWAP MODIFY,READ,UP
9776 034360  011100                  1$:     MOV     (R1),R0 ;V177640
9777 034362  020200                          CMP     R2,R0   ;V177642
9778 034364  001401                          BEQ     2$      ;V177644
9779 034366  104443                          PERR17          ;V177646
9780
9781 034370  000311                  2$:     SWAB    (R1)    ;V177650
9782 034372  011100                          MOV     (R1),R0 ;V177652
9783 034374  020300                          CMP     R3,R0   ;V177654
9784 034376  001401                          BEQ     3$      ;V177656
9785 034400  104444                          PERR20          ;V177660
9786
9787 034402  062701  000002          3$:     ADD     #2,R1   ;V177662
9788 034406  020501                          CMP     R5,R1   ;V177666        ;DONE?
9789 034410  001363                          BNE     1$      ;V177670        ;NO   LOOP
9790 034412  000207                          RETURN          ;V177672        ;YES   RETURN
```

```
9795 034414                        MTPD21: ;READ,BYTESWAP MODIFY,READ,DOWN
9794 034414  014100                1$:     MOV     (R1),R0 ;V177640
9795 034416  020300                        CMP     R3,R0   ;V177642
9796 034420  001401                        BEQ     2$      ;V177644
9797 034422  104444                        PERR20          ;V177646
9798
9799 034424  000311                2$:     SWAB    (R1)    ;V177650
9800 034426  011100                        MOV     (R1),R0 ;V177652
9801 034430  020200                        CMP     R2,R0   ;V177654
9802 034432  001401                        BEQ     3$      ;V177656
9803 034434  104443                        PERR17          ;V177660
9804
9805 034436  020401                3$:     CMP     R4,R1   ;V177662        ;DONE?
9806 034440  001365                        BNE     1$      ;V177664        ;NO   LOOP
9807 034442  000207                        RETURN          ;V177666        ;YES  RETURN
9808
```

```
9811 034444                    MTP022: SUBTST  <<MTP022        REFRESH & SHIFTING DIAGONAL TEST>>
                               ;*****************************************************************************
                               ;*SUBTEST       MTP022  REFRESH & SHIFTING DIAGONAL TEST
                               ;*****************************************************************************
9812                                   ;(1)    WE WRITE A DIAGONAL PATTERN IN MEMORY (WITH CACHE ON).
9813                                   ;(2)    IF A REFRESH TEST WE DISTURB ALL ROWS FOR > 2 MS (WITH CACHE ON).
9814                                   ;(3)    WE READ & CHECK FOR CORRECTNESS THE DIAGONAL PATTERN
9815                                   ;       (WITH CACHE OFF).
9816        000010             KDIAG=8.                    ;HOW OFTEN A DIAGONAL STRIPE OCCURS (MUST BE A POWER OF 2)
9817 034444                    FOR EVEN := #1 TO #2                        ;FOR DATA & COMPLEMENT DATA
9818 034452                      IF EVEN EQ #1
9819 034462                        LET R2 := ZEROS
9820 034466                        LET R3 := ONES
9821 034472                      ELSE
9822 034474                        LET R2 := ONES
9823 034500                        LET R3 := ZEROS
9824 034504                      END ;OF IF EVEN
9825 034504                      FOR STRIPES := #0 TO #KDIAG-1          ;FOR THE NUMBER OF STRIPES
9826
9827
9828 034510  104423                  ;WRITE LOOP
                                     CACHON                          ;TURN CACHE ON
9829 034512                          LET COUNT := STRIPES
9830 034520                          LET R1 := #FIRST
9831 034524                          WHILE R1 LOS #LAST
9832 034532                            IF COUNT LT #0 THEN LET COUNT := #KDIAG-1
9833 034546                            IF #374 OFF.IN R1 THEN LET COUNT := COUNT   #1
9834 034560                            IF COUNT NE #0
9835 034566                              LET (R1) := R2
9836 034570                              LET 2(R1) := R2
9837 034574                            ELSE
9838 034576                              LET (R1) := R3
9839 034600                              LET 2(R1) := R3
9840 034604                            END ;OF IF COUNT
9841 034604                            LET COUNT := COUNT   #1
9842 034610                            LET R1 := R1 + #4
9843 034614                          END ;OF WHILE
9844                                 ;END OF WRITE LOOP
9845
9846 034616                          IF DIAGFLAG IS FALSE THEN ;CALL REFRESH
9847                                 ;READ LOOP
9848 034630                          LET COUNT := STRIPES
9849 034636                          LET R1 := #FIRST
9850 034642  104424                 CACHOFF                         ;TURN CACHE OFF
```

```
9852 034644                                     WHILE R1 LOS @LAST
9853 034652                                       IF COUNT LT @0 THEN LET COUNT :* @#DIAG 1
9854 034666                                       IF @374 OFF.IN R1 THEN LET COUNT :* COUNT   #1
9855 034700                                       IF COUNT NE @0
9856 034706                                         LET R0 :* (R1)
9857 034710                                         IF R2 NE R0
9858 034714  104443                                   PERR17
9859 034716                                         END ;OF IF R2
9860 034716                                         LET R0 :* 2(R1)
9861 034722                                         IF R2 NE R0
9862 034726  104443                                   PERR17
9863 034730                                         END ;OF IF R2
9864 034730                                       ELSE
9865 034732                                         LET R0 :* (R1)
9866 034734                                         IF R3 NE R0
9867 034740  104444                                   PERR20
9868 034742                                         END ;OF IF R3
9869 034742                                         LET R0 :* 2(R1)
9870 034746                                         IF R3 NE R0
9871 034752  104444                                   PERR20
9872 034754                                         END ;OF IF R3
9873 034754                                       END ;OF IF COUNT
9874 034754                                       LET COUNT :* COUNT   #1
9875 034760                                       LET R1 :* R1 + #4
9876 034764                                     END ;OF WHILE
9877                                            ;END OF READ LOOP
9878
9879 034766                                   END ;OF FOR STRIPES
9880 035002                                 END ;OF FOR EVEN
9881 035016  000207                         RETURN
9882
9883 035020                           REFRESH:SUBTST  <<SUBR  REFRESH DELAY>>
                                      ;*****************************************************************************
                                      ;*SUBTEST       SUBR    REFRESH DELAY
                                      ;*****************************************************************************
9884                                          ;DISTURB EACH ROW FOR > 3.2 MS
9885 035020                                   FOR R0 :* @FIRST TO @FIRST+374 BY #4
9886 035024  004737  035070                     CALL REFSUB
9887 035030                                   END ;OF FOR R0
9888 035042                                   LET R0 :* @FIRST+BIT14
9889 035046                                   WHILE R0 LOS @LAST+BIT14+374
9890 035054  004737  035070                     CALL REFSUB
9891 035060                                     LET R0 :* R0 + #4
9892 035064                                   END ;OF WHILE
9893 035066  000207                           RETURN
9894 035070  012704  000640          REFSUB:  MOV     #640,R4                ;TIME FOR A > 3.2 MS LOOP
9895 035074  062700  000002                   ADD     #2,R0
9896 035100  005140                  1$:      COM      (R0)
9897 035102  005120                           COM     (R0)+
9898 035104  005110                           COM     (R0)
9899 035106  005110                           COM     (R0)
9900 035110  077405                           SOB     R4,1$
9901 035112  162700  000002                   SUB     #2,R0
9902 035116  000207                           RETURN
```

```
9906 035120                    MTPA24: SUBTST  <<MTPA24        FAST GALLOPING PATTERN TEST>>
                               ;*****************************************************************************
                               ;*SUBTEST       MTPA24  FAST GALLOPING PATTERN TEST
                               ;*****************************************************************************
9907                                   ;THE TOTAL TEST (INCLUDING SETUP) IS AS FOLLOWS
9908                                   ;*(1)   THIS TEST WRITES THE MEMORY WITH A BACK GROUND PATTERN
9909                                   ;*      STORED AT LOCATION BAKPAT
9910                                   ;*(2)   TEST BEGINS AT LOWEST LOCATION BEING TESTED
9911                                   ;*      (LETS NAME IT 'A')
9912                                   ;*(3)   LETS NAME THE 1ST LOCATION IN THE ROW/COLUMN UNDER TEST AS 'B'.
9913                                   ;*(4)   SWAPS BYTES FOR LOCATION 'A .
9914                                   ;*(5)   READS  A', READS 'B'
9915                                   ;*(6)    B' = B'+400   (ADDS 64 DOUBLE WORDS TO 'B')
9916                                   ;*(7)   REPEATS STEPS 5 AND 6 UNTIL 'B  IS GREATER THAN THE
9917                                   ;*(8)   END OF THE BANK  A+2
9918                                   ;*(9)   REPEATS  STEPS 3-8 UNTILL 'A' REACHES THE END OF THE BANK
9919                                   ;*(10)  AFTER EXECUTING THE TEST  DATA IS COMPLEMENTED
9920                                   ;*      AND STEPS 1 9 ARE REPEATED
9921                                   ;REGISTERS ARE USED AS FOLLOWS
9922                                   ;R0     TEST DATA
9923                                   ;R1     'A'
9924                                   ;R2     'B'
9925                                   ;R3     BAKPAT
9926                                   ;R4     SWAPAT
9927                                   ;R5     LAST
9928
9929                                   ;NOTE THE PATTERN STARTS AT MTPB24!!!!!!!!!!!!!!!
9930
9931                                   ;UIPAR'S
9932 035120  011100           1$:     MOV     (R1),R0         ;V177640        ;READ 'A
9933 035122  020004                   CMP     R0,R4           ;V177642        ;CHECK  A
9934 035124  001401                   BEQ     2$              ;V177644        ;BR IF OK
9935 035126  104447                   PERR23                  ;V177646        ;REPORT ERROR
9936
9937 035130  011200           2$:     MOV     (R2),R0         ;V177650        ;READ 'B'
9938 035132  020003                   CMP     R0,R3           ;V177652        ;CHECK  B'
9939 035134  001401                   BEQ     3$              ;V177654        ;BR IF OK
9940 035136  104450                   PERR24                  ;V177656        ;REPORT ERROR
9941
9942 035140  062702  000400    3$:     ADD     #400,R2         ;V177660        ;BUMP 'B'
9943 035144  020205                   CMP     R2,R5           ;V177664        ;AT END YET?
9944 035146  101764                   BLOS    1$              ;V177666        ;BR IF NO
9945
9946 035150  062701  000002           ADD     #2,R1           ;V177670        ;BUMP 'A'
9947 035154  000137  035160           JMP     @#MTPB24        ;V177674        ;GOTO V177260
```

```
9950 035160                          MTPB24: SUBTST  <<MTPB24        FAST GALLOP PART B>>
                                     ;*********************************************************************
                                     ;*SUBTEST        MTPB24  FAST GALLOP PART B
                                     ;*********************************************************************
9951                                         ;SDPAR S
9952 035160  010411                          MOV     R4,(R1)         ;V172260        ;WRITE  A
9953 035162  020105                          CMP     R1,R5           ;V172262        ;DONE?
9954 035164  001001                          BNE     1$              ;V172264        ;BR IF NO
9955 035166  000207                          RETURN                  ;V172266        ;YES   RETURN
9956 035170  000137  035174          1$:     JMP     @#MTPC24        ;V172270        ;GOTO V172360
9957
9958 035174                          MTPC24: SUBTST  <<MTPC24        FAST GALLOP PART C>>
                                     ;*********************************************************************
                                     ;*SUBTEST        MTPC24  FAST GALLOP PART C
                                     ;*********************************************************************
9959                                         ;KDPAR'S
9960 035174  010102                          MOV     R1,R2           ;V172360        ;RESET 'B' <      'A'
9961 035176  011100                          MOV     (R1),R0         ;V172362        ;READ  A'
9962 035200  020004                          CMP     R0,R4           ;V172364        ;CHECK 'A'
9963 035202  001401                          BEQ     1$              ;V172366        ;BR IF OK
9964 035204  104447                          PERR23                  ;V172370        ;REPORT ERROR
9965 035206  000137  035140          1$:     JMP     @#MTPA24+20     ;V172372        ;GOTO V177660
```

```
 9968 035212                                MTP025: SUBTST  <<MTP025          INTERRUPT ENABLE TEST>>
                                            ;*****************************************************************************
                                            ;*SUBTEST        MTP025  INTERRUPT ENABLE TEST
                                            ;*****************************************************************************
 9969 035212  005037  002242                        CLR     TSTDAT          ;GENERATE CHECKBITS ON 0,,0
 9970 035216  005037  002244                        CLR     TSTDAT+2
 9971 035222  012737  002242  002274                MOV     #TSTDAT,SOURCE
 9972 035230  004737  041724                        CALL    CHKGEN
 9973 035234  012737  000003  002074                MOV     #3,NOPAR                ;SETUP PARITY ACTION
 9974 035242  012701  002364                        MOV     #TESTADD,R1     ;FIRST TEST ADDRESS
 9975 035246  012737  035306  002266                MOV     #1$,PARTHERE    ;SETUP TRAP DESTINATION
 9976 035254  004737  035530                        CALL    MTPA25          ;WRITE DATA & CHECKBITS
 9977 035260  104473                                ECC1INIT                ;INITIALIZE 1 SELECTED MK11 CSR
 9978 035262  005771  000000                        TST     @(R1)           ;ACCESS LOCATIONS FOR DBE TRAPS
 9979 035266  005771  000002                        TST     @2(R1)
 9980                                               ;NONE   GOOD - ACCESS FOR SBE TRAPS
 9981 035272  104507                                ENA1SBE                 ;DISABLE TRAPS ON SBE S FROM 1 SELECTED CSR
 9982 035274  005771  000000                        TST     @(R1)
 9983 035300  005771  000002                        TST     @2(R1)
 9984 035304  000404                                BR      2$              ;NONE - GOOD   SKIP
 9985 035306  104426                        1$:     READCSR
 9986 035310                                        FATAL   27
 9987 035316  005237  002242                2$:     INC     TSTDAT          ;CHECK FOR CORRECT ACTION ON SBE'S
 9988 035322  004737  035456                        CALL    MTPD25          ;IN ALL 4 BYTES
 9989 035326  012737  000400  002242                MOV     #400,TSTDAT
 9990 035334  004737  035456                        CALL    MTPD25
 9991 035340  005037  002242                        CLR     TSTDAT
 9992 035344  005237  002244                        INC     TSTDAT+2
 9993 035350  004737  035456                        CALL    MTPD25
 9994 035354  012737  000400  002244                MOV     #400,TSTDAT+2
 9995 035362  004737  035456                        CALL    MTPD25
 9996
 9997 035366  005037  002244                        CLR     TSTDAT+2        ;CHECK FOR CORRECT ACTION ON DBE'S
 9998 035372  012737  000003  002242                MOV     #3,TSTDAT       ;IN ALL 4 BYTES
 9999 035400  004737  035500                        CALL    MTPE25
10000 035404  012737  001400  002242                MOV     #1400,TSTDAT
10001 035412  004737  035500                        CALL    MTPE25
10002 035416  005037  002242                        CLR     TSTDAT
10003 035422  012737  000003  002244                MOV     #3,TSTDAT+2
10004 035430  004737  035500                        CALL    MTPE25
10005 035434  012737  001400  002244                MOV     #1400,TSTDAT+2
10006 035442  004737  035500                        CALL    MTPE25
10007 035446  104503                                CLR1CSR                 ;CLEAR 1 SELECTED MK11 CSR
10008 035450  005037  002074                        CLR     NOPAR           ;INDICATE PARITY ACTION
10009 035454  000207                                RETURN
10010
10011 035456  004737  035530                MTPD25: CALL    MTPA25          ;WRITE DATA & CHECKBITS
10012 035462  104471                                ECC1DIS                 ;DISABLE ECC ON 1 SELECTED CSR
10013 035464  004737  035552                        CALL    MTPB25          ;CHECK FOR NO TRAPS
10014 035470  104507                                ENA1SBE                 ;DISABLE TRAPS ON SBE'S FROM 1 SELECTED CSR
10015 035472  004737  035612                        CALL    MTPC25          ;CHECK FOR EXPECTED TRAP
10016 035476  000207                                RETURN
```

```
10019 035500  004737  035530          MTPE25: CALL    MTPA25          ;WRITE DATA & CHECKBITS
10020 035504  104471                          ECC1DIS                 ;DISABLE ECC ON 1 SELECTED CSR
10021 035506  004737  035552                  CALL    MTPB25          ;CHECK FOR NO TRAPS
10022                                         ;ENABLE DBE TRAPS
10023 035512  104473                          ECC1INIT                ;INITIALIZE 1 SELECTED MX11 CSR
10024 035514  004737  035612                  CALL    MTPC25          ;CHECK FOR EXPECTED TRAP
10025 035520  104507                          ENA1SBE                 ;DISABLE TRAPS ON SBE'S FROM 1 SELECTED CSR
10026 035522  004737  035612                  CALL    MTPC25          ;CHECK FOR EXPECTED TRAP
10027 035526  000207                          RETURN
10028
10029                                         ;WRITE TSTDAT & TSTDAT+2 & CHECKBITS
10030 035530  104471          MTPA25: ECC1DIS                         ;DISABLE ECC ON 1 SELECTED CSR
10031 035532  013771  002242  000000          MOV     TSTDAT,@(R1)    ;WRITE FIRST 16 BITS
10032 035540  104475                          CB1CSR                  ;WRITE GENERATED CHECKBITS IN 1 SELECTED CSR
10033 035542  013771  002244  000002          MOV     TSTDAT+2,@2(R1) ;WRITE 2ND 16 BITS & CHECKBITS
10034 035550  000207                          RETURN
10035
10036                                         ;CHECK FOR NO TRAP OCCURING CONDITION
10037 035552  012737  035572  002266  MTPB25: MOV     #1$,PARTHERE    ;SETUP TRAP DESTINATION
10038 035560  005771  000000                  TST     @(R1)           ;ACCESS LOCATIONS
10039 035564  005771  000002                  TST     @2(R1)
10040 035570  000207                          RETURN                  ;NO TRAP   GOOD   RETURN
10041
10042 035572  104426          1$:     READCSR
10043 035574  011137  002032                  MOV     (R1),ADDRESS    ;SAVE VIRTUAL ADDRESS
10044 035600  104024                          ERROR   .24
10045 035602                                  SET     HEADER
10046 035610  000207                          RETURN
10047
10048                                         ;TRAP SHOULD OCCURE TEST
10049 035612  012737  035626  002266  MTPC25: MOV     #1$,PARTHERE    ;SETUP TRAP DESTINATION
10050 035620  005771  000000                  TST     @(R1)           ;ACCESS 1ST LOCATION
10051 035624  000405                          BR      2$              ;NO TRAP   BAD NEWS   SKIP
10052 035626  012737  035656  002266  1$:     MOV     #3$,PARTHERE    ;SETUP TRAP DESTINATION
10053 035634  005771  000002                  TST     @2(R1)          ;ACCESS 2ND LOCATION
10054 035640  104426          2$:     READCSR                         ;NO TRAP   BAD NEWS
10055 035642  011137  002032                  MOV     (R1),ADDRESS    ;SAVE VIRTUAL ADDRESS
10056 035646  104025                          ERROR   .25
10057 035650                                  SET     HEADER
10058 035656  000207          3$:     RETURN
10059
```

```
10062 035660                          MTPA26: SUBTST  <<MTPA26         RANDOM DATA (WRITE)>>
                                      ;*****************************************************************************
                                      ;*SUBTEST        MTPA26  RANDOM DATA (WRITE)
                                      ;*****************************************************************************
10063 035660  000137  035730          1$:     JMP     @#MTPC26        ;V177640        GOTO V172360
10064 035664  010221                          MOV     R2,(R1)+        ;V177644
10065 035666  010321                          MOV     R3,(R1)+        ;V177646
10066 035670  077005                          SOB     R0,1$           ;V177650
10067 035672  000207                          RETURN                  ;V177652
10068
10069 035674                          MTPB26: SUBTST  <<MTPB26         RANDOM DATA (READ)>>
                                      ;*****************************************************************************
                                      ;*SUBTEST        MTPB26  RANDOM DATA (READ)
                                      ;*****************************************************************************
10070                                         .DSABL  AMA
10071                                         .ENABL  LSB
10072 035674  000137  035730          1$:     JMP     @#MTPC26        ;V177640        GOTO V172360
10073 035700  020221                          CMP     R2,(R1)+        ;V177644
10074 035702  001401                          BEQ     2$              ;V177646
10075 035704  104451                          PERR25                  ;V177650
10076 035706  005127                  2$:     COM     (PC)+           ;V177652
10077 035710  000000                  RANODD: 0                       ;V177654        FOR ERROR REPORTING
10078 035712  020321                          CMP     R3,(R1)+        ;V177656
10079 035714  001401                          BEQ     3$              ;V177660
10080 035716  104451                          PERR25                  ;V177662
10081 035720  005167  177764          3$:     COM     RANODD          ;V177664
10082 035724  077015                          SOB     R0,1$           ;V177670
10083 035726  000207                          RETURN                  ;V177672
10084                                         .DSABL  LSB
10085                                         .ENABL  AMA
10086
10087 035730                          MTPC26: SUBTST  <<RANDOM NUMBER SUBPROGRAM>>
                                      ;*****************************************************************************
                                      ;*SUBTEST        RANDOM NUMBER SUBPROGRAM
                                      ;*****************************************************************************
10088                                         ;CALLER MUST SETUP
10089                                         ;       MOV     SEEDLO,R3
10090                                         ;       MOV     SEEDHI,R2
10091                                         ;       MOV     R3,R5
10092                                         ;       MOV     R2,R4
10093 035730  073427  000007                  ASHC    #7,R4           ;V172360
10094 035734  060305                          ADD     R3,R5           ;V172364
10095 035736  005504                          ADC     R4              ;V172366
10096 035740  060204                          ADD     R2,R4           ;V172370
10097 035742  062705  001057                  ADD     #1057,R5        ;V172372
10098 035746  000240                          NOP                     ;V172376        GOTO V172260
10099
10100 035750                          MTPD26: SUBTST  <<RANDOM NUMBER SUBSUBPROGRAM>>
                                      ;*****************************************************************************
                                      ;*SUBTEST        RANDOM NUMBER SUBSUBPROGRAM
                                      ;*****************************************************************************
10101 035750  005504                          ADC     R4              ;V172260
10102 035752  062704  047401                  ADD     #47401,R4       ;V172262
10103 035756  010503                          MOV     R5,R3           ;V172266
10104 035760  010402                          MOV     R4,R2           ;V172270
10105 035762  000137  035664                  JMP     @#MTPA26+4      ;V172272        GOTO V177644
```

```
10108 035766                          MTP030: SUBTST  <<MTO030        FLUSH OUT DBE'S>>
                                      ;*****************************************************************************
                                      ;*SUBTEST       MTO030  FLUSH OUT DBE'S
                                      ;*****************************************************************************
10109 035766  011002                  1$:     MOV     (R0),R2         ;V177640
10110 035770  010220                          MOV     R2,(R0)+        ;V177642
10111 035772  077103                          SOB     R1,1$           ;V177644
10112 035774  000207                          RETURN                  ;V177646
10113
10114 035776                          MTP031: SUBTST  <<MTP031        SOB-A-LONG TEST>>
                                      ;*****************************************************************************
                                      ;*SUBTEST       MTP031  SOB-A-LONG TEST
                                      ;*****************************************************************************
10115                                         .DSABL  AMA
10116 035776  000000                          0                       ;MOVE TERMINATOR
10117 036000  077001                  1$:     SOB     R0,1$           ;SOB TILL R0 UNDERFLOWS
10118 036002  005167  177772                  COM     1$              ;WRITE COMPLEMENT OF SOB
10119 036006  020167  177766                  CMP     R1,1$           ;READ & CHECK FOR NOT 'SOB R0,DOT'
10120 036012  001403                          BEQ     2$              ;OK - SKIP
10121 036014  104454                          PERR30
10122 036016  010167  177756                  MOV     R1,1$
10123 036022  005167  177752          2$:     COM     1$              ;CORRECT SOB INSTRUCTION
10124 036026  010200                          MOV     R2,R0           ;REINITIALIZE SOB CONSTANT
10125                                         ;UPDATE MOVE REGISTERS
10126 036030  010503                          MOV     R5,R3
10127 036032  005725                          TST     (R5)+           ;BUMP (SAFELY) BY 2
10128 036034  010504                          MOV     R5,R4
10129 036036  020537  002474                  CMP     R5,@#LINK1      ;DONE?
10130 036042  001001                          BNE     3$              ;NO - SKIP
10131 036044  000207                          RETURN                  ;YES
10132
10133 036046  014344                  3$:     MOV     -(R3),-(R4)
10134 036050  001376                          BNE     3$
10135 036052  000752                          BR      1$
10136         000056                  SOBLENGTH=.-MTP031
10137                                         .ENABL  AMA
```

```
10165 036054                        MTP032: SUBTST  <<MTP032        WRITE RECOVERY TEST>>
                                    ;*********************************************************************
                                    ;*SUBTEST        MTP032  WRITE RECOVERY TEST
                                    ;*********************************************************************
10166                                       ;THE TEST ACTUALLY EXECUTED ALREADY IN THE MEMORY UNDER TEST.
10167                                       ;THIS CODE INSURES THAT IT CHANGED MEMORY TO HAVE
10168                                       ;1/2 BANK OF #5141      WHICH IS A "COM  (R1)' INSTRUCTION AND
10169                                       ;1/2 BANK OF #110       WHICH IS A  JMP (R0)"   INSTRUCTION.
10170
10171 036054  012401              1$:       MOV     (R4)+,R1        ;V177640        ;GET DATA FROM LOWER 1/2 BANK
10172 036056  020102                        CMP     R1,R2           ;V177642        ;IS IT #5141?
10173 036060  001401                        BEQ     2$              ;V177644        ;YES - SKIP
10174 036062  104430                        PERR02                  ;V177646        ;NO - TAKE ERROR TRAP
10175 036064  077305              2$:       SOB     R3,1$           ;V177650        ;LOOP FOR 1/2 BANK
10176 036066  013703  002474              MOV     @#LINK1,R3      ;V177652        ;RESTORE LOOP SIZE
10177 036072  012400              3$:       MOV     (R4)+,R0        ;V177656        ;GET DATA FROM UPPER 1/2 BANK
10178 036074  020005                        CMP     R0,R5           ;V177660        ;IS IT #110?
10179 036076  001401                        BEQ     4$              ;V177662        ;YES - SKIP
10180 036100  104427                        PERR01                  ;V177664        ;NO- TAKE ERROR TRAP
10181 036102  077305              4$:       SOB     R3,3$           ;V177666        ;LOOP FOR 1/2 BANK
10182 036104  000207                        RETURN
```

```
10185 036106                          MTP033: SUBTST  <<MTP033        BRANCH GOBBLE TEST>>
                                      ;*********************************************************************
                                      ;*SUBTEST        MTP033  BRANCH GOBBLE TEST
                                      ;*********************************************************************
10186                                         .DSABL  AMA
10187 036106  000000                  0                               ;MOVE TERMINATOR
10188 036110  000000                  BGTEST: 0                       ;TEST WORD (TWO BYTES)
10189 036112  000261                  BRGOBB: SEC                     ;SET CARRY (TO BE ADDED TO "BGTEST )
10190 036114  105511                          ADCB    (R1)            ;INCREMENT LOW BYTE OF "BGTEST
10191 036116  100402                          BMI     1$              ;BRANCH WHEN BIT7 IS SET
10192 036120  105212                          INCB    (R2)            ;INCREMENT HIGH BYTE OF  BGTEST
10193 036122  000773                          BR      BRGOBB          ;LOOP 128 TIMES
10194
10195                                         ;NOW CHECK FOR CORRECT CONDITION CODES
10196 036124  102401                  1$:     BVS     2$              ;BR IF V-BIT SET (SHOULD BE)
10197 036126  104461                          PERR35                  ;NO   REPORT ERROR AND ABORT TEST
10198                                                                 ;COND CODES NOT EQUAL TO 1010
10199 036130  000242                  2$:     CLV                     ;CLEAR V-BIT
10200 036132  105212                          INCB    (R2)            ;INCREMENT HIGH BYTE OF "BGTEST' ONCE MORE
10201 036134  103402                          BCS     3$              ;BR IF C BIT SET (SHOULD NOT BE)
10202 036136  102001                          BVC     3$              ;BR IF V BIT CLEAR (SHOULD NOT BE)
10203 036140  100401                          BMI     4$              ;BR IF N-BIT SET (SHOULD BE)
10204 036142  104461                  3$:     PERR35                  ;NO   REPORT ERROR AND ABORT TEST
10205                                                                 ;COND CODES NOT EQUAL TO 1010
10206
10207                                         ;UPDATE TEST POINTERS
10208 036144  010701                  4$:     MOV     PC,R1
10209 036146  162701  000036          5$:     SUB     #5$-BGTEST,R1
10210 036152  010102                          MOV     R1,R2
10211 036154  005202                          INC     R2
10212
10213                                         ;UPDATE MOVE REGISTERS
10214 036156  010503                          MOV     R5,R3
10215 036160  005725                          TST     (R5)+           ;BUMP (SAFELY) BY 2
10216 036162  010504                          MOV     R5,R4
10217
10218                                         ;DONE?
10219 036164  020537  002474                  CMP     R5,@#LINK1      ;DONE?
10220 036170  001001                          BNE     6$              ;NO   SKIP
10221 036172  000207                          RETURN                  ;YES   RETURN
10222
10223                                         ;MOVE CODE 1 LOCATION
10224 036174  014344                  6$:     MOV     -(R3), (R4)
10225 036176  001376                          BNE     6$
10226 036200  005011                          CLR     (R1)            ;CLEAR TEST WORD "BGTEST'
10227 036202  000743                          BR      BRGOBB          ;RUN MOVED CODE AGAIN
10228         000076                  GBLENGTH=. MTP033
10229                                         .ENABL  AMA
```

```
10231 036204                    MTP034: SUBTST  <<MTP034        SOFT ERROR   BACKROUND PATTERN TEST>>
                                ;*****************************************************************
                                ;*SUBTEST        MTP034  SOFT ERROR   BACKROUND PATTERN TEST
                                ;*****************************************************************
10232 036204  010220            1$:     MOV     R2,(R0)+        ;V177640
10233 036206  077102                    SOB     R1,MTP034       ;V177642
10234 036210  000207                    RETURN                  ;V177644
10235 036212  012401            2$:     MOV     (R4)+,R1        ;V177646
10236 036214  020102                    CMP     R1,R2           ;V177650
10237 036216  001402                    BEQ     3$              ;V177652
10238 036220  104430                    PERR02                  ;V177654
10239 036222  000240                    NOP                     ;V177656
10240 036224  077306            3$:     SOB     R3,2$           ;V177660
10241 036226  000207                    RETURN                  ;V177662
```

```
10245 036230                                    MTP035:SUBTST   <<MTP035        WORST CASE NOISE PARITY TEST>>
                                                ;*****************************************************************
                                                ;*SUBTEST        MTP035  WORST CASE NOISE PARITY TEST
                                                ;*****************************************************************
10244 036230  012737  000003  002074                  MOV     #3,NOPAR        ;SET PARITY TRAPS TO RETURN TO  PARTHERE
10245
10246 036236                                          FOR R0 := #FIRST TO #LAST BY #4000
10247 036242  012737  000005  002144                    MOV   #BIT2!BIT0,CSR  ;SET WRITE WRONG PARITY & PAR. TRAPS INTO CSR
10248 036250  104425                                    LOADCSR
10249 036252  012737  036306  002266                    MOV   #1$,PARTHERE
10250 036260  011010                                    MOV   (R0),(R0)       ;WMP TEST LOCATION
10251 036262  005710                                    TST   (R0)
10252 036264  010037  002032                            MOV   R0,ADDRESS
10253 036270  104050                                    ERROR .50
10254 036272  004737  054754                            CALL  PERBNK
10255 036276  032763  002000  002630                    BIT   #BIT10,CONFIG+2(R3)
10256 036304  001002                                    BNE   2$
10257 036306  104426                              1$:   READCSR
10258 036310  104512                                    ERRGEN
10259
10260 036312  104503                              2$:   CLR1CSR
10261 036314  011010                                    MOV   (R0),(R0)       ;CLEAR WRONG PARITY IN MEMORY
10262 036316  012737  000001  002144                    MOV   #BIT0,CSR
10263 036324  104425                                    LOADCSR
10264 036326  012737  036340  002266                    MOV   #3$,PARTHERE
10265 036334  005710                                    TST   (R0)
10266 036336  000405                                    BR    4$
10267 036340  010037  002032                      3$:   MOV   R0,ADDRESS
10268 036344  104050                                    ERROR .50
10269 036346  004737  054754                            CALL  PERBNK
10270 036352                                      4$:   END; OF FOR
10271
10272 036364  005037  002074                        CLR     NOPAR           ;RESET PARITY TRAP ACTION
10273 036370  000207                                  RETURN
```

```
10275                                           .SBTTL  MISC SUBROUTINES
10276
10277 036372                                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
                                            ;*****************************************************************
10278 036372  010004                            MOV     R0,R4
10279 036374  010103                            MOV     R1,R3
10280 036376  010205                            MOV     R2,R5
10281 036400  000207                            RETURN
10282
10283 036402                                FLIPWARN:SUBTST <<FLIP WARNING CONSTANTS IN WORST CASE NOISE TESTS>>
                                            ;*****************************************************************
                                            ;*SUBTEST        FLIP WARNING CONSTANTS IN WORST CASE NOISE TESTS
                                            ;*****************************************************************
10284 036402                                    PUSH    R0
10285 036404  005237  002560                      INC     FLIPLOC
10286 036410  042737  177774  002560              BIC     #^C3,FLIPLOC
10287 036416  022737  000001  002560              CMP     #1,FLIPLOC
10288 036424  001414                              BEQ     1$
10289 036426  022737  000002  002560              CMP     #2,FLIPLOC
10290 036434  001413                              BEQ     2$
10291 036436  022737  000003  002560              CMP     #3,FLIPLOC
10292 036444  001414                              BEQ     3$
10293 036446  005000                              CLR     R0
10294 036450  013704  002556                      MOV     ONES,R4
10295 036454  000414                              BR      4$
10296 036456                                1$:     CLEAR   R0,R4
10297 036462  000411                              BR      4$
10298 036464  012700  000401                2$:     MOV     #401,R0
10299 036470  013704  002556                      MOV     ONES,R4
10300 036474  000404                              BR      4$
10301 036476  012700  000401                3$:     MOV     #401,R0
10302 036502  012704  000401                        MOV     #401,R4
10303 036506  010037  027160                4$:     MOV     R0,WARN2
10304 036512  010037  027174                        MOV     R0,WARN3
10305 036516  010037  027220                        MOV     R0,WARN4
10306 036522  010037  027234                        MOV     R0,WARN5
10307 036526                                    POP     R0
10308 036530  000207                              RETURN
```

```
10310 036532                                BACKGND:SUBTST  <<SUBR  WRITE BACKGROUND>>
                                            ;*****************************************************************
                                            ;*SUBTEST        SUBR    WRITE BACKGROUND
                                            ;*****************************************************************
10311                                               ;WRITES DATA FROM R2
10312 036532  104415                                SAVREG
10313 036534  012700  060000                        MOV     #FIRST,R0
10314 036540  012701  040000                        MOV     #SIZE,R1
10315 036544  022737  000001  003720                CMP     #1,PROTYP
10316 036552  001415                                BEQ     WARN6B
10317 036554  012737  000207  027042  WARN6A:       MOV     #207,MTP000+4           ;WARNING PUTTING "RETURN" AFTER WRITE
10318 036562  012737  027036  002256                MOV     #MTP000,SUPDOADD
10319 036570  004737  026644                        CALL    SUPD03
10320 036574  012737  000240  027042                MOV     #240,MTP000+4           ;RESTORE "NOP" AFTER WRITE
10321 036602  104416                                RESREG
10322 036604  000207                                RETURN
10323 036606                                WARN6B: BMOV    MTP000
10324 036614  012737  000207  177644  WARN6:      MOV     #207,UIPAR2             ;WARNING PUTTING  RETURN" INSTRUCTION AFTER WRITE
10325 036622  004737  026466                        CALL    SUPD01
10326 036626  104416                                RESREG
10327 036630  000207                                RETURN
```

```
10330 036632                                    PCONFIG:SUBTST  <<SUBR  PRINT CONFIGURATION MAP>>
                                                ;*****************************************************************
                                                ;*SUBTEST        SUBR    PRINT CONFIGURATION MAP
                                                ;*****************************************************************
10331 036632                                            PUSH    TKVEC,TKVEC+2,R0
10332 036644  010637  037132                            MOV     SP,PCONFS               ;SAVE LAST GOOD SP
10333 036650  012737  037100  000060                     MOV     #PCONF2,TKVEC
10334 036656  012737  000340  000062                     MOV     #340,TKVEC+2
10335 036664  017700  143716                            MOV     @#TKB,R0                ;KILL ANY OLD INTERRUPT
10336 036670  042737  000200  177776                     BIC     #BIT7,PSW               ;LOWER CPU PRIORITY TO 140
10337 036676  052777  000100  143700                     BIS     #BIT6,@#TKS             ;ENABLE KEYBOARD INTERRUPTS
10338
10339 036704                                            TYPE    MSG001
10340 036710                                            TYPE    MSG002
10341 036714                                            TYPE    MSG003
10342 036720  022737  000060  002530                     CMP     #60,LASTBANK
10343 036726  002006                                    BGE     NOOJ
10344                                                   ;IF FAT PAPER ON TERMINAL GOTO 1$
10345 036730                                            IF #SW4 SET.IN @SWR THEN JUMPTO PCONF1
10346 036744  012700  000074                    NOOJ:   MOV     #60..R0
10347 036750  010004                                    MOV     R0,R4
10348 036752                                            CLEAR   R1,R3
10349 036756                                            TYPE    MSG004
10350 036762  004737  037134                            CALL    TCONFIG                 ;GO TYPE CONFIGURATION (1ST HALF)
10351 036766  022737  000060  002530                     CMP     #60,LASTBANK
10352 036774  002041                                    BGE     PCONF2
10353 036776                                            TYPE    $CRLF
10354 037002                                            TYPE    MSG017                  ;PRINT SPACE(S)
10355 037006                                            TYPE    MSG011
10356 037012                                            TYPE    $CRLF
10357 037016                                            TYPE    MSG017                  ;PRINT SPACE(S)
10358 037022                                            TYPE    MSG012
10359 037026  012701  000360                            MOV     #60.*2*2,R1
10360 037032  010103                                    MOV     R1,R3
10361 037034  004737  037134                            CALL    TCONFIG
10362 037040  000417                                    BR      PCONF2
10363
10364 037042  012700  000170                    PCONF1: MOV     #120..R0
10365 037046  010004                                    MOV     R0,R4
10366 037050                                            CLEAR   R1,R3
10367 037054                                            TYPE    MSG014                  ;SPACE
10368 037060                                            TYPE    MSG011
10369 037064                                            TYPE    MSG004
10370 037070                                            TYPE    MSG012
10371 037074  004737  037134                            CALL    TCONFIG
10372
10373 037100  013706  037132                    PCONF2: MOV     PCONFS,SP               ;RESTORE STACK
10374 037104  042777  000100  143472                     BIC     #BIT6,@#TKS
10375 037112  117700  143470                            MOVB    @#TKB,R0                ;READ CHAR TO KILL FLAG
10376 037116                                            POP     R0,TKVEC+2,TKVEC
10377 037130  000207                                    RETURN
10378
10379 037132  000000                    PCONFS: 0                                       ;STACK SAVED HERE!
```

```
                                        SUBTST  <<SUBR  TYPE CONFIGURATION>>
                                ;*****************************************************************
                                ;*SUBTEST        SUBR    TYPE CONFIGURATION
                                ;*****************************************************************
                                ;*****************************************************************
                                ;CALL:  MOV     #N,R0           ;N=NUMBER OF CHARACTERS
                                ;       MOV     R0,R4           ;BACKUP
                                ;       MOV     #K,R1           ;INDEX CONSTANT
                                ;       MOV     R1,R3           ;BACKUP
                                ;       CALL    TCONFIG         ;ACTUAL CALL
                                ;       RETURN                  ;ONLY RETURN
                                ;*****************************************************************

                                        ;***********
                                        ;** ERROR **
                                        ;***********
037134                          TCONFIG:TYPE    MSG005
037140  032761  000001  002626  1$:     BIT     #BIT0,CONFIG(R1)        ;ERROR ON THIS BANK?
037146  001403                          BEQ     2$                      ;NO   SKIP
037150                                  TYPE    MSG013                  ;PRINT "X"
037154  000402                          BR      3$
037156                          2$:     TYPE    MSG014                  ;PRINT SPACE
037162  062701  000004          3$:     ADD     #4,R1                   ;BUMP POINTER
037166  077014                          SOB     R0,1$                   ;LOOP TILL DONE
037170  010400                          MOV     R4,R0
037172  010301                          MOV     R3,R1

                                        ;***********
                                        ;** CPU S **
                                        ;***********
037174                                  TYPE    MSG008
037200  016105  002626          4$:     MOV     CONFIG(R1),R5
037204  006205                          ASR     R5                      ;GET CPU BITS
037206  042705  177760                  BIC     #^C17,R5                ;CLEAR NON INTERESTING BITS
037212  005705                          TST     R5                      ;IS THERE ANYTHING THERE?
037214  001003                          BNE     8$                      ;YES   BRANCH.
037216  112705  000040                  MOVB    #' ,R5                  ;NO   MOVE A BLANK INTO R5
037222  000402                          BR      9$                      ;BRANCH OVER NEXT INSTRUCTION
037224  062705  000060          8$:     ADD     #60,R5                  ;MAKE ASCII
037230  110537  071644          9$:     MOVB    R5,MSG015               ;PLUG INTO MEMORY
037234                                  TYPE    MSG015
037240  062701  000004                  ADD     #4,R1                   ;BUMP POINTER
037244  077023                          SOB     R0,4$                   ;LOOP TILL DONE
037246  010400                          MOV     R4,R0
037250  010301                          MOV     R3,R1
```

```
10431                                                   ;****************
10432                                                   ;** INTERLEAVE **
10433                                                   ;****************
10434 037252                                            TYPE    MSG007
10435                                                   ;THIS IS AN ENTRY POINT FROM ERROR REPORTS
10436 037256  032761  010000  002630  TCFIG1: BIT     @#BIT12,CONFIG+2(R1)
10437 037264  001014                        BNE     1$
10438 037266  032761  000002  002626        BIT     @#BIT1,CONFIG(R1)       ;IS THERE ANY MEMORY HERE?
10439 037274  001004                        BNE     18$                     ;BRANCH IF MEMORY PRESENT.
10440 037276  112737  000040  071644        MOVB    @' ,MSG015              ;MOVE A BLANK IN TO BE PRINTED
10441 037304  000424                        BR      16$                     ;BRANCH TO TYPE ROUTINE
10442 037306  112737  000055  071644  18$:   MOVB    @-,MSG015
10443 037314  000420                        BR      16$
10444 037316  016105  002626        1$:     MOV     CONFIG(R1),R5
10445 037322  042705  007777                BIC     @#C170000,R5            ;GET CSR INTERLEAVE
10446 037326  000305                        SWAB    R5
10447 037330  072527  177774                ASH     @-4,R5
10448 037334  022705  000012                CMP     @10.,R5
10449 037340  100002                        BPL     2$
10450 037342  062705  000007                ADD     @7,R5
10451 037346  062705  000060        2$:     ADD     @60,R5                  ;MAKE ASCII
10452 037352  110537  071644                MOVB    R5,MSG015               ;PLUG INTO MEMORY
10453 037356                        16$:    TYPE    MSG015
10454 037362                                IF NOTAB NE @0 THEN $RETURN
10455 037372  062701  000004                ADD     @4,R1                   ;BUMP POINTER
10456 037376  077051                        SOB     R0,TCFIG1               ;LOOP TILL DONE
10457 037400  010400                        MOV     R4,R0
10458 037402  010301                        MOV     R3,R1
10459
10460                                                   ;*****************
10461                                                   ;** MEMORY TYPE **
10462                                                   ;*****************
10463                                                   .ENABL  LSB
10464 037404                                            TYPE    MSG009
10465 037410  033761  002104  002626  TCFIG2: BIT     CPUBIT,CONFIG(R1)
10466 037416  001447                        BEQ     17$
10467 037420  016105  002630                MOV     CONFIG+2(R1),R5
10468 037424  000305                        SWAB    R5                      ;GET MEMORY TYPE
10469 037426  042705  177770                BIC     @#C7,R5                 ;CLEAR NON INTERESTING BITS
10470 037432  005705                        TST     R5
10471 037434  001440                        BEQ     17$
10472 037436  032705  000004                BIT     @#BIT2,R5
10473 037442  001004                        BNE     4$
10474 037444  112737  000102  071644        MOVB    @'B,MSG015
10475 037452  000434                        BR      8$
10476 037454  032705  000002        4$:     BIT     @#BIT1,R5
10477 037460  001013                        BNE     6$
10478 037462  032705  000001                BIT     @#BIT0,R5
10479 037466  001004                        BNE     5$
10480 037470  112737  000115  071644        MOVB    @'M,MSG015
10481 037476  000422                        BR      8$
10482 037500  112737  000113  071644  5$:   MOVB    @ K,MSG015
10483 037506  000416                        BR      8$
10484 037510  032705  000001        6$      BIT     @#BIT0,R5
10485 037514  001004                        BNE     7$
10486 037516  112737  000114  071644        MOVB    @ L,MSG015
10487 037524  000407                        BR      8$
```

```
10488 037526  112737  000120  071644  8$:     MOVB    #'P,MSG015
10489 037534  000403                          BR      8$
10490 037536  112737  000040  071644  10$:    MOVB    #' ,MSG015
10491 037544                          8$:     TYPE    MSG015
10492 037550                                  IF NOTAB NE #0 THEN $RETURN
10493 037560  062701  000004                  ADD     #4,R1                   ;BUMP POINTER
10494 037564  077067                          SOB     R0,TCFIG2               ;LOOP TILL DONE
10495 037566  010400                          MOV     R4,R0
10496 037570  010301                          MOV     R3,R1
10497                                         .DSABL  LSB
10498
10499                                         ;*********
10500                                         ;** CSR **
10501                                         ;*********
10502 037572                                   TYPE    MSG016
10503 037576  112737  000040  071644  TCFIG3: MOVB    #' ,MSG015
10504 037604  016105  002626                  MOV     CONFIG(R1),R5
10505 037610  032705  000002                  BIT     #BIT1,R5
10506 037614  001414                          BEQ     16$
10507 037616  042705  170377                  BIC     #^C7400,R5
10508 037622  000305                          SWAB    R5
10509 037624  022705  000012                  CMP     #10.,R5
10510 037630  100002                          BPL     10$
10511 037632  062705  000007                  ADD     #7,R5
10512 037636  062705  000060          10$:    ADD     #60,R5                  ;MAKE ASCII
10513 037642  110537  071644                  MOVB    R5,MSG015               ;PLUG INTO MEMORY
10514 037646                          16$:    TYPE    MSG015
10515 037652                                  IF NOTAB NE #0 THEN $RETURN
10516 037662  062701  000004                  ADD     #4,R1                   ;BUMP POINTER
10517 037666  077035                          SOB     R0,TCFIG3               ;LOOP TILL DONE
10518 037670  010400                          MOV     R4,R0
10519 037672  010301                          MOV     R3,R1
10520
10521                                         ;************ ***
10522                                         ;** PROTECTED **
10523                                         ;***************
10524 037674                                   TYPE    MSG010
10525 037700  105761  002626          11$:    TSTB    CONFIG(R1)              ;BANK PROTECTED?
10526 037704  100004                          BPL     12$                     ;NO   SKIP
10527 037706  112737  000120  071644          MOVB    #'P,MSG015
10528 037714  000407                          BR      13$
10529 037716  032761  000100  002626  12$:    BIT     #BIT6,CONFIG(R1)        ;PROTECTED REGION OF ECC?
10530 037724  001406                          BEQ     14$                     ;NO - SKIP
10531 037726  112737  000111  071644          MOVB    #'I,MSG015
10532 037734                          13$:    TYPE    MSG015
10533 037740  000402                          BR      15$
10534 037742                          14$:    TYPE    MSG014                  ;PRINT SPACE
10535 037746  062701  000004          15$:    ADD     #4,R1                   ;BUMP POINTER
10536 037752  077026                          SOB     R0,11$                  ;LOOP TILL DONE
10537 037754  010400                          MOV     R4,R0
10538 037756  010301                          MOV     R3,R1
10539 037760  000207                          RETURN
```

```
10542                                           .SBTTL  TRAP    PARITY ERROR HANDLER
10543                                   ;*******************************************************************
10544                                   ;VECTOR TO HERE FROM TRAPS TO 114
10545                                   ;IGNORE ERRORS BUT COUNT IF NOPAR FLAG = 1.
10546                                   ;*******************************************************************
10547                                   ;
10548                                   ;       CODE    ACTION
10549                                   ;
10550                                   ;         0     PRINT UNEXPECTED PARITY TRAP
10551                                   ;         1     COUNT ERROR
10552                                   ;         2     SET "ABORT" / SETUP  BADPC" / RETURN VIA PCBUMP
10553                                   ;         3     RETURN VIA  PARTHERE
10554                                   ;
10555 037762  022737  000001  002074  PARITY: CMP     #1,NOPAR                ;COUNTING PARITY ERRORS?
10556 037770  001003                          BNE     1$                      ;NO   SKIP
10557 037772  005237  002070                  INC     PARCNT                  ;PARITY ERROR COUNTER = 1
10558 037776  000002                          RTI
10559 040000  022737  000002  002074  1$:     CMP     #2,NOPAR                ;ACTION CODE = 2 ?
10560 040006  001013                          BNE     2$                      ;NO   SKIP
10561 040010                                  SET     ABORTFLAG               ;YES
10562 040016  004737  040170                  CALL    BADSTACK                ;FIND BAD SP,PC,PSW OFF STACK
10563 040022  063716  002300                  ADD     PCBUMP,(SP)             ;UPDATE RETURN PC
10564 040026  042766  000004  000002          BIC     #BIT2,2(SP)             ;SHOW FAILURE BY .NE.
10565 040034  000002                          RTI
10566 040036  022737  000003  002074  2$:     CMP     #3,NOPAR                ;ACTION CODE = 3 ?
10567 040044  001003                          BNE     3$                      ;NO   SKIP
10568 040046  013716  002266                  MOV     PARTHERE,(SP)
10569 040052  000002                          RTI
10570 040054  004737  040170          3$:     CALL    BADSTACK                ;FIND BAD SP,PC,PSW OFF STACK
10571 040060                                  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
                                ;*******************************************************************

040066  022737  000001  002076  NONEXIST:CMP     #1,NONEM                ;COUNTING NON-EXISTANT MEMORY ERRORS?
040074  001011                          BNE     2$                      ;NO - SKIP
040076  005237  002066                  INC     NEMCNT                  ;BUMP NON-EXISTANT MEMORY COUNTER
040102  022737  000001  002066          CMP     #1,NEMCNT               ;FIRST ERROR?
040110  001002                          BNE     1$                      ;NO - SKIP
040112  010037  002032                  MOV     R0,ADDRESS              ;ASSUME R0 CONTAINS THE ADDRESS ACCESSED
040116  000002                  1$:     RTI
040120  005237  002066          2$:     INC     NEMCNT                  ;BUMP NON-EXISTANT MEMORY COUNTER
040124  012701  000001                  MOV     #1,R1                   ;DUMMY UP R1 FOR A FORCED SOB EXIT
040130  000002                          RTI

                                ;*******************************************************************
                                        .SBTTL  TRAP    TIMEOUT (TRAP TO 4) HANDLER
040132  004737  040170          TIMEOUT:CALL    BADSTACK                ;FIND BAD SP,PC,PSW OFF STACK
040136                                  FATAL   6
                                ;*******************************************************************
                                        .SBTTL  TRAP    MEMORY MANAGEMENT (TRAP TO 250) HANDLER
040144  004737  040170          MMTRAP: CALL    BADSTACK                ;FIND BAD SP,PC,PSW OFF STACK
040150                                  FATAL   7
                                        .SBTTL  TRAP    RESERVED INSTRUCTION HANDLER
040156  004737  040170          PDP1105:CALL    BADSTACK                ;FIND BAD SP,PC,PSW OFF STACK
040162                                  FATAL   5


040170                          BADSTACK:SUBTST <<FIND BAD SP, PC, & PSW FROM STACK>>
                                ;*******************************************************************
                                ;*SUBTEST       FIND BAD SP, PC, & PSW FROM STACK
                                ;*******************************************************************
040170  010637  002024                  MOV     SP,BADSP
040174  062737  000002  002024          ADD     #2,BADSP
040202  016637  000002  002020          MOV     2(SP),BADPC
040210  016637  000004  002030          MOV     4(SP),BADPSW
040216  000207                          RETURN
```

```
10619                                           .SBTTL  TRAP    KERNEL TRAP HANDLER
10620                                   ;*******************************************************************************
10621                                   ;KERNEL IS A TRAP THAT COMES HERE
10622                                   ;*******************************************************************************
10623
10624 040220  042766  140000  000002  $KERNEL:        BIC     #140000,2(SP)
10625 040226  000002                          RTI
10626                                   ;*******************************************************************************
10627                                           .SBTTL  TRAP    ENERGIZE TRAP HANDLER
10628 040230  052737  000001  177572  $ENERGIZE:BIS   #BIT0,MMR0
10629 040236  000002                          RTI
10630                                   ;*******************************************************************************
10631                                           .SBTTL  TRAP    DEENERGIZE TRAP HANDLER
10632 040240  042737  000001  177572  $DEENERGIZE:BIC #BIT0,MMR0
10633 040246  000002                          RTI
10634                                   ;*******************************************************************************
10635                                           .SBTTL  TRAP    CACHON TRAP HANDLER
10636 040250  005737  002516          $CACHN: TST     CACHKN          ;IS THERE A CACHE
10637 040254  001406                          BEQ     1$              ;NO - RETURN
10638 040256  013737  002516  177746          MOV     CACHKN,CONTRL   ;SETUP CACHE AS PER CONSTANT (USUALLY 1 = FULLY ON)
10639 040264  052737  000001  177746          BIS     #BIT0,CONTRL    ;DISABLE TRAPS (BUT NOT ABORTS)
10640 040272  000002                  1$:     RTI
10641                                   ;*******************************************************************************
10642                                           .SBTTL  TRAP    CACHOFF TRAP HANDLER
10643 040274  005737  002516          $CACHF: TST     CACHKN          ;IS THERE A CACHE?
10644 040300  001403                          BEQ     1$              ;NO - RETURN
10645                                           ;DISABLE TRAPS (NOT ABORTS), FORCE MISSES, FLUSH, BYPASS
10646 040302  053737  002522  177746          BIS     CACHKF,CONTRL
10647 040310  000002                  1$:     RTI
```

```
10650                                                   .SBTTL  TRAP    LOAD CSR TRAP HANDLER
10651                                                   ;LOAD CORRECT CSR WITH DATA IN CSR
10652                                                   ;PROGRAM CSR S ASSERT INHIBIT MODE POINTER WHEN LOADED
10653 040312                                  $LOADC:   PUSH    R0,R1                       ;SAVE REGISTERS
10654 040316  013700  002146                            MOV     CSRNO,R0                    ;CREATE CSR ADDRESS
10655 040322                                            IF INHECC IS TRUE THEN GOTO 3$      ;DON'T WANT INH. MODE POINTER ON
10656 040330  005737  002504                            TST     PGMCSR                      ;PROGRAM IN INTERLEAVED SPACE?
10657 040334  100007                                    BPL     1$                          ;BRANCH IF NOT
10658 040336  113701  002505                            MOVB    PGMCSR+1,R1                 ;CHECK SECOND CSR
10659 040342  042701  177740                            BIC     #^C37,R1                    ;CLEAR UNNECESSARY BITS
10660 040346  020137  002146                            CMP     R1,CSRNO                    ;IS THIS THE CURRENT CSR?
10661 040352  001404                                    BEQ     2$                          ;BRANCH IF IT IS
10662 040354  123737  002504  002146  1$:               CMPB    PGMCSR,CSRNO                ;IS THIS THE CURRENT CSR?
10663 040362  001003                                    BNE     3$                          ;BRANCH IF NOT
10664 040364  052737  020000  002144  2$:               BIS     #BIT13,CSR                  ;SET THE INHIBIT MODE POINTER TO  1ST 16K
10665 040372  013760  002144  172100  3$:               MOV     CSR,CSRADD(R0)              ;LOAD THE CSR
10666 040400                                            POP     R1,R0                       ;RESTORE REGISTERS
10667 040404  000002                                    RTI
10668
10669                                                   .SBTTL  TRAP    READ CSR TRAP HANDLER
10670                                                   ;READ THE CORRECT CSR INTO LOCATIONS CSR
10671 040406                                  $READC:   PUSH    R0
10672 040410  013700  002146                            MOV     CSRNO,R0
10673 040414  016037  172100  002144                    MOV     CSRADD(R0),CSR              ;READ IT
10674 040422                                            POP     R0
10675 040424  000002                                    RTI
```

```
10677                                                   .SBTTL  TRAP    TEST (R1) & READ CSR CAREFULLY
10678 040426                                    $TSTRD: PUSH    R0,R2,R3
10679 040434  012700  172100                            MOV     #CSRADD,R0              ;CREATE CSR ADDRES
10680 040440  063700  002146                            ADD     CSRNO,R0
10681 040444  005002                                    CLR     R2
10682 040446  005737  002504                            TST     PGMCSR
10683 040452  100007                                    BPL     1$
10684 040454  113703  002505                            MOVB    PGMCSR+1,R3
10685 040460  042703  000200                            BIC     #BIT7,R3
10686 040464  020337  002146                            CMP     R3,CSRNO
10687 040470  001404                                    BEQ     2$
10688 040472  123737  002504  002146  1$:       CMPB    PGMCSR,CSRNO
10689 040500  001002                                    BNE     3$
10690 040502  012702  020000            2$:     MOV     #BIT13,R2
10691 040506  022737  000001  003720  3$:       CMP     #1,PROTYP       ;IS THIS AN 11/44?
10692 040514  001403                                    BEQ     4$              ;BRANCH IF IT IS
10693 040516  004737  040604                            CALL    TSTRD1
10694 040522  000405                                    BR      5$
10695 040524                                    4$:     BMOV    TSTRD1
10696 040532  004737  177640                            CALL    FASTCITY                ;CALL TO THE USER INSTRUCTION PAR S
10697                                                   ;IF SINGLE BIT ERROR ONLY   SET CARRY BIT
10698 040536                                    5$:     POP     R3,R2,R0
10699 040544                                            IF #BIT4 SET.IN CSR AND #BIT15 OFF.IN CSR
10700 040564  052766  000001  000002                      BIS   #BIT0,2(SP)
10701 040572                                            ELSE
10702 040574  042766  000001  000002                      BIC   #BIT0,2(SP)
10703 040602                                            END ;OF IF #BIT4
10704 040602  000002                                    RTI
10705
10706 040604  010210                            TSTRD1: MOV     R2,(R0)                 ;V177640
10707 040606                                            TESTAREA                        ;V177642 ;ENTER SUPERVISOR MODE
10708 040614  105711                                    TSTB    (R1)                    ;V177646
10709 040616  042737  140000  177776                    BIC     #BIT15!BIT14,PSW        ;V177650
10710 040624  011037  002144                            MOV     (R0),CSR                ;V177656
10711 040630  000207                                    RETURN                          ;V177662
```

```
10714                                                   .SBTTL  TRAP    ECC DISABLE ALL CSR S TRAP HANDLER
10715 040632  012737  000002  002144  $ECCDIS:MOV       #BIT1,CSR
10716 040640  004737  041356                  CALL      CSROUT
10717 040644  000002                          RTI
10718                                                   .SBTTL  TRAP    ECC DISABLE OF 1 SELECTED CSR TRAP HANDLER
10719 040646  012737  000002  002144  $ECC1DIS:MOV      #BIT1,CSR
10720 040654  104425                  LOADCSR
10721 040656  000002                          RTI
10722                                                   .SBTTL  TRAP    INITIALIZE ALL CSR S TRAP HANDLER
10723 040660  012737  000001  002144  $ECCINIT:MOV      #BIT0,CSR
10724 040666  004737  041356                  CALL      CSROUT
10725 040672  000002                          RTI
10726                                                   .SBTTL  TRAP    INITIALIZE 1 SELECTED CSR TRAP HANDLER
10727 040674  012737  000001  002144  $ECC1INIT:MOV     #BIT0,CSR
10728 040702  104425                  LOADCSR
10729 040704  000002                          RTI
10730                                                   .SBTTL  TRAP    ENABLE SBE PARITY TRAPS ON ALL CSR'S
10731 040706  012737  000003  002144  $ENASBE:MOV       #BIT0!BIT1,CSR
10732 040714  004737  041356                  CALL      CSROUT
10733 040720  000002                          RTI
10734                                                   .SBTTL  TRAP    ENABLE SBE PARITY TRAPS ON 1 SELECTED CSR
10735 040722  012737  000003  002144  $ENA1SBE:MOV      #BIT0!BIT1,CSR
10736 040730  104425                  LOADCSR
10737 040732  000002                          RTI
10738                                                   .SBTTL  TRAP    WRITE CHECKBITS THRU ALL CSR'S TRAP HANDLER
10739 040734  013737  002276  002144  $CBCSR:  MOV      CHECK,CSR               ;BITS 11-5
10740 040742  052737  000006  002144           BIS      #BIT1!BIT2,CSR          ;CHECK MODE
10741 040750  004737  041356                  CALL CSROUT
10742 040754  000002                          RTI
10743                                                   .SBTTL  TRAP    WRITE CHECKBITS THRU 1 SELECTED CSR TRAP HANDLER
10744 040756  013737  002276  002144  $CB1CSR:MOV       CHECK,CSR               ;BITS 11-5
10745 040764  052737  000006  002144           BIS      #BIT1!BIT2,CSR          ;CHECK MODE
10746 040772  104425                  LOADCSR
10747 040774  000002                          RTI
```

```
10750                                                   .SBTTL  TRAP    WAS THERE A SBE ON ANY CSR TRAP HANDLER
10751 040776                                    $WASSBE:PUSH    R1,R4
10752 041002  013701  002220                              MOV     TOTCSRS,R1      ;GET CSR'S BYTE
10753 041006  005004                                      CLR     R4
10754 041010                                              BEGIN LWSBE
10755 041010                                                FOR CSRNO := #0 TO #36 BY #2
10756 041014  006301                                          ASL       R1
10757 041016                                                  ON.ERROR
10758 041020  104426                                            READCSR
10759 041022                                                    IF #BIT4 SET IN CSR
10760 041032                                                      SET     R4
10761 041036                                                      LEAVE LWSBE
10762 041040                                                    END ;OF IF #BIT4
10763 041040                                                  END ;OF ON.ERROR
10764 041040                                                  IF R1 EQ #0 THEN LEAVE LWSBE
10765 041044                                                FI ;OF FOR CSRNO
10766 041062                                              END LWSBE
10767 041062  006004                                      ROR     R4              ;SET C BIT FOR ERROR
10768 041064                                              POP     R4,R1
10769 041070                                              ON.ERROR
10770 041072  052766  000001  000002                        BIS   #BIT0,2(SP)
10771 041100                                              ELSE
10772 041102  042766  000001  000002                        BIC   #BIT0,2(SP)
10773 041110                                              END ;OF ON.ERROR
10774 041110  000002                                      RTI
10775                                                   .SBTTL  TRAP    WAS THERE A SBE IN 1 SELECTED CSR TRAP HANDLER
10776                                                   ;ON RETURN IF CARRY IS SET THERE WAS A SBE
10777 041112  104426                            $WAS1SBE:READCSR
10778 041114  042766  000001  000002                    BIC     #BIT0,2(SP)     ;CLR C BIT ON STACK
10779 041122  032737  000020  002144                    BIT     #BIT4,CSR
10780 041130  001403                                    BEQ     1$
10781 041132  052766  000001  000002                    BIS     #BIT0,2(SP)     ;SET C BIT ON STACK
10782 041140  000002                            1$:     RTI
```

```
10785                                           .SBTTL  TRAP    WAS THERE A DBE ON ANY CSR TRAP HANDLER
10786 041142                            $WASDBE:PUSH    R1,R4
10787 041146  013701  002220                      MOV     TOTCSRS,R1      ;GET CSR'S BYTE
10788 041152  005004                              CLR     R4
10789 041154                                      BEGIN LWDBE
10790 041154                                        FOR CSRNO := #0 TO #36 BY #2
10791 041160  006301                                  ASL       R1
10792 041162                                          ON.ERROR
10793 041164  104426                                    READCSR
10794 041166                                            IF #BIT15 SET.IN CSR
10795 041176                                              SET     R4
10796 041202                                              LEAVE LWDBE
10797 041204                                            END ;OF IF #BIT4
10798 041204                                          END ;OF ON.ERROR
10799 041204                                        IF R1 EQ #0 THEN LEAVE LWDBE
10800 041210                                        END ;OF FOR CSRNO
10801 041226                                      END LWDBE
10802 041226  006004                              ROR     R4              ;SET C BIT FOR ERROR
10803 041230                                      POP     R4,R1
10804 041234                                      ON.ERROR
10805 041236  052766  000001  000002                BIS   #BIT0,2(SP)
10806 041244                                      ELSE
10807 041246  042766  000001  000002                BIC   #BIT0,2(SP)
10808 041254                                      END ;OF ON.ERROR
10809 041254  000002                              RTI
10810                                           .SBTTL  TRAP    WAS THERE A DBE ON 1 SELECTED CSR TRAP HANDLER
10811                                           ;ON RETURN IF CARRY IS SET THERE WAS A DBE
10812 041256  104426                    $WAS1DBE:READCSR
10813 041260  005737  002144                      TST     CSR             ;DBE?
10814 041264  100004                              BPL     3$              ;NO   SKIP
10815 041266  052766  000001  000002              BIS     #BIT0,2(SP)     ;SET C BIT ON STACK
10816 041274  000002                              RTI
10817 041276  042766  000001  000002  3$:         BIC     #BIT0,2(SP)     ;CLR C BIT ON STACK
10818 041304  000002                              RTI
```

```
10821                                                   .SBTTL  TRAP    CLEAR ALL ECC CSR S TRAP HANDLER
10822 041306                                    $CLRCSR:CLEAR   CSR
10823 041312  004737  041356                            CALL    CSROUT
10824 041316  000002                                    RTI
10825                                                   .SBTTL  TRAP    CLEAR 1 SELECTED CSR TRAP HANDLER
10826 041320                                    $CLR1CSR:CLEAR  CSR
10827 041324  104425                                    LOADCSR
10828 041326  000002                                    RTI
10829                                                   .SBTTL  TRAP    ECC DISABLE, CHECK MODE, & WRITE CHECKBITS IN ALL CSR S TRAP HANDLER
10830                                                   ;CHECKBITS ALREADY IN LOC "CSR"
10831 041330  052737  000006  002144              $CHKDIS:BIS     #BIT1!BIT2,CSR          ;ECC DISABLE & DIAG CHECK MODE
10832 041336  004737  041356                            CALL    CSROUT
10833 041342  000002                                    RTI
10834                                                   .SBTTL  TRAP    ECC DISABLE, CHECK MODE, & WRITE CHECKBITS IN 1 SELECTED CSR
10835                                                   ;CHECKBITS ALREADY IN LOC "CSR"
10836 041344  052737  000006  002144              $CHK1DIS:BIS    #BIT1!BIT2,CSR          ;ECC DISABLE & DIAG CHECK MODE
10837 041352  104425                                    LOADCSR
10838 041354  000002                                    RTI
```

```
10841 041356                          CSROUT: SUBTST  <<SUBR  WRITE IN ALL CSR'S>>
                                      ;*****************************************************************************
                                      ;*SUBTEST        SUBR    WRITE IN ALL CSR'S
                                      ;*****************************************************************************
10842 041356                                  PUSH    R1
10843 041360  013701  002220                    MOV     TOTCSRS,R1      ;GET CSR'S BYTE
10844 041364                                  BEGIN LCSROUT
10845 041364                                    FOR CSRNO := #0 TO #36 BY #2
10846 041370  006301                              ASL           R1
10847 041372                                      ON.ERROR
10848 041374  104425                                LOADCSR
10849 041376                                      END ;OF ON.ERROR
10850 041376                                    IF R1 EQ #0 THEN LEAVE LCSROUT
10851 041402                                    END ;OF FOR CSRNO
10852 041420                                  END LCSROUT
10853 041420                                  POP     R1
10854 041422  000207                          RETURN
10855
10856 041424                          $INVALID:       SUBTST  <<TRAP  INVALIDATE BACKGROUND PATTERN>>
                                      ;*****************************************************************************
                                      ;*SUBTEST        TRAP    INVALIDATE BACKGROUND PATTERN
                                      ;*****************************************************************************
10857 041424                                  PUSH    R0,R1
10858 041430  013701  002100                    MOV     BANK,R1
10859 041434  006301                            ASL     R1
10860 041436  006301                            ASL     R1
10861 041440  042761  020000  002630            BIC     #BIT13,CONFIG+2(R1)
10862 041446                                  POP     R1,R0
10863 041452  000002                          RTI
```

```
10865 041454                                    $ERRGEN:        SUBTST<<TRAP    GENERATE AND TEST ERROR ADDRESS>
                                                ;*****************************************************************
                                                ;*SUBTEST        TRAP    GENERATE AND TEST ERROR ADDRESS
                                                ;*****************************************************************
10866 041454                                            PUSH    R0,R1,R2,R3
10867 041464  013703  002102                      MOV     BANKINDEX,R3
10868 041470  005737  002430                      TST     NOSUPER
10869 041474  001003                              BNE     6$
10870 041476  013700  172246                      MOV     SIPAR3,R0               ;GENERATE WHAT ERROR ADDR SHOULD BE
10871 041502  000402                              BR      7$
10872 041504  013700  177646              6$:     MOV     UIPAR3,R0
10873 041510  072027  177773              7$:     ASH     #-5,R0
10874 041514  005737  002130                      TST     EUFLAG
10875 041520  001002                              BNE     1$
10876 041522  042700  177600                      BIC     #^C177,R0
10877 041526  000301                      1$:     SWAB    R1                      ;GET CURRENT ADDRESS BITS 11 AND 12
10878 041530  006201                              ASR     R1
10879 041532  006201                              ASR     R1
10880 041534  006201                              ASR     R1
10881 041536  042701  177775                      BIC     #^C2,R1
10882 041542  060100                              ADD     R1,R0                   ;ADD THEM TO THE ADJUSTED PAR VALUE
10883                                           ;GET ERROR ADDRESS FROM CSR UNDER TEST
10884 041544  013701  002144                      MOV     CSR,R1
10885 041550  072127  177773                      ASH     #-5,R1
10886 041554  042701  177600                      BIC     #^C177,R1
10887 041560  005737  002426                      TST     NO22BIT                 ;IS THIS AN 11/44 OR 11/24?
10888 041564  001024                              BNE     2$                      ;BRANCH IF NOT NECESSARY
10889 041566  005737  002130                      TST     EUFLAG                  ;IS IT EUB?
10890 041572  001421                              BEQ     2$                      ;BRANCH IF NOT
10891 041574                                      PUSH    R0                      ;SAVE GENERATED ERROR ADDRESS
10892 041576  013702  002146                      MOV     CSRNO,R2                ;GET CSR NUMBER
10893 041602  052762  040000  172100              BIS     #BIT14,CSRADD(R2)       ;TURN ON EUB BIT CAREFULLY
10894 041610  016200  172100                      MOV     CSRADD(R2),R0           ;GET CSR CONTENTS
10895 041614  042762  040000  172100              BIC     #BIT14,CSRADD(R2)       ;TURN OFF EUB BIT CAREFULLY
10896 041622  042700  177037                      BIC     #^C740,R0               ;CLEAR EVERYTHING BUT ERROR ADDR
10897 041626  006300                              ASL     R0                      ;SHIFT ADDR BITS 18-21 INTO POSITION
10898 041630  006300                              ASL     R0
10899 041632  060001                              ADD     R0,R1                   ;ADD TO CURRENT ERROR ADDRESS
10900 041634                                      POP     R0
10901 041636  020001                      2$:     CMP     R0,R1                   ;COMPARE REAL AND GENERATED ERR. ADDR.
10902 041640  001420                              BEQ     5$                      ;BRANCH IF THEY ARE THE SAME
10903 041642  005737  002134                      TST     INTFLAG                 ;INTERLEAVED?
10904 041646  001411                              BEQ     3$                      ;NO - WE HAVE AN ERROR
10905 041650  062700  000100                      ADD     #100,R0
10906 041654  005737  002136                      TST     INT64K                  ;64K INTERLEAVED MEMORY?
10907 041660  001002                              BNE     4$
10908 041662  062700  000100                      ADD     #100,R0
10909 041666  020001                      4$:     CMP     R0,R1
10910 041670  001404                              BEQ     5$
10911 041672  005737  002064              3$:     TST     SKPERR                  ;ARE WE SUPPOSED TO SKIP ERROR P.O.?
10912 041676  001001                              BNE     5$                      ;YES - SKIP ERROR PRINTOUT
10913 041700  104462                              PERR36                          ;ELSE PRINT ERROR ADDRESS ERROR
10914 041702  010137  002432              5$:     MOV     R1,ERRADD               ;SAVE CSR'S ERROR ADDRESS
10915 041706  005037  002064                      CLR     SKPERR                  ;ENABLE THE ERROR PRINTOUT AGAIN
10916 041712                                      POP     R3,R2,R1,R0             ;RESTORE REGISTERS
10917 041722  000002                              RTI
```

```
10920 041724                          CHKGEN: SUBTST<<SUBR    GENERATE CHECK BITS>>
                                      ;*****************************************************************************
                                      ;*SUBTEST        SUBR    GENERATE CHECK BITS
                                      ;*****************************************************************************
10921                                         ;CHECK BIT GENERATOR ROUTINE
10922                                         ;CALLING SEQUENCE IS:
10923                                         ;       MOV     #WORD1,SOURCE   ;SOURCE = ADDRESS OF DATA
10924                                         ;       CALL    CHKGEN
10925                                         ;
10926                                         ;CHECK BITS RETURNED IN BITS 11 5 OF LOCATION CHECK
10927                                         ;
10928 041724                                  PUSH    R0,R1,R2,R3,R4,R5
10929 041740  012702  000077                  MOV     #77,R2          ;DEFAULT CHECKBITS FOR DOUBLE WORD OF ZEROS
10930 041744  012703  042032                  MOV     #CHKTAB,R3      ;ADDRESS OF CHECKBIT TABLE
10931 041750  013705  002274                  MOV     SOURCE,R5       ;GET SOURCE ADDRESS
10932 041754  012501                          MOV     (R5)+,R1        ;GET LSB'S
10933 041756  011500                          MOV     (R5),R0         ;GET MSB'S
10934
10935 041760  006704                  1$:     SXT     R4              ;EXTEND SIGN OF DOUBLE WORD TO R4
10936 041762  142304                          BICB    (R3)+,R4        ;ELIMINATE BITS THAT DON'T COUNT
10937 041764  074402                          XOR     R4,R2           ;COMPLEMENT MASKED BITS IN CHECKBITS
10938 041766  073027  000001                  ASHC    #1,R0           ;DOUBLE PRECISION LEFT SHIFT R0,,R1
10939 041772  001372                          BNE     1$              ;LOOP TILL ALL BITS ARE CHECKED
10940
10941 041774  042702  177600                  BIC     #^C177,R2               ;KILL ALL JUNK BITS
10942 042000  000302                          SWAB    R2              ;POSITION CHECKBITS IN BITS 11 5
10943 042002  006202                          ASR     R2
10944 042004  006202                          ASR     R2
10945 042006  006202                          ASR     R2
10946 042010  010237  002276                  MOV     R2,CHECK
10947 042014                                  POP     R5,R4,R3,R2,R1,R0
10948 042030  000207                          RETURN
```

```
10951 042032                    CHKTAB: ;BYTE #3
10952 042032 200                        .BYTE   ^C177           ;BIT 31
10953 042033 301                        .BYTE   ^C076           ;BIT 30
10954 042034 302                        .BYTE   ^C075           ;BIT 29
10955 042035 203                        .BYTE   ^C174           ;BIT 28
10956 042036 304                        .BYTE   ^C073           ;BIT 27
10957 042037 205                        .BYTE   ^C172           ;BIT 26
10958 042040 206                        .BYTE   ^C171           ;BIT 25
10959 042041 307                        .BYTE   ^C070           ;BIT 24
10960                                   ;BYTE #2
10961 042042 310                        .BYTE   ^C067           ;BIT 23
10962 042043 211                        .BYTE   ^C166           ;BIT 22
10963 042044 212                        .BYTE   ^C165           ;BIT 21
10964 042045 313                        .BYTE   ^C064           ;BIT 20
10965 042046 214                        .BYTE   ^C163           ;BIT 19
10966 042047 315                        .BYTE   ^C062           ;BIT 18
10967 042050 316                        .BYTE   ^C061           ;BIT 17
10968 042051 217                        .BYTE   ^C160           ;BIT 16
10969                                   ;BYTE #1
10970 042052 320                        .BYTE   ^C057           ;BIT 15
10971 042053 221                        .BYTE   ^C156           ;BIT 14
10972 042054 222                        .BYTE   ^C155           ;BIT 13
10973 042055 323                        .BYTE   ^C054           ;BIT 12
10974 042056 224                        .BYTE   ^C153           ;BIT 11
10975 042057 325                        .BYTE   ^C052           ;BIT 10
10976 042060 326                        .BYTE   ^C051           ;BIT  9
10977 042061 227                        .BYTE   ^C150           ;BIT  8
10978                                   ;BYTE #0
10979 042062 340                        .BYTE   ^C037           ;BIT  7
10980 042063 241                        .BYTE   ^C136           ;BIT  6
10981 042064 242                        .BYTE   ^C135           ;BIT  5
10982 042065 343                        .BYTE   ^C034           ;BIT  4
10983 042066 244                        .BYTE   ^C133           ;BIT  3
10984 042067 345                        .BYTE   ^C032           ;BIT  2
10985 042070 346                        .BYTE   ^C031           ;BIT  1
10986 042071 247                        .BYTE   ^C130           ;BIT  0
```

```
10989 042072                            SUBTST<<SUBR    MAPPER>>
                                ;*********************************************************************
                                ;*SUBTEST        SUBR    MAPPER
                                ;*********************************************************************
10990                           ;THIS SUBROUTINE MAPS THE MEMORY BANK (16K WORDS = 1 BANK)
10991                           ;IN R3 TO THE TEST PATTERN AREA (SUPERVISOR VIRTUAL (60000 - 157777) FOR
10992                           ;THE 11/44 AND 11/45 55; USER VIRTUAL (60000 - 157777) FOR ALL OTHER
10993                           ;PDP-11'S).
10994                           ;
10995                           ;CALL   MOV     BANKNO,R3               ;SET UP BANK ARGUEMENT
10996                           ;       CALL    MAPPER                  ;ACTUAL CALL
10997                           ;       RETURN                          ;ONLY RETURN
10998
10999                           ;SET SUPERVISOR/USER UP FOR 1 TO 1 MAP
11000 042072                    MAPPER: PUSH    R0,R1,R2,R4,R5
11001 042104  012700  172340            MOV     #KIPAR0,R0              ;FIRST AREA TO MAP TO
11002 042110  012701  172240            MOV     #SIPAR0,R1              ;FIRST ADDRESS REGISTER
11003 042114  012704  172200            MOV     #SIPDR0,R4              ;FIRST DESCRIPTOR REGISTER
11004 042120  005737  002430            TST     NOSUPER                 ;CAN WE USE SUPERVISOR MODE?
11005 042124  001404                    BEQ     4$                      ;YES, BRANCH
11006 042126  012701  177640            MOV     #UIPAR0,R1              ;FIRST ADDRESS REGISTER
11007 042132  012704  177600            MOV     #UIPDR0,R4              ;FIRST DESCRIPTOR REGISTER
11008 042136  012702  077406    4$.     MOV     #77406,R2               ;CONSTANT FOR 4K PAGE, UP, R/W
11009 042142  012705  000010            MOV     #8.,R5                  ;COUNTER
11010 042146  012021            1$:     MOV     (R0)+,(R1)+             ;PUT IN SUPERVISOR ADDRESS
11011 042150  010224                    MOV     R2,(R4)+                ;PUT IN SUPERVISOR DESCRIPTOR
11012 042152  077503                    SOB     R5,1$                   ;LOOP TILL DONE
11013 042154  012741  177600            MOV     #177600, (R1)           ;CORRECT LAST FIELD FOR PERIPHERALS PAGE
11014
11015                           ;SET UP SUPERVISOR/USER FOR TEST AREA
11016 042160  022703  000170            CMP     #120.,R3                ;MAP NOTHING (1 TO 1)?
11017 042164  001516                    BEQ     3$                      ;YES - SKIP
11018 042166  072327  000011            ASH     #9.,R3                  ;BANK 1 STARTS AT 100,000 LESS 6 LSB'S
11019                                                                   ;FOR MEMORY MANAGEMENT = 1000
11020 042172  012701  172246            MOV     #SIPAR3,R1              ;SETUP FOR AUTO INCREMENTING
11021 042176  005737  002430            TST     NOSUPER                 ;DO WE HAVE SUPERVISOR MODE?
11022 042202  001402                    BEQ     5$                      ;YES - BRANCH
11023 042204  012701  177646            MOV     #UIPAR3,R1              ;SETUP FOR AUTO INCREMENTING
11024 042210  012702  000004    5$:     MOV     #4,R2                   ;COUNTER
11025 042214  010321            2$:     MOV     R3,(R1)+                ;PLUG IN PAR INFO
11026 042216  062703  000200            ADD     #200,R3                 ;BUMP ADDRESS 4K
11027 042222  077204                    SOB     R2,2$                   ;LOOP TILL DONE
11028 042224  005737  002234            TST     SPLTCSR
11029 042230  001442                    BEQ     9$
11030 042232  162701  000010            SUB     #10,R1
11031 042236  010102                    MOV     R1,R2
11032 042240  062702  000004            ADD     #4,R2
11033 042244  022737  000001  002234    CMP     #1,SPLTCSR
11034 042252  001403                    BEQ     10$
11035 042254  010200                    MOV     R2,R0
11036 042256  010102                    MOV     R1,R2
11037 042260  010001                    MOV     R0,R1
11038 042262  012122            10$:    MOV     (R1)+,(R2)+
11039 042264  011112                    MOV     (R1),(R2)
11040 042266  013700  002102            MOV     BANKINDEX,R0
11041 042272  005737  002136            TST     INT64K
11042 042276  001403                    BEQ     11$
```

```
11043 042300 012700 004000          MOV     #4000,R0
11044 042304 000402                 BR      12$
11045 042306 012700 010000      11$:MOV     #10000,R0
11046 042312 005737 002430      12$:TST     NOSUPER
11047 042316 001403                 BEQ     13$
11048 042320 012701 177652          MOV     #UIPAR5,R1
11049 042324 000402                 BR      14$
11050 042326 012701 172252      13$:MOV     #SIPAR5,R1
11051 042332 060021             14$:ADD     R0,(R1)+
11052 042334 060011                 ADD     R0,(R1)
11053                               ;IF WE ONLY HAVE AN 124K SYSTEM, WE DON T WANT TO TEST THE
11054                               ;LAST 4K, WHERE THE UNIBUS DEVICE PAGE IS. INSTEAD, THE
11055                               ;PROGRAM WILL REMAP THE LAST 4K TO 8-12K. ALSO, IF THERE
11056                               ;IS A BANK 177 ON AN 11/44, THE PROGRAM WILL REMAP THE LAST
11057                               ;4K TO 8-12K FOR THE SAME REASON.
11058 042336 022737 000007 002530 9$:   CMP     #7,LASTBANK
11059 042344 001010                 BNE     7$
11060 042346 005737 002426          TST     NO22BIT                 ;11/44 OR 24?
11061 042352 001423                 BEQ     3$                      ;BRANCH IF SO
11062 042354 022737 000007 002100       CMP     #7,BANK             ;BANK 7?
11063 042362 001017                 BNE     3$                      ;NO   BRANCH
11064 042364 000404                 BR      8$
11065 042366 022737 000177 002530 7$:   CMP     #177,LASTBANK
11066 042374 001012                 BNE     3$
11067 042376 005737 002430      8$: TST     NOSUPER
11068 042402 001404                 BEQ     6$
11069 042404 013737 177652 177654       MOV     UIPAR5,UIPAR6
11070 042412 000403                 BR      3$
11071 042414 013737 172252 172254 6$:   MOV     SIPAR5,SIPAR6
11072 042422                    3$: POP     R5,R4,R2,R1,R0
11073 042434 000207                 RETURN
11074
11075 042436                        .SBTTL  TRAP    MAP KERNEL (ALMOST 1 TO 1) TRAP HANDLER
11076 042450 005000             $KMAP:  PUSH    R0,R1,R2,R3,R4
11077 042452 012701 172340              CLR     R0                      ;1ST AREA TO MAP TO
11078 042456 012702 077406              MOV     #KIPAR0,R1              ;FIRST ADDRESS
11079 042462 012703 172300              MOV     #77406,R2               ;CONSTANT FOR 4K PAGE,UP,R/W
11080 042466 012704 000010              MOV     #KIPDR0,R3              ;1ST PAGE DESCRIPTOR REGISTER
11081 042472 010021                     MOV     #8.,R4                  ;COUNTER
11082 042474 010223             1$:     MOV     R0,(R1)+                ;PUT IN KERNEL ADDRESS
11083 042476 062700 000200              MOV     R2,(R3)+                ;PUT IN KERNEL DISCRIPTOR
11084 042502 077405                     ADD     #200,R0                 ;ADD ADDRESS CONSTANT FOR 4K CHANGE
11085 042504 012741 177600              SOB     R4,1$                   ;LOOP TILL DONE
11086 042510 012741 177400              MOV     #177600,-(R1)           ;THE PERIPHERALS PAGE TO KIPAR7
11093 042514                            MOV     #177400,-(R1)           ;AND NEXT LOWER PAGE TO KIPAR6
11094 042526 000002                     POP     R4,R3,R2,R1,R0
                                        RTI
```

```
11097 042530                                    RELOCATE:SUBTST <<RELOCATE PROGRAM>>
                                                ;*****************************************************************************
                                                ;*SUBTEST       RELOCATE PROGRAM
                                                ;*****************************************************************************
11098 042530                                            IF @SW12 SET.IN @SWR THEN $RETURN ERROR
11099 042544                                            IF APTFLAG IS TRUE OR ACTFLAG IS TRUE
11100 042560                                              IF $PASS NE #0 THEN $RETURN ERROR
11101 042572                                            END; OF IF APTFLAG
11102 042572                                            BEGIN LOADERBANK
11103 042572                                              FOR BANK := #1 TO LASTBANK
11104 042600  004737  044302                                CALL EXBANK
11105 042604                                                IF ACFLAG IS TRUE AND PFLAG IS FALSE AND BMFLAG IS FALSE
11106 042626  013700  002100                                  MOV       BANK,R0
11107 042632  010037  002404                                  MOV       R0,LOADBANK
11108 042636  013701  002540                                  MOV       LOADHOME,R1
11109 042642  004737  043752                                  CALL      BANKMOV
11110 042646  004737  044250                                  CALL      NEWLOAD          ;MAP NEW LOADER BANK IN KERNEL
11111 042652  013701  002102                                  MOV       BANKINDEX,R1
11112 042656  052761  100000  002630                          BIS       #BIT15,CONFIG+2(R1)     ;MARK LOADER
11113 042664  042761  020000  002630                          BIC       #BIT13,CONFIG+2(R1)     ;INVALIDATE BACKGROUND PAT'ERN
11114 042672                                                  LEAVE LOADERBANK
11115 042674                                                END ;OF IF ACFLAG
11116 042674                                              END ;OF FOR BANK
11117 042710                                              IF #SW13 OFF.IN @SWR
11118 042720                                                TYPE      MSG075           ;RELOCATION NOT POSSIBLE
11119 042724                                              END ;OF IF #SW13
11120 042724                                              $RETURN ERROR
11121 042730                                            END LOADERBANK
11122 042730                                            BEGIN FINDBANK
11123 042730  013702  002530                            MOV     LASTBANK,R2
11124 042734  006302                                    ASL     R2
11125 042736  006302                                    ASL     R2                ;R2 <  R2 * 4
11126 042740                                              FOR R1 := #2+2 TO R2 BY #4
11127 042744                                                IF #BIT7!BIT0 OFF.IN CONFIG(R1) ;IF NO ERRORS & NOT PROGRAM SPACE
11128 042754                                                  IF #BIT15 OFF.IN CONFIG+2(R1) ;IF NOT LOADER BANK
11129 042764                                                    IF CPUBIT SET.IN CONFIG(R1) ;IF ACCESSABLE
11130 042774                                                      IF #BIT9 SET.IN CONFIG+2(R1) THEN LEAVE FINDBANK ;IF PARITY
11131 043004                                                        IF #BIT6 SET.IN CONFIG(R1) AND #BIT7 OFF.IN CONFIG(R1)
11132                                                                 ;IF 1ST PROTECTABLE ECC BANK
11133 043024                                                          LEAVE FINDBANK
11134 043026                                                        END ;OF IF #BIT6
11135 043026                                                        IF INHECC IS FALSE
11136 043034                                                          SET   INHECC
11137 043042  010137  002512                                        MOV   R1,INHBANK
11138 043046                                                        END; OF IF INHECC
11139 043046                                                    END ;OF IF CPUBIT
11140 043046                                                  END ;OF IF #BIT15
11141 043046                                                END ;OF IF #BIT7
11142 043046                                              END ;OF FOR
11143 043056                                              IF FULLREL IS FALSE
11144 043064                                                IF INHECC IS TRUE
11145 043072  013701  002512                                  MOV     INHBANK,R1
11146 043076  023727  002262  000030                          CMP     REALPAT,#30       ;IS THIS PATTERN 30?
11147 043104  001423                                          BEQ     RELENT1           ;YES   SKIP MESSAGE
11148 043106                                                  TYPE    MSG123
11149 043112  000420                                          BR      RELENT1
11150 043114                                                END; OF IF INHECC
```

```
11151 043114                                        END; OF IF FULLREL
11152 043114  005037  002510                        CLR   INHECC                    ;MAKE SURE FLAG IS TURNED OFF!
11153 043120                                        IF @SW13 OFF.IN @SWR
11154 043130  023727  002262  000030                  CMP         REALPAT,@30       ;IS THIS PATTERN 30?
11155 043136  001402                                  BEQ         SKUB              ;YES   SKIP MESSAGE
11156 043140                                          TYPE        MSG075            ;RELOCATION NOT POSSIBLE
11157 043144                                        END ;OF IF @SW13
11158 043144                              SKUB:     $RETURN ERROR
11159 043150                                      END FINDBANK
11160 043150                                      CLEAR   INHECC                    ;IF WE RELOCATED PROPERLY, THIS SHOULD BE OFF!
11161 043154  042761  020000  002630    RELENT1: BIC     @BIT13,CONFIG+2(R1)        ;INVALIDATE BACKGROUND PATTERN
11162 043162  005000                              CLR     R0
11163 043164  071027  000004                      DIV     @4,R0
11164 043170                              RELOC1: LET NEWBANK := R0
11165 043174  013737  002504  002506              MOV     PGMCSR,PGMCSR+2           ;SAVE CURRENT PGM. CSR
11166 043202  004737  044120                      CALL    USERMAP                   ;MAP NEWBANK TO USER PAR
11167 043206                                      USER                              ;ENTER USER MODE
11168 043214                                      BMOV    0,100000,SIZE             ;MOVE PROGRAM
11169 043226  104417                              KERNEL                            ;ENTER KERNEL MODE
11170 043230  022737  000001  003720              CMP     @1,PROTYP                 ;IS THIS AN 11/44 ?
11171 043236  001021                              BNE     JMPRL1                    ;JUMP IF NOT
11172 043240  042737  000040  172516              BIC     @BIT5,MMR3                ;TURN OFF UNIBUS MAP
11173 043246  013700  002272                      MOV     NEWBANK,R0
11174 043252  006200                              ASR     R0
11175 043254                                      ON.ERROR
11176 043256  012737  100000  170200                MOV   @BIT15,MAPL0
11177 043264                                      END ;OF ON.ERROR
11178 043264  010037  170202                      MOV     R0,MAPH0
11179 043270  004737  043706                      CALL    LOWMAP                    ;SETUP LOWER 16K IN UNIBUS MAP
11180 043274  052737  000040  172516              BIS     @BIT5,MMR3                ;ENERGIZE UNIBUS MAP
11181 043302  042737  000001  177572    JMPRL1: BIC     @BIT0,MMR0                ;DEENERGIZE MEMORY MANAGEMENT
11182 043310  004737  044202                      CALL    NEWKERNEL
11183 043314  013700  002272                      MOV     NEWBANK,R0
11184 043320  006300                              ASL     R0
11185 043322  006300                              ASL     R0                        ;R0 <- R0 * 4
11186 043324  016002  002626                      MOV     CONFIG(R0),R2
11187 043330  000302                              SWAB    R2
11188 043332  042702  177760                      BIC     #^C17,R2
11189 043336  006302                              ASL     R2
11190 043340  052737  000001  177572              BIS     @BIT0,MMR0                ;ENERGIZE MEMORY MANAGEMENT
11191 043346  010237  002504                      MOV     R2,PGMCSR                 ;PUT NEW PGM. CSR INTO PGMCSR
11192 043352  032760  010000  002630              BIT     @BIT12,CONFIG+2(R0)       ;IS THE NEW BANK INTERLEAVED?
11193 043360  001412                              BEQ     1$                        ;BRANCH IF NOT INTERLEAVED
11194 043362  016002  002626                      MOV     CONFIG(R0),R2
11195 043366  042702  007777                      BIC     #^C170000,R2
11196 043372  072227  177775                      ASH     #-3,R2
11197 043376  052702  100000                      BIS     @BIT15,R2
11198 043402  050237  002504                      BIS     R2,PGMCSR
11199 043406                                1$:   SET     RLFLAG
11200 043414                                      $RETURN NOERROR
```

```
11203 043420                                UNRELOCATE:SUBTST       <<UNRELOCATE PROGRAM>>
                                            ;*****************************************************************************
                                            ;*SUBTEST        UNRELOCATE PROGRAM
                                            ;*****************************************************************************
11204                                               ;RESTORE LOADERS
11205 043420                                        PUSH    R0
11206 043422  013701  002404                        MOV     LOADBANK,R1
11207 043426  013700  002540                        MOV     LOADHOME,R0
11208 043432  004737  043752                        CALL    BANKMOV
11209 043436  004737  044250                        CALL    NEWLOAD                 ;MAP NEW LOADER BANK IN KERNEL SPACE
11210 043442                                        PUSH    BANK
11211 043446  013737  002404  002100                MOV     LOADBANK,BANK
11212 043454  004737  044302                        CALL    EXBANK
11213 043460  013701  002102                        MOV     BANKINDEX,R1
11214 043464  042761  100000  002630                BIC     #BIT15,CONFIG+2(R1)     ;CLEAR LOADER FLAG
11215 043472  013737  002540  002100                MOV     LOADHOME,BANK
11216 043500  004737  044302                        CALL    EXBANK
11217 043504  013701  002102                        MOV     BANKINDEX,R1
11218 043510  042761  020000  002630                BIC     #BIT13,CONFIG+2(R1)     ;INVALIDATE BACKGROUND PATTERN
11219 043516                                        POP     BANK
11220 043522                                        CLEAR   INHECC                  ;MAKE SURE ECC TESTS ARE NOT INHIBITED!
11221
11222                                               ;RESTORE BANK 0
11223 043526  042737  020000  002630                BIC     #BIT13,CONFIG+2         ;INVALIDATE BACKGROUND PATTERN
11224 043534                                        LET NEWBANK := #0
11225 043540  004737  044120                        CALL    USERMAP                 ;MAP NEWBANK TO USER PAR
11226 043544                                        USER                            ;ENTER USER MODE
11227 043552                                        BMOV    0,100000,SIZE           ;MOVE PROGRAM
11228 043564  104417                                KERNEL                          ;ENTER KERNEL MODE
11229 043566  042737  000001  177572                BIC     #BIT0,MMR0              ;DEENERGIZE MEMORY MANAGEMENT
11230 043574  004737  044202                        CALL    NEWKERNEL
11231 043600  013737  002506  002504                MOV     PGMCSR+2,PGMCSR         ;RESTORE PREVIOUS PGM. CSR
11232 043606  052737  000001  177572                BIS     #BIT0,MMR0              ;ENERGIZE MEMORY MANAGEMENT
11233 043614  005037  002124                        CLR     RLFLAG
11234 043620  022737  000001  003720                CMP     #1,PROTYP               ;IS THIS AN 11/44 ?
11235 043626  001014                                BNE     1$
11236 043630  042737  000040  172516                BIC     #BIT5,MMR3              ;TURN OFF UNIBUS MAP
11237 043636                                        CLEAR   MAPLO,MAPHO
11238 043646  004737  043706                        CALL    LOWMAP                  ;SETUP LOWER 16K OF UNIBUS MAP
11239 043652  052737  000040  172516                BIS     #BIT5,MMR3              ;ENERGIZE UNIBUS MAP
11240 043660  012700  002630          1$:           MOV     #CONFIG+2,R0    ;MOVE 2ND WORD OF CONFIG TO R0
11241 043664  042710  020000          2$:           BIC     #BIT13,(R0)     ;CLEAR BACKGROUND VALID BIT
11242 043670  062700  000004                        ADD     #4,R0           ;INCREMENT TO NEXT BANK
11243 043674  020027  003620                        CMP     R0,#3620        ;DONE?
11244 043700  003771                                BLE     2$              ;NO   BRANCH
11245 043702                                        POP     R0
11246 043704  000207                                RETURN
11247
11248 043706                                LOWMAP: SUBTST  <<SETUP LOWER 16K OF UNIBUS MAP>>
                                            ;*****************************************************************************
                                            ;*SUBTEST        SETUP LOWER 16K OF UNIBUS MAP
                                            ;*****************************************************************************
11249 043706                                        PUSH    R0,R1,R2
11250 043714  012700  170200                        MOV     #MAPLO,R0
11251 043720  012701  170204                        MOV     #MAPL1,R1
11252 043724  012702  000003                        MOV     #3,R2
11253 043730  012011                          1$:   MOV     (R0)+,(R1)
```

SETUP LOWER 16K OF UNIBUS MAP

```
11254 043732  062721  020000          ADD     #BIT13,(R1)+
11255 043736  012021                  MOV     (R0)+,(R1)+
11256 043740  077205                  SOB     R2,1$
11257 043742                          POP     R2,R1,R0
11258 043750  000207                  RETURN
```

```
11261 043752                          BANKMOV:SUBTST  <<MOVE BANKS>>
                                      ;*****************************************************************
                                      ;*SUBTEST        MOVE BANKS
                                      ;*****************************************************************
11262                                         ;MOVE 3/4 OF A BANK
11263                                         ;CALLING SEQUENCE
11264                                         ;R0 - DESTINATION BANK
11265                                         ;R1 - SOURCE BANK
11266 043752  104415                          SAVREG
11267 043754  004737  044120                  CALL    USERMAP
11268 043760  104416                          RESREG
11269 043762  104415                          SAVREG
11270 043764  072027  000011                  ASH     #9.,R0
11271 043770  072127  000011                  ASH     #9.,R1
11272 043774  012702  177650                  MOV     #UIPAR4,R2
11273 044000  012703  000200                  MOV     #200,R3
11274
11275 044004  010122                          MOV     R1,(R2)+                ;MAP 1ST HALF BANK
11276 044006  060301                          ADD     R3,R1                   ;BUMP BY 4K
11277 044010  010122                          MOV     R1,(R2)+
11278 044012  060301                          ADD     R3,R1
11279
11280 044014  010022                          MOV     R0,(R2)+
11281 044016  060300                          ADD     R3,R0
11282 044020  010022                          MOV     R0,(R2)+
11283 044022  060300                          ADD     R3,R0
11284
11285 044024                                   USER
11286 044032                                   BMOV    100000,140000,SIZE/2    ;MOV 1ST HALF BANK
11287 044044  104417                          KERNEL                          ;ENTER KERNEL MODE
11288
11289 044046  012702  177650                  MOV     #UIPAR4,R2
11290
11291 044052  010122                          MOV     R1,(R2)+                ;MAP 2ND HALF BANK
11292 044054  060301                          ADD     R3,R1                   ;BUMP BY 4K
11293 044056  010122                          MOV     R1,(R2)+
11294 044060  060301                          ADD     R3,R1
11295
11296 044062  010022                          MOV     R0,(R2)+
11297 044064  060300                          ADD     R3,R0
11298 044066  010022                          MOV     R0,(R2)+
11299 044070  060300                          ADD     R3,R0
11300
11301 044072                                   USER
11302 044100                                   BMOV    100000,140000,SIZE/4    ;MOV 3ND FOURTH OF BANK
11303 044112  104417                          KERNEL                          ;ENTER KERNEL MODE
11304
11305 044114  104416                          RESREG
11306 044116  000207                          RETURN
```

```
044120                                  USERMAP:SUBTST  <<SUBR  MAP USER TO NEW BANK>>
                                        ;*********************************************************************
                                        ;*SUBTEST        SUBR    MAP USER TO NEW BANK
                                        ;*********************************************************************
044120  012701  177640                          MOV     #UIPAR0,R1              ;COPY KERNEL PAR'S & PDR'S (0 3)
044124  012702  172340                          MOV     #KIPAR0,R2
044130  012703  177600                          MOV     #UIPDR0,R3
044134  012704  172300                          MOV     #KIPDR0,R4
044140  012705  000004                          MOV     #4,R5
044144  012221                          1$:     MOV     (R2)+,(R1)+
044146  011423                                  MOV     (R4),(R3)+
044150  077503                                  SOB     R5,1$

044152  013700  002272                          MOV     NEWBANK,R0
044156  072027  000011                          ASH     #9.,R0                  ;BANK 1 STARTS AT 100,000 LESS 6 LSB S
                                                                                ;FOR MEMORY MANAGEMENT = 1000
044162  012705  000004                          MOV     #4,R5
044166  010021                          2$:     MOV     R0,(R1)+                ;SETUP UIPAR(4 7)
044170  062700  000200                          ADD     #200,R0                 ;BUMP ADDRESS 4K
044174  011423                                  MOV     (R4),(R3)+              ;SETUP UIPDR(4 7)
044176  077505                                  SOB     R5,2$
044200  000207                                  RETURN

044202                                  NEWKERNEL:SUBTST        <<SUBR  SETUP KERNEL PAR'S FOR NEW BANK>>
                                        ;*********************************************************************
                                        ;*SUBTEST        SUBR    SETUP KERNEL PAR'S FOR NEW BANK
                                        ;*********************************************************************
044202                                          PUSH    R0,R1,R5
044210  012700  172340                          MOV     #KIPAR0,R0
044214  013701  002272                          MOV     NEWBANK,R1
044220  072127  000011                          ASH     #9.,R1                  ;BANK 1 STARTS AT 100,000 LESS 6 LSB'S
                                                                                ;FOR MEMORY MANAGEMENT = 1000
044224  012705  000004                          MOV     #4,R5
044230  010120                          1$:     MOV     R1,(R0)+                ;SETUP KIPAR(0-3)
044232  062701  000200                          ADD     #200,R1
044236  077504                                  SOB     R5,1$
044240                                          POP     R5,R1,R0
044246  000207                                  RETURN

044250                                  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
044250                                          PUSH    R0,R1
044254  012701  172350                          MOV     #KIPAR4,R1
044260  072027  000011                          ASH     #9.,R0                  ;BANK 1 STARTS AT 100000 LESS 6 LSB S (1000)
044264  010021                                  MOV     R0,(R1)+                ;SETUP KIPAR4
044266  062700  000200                          ADD     #200,R0
044272  010021                                  MOV     R0,(R1)+                ;SETUP KIPAR5
044274                                          POP     R1,R0
044300  000207                                  RETURN
```

```
11354 044502                          EXBANK: SUBTST  <<SUBR  EXAMINE BANK>>
                                      ;*****************************************************************************
                                      ;*SUBTEST        SUBR    EXAMINE BANK
                                      ;*****************************************************************************
11355                                 ;DOES THE FOLLOWING:
11356                                 ;(1)  SETS UP "BANKINDEX" AND R1 BASED ON VALUE OF  BANK .
11357                                 ;(2)  SETS THE "MKFLAG" IF THE BANK IS ECC.
11358                                 ;(3)  SETS THE "KFLAG" IF THE BANK IS MF11S K.
11359                                 ;(4)  SETS THE "KPFLAG" IF THE BANK IS THE PROTECTED REGION OF ECC MEMORY.
11360                                 ;(5)  SETS THE "ACFLAG" IF THE BANK CAN BE ACCESSED BY THIS CPU.
11361                                 ;(6)  SETS THE "PFLAG" IF THE BANK IS IN PROGRAM SPACE.
11362                                 ;(7)  SETS THE  "RRFLAG" IF RELOCATION IS REQUIRED TO TEST THIS BANK; HOWEVER,
11363                                 ;     IT COMPLEMENTS THIS FLAG IF THE RELOCATION FLAG 'RLFLAG" IS SET (THIS IS
11364                                 ;     NECESSARY FOR THE USE OF THE RECURSIVE "MODE" SUBROUTINES). THE "RRFLAG
11365                                 ;     IS ALWAYS SET TO DISABLE TESTING IF FIELD SERVICE MODE  "SELECTED BANKS
11366                                 ;     ARE BEING TESTED AND THIS BANK IS NOT SELECTED.
11367                                 ;(8)  SETS THE "BMFLAG" IF THE BANK IS A BAD MEMORY;  HOWEVER, IT  COMPLEMENTS
11368                                 ;     THIS FLAG IF THE "WORST" FLAG IS NOT SET  (THIS IS NECESSARY FOR THE USE
11369                                 ;     OF THE RECURSIVE "MODE" SUBROUTINES).
11370                                 ;(9)  SETS THE "EUFLAG" IF THE BANK HAS EXTENDED UNIBUS MEMORY.
11371                                 ;(10) SETS THE "INTFLAG" IF THE BANK IS INTERLEAVED.
11372                                 ;(11) SETS THE "INT64K" FLAG IF THE BANK IS INTERLEAVED ON 64K WORD BOUNDS.
11373                                 ;(12) SETS THE "SKIPMK" FLAG IF THIS BANK IS INTERLEAVED, AND HAS ALREADY
11374                                 ;     BEEN TESTED.
11375                                 ;
11376 044302                                 PUSH    R0,R1,R2
11377 044310                                 CLEAR   MKFLAG,KPFLAG,KFLAG,EUFLAG
11378 044330                                 SET     ACFLAG
11379 044336                                 CLEAR   PFLAG,RRFLAG,BMFLAG
11380 044352                                 CLEAR   INTFLAG,INT64K,SKIPMK
11381 044366  013701  002100                 MOV     BANK,R1
11382 044372  006301                         ASL     R1
11383 044374  006301                         ASL     R1                      ;R1 <  R1 * 4
11384 044376  010137  002102                 MOV     R1,BANKINDEX
11385 044402  032761  000100  002626         BIT     #BIT6,CONFIG(R1)        ;PROTECTED REGION OF ECC MEMORY?
11386 044410  001403                         BEQ     1$                      ;NO   SKIP
11387 044412                                 SET     KPFLAG
11388 044420  012700  000002          1$:    MOV     #BIT1,R0
11392 044424                                 IF R0 SET.IN CPUBIT AND R0 OFF.IN CONFIG(R1)
11393 044440  005037  002114                   CLR    ACFLAG
11394 044444                                 END ;OF IF R0
11399 044444  005737  002114                 TST     ACFLAG                  ;ACTIVE MEMORY?
11400 044450  001415                         BEQ     12$                     ;BRANCH IF NOT
11401 044452  016102  002630                 MOV     CONFIG+2(R1),R2
11402 044456  000302                         SWAB    R2
11403 044460  042702  177770                 BIC     #^C7,R2                 ;ISOLATE MEM TYPE BITS
11404 044464  020227  000002                 CMP     R2,#2                   ;IS THIS AN ILLEGAL MEM TYPE?
11405 044470  003005                         BGT     12$                     ;BRANCH IF NOT
11406 044472                                 SET     BMFLAG                  ;SET BAD BANK FLAG
11407 044500  000137  044742                 JMP     ENEXBK                  ;JUMP OVER REST OF FLAG TESTS
11408 044504  032761  000400  002630  12$:   BIT     #BIT8,CONFIG+2(R1)      ;IS THIS EUB?
11409 044512  001003                         BNE     2$                      ;BRANCH IF NOT
11410 044514                                 SET     EUFLAG                  ;YES - SET EUB FLAG
11411 044522  032761  001000  002630  2$:    BIT     #BIT9,CONFIG+2(R1)      ;IS THERE ECC THERE?
11412 044530  001012                         BNE     3$                      ;NO - SKIP
11413 044532                                 SET     MKFLAG          ;YES    SET MKFLAG
11414 044540  032761  000400  002630         BIT     #BIT8,CONFIG+2(R1)      ;IS THIS MF11S K MEMORY
```

```
11415 044546 001407                    BEQ     3$                      ;NO   IT S MS11 M
11416 044550                           SET     KFLAG                   ;YES   SET KFLAG
11417 044556 032761 000200 002620  3$: BIT     #BIT7,CONFIG(R1)        ;BANK - PROGRAM SPACE
11418 044564 001406                    BEQ     5$                      ;NO   SKIP
11419 044566                           SET     PFLAG,RRFLAG
11420 044602 005737 002124         5$: TST     RLFLAG                  ;IS PROGRAM RELOCATED?
11421 044606 001402                    BEQ     6$                      ;NO   SKIP
11422 044610 005137 002122             COM     RRFLAG                  ;YES   COMPLEMENT RELOCATION REQUIRED FLAG
11423 044614 032761 000001 002626  6$: BIT     #BIT0,CONFIG(R1)        ;ERRORS PRESENT IN THIS BANK?
11424 044622 001403                    BEQ     8$                      ;NO   SKIP
11425 044624                           SET     BMFLAG
11426 044632 005737 002542         8$: TST     WORST                   ;IS THIS A WORST FIRST PASS?
11427 044636 001002                    BNE     9$                      ;YES   SKIP
11428 044640 005137 002126             COM     BMFLAG                  ;NO   COMPLEMENT BAD MEMORY FLAG
11429 044644                       9$: IF SELONLY IS TRUE AND #BIT14 OFF.IN CONFIG+2(R1)
11430 044662                             SET   RRFLAG
11431 044670                           END ;OF IF SELONLY
11432 044670 032761 010000 002630      BIT     #BIT12,CONFIG+2(R1)     ;IS THIS BANK INTERLEAVED?
11433 044676 001421                    BEQ     ENEXBK                  ;BRANCH IF IT IS NOT
11434 044700                           SET     INTFLAG
11435 044706 032761 004000 002630      BIT     #BIT11,CONFIG+2(R1)     ;IS THIS BANK INTERLEAVED WITH 64K BOARDS?
11436 044714 001403                    BEQ     10$                     ;BRANCH IF IT IS NOT
11437 044716                           SET     INT64K
11438 044724 032761 000040 002626 10$: BIT     #BIT5,CONFIG(R1)        ;SHOULD THIS BANK BE TESTED?
11439 044732 001403                    BEQ     ENEXBK                  ;BRANCH IF IT SHOULD
11440 044734                           SET     SKIPBK
11441 044742                       ENEXBK: POP     R2,R1,R0                ;RESTORE REGISTERS
11442 044750 000207                    RETURN
```

```
11445 044752                          BANKOK: SUBTST  <<SUBR  BANK OK?>>
                                      ;*****************************************************************************
                                      ;*SUBTEST        SUBR    BANK OK?
                                      ;*****************************************************************************
11446                                         ;TEST TO INSURE THAT THE TYPE OF MEMORY IN THE PRESENT BANK
11447                                         ;IS OF THE TYPE WE ARE TESTING "TMFLAG".
11448                                         ;RESULT IS RETURNED IN THE CONDITION CODES (OK = (=0)).
11449 044752  013700  002132                  MOV     TMFLAG,R0
11450 044756  005100                        COM     R0
11451 044760  013701  002116                MOV     MKFLAG,R1
11452 044764  074001                        XOR     R0,R1
11453 044766  000207                        RETURN                          ;OK = (=OK)
11454
11455 044770                          INCRPT:
11456 044770                          INCPAT: SUBTST  <<SUBR  INCREMENT PATTERN TESTING >>
                                      ;*****************************************************************************
                                      ;*SUBTEST        SUBR    INCREMENT PATTERN TESTING
                                      ;*****************************************************************************
11457                                         ;INCREMENT THE PATTERN & SET UP THE CONDITION CODES
11458                                         ;RESULT - Z BIT SET INDICATES OVERFLOW
11459 044770  005237  002110                INC     PATTERN
11460 044774  022737  000030  002110        CMP     #30,PATTERN             ;SET UP CONDITION CODES
11461 045002  000207                        RETURN                          ;NOT EQUAL TO ZERO IS GOOD (NO OVERFLOW)
11462
11463 045004                          SETPAT:
11464 045004                          HIPAT:  SUBTST  <<SUBR  SET HIGHEST PATTERN TESTING TYPE>>
                                      ;*****************************************************************************
                                      ;*SUBTEST        SUBR    SET HIGHEST PATTERN TESTING TYPE
                                      ;*****************************************************************************
11465 045004  012737  000027  002110        MOV     #27,PATTERN             ;SET HIGHEST PATTERN
11466 045012  000207                        RETURN
11467
11468 045014                          INCBNK: SUBTST  <<SUBR  INCREMENT BANK & TEST>>
                                      ;*****************************************************************************
                                      ;*SUBTEST        SUBR    INCREMENT BANK & TEST
                                      ;*****************************************************************************
11469                                         ;RESULTS RETURNED IN CONDITION CODES
11470 045014  005237  002100                INC     BANK
11471 045020  023737  002530  002100        CMP     LASTBANK,BANK           ;TOO FAR?
11472 045026  000207                        RETURN
```

```
11475 045030                          BOOT:   SUBTST  <<BOOTSTRAP ROUTINE>>
                                      ;*****************************************************************
                                      ;*SUBTEST       BOOTSTRAP ROUTINE
                                      ;*****************************************************************
11476                                         ;INITIALIZE ALL CSR'S
11477                                         ;UNRELOCATE IF NECESSARY
11478                                         ;FLUSH OUT ANY DBE'S
11479                                         ;TURN OFF MEMORY MANAGEMENT
11480                                         ;TURN OFF THE UNIBUS MAP
11481                                         ;BOOT RK0 OR RK1
11482 045030  104472                        ECCINIT         ;TRAP ON DOUBLE BIT ERRORS (NORMAL)
11483 045032                                SET4    #BOOT1          ;TRAPS TO 4 GOTO BOOT1
11484 045040                                IF RLFLAG IS TRUE THEN $CALL UNRELOCATE
11485 045052  004737  024720                CALL    MT0030          ;FLUSH OUT DBE'S
11486 045056  104421                        DEENERGIZE              ;TURN OFF MEMORY MANAGEMENT
11487 045060  005737  002426                TST     N022BIT                 ;IS THIS AN 11/44 OR 11/24?
11488 045064  001003                        BNE     BOOT1
11489 045066  042737  000040  172516        BIC     #BIT5,MMR3                      ;TURN OFF THE UNIBUS MAP
11490 045074  005001              BOOT1:    CLR     R1
11491 045076  000005              1$:       RESET
11492 045100  012700  177406                MOV     #177406,R0
11493 045104  010160  000004                MOV     R1,4(R0)
11494 045110  012710  177400                MOV     #177400,(R0)
11495 045114  012740  000005                MOV     #5,-(R0)
11496 045120  105710              2$:       TSTB    (R0)
11497 045122  100376                        BPL     2$
11498 045124  062701  020000                ADD     #BIT13,R1
11499 045130  005710                        TST     (R0)
11500 045132  100761                        BMI     1$
11501 045134  005007                        CLR     PC
```

```
11504 045136                          EXIT:   SUBTST  <<HALT PROGRAM>>
                                      ;*****************************************************************************
                                      ;*SUBTEST       HALT PROGRAM
                                      ;*****************************************************************************
:11505 045136  004737  045170                 CALL    SHUTUP
11506 045142                          EXIT2:  IF APTFLAG IS TRUE OR ACTFLAG IS TRUE
11507 045156  000777                            BR      .
11508 045160                                  ELSE
11509 045162  000000                  $EXHALT:  HALT
11510 045164  000137  003632                   JMP     START
11511 045170                                  END ;OF IF APTFLAG
11512
11513 045170                          SHUTUP: SUBTST  <<SHUTDOWN DIAGNOSTIC>>
                                      ;*****************************************************************************
                                      ;*SUBTEST       SHUTDOWN DIAGNOSTIC
                                      ;*****************************************************************************
11514                                         ;INITIALIZE ALL CSR'S
11515                                         ;UNRELOCATE
11516                                         ;FLUSH OUT DBE'S
11517                                         ;RESTORE LOADERS
11518                                         ;TURN OFF MEMORY MANAGEMENT
11519                                         ;UNMAP THE UNIBUS MAP
11523 045170  104472                        ECCINIT         ;TRAP ON DOUBLE BIT ERRORS (NORMAL)
11524 045172                                IF RLFLAG IS TRUE THEN $CALL UNRELOCATE
11525 045204                                IF QUICK IS FALSE
11526 045212  004737  024720                  CALL  MT0030          ;FLUSH OUT DBE'S
11527 045216                                END ;OF IF QUICK
11528 045216  012700  000001                MOV     #1,R0           ;DESTINATION BANK
11529 045222  013701  002540                MOV     LOADHOME,R1     ;SOURCE BANK
11530 045226  004737  043752                CALL    BANKMOV
11531 045232  104421                        DEENERGIZE              ;TURN OFF MEMORY MANAGEMENT
11532 045234  005737  002426                TST     NO22BIT         ;DOES THIS PDP-11 HAVE 22-BIT ADDR?
11533 045240  001003                        BNE     1$              ;BRANCH IF NOT
11534 045242  042737  000040  172516        BIC     #BIT5,MMR3              ;TURN OFF UNIBUS MAP
11538 045250  000207                  1$:     RETURN
11539
11540 045252                          APTDOWN:SUBTST  <<APT SHUTDOWN SEQUENCE>>
                                      ;*****************************************************************************
                                      ;*SUBTEST       APT SHUTDOWN SEQUENCE
                                      ;*****************************************************************************
11541 045252                                  MAP     #0                      ;MAP SUPERVISOR SPACE (TEST AREA) TO BANK #0
11542 045266                                  TESTAREA                ;ENTER TEST MODE
11543 045274  012737  045252  060024          MOV     #APTDOWN,FIRST+24
11544 045302  012737  000340  060026          MOV     #340,FIRST+26
11545 045310  012737  000000  125252          MOV     #0,FIRST+APTDOWN
11546 045316  104417                          KERNEL                  ;ENTER KERNEL MODE
11547 045320  000000                  APTHLT: HALT
```

```
11550 045322                                    SUBTST  <<BLOCK MOVE SUBROUTINE>>
                                        ;*****************************************************************************
                                        ;*SUBTEST        BLOCK MOVE SUBROUTINE
                                        ;*****************************************************************************
11551                                           ;BLOCK3 HAS 3 ARGUEMENTS
11552                                           ;BLOCK2 HAS 2 ARGUEMENTS
11553                                           ;BLOCK1 HAS 1 ARGUEMENTS
11554                                           ;
11555                                           ;ALL ARE CALLED BY THE BMOV MACRO
11556                                           .ENABL  LSB
11557 045322                            BLOCK1: PUSH    R0,R1,R2
11558 045330  012702  177640                    MOV     #FASTCITY,R2
11559 045334  012701  000020                    MOV     #16.,R1
11560 045340  000413                            BR      3$
11561
11562 045342                            BLOCK2: PUSH    R0,R1,R2
11563 045350  012701  000020                    MOV     #16.,R1
11564 045354  000404                            BR      2$
11565
11566 045356                            BLOCK3: PUSH    R0,R1,R2
11567 045364  012501                            MOV     (R5)+,R1
11568 045366  012502                    2$:     MOV     (R5)+,R2
11569 045370  012500                    3$:     MOV     (R5)+,R0
11570
11571 045372  012022                    1$:     MOV     (R0)+,(R2)+
11572 045374  077102                            SOB     R1,1$
11573 045376                                    POP     R2,R1,R0
11574 045404  000205                            RTS     R5
11575                                           .DSABL  LSB
```

```
                                        SBTTL  FIELD SERVICE MODE

045406                          FIELDSERVICE:SUBTST     <<SUBR  FIELD SERVICE COMMAND MODE>>
                                ;*****************************************************************
                                ;*SUBTEST        SUBR    FIELD SERVICE COMMAND MODE
                                ;*****************************************************************
045406  104415                          SAVREG
045410                                  TYPE    MSG020          ;FIELD SERVICE COMMAND MODE
045412
045414                                  IF RLFLAG IS TRUE OR NOFSMODE IS TRUE
045430                                    TYPE  MSG048          ;NOT AVAILABLE NOW - TRY LATER!
045434  104416                            RESREG
045436  000207                            RETURN
045440                                  END ;OF IF RLFLAG
045440  005737  002516                  TST     CACHKN
045444  001402                          BEQ     1$
045446                                  PUSH    CONTRL          ;SAVE CACHE STATUS
045452                          1$:     PUSH    CSRNO,KAMIKAZE  ;SAVE CSR & KAMIKAZE STATUS
045462  104424                          CACHOFF                 ;TURN CACHE OFF
045464                                  SET     KAMIKAZF
045472                          FS1:    TYPE    MSG026          ;COMMAND:
045476  104414                          RDDEC                   ;READ A DECIMAL NUMBER
045500                                  POP     R0              ;COMMAND    > R0
045502  020027  000022                  CMP     R0,#18.
045506  101403                          BLOS    1$
045510                                  TYPE    MSG021
045514  000766                          BR      FS1
045516                          1$:     CASE R0
045526  045604                            FSCMD0                ;EXIT FIELD SERVICE COMMANDS
045530  045706                            FSCMD1                ;READ CSR
045532  046016                            FSCMD2                ;LOAD CSR
045534  046164                            FSCMD3                ;EXAMINE MEMORY
045536  046440                            FSCMD4                ;MODIFY MEMORY
045540  046760                            FSCMD5                ;SELECT BANK & PATTERN
045542  047650                            FSCMD6                ;TYPE CONFIGURATION MAP
045544  047656                            FSCMD7                ;SOB-A-LONG TEST
045546  050150                            FSCMD8                ;ERROR SUMMARY
045550  050354                            FSCMD9                ;REFRESH TEST
045552  050646                            FCMD10                ;SET FILL COUNT
045554  050674                            FCMD11                ;ENTER KAMIKAZE MODE
045556  050716                            FCMD12                ;EXIT KAMIKAZE MODE
045560  050736                            FCMD13                ;TURN CACHE OFF
045562  050760                            FCMD14                ;TURN CACHE ON
045564  050776                            FCMD15                ;TEST ONLY SELECTED BANKS
045566  051062                            FCMD16                ;RESUME TESTING ALL BANKS
045570  051124                            FCMD17                ;ENABLE TRACE
045572  051140                            FCMD18                ;DISABLE TRACE
045574                                  END ;OF CASE
045602  000733                          BR      FS1
```

```
11629 045604                                FSCMD0: SUBTST  <<COMMAND 0     EXIT>>
                                            ;*****************************************************************************
                                            ;*SUBTEST        COMMAND 0       EXIT
                                            ;*****************************************************************************
11630 045604                                        TYPE    MSG103          ;LEAVING FIELD SERVICE MODE
11631 045610  062706  000002                  ADD     #2,SP
11632 045614                                        IF SKIPKAMI IS TRUE
11633 045622  062706  000002                    ADD   #2,SP                   ;THROW AWAY OLD KAMIKAZE FLAG
11634 045626  005037  002006                    CLR   SKIPKAMI
11635 045632                                        ELSE
11636 045634                                      POP   KAMIKAZE              ;RESTORE OLD KAMIKAZE FLAG
11637 045640                                        END ;OF IF SKIPKAMI
11638 045640                                        POP     CSRNO
11639 045644  005737  002516                  TST     CACHKN
11640 045650  001414                          BEQ     RES0
11641 045652                                  IF CACHKN EQ CACHKF           ;IF CACHE IS OFF
11642 045662  062706  000002                    ADD   #2,SP                   ;THROW AWAY CACHE STATUS
11643 045666                                        ELSE
11644 045670  005737  002516                      TST   CACHKN
11645 045674  001402                              BEQ   RES0
11646 045676                                      POP   CONTRL                ;RESTORE CACHE STATUS
11647 045702                                        END ;OF IF CACHKN
11648 045702  104416                    RES0:   RESREG
11649 045704  000207                          RETURN
11650
11651 045706                                FSCMD1: SUBTST  <<FS    COMMAND 1       READ CSR>>
                                            ;*****************************************************************************
                                            ;*SUBTEST        FS      COMMAND 1       READ CSR
                                            ;*****************************************************************************
11652 045706  004737  051152                  CALL    WHICHCSR
11653 045712  010637  002270                  MOV     SP,FSSTACK
11654 045716                                  SET4    #RES1           ;TRAPS TO 4 GOTO RES1
11655 045724  104426                          READCSR
11656 045726                                  SET     NOERROR
11657 045734  104026                          ERROR   +26             ;USE ERROR ROUTINE FOR PRINTOUT
11658 045736                                  RES4                    ;RESET TRAPS TO 4 TO DEFAULT
11659 045760  000207                          RETURN
11660 045762                            RES1: TYPE    MSG025          ;THIS CSR DOES NOT EXIST
11661 045766  013706  002270                  MOV     FSSTACK,SP
11662 045772                                  RES4                    ;RESET TRAPS TO 4 TO DEFAULT
11663 046014  000207                          RETURN
```

```
11666 046016                          FSCMD2: SUBTST  <<FS    COMMAND 2       LOAD CSR>>
                                      ;*********************************************************************
                                      ;*SUBTEST        FS      COMMAND 2       LOAD CSR
                                      ;*********************************************************************
11667 046016  004737  051152                  CALL    WHICHCSR
11668 046022  010637  002270                  MOV     SP,FSSTACK
11669 046026                                  SET4    #RES2           ;TRAPS TO 4 GOTO RES2
11670 046034  104426                          READCSR
11671 046036                                  TYPE    MSG027
11672 046042                                  SET     NOERROR
11673 046050  104026                          ERROR   .26             ;USE ERROR ROUTINE FOR PRINTOUT
11674 046052                                  RES4                    ;RESET TRAPS TO 4 TO DEFAULT
11675 046074                                  TYPE    MSG023          ;FIRST CSR WORD
11676 046100  104413                          RDOCT                   ;READ AN OCTAL NUMBER
11677 046102                                  POP     CSR             ;PUT IN IN LOC 'CSR
11678 046106  104425                          LOADCSR
11679 046110  104426                          READCSR
11680 046112                                  TYPE    MSG028
11681 046116                                  SET     NOERROR
11682 046124  104026                          ERROR   .26             ;USE FOR PRINTOUT   NOT AN ERROR
11683 046126  000207                          RETURN
11684 046130                          RES2:   TYPE    MSG025          ;THIS CSR DOES NOT EXIST
11685 046134  013706  002270                  MOV     FSSTACK,SP
11686 046140                                  RES4                    ;RESET TRAPS TO 4 TO DEFAULT
11687 046162  000207                          RETURN
```

```
11690 046164                                FSCMD3: SUBTST <<FS    COMMAND 3       EXAMINE MEMORY>>
                                            ;*****************************************************************************
                                            ;*SUBTEST       FS      COMMAND 3       EXAMINE MEMORY
                                            ;*****************************************************************************
11691 046164                                        PUSH    BANK,NOPAR,PARTHERE,4
11692 046204  012737  000002  002074                MOV     #2,NOPAR            ;INDICATE PARITY ACTION
11693 046212                                        TYPE    MSG029              ;EXAMINE MEMORY
11694 046216                                1$:     TYPE    MSG031              ;PHYSICAL ADDRESS (0-17775776)??
11695 046222  104413                              RDOCT                       ;READ OCTAL NUMBER UNTO STACK & $HIOCT
11696 046224  013737  062376  002100              MOV     $HIOCT,BANK         ;PUT MSB'S IN BANK
11697 046232                                      POP     R0                  ;PUT LSB'S IN R0
11698 046234  000241                              CLC
11699 046236  006100                              ROL     R0
11700 046240  006137  002100                      ROL     BANK
11701 046244  000241                              CLC
11702 046246  006000                              ROR     R0
11703 046250  023737  002100  002530              CMP     BANK,LASTBANK       ;CHECK FOR BANK TOO HIGH
11704 046256  003357                              BGT     1$                  ;BRANCH IF TRUE
11705 046260  062700  060000                      ADD     #FIRST,R0
11706 046264  032700  000001                      BIT     #BIT0,R0            ;CHECK FOR ODD ADDRESS
11707 046270  001352                              BNE     1$                  ;BRANCH IF ODD ADDRESS
11708 046272  020027  157776                      CMP     R0,#LAST            ;CHECK FOR ADDRESS OVER 16K
11709 046276  101347                              BHI     1$                  ;BRANCH IF OVER 16K
11710 046300  012737  046352  002266              MOV     #3$,PARTHERE        ;INCASE OF ABORTS
11711 046306                                      SET4    #4$                 ;TRAPS TO 4 GOTO 4$
11712 046314                                      MAP     BANK                        ;MAP SUPERVISOR SPACE (TEST AREA) TO BANK
11713 046330                                      TESTAREA                    ;ENTER TEST MODE
11714 046336  011001                              MOV     (R0),R1
11715 046340  104417                              KERNEL                      ;ENTER KERNEL MODE
11716 046342                                      TYPOCS  R1
11717 046350  000410                              BR      EXCMD3
11718
11719 046352                                3$:     TYPE    MSG032                      ;PARITY ABORT
11720 046356  000405                              BR      EXCMD3
11721
11722 046360  062706  000004                4$:     ADD     #4,SP                       ;FIX STACK
11723 046364                                      TYPE    MSG033                      ;TIMEOUT TRAP
11724 046370  000400                              BR      EXCMD3
11725
11726 046372  104417                      EXCMD3: KERNEL                      ;ENTER KERNEL MODE
11727 046374                                      POP     4,PARTHERE,NOPAR,BANK
11728 046414                                      RES4                        ;RESET TRAPS TO 4 TO DEFAULT
11729 046436  000207                              RETURN
```

```
11732 046440                                FSCMD4: SUBTST  <<FS    COMMAND 4       MODIFY MEMORY>>
                                                    ;*****************************************************************************
                                                    ;*SUBTEST        FS      COMMAND 4       MODIFY MEMORY
                                                    ;*****************************************************************************
11733 046440                                        PUSH    BANK,NOPAR,PARTHERE,4
11734 046460  012737  000003  002074                MOV     #3,NOPAR                ;INDICATE PARITY ACTION
11735 046466                                        TYPE    MSG036                  ;MODIFY MEMORY
11736 046472                                1$:     TYPE    MSG031          ;PHYSICAL ADDRESS (O 17775776)??
11737 046476  104413                              RDOCT                   ;READ OCTAL NUMBER ONTO STACK & $HIOCT
11738 046500  013737  062376  002100              MOV     $HIOCT,BANK     ;PUT MSB'S IN BANK
11739 046506                                      POP     R0              ;PUT LSB'S IN R0
11740 046510  000241                              CLC
11741 046512  006100                              ROL     R0
11742 046514  006137  002100                      ROL     BANK
11743 046520  000241                              CLC
11744 046522  006000                              ROR     R0
11745 046524                                      IF BANK GT LASTBANK THEN GOTO 1$  ;CHECK FOR BANK TOO HIGH
11746 046534  062700  060000                      ADD     #FIRST,R0
11747 046540                                      IF #BIT0 SET.IN R0 THEN GOTO 1$ ;CHECK FOR ODD ADDRESS
11748 046546                                      IF R0 HI #LAST THEN GOTO 1$     ;CHECK FOR ADDRESS OVER 16K
11749 046554  012737  046622  002266              MOV     #3$,PARTHERE            ;INCASE OF ABORTS
11750 046562                                      SET4    #4$             ;TRAPS TO 4 GOTO 4$
11751 046570                                      MAP     BANK                    ;MAP SUPERVISOR SPACE (TEST AREA) TO BANK
11752 046604  104511                              INVALIDATE
11753 046606                                      TESTAREA                ;ENTER TEST MODE
11754 046614  011001                              MOV     (R0),R1
11755                                             ;GETTING HERE MEANS WE GOT LUCKY - NO TRAPS
11756 046616  104417                              KERNEL                  ;ENTER KERNEL MODE
11757 046620  000410                              BR      5$
11758
11759 046622                                3$:     TYPE    MSG032                  ;PARITY ABORT
11760 046626  000431                              BR      EXCMD4                  ;EXIT
11761
11762 046630  062706  000004                4$:     ADD     #4,SP           ;FIX STACK
11763 046634                                      TYPE    MSG033          ;TIMEOUT TRAP
11764 046640  000424                              BR      EXCMD4                  ;EXIT
11765
11766 046642                                5$:     TYPE    MSG037                  ;OLD DATA WAS
11767 046646                                      TYPOCS  R1                      ;PRINT IT
11768 046654                                      TYPE    MSG039                  ;INPUT NEW DATA
11769 046660  104413                              RDOCT                           ;READ ON OCTAL NUMBER ONTO THE STACK
11770 046662                                      POP     R1                      ;GET NEW NUMBER
11771 046664                                      TESTAREA                ;ENTER TEST MODE
11772 046672  010110                              MOV     R1,(R0)                 ;PUT IT IN MEMORY
11773 046674  011001                              MOV     (R0),R1                 ;READ IT AGAIN
11774 046676  104417                              KERNEL                  ;ENTER KERNEL MODE
11775 046700                                      TYPE    MSG038                  ;DATA IS NOW
11776 046704                                      TYPOCS  R1                      ;PRINT IT
11777
11778 046712  104417                        EXCMD4: KERNEL                  ;ENTER KERNEL MODE
11779 046714                                      POP     4,PARTHERE,NOPAR,BANK
11780 046734                                      RES4                    ;RESET TRAPS TO 4 TO DEFAULT
11781 046756  000207                              RETURN
```

```
11784 046760                        FSCMD5: SUBTST  <<FS    COMMAND 5       SELECT BANK & PATTERN..
                                    ;*****************************************************************
                                    ;*SUBTEST       FS      COMMAND 5       SELECT BANK & PATTERN
                                    ;*****************************************************************
11785 046760                                PUSH    BANK,PATTERN,TESTADD,PCBUMP,TKVEC,TKVEC+2
11786 047010  010637  002270                MOV     SP,FSSTACK              ;SAVE LAST GOOD STACK POINTER
11787 047014                                TYPE    MSG040                  ;SELECT BANK & PATTERN TEST
11788 047020                        18:     TYPE    MSG030                  ;BANK(0-177)?
11789 047024  104413                        RDOCT                           ;READ AN OCTAL NUMBER ONTO THE STACK
11790 047026                                POP     BANK                    ;PUT IT IN BANK
11791 047032                                IF BANK GT LASTBANK THEN GOTO 18  ;CHECK FOR BANK TOO HIGH
11792
11793 047042  013701  002100                MOV     BANK,R1
11794 047046  006301                        ASL     R1
11795 047050  006301                        ASL     R1
11796 047052                                IF CPUBIT OFF.IN CONFIG(R1)
11797 047062                                  TYPE  MSG041                  ;BANK NOT ACCESSABLE
11798 047066                                  GOTO 18
11799 047070                                END ;OF IF
11800
11801 047070                        28:     TYPE    MSG042                  ;PATTERN(0 35)?
11802 047074  104413                        RDOCT                           ;READ AN OCTAL NUMBER ONTO THE STACK
11803 047076                                POP     PATTERN                 ;PUT IT IN PATTERN
11804 047102                                IF PATTERN GT #35 THEN GOTO 28  ;CHECK FOR PATTERN TO HIGH
11805 047112                                IF PATTERN EQ #0
11806 047120                                  TYPE  MSG043                  ;PATTERN 0 DATA IS?
11807 047124  104413                          RDOCT                         ;READ AN OCTAL NUMBER ONTO THE STACK
11808 047126                                  POP   R2                      ;PUT IT IN R2
11809 047130                                END ;OF IF
11810
11811
11812 047130                                MAP     BANK                    ;MAP SUPERVISOR SPACE (TEST AREA) TO BANK
11813 047144  104511                        INVALIDATE
11814 047146  004737  044302                CALL    EXBANK                  ;SET NEW MARGINS
11815 047152                                IF RRFLAG IS TRUE
11816 047160                                  TYPE  MSG049                  ;BANK REQUIRES RELOCATION
11817 047164                                  GOTO CMDSC
11818 047166                                END ;OF IF RRFLAG
11819 047166                                TYPE    MSG046                  ;TO ESCAPE TYPE ANY KEY!
11820                               ;
11821                               ; Alter CsrNo to chosen Bank (This section added Rev D)         ;D
11822                               ;
11823 047172  013737  002146  002150        Mov     CsrNo,SavCsr            ;Save Old CSR Number
11824 047200  013702  002100                Mov     Bank,r2
11825 047204  072227  000002                Ash     #2,r2                   ;Generate index into Config table
11826 047210  016203  002626                Mov     Config(r2),r3           ;R3 - low word of config for this bank
11827 047214  072327  177770                Ash     #-10,r3                 ;Position the Csr Code to Bits 0   3
11828 047220  042703  177760                Bic     #^C17,r3                ;Clear all but CSR Code
11829 047224  006303                        Asl     r3                      ;Adjust
11830 047226  010337  002146                Mov     r3,CsrNo
11831                               ;
11832 047232  012737  047552  000060        MOV     #CMDSC,TKVEC
11833 047240  012737  000340  000062        MOV     #340,TKVEC+2
11834 047246  017700  133334                MOV     @#TKB,R0                ;KILL ANY OLD INTERRUPT
11835 047252  042737  000200  177776        BIC     #BIT7,PSW               ;LOWER CPU PRIORITY TO 140
11836 047260  052777  000100  133316        BIS     #BIT6,@#TKS             ;ENABLE KEYBOARD INTERRUPTS
11837
```

```
11838
11839 047266                                    SET     HEADER,MUT
11840 047302  013701  002100            CMD5B: MOV     BANK,R1
11841 047306  006301                            ASL     R1
11842 047310  006301                            ASL     R1
11843 047312  005037  002234                    CLR     SPLTCSR
11844 047316  005037  002260                    CLR     PASFLG
11845 047322  012737  060000  002364            MOV     @FIRST,TESTADD
11846 047330  012737  060002  002366            MOV     @FIRST+2,TESTADD+2
11847 047336                                    IF @BIT12 SET.IN CONFIG+2(R1)
11848 047346  005237  002234                      INC   SPLTCSR
11849 047352                                      MAP   BANK
11850 047366  012737  120000  002366              MOV   #120000,TESTADD+2
11851 047374                                    END; OF IF @BIT12
11852 047374                                    IF @SW0 SET.IN @SWR
11853 047404  104470                              ECCDIS                    ;DISABLE ERROR CORRECTION
11854 047406                                    ELSE
11855 047410                                      PUSH  CSRNO
11856 047414  104502                              CLRCSR                    ;CLEAR CSRS
11857 047416                                      POP   CSRNO
11858 047422                                    END ;OF IF
11859 047422  012737  000002  002074            MOV     #2,NOPAR            ;PARITY ACTION
11860 047430  012737  000002  002300            MOV     #2,PCBUMP           ;TRAPS ADD 2 TO PC
11861 047436  013700  002110                    MOV     PATTERN,R0
11862 047442  006300                            ASL     R0
11863 047444  004770  047456                    CALL    @FSPAT(R0)
11864 047450  005037  002074                    CLR     NOPAR
11865 047454  000712                            BR      CMD5B               ;LOOP TILL KEYBOARD INTERRUPT
11866
11867 047456  020124                    FSPAT:  MT0000  ;<1 SEC     DATA PATTERN TEST
11868 047460  020204                            MT0001  ;<1 SEC     ADDRESS TEST
11869 047462  020324                            MT0002  ;<1 SEC     COMPLEMENT ADDRESS TEST
11870 047464  020464                            MT0003  ; 1 SEC     3 XOR 9 WORST CASE NOISE TEST
11871 047466  020716                            MT0004  ; 1 SEC     ROTATING ZEROS TEST
11872 047470  021040                            MT0005  ; 1 SEC     ROTATING ONES TEST
11873 047472  021174                            MT0006  ;<1 SEC     INITIAL DATA TEST
11874 047474  021230                            MT0007  ;<1 SEC     ADDRESS BIT TEST
11875 047476  021272                            MT0010  ;<1 SEC     BYTE ADDRESSING TEST
11876 047500  021326                            MT0011  ;<2 SEC     CREATE SINGLE BIT ERROR TEST
11877 047502  021374                            MT0012  ;<1 SEC     WRITE BYTE CLEARS SBE TEST
11878 047504  021470                            MT0013  ; 1 SEC     CREATE DOUBLE BIT ERROR TEST
11879 047506  021544                            MT0014  ; 1 SEC     WRITE INHIBIT DURING DATIP WITH DBE
11880 047510  021622                            MT0015  ; 1 SEC     WRITE INHIBIT OF BYTE WITH DBE
11881 047512  021670                            MT0016  ;<1 SEC     WRITE INHIBIT OF WORD WITH DBE
11882 047514  021736                            MT0017  ;<1 SEC     HOLDING 1'S & 0'S TEST
11883 047516  021760                            MT0020  ;<1 SEC     MARCHING 1'S & 0'S IN CHECK BITS
11884 047520  023050                            MT0021  ; 1 SEC     MARCHING 0'S & 1'S TEST
11885 047522  023322                            MT0022  ;10 SEC     REFRESH & SHIFTING DIAGONAL TEST
11886 047524  023354                            MT0023  ;10 SEC     SHIFTING DIAGONAL TEST
11887 047526  023420                            MT0024  ;20 SEC     FAST GALLOPING PATTERN TEST
11888 047530  023664                            MT0025  ;<1 SEC     INTERRUPT ENABLE TEST
11889 047532  023732                            MT0026  ;<1 SEC     RANDOM DATA TEST
11890 047534  024234                            MT0027  ; 1 SEC     UNIQUE BANK TEST
11891 047536  024720                            MT0030  ; 1 SEC     FLUSH OUT DBE'S TEST
11892 047540  025222                            MT0031  ; 3 SEC     SOB-A-LONG TEST
11893 047542  025412                            MT0032  ;<1 SEC     WRITE RECOVERY TEST
11894 047544  025744                            MT0033  ;35 SEC     BRANCH GOBBLE TEST
```

```
11895 047546  026132                        MT0034  ; 1 SEC          SOFT ERROR TEST
11896 047550  026304                        MT0035  ;<1 SEC          WORST CASE NOISE PARITY TEST
11897
11898 047552  013706  002270          CMD5C:  MOV     FSSTACK,SP              ;RECOVER OLD STACK POINTER
11899 047556  042777  000100  133020        BIC     #BIT6,@#TKS
11900 047564                                POP     TKVEC+2,TKVEC
11901 047574  117700  133006                MOVB    @#TKB,R0                ;GET CHARACTER TO GET RID OF FLAG
11902 047600                                POP     PCBUMP,TESTADD
11903 047610                                POP     PATTERN,BANK
11904 047620                                MAP     BANK                    ;REMAP OLD BANK
11905 047634  004737  044302                CALL    EXBANK
11906                               ;
11907                               ; Restore CsrNo (Addition to Rev D)                     ;D
11908                               ;
11909 047640  013737  002150  002146        Mov     SavCsr,CsrNo
11910                               ;
11911 047646  000207                        RETURN
```

```
11913 047650                        FSCMD6: SUBTST <<FS    COMMAND 6       TYPE CONFIGURATION MAP>>
                                    ;*****************************************************************
                                    ;*SUBTEST       FS      COMMAND 6       TYPE CONFIGURATION MAP
                                    ;*****************************************************************
11914 047650  004737  036632                CALL    PCONFIG
11915 047654  000207                RETURN
11916
```

```
11919 047656                                FSCMD7: SUBTST  <<FS    COMMAND 7       SOB-A-LONG TEST>>
                                                    ;*****************************************************************
                                                    ;*SUBTEST       FS      COMMAND 7       SOB-A LONG TEST
                                                    ;*****************************************************************
11920 047656                                        PUSH    BANK,PATTERN,TKVEC,TKVEC+2,NOPAR
11921 047702  010637  002270                        MOV     SP,FSSTACK              ;SAVE LAST GOOD STACK POINTER
11922 047706                                        TYPE    MSG055                  ;SOB-A LONG TEST
11923
11924 047712                                        IF @SW0 SET.IN @SWR
11925 047722  104470                                  ECCDIS                        ;DISABLE ERROR CORRECTION
11926 047724                                        ELSE
11927 047726  104502                                  CLRCSR                        ;CLEAR CSRS
11928 047730                                        END ;OF IF
11929 047730                                        TYPE    MSG056                  ;BELL * EACH PASS COMPLETE
11930
11931 047734                                        TYPE    MSG046                  ;TO ESCAPE TYPE ANY KEY!
11932 047740  012737  050064  000060                MOV     #CMD7C,TKVEC
11933 047746  012737  000340  000062                MOV     #340,TKVEC+2
11934 047754  017700  132626                        MOV     @#TKB,R0                ;KILL ANY OLD INTERRUPT
11935 047760  042737  000200  177776                BIC     #BIT7,PSW               ;LOWER CPU PRIORITY TO 140
11936 047766  052777  000100  132610                BIS     #BIT6,@#TKS             ;ENABLE KEYBOARD INTERRUPTS
11937
11938
11939 047774                                        SET     HEADER,MUT
11940
11941 050010                                CMD7B:  FOR BANK := #0 TO LASTBANK
11942 050014  004737  044302                          CALL EXBANK
11943 050020                                          IF ACFLAG IS TRUE AND RRFLAG IS FALSE
11944 050034  104511                                    INVALIDATE
11945 050036  004737  025222                            CALL MT0031
11946 050042                                          END ;OF IF ACFLAG
11947 050042                                        END ;OF FOR BANK
11948 050056                                        TYPE    $BELL                   ;RING BELL
11949 050062                                        GOTO    CMD7B
11950
11951 050064  013706  002270                CMD7C:  MOV     FSSTACK,SP              ;RECOVER OLD STACK POINTER
11952 050070  042777  000100  132506                BIC     #BIT6,@#TKS
11953 050076  117700  132504                        MOVB    @#TKB,R0                ;READ CHAR TO KILL FLAG
11954 050102                                        POP     NOPAR,TKVEC+2,TKVEC,PATTERN,BANK
11955 050126                                        MAP     BANK                    ;MAP SUPERVISOR SPACE (TEST AREA) TO BANK
11956 050142  004737  044302                        CALL    EXBANK
11957 050146  000207                                RETURN
```

```
 11960 050150                                FSCMD8: SUBTST  <<FS    COMMAND 8       ERROR SUMMARY>>
                                             ;*****************************************************************************
                                             ;*SUBTEST        FS      COMMAND 8       ERROR SUMMARY
                                             ;*****************************************************************************
 11961 050150                                        PUSH    R0,R2,R3,BANK
 11962 050162  013737  063104  002406              MOV     $PASS,TEMP
 11963 050170  005337  002406                      DEC     TEMP
 11964 050174                                      TYPDEC  TEMP
 11965 050202                                      TYPE    MSG125                  ;PASSES COMPLETED
 11966 050206                                      TYPDEC  $ERTTL
 11967 050214                                      TYPE    MSG079                  ;ERROR(S) DETECTED
 11968 050220                                      IF $ERTTL NE #0
 11969 050226  005037  002306                        CLR   SUCCESS
 11970 050232                                        FOR BANK := #0 TO LASTBANK
 11971 050236  013703  002100                          MOV BANK,R3
 11972 050242  070327  000004                          MUL #4,R3
 11973 050246                                          IFB CONFIG+2(R3) NE #0
 11974 050254                                            IF SUCCESS IS FALSE
 11975 050262                                              TYPE    MSG076        ;BANK  ERRORS
 11976 050266                                              SET SUCCESS
 11977 050274                                            END ;OF IF SUCCESS
 11978 050274                                            TYPOCS  BANK,3
 11979 050304  116300  002630                            MOVB    CONFIG+2(R3),R0
 11980 050310  042700  177400                            BIC     #^C377,R0
 11981 050314                                            TYPDEC  R0
 11982 050320                                            TYPE    $CRLF
 11983 050324                                          END ;OF IFB CONFIG(R3)
 11984 050324                                        END ;OF FOR BANK
 11985 050340                                      END ;OF IF $ERTTL
 11986 050340                                      POP     BANK,R3,R2,R0
 11987 050352  000207                              RETURN
```

```
11990 050354                                    FSCMD9: SUBTST ;<FS    COMMAND 9       REFRESH TEST>
                                                ;*****************************************************************************
                                                ;*SUBTEST       FS      COMMAND 9       REFRESH TEST
                                                ;*****************************************************************************
11991 050354                                            PUSH    BANK,PATTERN,TKVEC,TKVEC+2,NOPAR
11992 050400  010637  002270                            MOV     SP,FSSTACK              ;SAVE LAST GOOD STACK POINTER
11993 050404                                            TYPE    MSG073                  ;REFRESH TEST
11994
11995 050410                                            IF @SW0 SET.IN @SWR
11996 050420  104470                                      ECCDIS                        ;DISABLE ERROR CORRECTION
11997 050422                                            ELSE
11998 050424  104502                                      CLRCSR                        ;CLEAR CSRS
11999 050426                                            END ;OF IF
12000 050426                                            TYPE    MSG056                  ;BELL + EACH PASS COMPLETE
12001
12002 050432                                            TYPE    MSG046                  ;TO ESCAPE TYPE ANY KEY!
12003 050436  012737  050562  000060                    MOV     #CMD9C,TKVEC
12004 050444  012737  000340  000062                    MOV     #340,TKVEC+2
12005 050452  017700  132130                            MOV     @#TKB,R0                ;KILL ANY OLD INTERRUPT
12006 050456  042737  000200  177776                    BIC     #BIT7,PSW               ;LOWER CPU PRIORITY TO 140
12007 050464  052777  000100  132112                    BIS     #BIT6,@#TKS             ;ENABLE KEYBOARD INTERRUPTS
12008
12009 050472                                            SET     HEADER,MUT
12010
12011 050506                                    CMD9B:  FOR BANK := #0 TO LASTBANK
12012 050512  004737  044302                              CALL EXBANK
12013 050516                                              IF ACFLAG IS TRUE AND RRFLAG IS FALSE
12014 050532  104511                                        INVALIDATE
12015 050534  004737  023322                                CALL MT0022
12016 050540                                              END ;OF IF ACFLAG
12017 050540                                            END ;OF FOR BANK
12018 050554                                            TYPE    #BELL                   ;RING BELL
12019 050560                                            GOTO    CMD9B
12020
12021 050562  013706  002270                    CMD9C:  MOV     FSSTACK,SP              ;RECOVER OLD STACK POINTER
12022 050566  042777  000100  132010                    BIC     #BIT6,@#TKS
12023 050574  117700  132006                            MOVB    @#TKB,R0                ;READ CHAR TO KILL FLAG
12024 050600                                            POP     NOPAR,TKVEC+2,TKVEC,PATTERN,BANK
12025 050624                                            MAP     BANK                    ;MAP SUPERVISOR SPACE (TEST AREA) TO BANK
12026 050640  004737  044302                            CALL    EXBANK
12027 050644  000207                                    RETURN
12028
```

```
12031 050646                        FCMD10: SUBTST  <<FS    COMMAND 10      SET FILL COUNT>>
                                    ;*****************************************************************
                                    ;*SUBTEST        FS      COMMAND 10      SET FILL COUNT
                                    ;*****************************************************************
12032 050646                                PUSH    R0
12033 050650                                TYPE    MSG085                  ;FILL COUNT(OCTAL)?
12034 050654  104413                          RDOCT
12035 050656                                POP     R0
12036 050660  042700  177760                BIC     #^C17,R0
12037 050664  110037  002331                MOVB    R0,@FILLS
12038 050670                                POP     R0
12039 050672  000207                        RETURN
12040
12041 050674                        FCMD11: SUBTST  <<FS    COMMAND 11      ENTER KAMIKAZE MODE>>
                                    ;*****************************************************************
                                    ;*SUBTEST        FS      COMMAND 11      ENTER KAMIKAZE MODE
                                    ;*****************************************************************
12042 050674                                TYPE    MSG101          ;ENTERING KAMIKAZE MODE
12043 050700                                SET     KAMIKAZE,SKIPKAMI
12044 050714  000207                        RETURN
12045
12046 050716                        FCMD12: SUBTST  <<FS    COMMAND 12      EXIT KAMIKAZE MODE>>
                                    ;*****************************************************************
                                    ;*SUBTEST        FS      COMMAND 12      EXIT KAMIKAZE MODE
                                    ;*****************************************************************
12047 050716                                TYPE    MSG102          ;LEAVING KAMIKAZE MODE
12048 050722  005037  002004                CLR     KAMIKAZE
12049 050726                                SET     SKIPKAMI
12050 050734  000207                        RETURN
12051
12052 050736                        FCMD13: SUBTST  <<FS    COMMAND 13      TURN CACHE OFF>>
                                    ;*****************************************************************
                                    ;*SUBTEST        FS      COMMAND 13      TURN CACHE OFF
                                    ;*****************************************************************
12053 050736                                TYPE    MSG106                  ;CACHE IS OFF
12054 050742  104424                        CACHOFF                         ;TURN CACHE OFF
12055 050744  013737  002516  002520        MOV     CACHKN,CACHKN+2         ;SAVE OLD CACHE ON STATE
12056 050752  005037  002516                CLR     CACHKN                  ;KEEP CACHE OFF
12057 050756  000207                        RETURN
12058
12059 050760                        FCMD14: SUBTST  <<FS    COMMAND 14      TURN CACHE ON>>
                                    ;*****************************************************************
                                    ;*SUBTEST        FS      COMMAND 14      TURN CACHE ON
                                    ;*****************************************************************
12060 050760                                TYPE    MSG107                  ;CACHE IS ON (EXCEPT DURING ACTUAL PATTERNS)
12061 050764  013737  002520  002516        MOV     CACHKN+2,CACHKN         ;RESTORE OLD CACHE ON STATE
12062 050772  104423                        CACHON                          ;TURN CACHE ON
12063 050774  000207                        RETURN
12064
```

```
12077
12078 050776                          FCMD15: SUBTST  <<FS    COMMAND 15      TEST ONLY SELECTED BANKS>>
                                      ;*****************************************************************
                                      ;*SUBTEST       FS      COMMAND 15      TEST ONLY SELECTED BANKS
                                      ;*****************************************************************
12079 050776                                  TYPE    MSG105          ;ENTER BANKS IN OCTAL  USE NUMBER OUTSIDE RANGE TO TERMINAT
E (177)
12080 051002  004737  051072                  CALL    CMD16A                  ;ERASE OLD SELECTIONS
12081 051006                                  BEGIN CMD16LOOP
12082 051006                                    REPEAT
12083 051006                                      TYPE        MSG030          ;BANK(0 177)?
12084 051012  104413                              RDOCT                       ;READ AN OCTAL NUMBER ONTO THE STACK
12085 051014                                      POP R1                      ;PUT IT IN R1
12086 051016                                      IF R1 GT #177 OR R1 LT #0
12087 051030                                        LEAVE CMD16LOOP
12088 051032                                      END ;OF IF R1
12089 051032  006301                              ASL  R1
12090 051034  006301                              ASL  R1                     ;R1 <  R1 * 4
12091 051036  052761  040000  002630              BIS #BIT14,CONFIG+2(R1)
12092 051044                                    END ;OF REPEAT
12093 051046                                  END CMD16LOOP
12094 051046                                  TYPE    MSG110                  ;ONLY SELECTED BANKS WILL BE TESTED
12095 051052                                  SET     SELONLY
12096 051060  000207                          RETURN
12097
12098 051062                          FCMD16: SUBTST  <<FS    COMMAND 16      RESUME TESTING ALL BANKS>>
                                      ;*****************************************************************
                                      ;*SUBTEST       FS      COMMAND 16      RESUME TESTING ALL BANKS
                                      ;*****************************************************************
12099 051062                                  TYPE    MSG111                  ;ALL BANKS WILL BE TESTED
12100 051066  005037  002000                  CLR     SELONLY
12101
12102                                         ;ENTRY POINT FROM CMD15
12103 051072  013702  002530          CMD16A: MOV     LASTBANK,R2
12104 051076  006302                          ASL     R2
12105 051100  006302                          ASL     R2
12106 051102                                  FOR R1 := #0 TO R2 BY #4
12107 051104  042761  040000  002630            BIC   #BIT14,CONFIG+2(R1)
12108 051112                                  END ;OF FOR R1
12109 051122  000207                          RETURN
```

```
 12112 051124                          FCMD17: SUBTST  <<FS    COMMAND 17      ENABLE TRACE>>
                                       ;*****************************************************************
                                       ;*SUBTEST       FS      COMMAND 17      ENABLE TRACE
                                       ;*****************************************************************
 12113 051124                                 TYPE    MSG127
 12114 051130  012737  177777  006140         MOV     #1,TRACE
 12115 051136  000207                         RETURN
```

```
12118 051140                          ;CMD18: SUBTST <<FS    COMMAND 18      DISABLE TRACE>>
                                      ;*********************************************************************
                                      ;*SUBTEST        FS      COMMAND 18      DISABLE TRACE
                                      ;*********************************************************************
12119 051140                                  TYPE    MSG128
12120 051144  005037  006140                  CLR     TRACE
12121 051150  000207                          RETURN
```

```
12124 051152                                    WHICHCSR:SUBTST <<SUBR  DETERMINE CORRECT CSR>>
                                                ;*****************************************************************
                                                ;*SUBTEST        SUBR    DETERMINE CORRECT CSR
                                                ;*****************************************************************
12125 051152  013700  002220                            MOV     TOTCSRS,R0      ;GET CSR'S FLAG
12126 051156  022700  100000                            CMP     #BIT15,R0       ;CSR 0?
12127 051162  001003                                    BNE     1$              ;NO   SKIP
12128 051164  005037  002146                            CLR     CSRNO           ;YES   SET IT UP
12129 051170  000207                                    RETURN
12130
12131 051172                                    1$:     TYPE    MSG022          ;WHICH CSR(O F)
12132 051176  104412                                    RDLIN                   ;GET CHARACTER
12133 051200                                            POP     R0              ;PUT IN R0
12134 051202  011000                                    MOV     (R0),R0         ;PUT CHAR IN R0
12135 051204  020027  000106                            CMP     R0,#106         ;CHECK LIMIT
12136 051210  101370                                    BHI     1$              ;IF BAD LOOP TILL HE TYPES IT RIGHT
12137 051212  022700  000101                            CMP     #'A,R0
12138 051216  103002                                    BHIS    2$
12139 051220  162700  000007                            SUB     #7,R0
12140 051224  162700  000060                    2$:     SUB     #60,R0
12141 051230  006300                                    ASL     R0
12142 051232  010037  002146                            MOV     R0,CSRNO
12143 051236  000207                                    RETURN
```

```
12692                                           .SBTTL  ERROR DATA (SUPERVISOR) SETUP STUFF
12693 051240                            $PER25: LET ADDRESS := R1    #2
12694 051252                                    IF ABORTFLAG IS FALSE
12695 051260                                      TESTAREA                  ;ENTER TEST MODE
12696 051266                                      LET BAD := -2(R1)
12697 051274  104417                              KERNEL                    ;ENTER KERNEL MODE
12698 051276                                    END ;OF IF ABORTFLAG
12699 051276                                    IF 177654 EQ #0
12700 051304                                      LET GOOD := R2
12701 051310                                    ELSE
12702 051312                                      LET GOOD := R3
12703 051316                                    END ;OF IF
12704 051316  000137  054520                    JMP     PERRAW
12705
12706 051322                            PERRA3: SUBTST  <<DATA WAS 3 WORDS>>
                                        ;*************************************************************************
                                        ;*SUBTEST       DATA WAS 3 WORDS
                                        ;*************************************************************************
12707 051322                                    IF BADPC EQ #0 THEN $CALL BADSTACK
12708 051334                                    PUSH    R0
12709 051336  005037  002144                    CLR     CSR               ;MAKE SURE CSR BIT HOLDER IS CLEAR
12710 051342  104505                            CHK1DIS                   ;DISABLE ECC & WRITE CHECKBITS FROM 1 SELECTED CSR
12711 051344                                    TESTAREA
12712 051352  005711                            TST     (R1)              ;READ LOCATION TO READ CHECKBITS INTO CSR
12713 051354  104417                            KERNEL
12714 051356  104426                            READCSR                   ;GET CSR CONTENTS
12715 051360  013700  002144                    MOV     CSR,R0            ;SAVE CSR CONTENTS IN R0
12716 051364  104503                            CLR1CSR                   ;RETURN CSR TO NORMAL MODE
12717 051366  072027  177773                    ASH     #-5,R0            ;MOVE CHECK BITS TO BOTTOM OF WORD
12718 051372  042700  177600                    BIC     #^C177,R0         ;CLEAR OFF EXTRANEOUS GARBAGE
12719 051376                                    LET ADDRESS := R1         ;SAVE VIRTUAL ADDRESS FOR PRINTOUT
12720 051402  005037  002042                    CLR     GOOD              ;FIRST TEST WORD WRITTEN SHOULD ALWAYS BE ZERO
12721 051406                                    TESTAREA                  ;ENTER TEST MODE
12722 051414  011137  002050                    MOV     (R1),BAD          ;GET BAD DATA FROM MUT   FIRST WORD
12723 051420  011437  002052                    MOV     (R4),BAD2         ;AND SECOND WORD
12724 051424  104417                            KERNEL                    ;ENTER KERNEL MODE
12725 051426  110037  002054                    MOVB    R0,BAD3           ;MOVE BAD CHECKBITS FOR PRINTOUT
12726 051432  105037  002055                    CLRB    BAD3+1            ;CLEAR OFF THE OTHER UNUSED BITS
12727 051436  004737  054754                    CALL    PERBNK            ;MARK BANK AS BAD IN CONFIG TABLE
12728 051442  104033                            ERROR   +33
12729 051444                                    POP     R0                ;RESTORE R0
12730 051446                                    IF #SW0 SET.IN @SWR
12731 051456  104506                              ENASBE                  ;TRAP ON SINGLE BIT ERRORS
12732 051460                                    ELSE
12733 051462  104472                              ECCINIT                 ;TRAP ON UNCORRECTABLE ERRORS
12734 051464                                    END; OF IF #SW0
12735 051464  000002                            RTI
```

DATA WAS 3 WORDS

```
12738 051466                                  $PER30: LET GOOD := R1
12739 051472                                          LET ADDRESS := (SP)+16
12740 051504                                          IF ABORTFLAG IS FALSE
12741 051512                                            TESTAREA                ;ENTER TEST MODE
12742 051520                                            LET BAD := BADDRESS
12743 051526  104417                                    KERNEL                  ;ENTER KERNEL MODE
12744 051530                                          END ;OF IF ABORTFLAG
12745 051530  000137  054520                          JMP PERRAW
12746
12747 051534                                  GETDATA:SUBTST  <<GET DATA FROM ABORTED AREA IF POSSIBLE>>
                                              ;*****************************************************************
                                              ;*SUBTEST       GET DATA FROM ABORTED AREA IF POSSIBLE
                                              ;*****************************************************************
12748 051534                                          PUSH    R0,4,114
12749 051546  010637  051632                          MOV     SP,GETDA1
12750 051552  012737  051612  000004                  MOV     #18.,4
12751 051560  012737  051612  000114                  MOV     #18.,114
12752 051566  013700  002032                          MOV     ADDRESS,R0
12753 051572                                          TESTAREA
12754 051600  011037  002050                          MOV     (R0),BAD
12755 051604  104417                                  KERNEL
12756 051606  005037  002140                          CLR     ABORTFLAG
12757 051612  013706  051632                  18:     MOV     GETDA1,SP               ;RESTORE KNOWN GOOD STACK POINTER
12758 051616                                          POP     114,4,R0
12759 051630  000207                                  RETURN
12760 051632  000000                          GETDA1: 0
```

```
                                        .SBTTL  POWER FAIL AUTO RESTART
                                        .SBTTL  ROUTINE POWER DOWN AND UP
                                ;***************************************************************************
                                ;POWER DOWN ROUTINE
051634                          $PWRDN:
                                        ;SAVE CACHE STATUS
051634  005737  002516                  TST     CACHKN
051640  001403                          BEQ     5$
051642                                  PUSH    CONTRL
051646  104423                          CACHON                          ;TURN CACHE ON
051650  012737  052606  000024  5$:     MOV     #$ILLUP,PWRVEC  ;;SET FOR FAST UP
051656  012737  000340  000026          MOV     #340,PWRVEC+2   ;;PRIO:7
051664                                  PUSH    R0,R1,R2,R3,R4,R5,CSRNO
                                        ;SAVE USER PAR'S & PDR7
051704  012700  177700                  MOV     #177700,R0
051710  012701  000021                  MOV     #17.,R1
051714                          1$:     PUSH    -(R0)
051716  077102                          SOB     R1,1$
                                        ;SAVE SUPERVISOR PAR'S
051720  005737  002430                  TST     NOSUPER
051724  001013                          BNE     PD1
051726  012700  172300                  MOV     #172300,R0
051732  012701  000020                  MOV     #16.,R1
051736                          2$:     PUSH    -(R0)
051740  077102                          SOB     R1,2$
051742                                  IF RLFLAG IS TRUE THEN $CALL WOOPS
                                        ;COPY KERNEL MAP TO USER & SUPERVISOR
051754  012700  172300          PD1:    MOV     #KIPDR0,R0
051760  012701  177600                  MOV     #UIPDR0,R1
051764  012702  172200                  MOV     #SIPDR0,R2
051770  012703  000040                  MOV     #32.,R3
051774  011021                  3$:     MOV     (R0),(R1)+
051776  012022                          MOV     (R0)+,(R2)+
052000  077303                          SOB     R3,3$
```

```
12805
12806 052002
12807 052010  010600                    ;SAVE USER & SUPERVISOR STACK POINTERS
12808 052012  104417                    USER
12809 052014                            MOV     USP,R0
12810 052016  005737  002430            KERNEL                  ;ENTER KERNEL MODE
12811 052022  001006                    PUSH    R0
12812 052024                            TST     NOSUPER
12813 052032  010600                    BNE     7$
12814 052034  104417                    SUPERVISOR              ;ENTER SUPERVISOR MODE
12815 052036                            MOV     SSP,R0
12816                                   KERNEL                  ;ENTER KERNEL MODE
12817 052040  013701  002220      7$:   PUSH    R0
12818 052044                            ;SAVE ECC REGISTERS
12819 052044                            MOV     TOTCSRS,R1      ;GET CSR'S
12820 052050  006301                    BEGIN   LCSRSAVE
12821 052052                              FOR CSRNO := #0 TO #36 BY #2
12822 052054  104426                        ASL     R1
12823 052056                                ON.ERROR
12824 052062                                  READCSR
12825 052062                                  PUSH    CSR
12826 052066                                END ;OF ON.ERROR
12827 052104                              IF R1 EQ #0 THEN LEAVE LCSRSAVE
12828                                     END ;OF FOR CSRNO
12829 052104                            END LCSRSAVE
12830 052120  005737  002430            ;SAVE MMR0,1,2,3
12831 052124  001002                    PUSH    MMR0,MMR1,MMR2
12832 052126                            TST     NOSUPER
12833                                   BNE     8$
12834 052132  012700  172400      8$:   PUSH    MMR3
12835 052136  012701  000020            ;SAVE KERNEL PAR'S
12836 052142                      4$:   MOV     #172400,R0
12837 052144  077102                    MOV     #16.,R1
12838                                   PUSH    -(R0)
12839 052146  022737  000001  003720    SOB     R1,4$
12840 052154  001004                    ;SAVE UNIBUS MAP REGISTERS
12841 052156                            CMP     #1,PROTYP               ;IS THIS AN 11/44?
12842                                   BNE     9$                      ;BRANCH IF NOT
12843 052166                      9$:   PUSH    MAPHO,MAPLO
12844                                   ;SAVE POSSIBLE SOFTWARE SWITCH REGISTER
12845 052172  010637  052612            PUSH    @SWR
12846                                   ;SAVE STACK POINTER
12847 052176  012737  052210  000024    MOV     SP,$SAVR6       ;;SAVE SP
12848 052204  000000              $DOWN:  ;NOW SET UP REAL VECTOR
12849 052206  000776                    MOV     #$PWRUP,PWRVEC  ;;SET UP VECTOR
                                        HALT
                                        BR      $DOWN           ;;HANG UP
```

```
12852                                   ;*****************************************************************************
12853                                   ;POWER UP ROUTINE
12854 052210                            $PWRUP:
12858 052210  012737  052606  000024              MOV     #$ILLUP,PWRVEC ;;SET UP FOR FAST DOWN
12859                                   ;RESTORE STACK POINTER
12860 052216  013706  052612                      MOV     $SAVR6,SP       ;;GET SP
12861 052222  005037  052612                      CLR     $SAVR6          ;;WAIT LOOP FOR THE "1"
12862 052226  005237  052612            1$:       INC     $SAVR6          ;;WAIT FOR THE INC
12863 052232  001375                              BNE     1$              ;;OF A WORD
12864                                   ;RESTORE POSSIBLE SOFTWARE SWITCH REGISTER
12865 052234                                      POP     @SWR
12866                                   ;RESTORE UNIBUS MAP
12867 052240  022737  000001  003720              CMP     #1,PROTYP               ;IS THIS AN 11/44?
12868 052246  001006                              BNE     10$
12869 052250                                      POP     MAPLO,MAPHO
12870 052260  004737  043706                      CALL    LOWMAP          ;SETUP LOWER 16K OF UNIBUS MAP
12871                                   ;RESTORE KERNEL PAR'S & PDR'S
12872 052264  012700  172340            10$:      MOV     #172340,R0
12873 052270  012702  172300                      MOV     #KIPDR0,R2
12874 052274  012701  000020                      MOV     #16.,R1
12875 052300                            6$:       POP     (R0)+
12876 052302  012722  077406                      MOV     #77406,(R2)+
12877 052306  077104                              SOB     R1,6$
12878                                   ;RESTORE MMR3,2,1,0
12879 052310  005737  002430                      TST     NOSUPER
12880 052314  001002                              BNE     11$
12881 052316                                      POP     MMR3
12882 052322                            11$:      POP     MMR2,MMR1,MMR0
12883                                   ;RESTORE ECC REGISTERS
12884 052336  013701  002220                      MOV     TOTCSRS,R1      ;GET CSR'S
12885 052342  042701  177400                      BIC     #177400,R1
12886 052346                                      BEGIN LCSRRESTORE
12887 052346                                        FOR CSRNO := #36 DOWNTO #0 BY #2
12888 052354  006201                                  ASR         R1
12889 052356                                          ON.ERROR
12890 052360                                            POP         CSR
12891 052364  104425                                    LOADCSR
12892 052366                                          END ;OF ON.ERROR
12893 052366                                        IF R1 EQ #0 THEN LEAVE LCSRRESTORE
12894 052372                                        END ;OF FOR CSRNO
12895 052410                                      END LCSRRESTORE
12896                                   ;COPY KERNEL MAP TO USER & SUPERVISOR
12897 052410  012700  172300                      MOV     #KIPDR0,R0
12898 052414  012701  177600                      MOV     #UIPDR0,R1
12899 052420  012702  172200                      MOV     #SIPDR0,R2
12900 052424  012703  000040                      MOV     #32.,R3
12901 052430  011021                    3$:       MOV     (R0),(R1)+
12902 052432  012022                              MOV     (R0)+,(R2)+
12903 052434  077303                              SOB     R3,3$
```

```
12905                                                   ;RESTORE SUPERVISOR & USER STACK POINTERS
12906 052436  005737  002430                            TST     NOSUPER
12907 052442  001006                                    BNE     13$
12908 052444                                            POP     R0
12909 052446                                            SUPERVISOR              ;ENTER SUPERVISOR MODE
12910 052454  010006                                    MOV     R0,SSP
12911 052456  104417                                    KERNEL                  ;ENTER KERNEL MODE
12912 052460                                    13$:    POP     R0
12913 052462                                            USER
12914 052470  010006                                    MOV     R0,USP
12915 052472  104417                                    KERNEL                  ;ENTER KERNEL MODE
12916                                                   ;RESTORE SUPERVISOR PAR'S
12917 052474  012700  172240                            MOV     #172240,R0
12918 052500  012701  000020                            MOV     #16.,R1
12919 052504                                    7$:     POP     (R0)+
12920 052506  077102                                    SOB     R1,7$
12921                                                   ;RESTORE USER PAR'S & PDR7
12922 052510  012700  177636                            MOV     #177636,R0
12923 052514  012701  000021                            MOV     #17.,R1
12924 052520                                    8$:     POP     (R0)+
12925 052522  077102                                    SOB     R1,8$
12926                                                   ;RESTORE POSSIBLE SOFTWARE DISPLAY REGISTER
12927 052524  013777  002010  130050                    MOV     @PATMAR,@DISPLAY
12928 052532  013737  002010  000174                    MOV     @PATMAR,DISPREG
12929 052540                                            POP     CSR60,R5,R4,R3,R2,R1,R0
12930 052560  012737  051634  000024                    MOV     #$PWRDN,PWRVEC  ;;SET UP THE POWER DOWN VECTOR
12931 052566                                            TYPE    MSG051          ;REPORT THE POWER FAILURE
12932                                                   ;RESTORE CACHE STATUS
12933 052572  005737  002516                            TST     CACHKN
12934 052576  001402                                    BEQ     9$
12935 052600                                            POP     CONTRL
12936 052604  000002                                    RTI
12937 052606  000000                            $ILLUP: HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
12938 052610  000776                                    BR      $ILLUP          ;; BEFORE THE POWER DOWN WAS COMPLETE
12939 052612  000000                            $SAVR6: 0                       ;;PUT THE SP HERE
12940                                                   .EVEN
```

```
12952 052614                                    WOOPS:  SUBTST  <<POWER FAIL WHILE RELOCATED>>
                                                ;*****************************************************************
                                                ;*SUBTEST        POWER FAIL WHILE RELOCATED
                                                ;*****************************************************************
12953 052614                                            PUSH    BANK
12954 052620  005037  002100                              CLR     BANK
12955 052624                                            MAP     BANK                    ;MAP SUPERVISOR SPACE (TEST AREA) TO BANK
12956 052640                                            SUPERVISOR                      ;ENTER SUPERVISOR MODE
12957 052646  013737  060024  053212                    MOV     FIRST+PWRVEC,WOOPSAV
12958 052654  013737  060026  053214                    MOV     FIRST+PWRVEC+2,WOOPSAV+2
12959 052662                                            BMOV    FIRST+WOOPUP,WOOPSAV+4,WOOPEND WOOPUP/2+12.
12960 052674  012737  053000  060024                    MOV     #WOOPUP,FIRST+PWRVEC
12961 052702  012737  000340  060026                    MOV     #340,FIRST+PWRVEC+2
12962 052710                                            BMOV    WOOPUP,FIRST+WOOPUP,WOOPEND WOOPUP/2
12963 052722  012700  172340                            MOV     #KIPAR0,R0
12964 052726  012701  133162                            MOV     #FIRST+WOOPEND,R1
12965 052732  012702  000010                            MOV     #8.,R2
12966 052736  012021                              1$:   MOV     (R0)+,(R1)+
12967 052740  077202                                    SOB     R2,1$
12968 052742  005737  002430                            TST     NOSUPER
12969 052746  001002                                    BNE     2$
12970 052750  013721  172516                            MOV     MMR3,(R1)+
12971 052754  013721  177576                      2$:   MOV     MMR2,(R1)+
12972 052760  013721  177574                            MOV     MMR1,(R1)+
12973 052764  013721  177572                            MOV     MMR0,(R1)+
12974 052770  104417                                    KERNEL                          ;ENTER KERNEL MODE
12975 052772                                            POP     BANK
12976 052776  000207                                    RETURN
```

```
12979 053000                                    WOOPUP: SUBTST  <<POWER UP FROM BANK 0 TO RELOCATION>>
                                                ;*****************************************************************
                                                ;*SUBTEST       POWER UP FROM BANK 0 TO RELOCATION
                                                ;*****************************************************************
12980 053000  012700  053162                            MOV     #WOOPEND,R0
12981 053004  012701  172340                            MOV     #KIPAR0,R1
12982 053010  012703  172300                            MOV     #KIPDR0,R3
12983 053014  012702  000010                            MOV     #8.,R2
12984 053020  012021                            1$:     MOV     (R0)+,(R1)+
12985 053022  012723  077406                            MOV     #77406,(R3)+
12986 053026  077204                                    SOB     R2,1$
12987 053030  005737  002430                            TST     NOSUPER
12988 053034  001002                                    BNE     3$
12989 053036  012037  172516                            MOV     (R0)+,MMR3
12990 053042  012037  177576                    3$:     MOV     (R0)+,MMR2
12991 053046  012037  177574                            MOV     (R0)+,MMR1
12992 053052  012037  177572                            MOV     (R0)+,MMR0
12993 053056  013706  052612                            MOV     $SAVR6,SP
12994 053062                                            PUSH    BANK
12995 053066  005037  002100                            CLR     BANK
12996 053072                                            MAP     BANK                    ;MAP SUPERVISUR SPACE (TES" AREA) TO BANK
12997 053106                                            SUPERVISOR              ;ENTER SUPERVISOR MODE
12998 053114  013737  053212  060024                    MOV     WOOPSAV,FIRST+PWRVEC
12999 053122  013737  053214  060026                    MOV     WOOPSAV+2,FIRST+PWRVEC+2
13000                                                   ;SIMULATE THE FOLLOWING BLOCK MOV BUT WITH NO STACK ACCESSES
13001                                                   ;BMOV   WOOPSAV+4,FIRST+WOOPUP,WOOPEND WOOPUP/2+12.
13002 053130  012700  053216                            MOV     #WOOPSAV+4,R0
13003 053134  012701  000105                            MOV     #WOOPEND-WOOPUP/2+12.,R1
13004 053140  012702  133000                            MOV     #FIRST+WOOPUP,R2
13005 053144  012022                            2$:     MOV     (R0)+,(R2)+
13006 053146  077102                                    SOB     R1,2$
13007
13008 053150  104417                                    KERNEL                          ;ENTER KERNEL MODE
13009 053152                                            POP     BANK
13010 053156  000137  052210                            JMP     $PWRUP
13011 053162  000014                    WOOPEND:.REPT   12.
13014 053212  000107                    WOOPSAV:.REPT   WOOPEND WOOPUP/2+12.+2
```

```
13019                                   .SBTTL  IO SUBROUTINES
13020
13021                                   .SBTTL  ROUTINE TYPE
13022
13023                           ;*******************************************************************************
13024                           ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
13025                           ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
13026                           ;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
13027                           ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
13028                           ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
13029                           ;*
13030                           ;*CALL:
13031                           ;*1) USING A TRAP INSTRUCTION
13032                           ;*      TYPE    MESADR          ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
13033                           ;*OR
13034                           ;*      TYPE
13035                           ;*      MESADR
13036                           ;*
13037
13038 053430  105737  002332    $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
13039 053434  100407            BMI     6$              ;BR IF NO
13040 053436  010046    1$:     MOV     R0,-(SP)        ;;SAVE R0
13041 053440  017600  000002    MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
13042 053444  112046    4$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
13043 053446  001005            BNE     7$              ;;BR IF IT ISN'T THE TERMINATOR
13044 053450  005726            TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
13045 053452  012600    5$:     MOV     (SP)+,R0        ;;RESTORE R0
13046 053454  062716  000002    6$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
13047 053460  000002            RTI                     ;;RETURN
13048 053462  122716  000011    7$:     CMPB    #HT,(SP)        ;BRANCH IF NOT <HT>
13049 053466  001002            BNE     11$
13050 053470  112716  000040            MOVB    #' ,(SP)        ;REPLACE TAB WITH SPACE
13051 053474  122716  000200    11$:    CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
13052 053500  001006            BNE     8$
13053 053502  005726            TST     (SP)+           ;;POP  <CR><LF> EQUIV
13054 053504                    TYPE                    ;;TYPE A CR AND LF
13055 053506  002622            $CRLF
13056 053510  105037  053742            CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
13057 053514  000753            BR      4$              ;;GET NEXT CHARACTER
13058 053516  004737  053556    8$:     CALL    $TYPEC          ;;GO TYPE THIS CHARACTER
13059 053522  123726  002614    9$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
13060 053526  001346            BNE     4$              ;;IF NO GO GET NEXT CHAR.
13061 053530  013746  002330            MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
13062                                                   ;;AND THE NULL CHAR.
13063 053534  105366  000001    10$:    DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
13064 053540  002770            BLT     9$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
13065 053542  004737  053556            CALL    $TYPEC          ;;GO TYPE A NULL
13066 053546  105337  053742            DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
13067 053552  000770            BR      10$             ;;LOOP
13068 053554  000000    XOCHAR: .WORD 0
13069 053556            $TYPEC: PUSH    R1
13070 053560  116601  000004            MOVB    4(SP),R1
13071 053564  005737  002516            TST     CACHKN
13072 053570  001402            BEQ     2$
13073 053572                    PUSH    CONTRL
13074 053576            2$:     PUSH    R0
13075 053600  104424            CACHOFF                 ;TURN CACHE OFF
```

```
13100 053602  105777  127002          3$:     TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
13101 053606  100375                          BPL     3$
13102 053610  005037  053554                  CLR     XOCHAR
13103 053614  105777  126764                  TSTB    @$TKS           ;;CHECK FOR XOFF
13104 053620  100032                          BPL     NC              ;;SKIP IF NO CHARACTER
13105 053622  117737  126760  053554          MOVB    @$TKB,XOCHAR    ;;SAVE THE CHARACTER
13106 053630  042737  177600  053554          BIC     #^C177,XOCHAR   ;;STRIP OFF ASCII
13107 053636  023727  053554  000023          CMP     XOCHAR,#023     ;;WAS IT A CONTROL S?
13108 053644  001020                          BNE     NC              ;;BRANCH IF NOT
13109 053646  105777  126732          CONTS3: TSTB    @$TKS           ;;WAIT FOR CHARACTER
13110 053652  100375                          BPL     CONTS3
13111 053654  117737  126726  053554          MOVB    @$TKB,XOCHAR    ;;GET CHARACTER
13112 053662  042737  177600  053554          BIC     #^C177,XOCHAR   ;;STRIP OFF ASCII
13113 053670                                  IF XOCHAR EQ #21        ;; IF IT IS A ^Q
13114 053700  000402                            BR    NC
13115 053702                                  ELSE
13116 053704  000760                            BR    CONTS3
13117 053706                                  END ;OF IF XOCHAR
13118 053706  110177  126700          NC:       MOVB  R1,@$TPB        ;;LOAD CHAR TO BE TYPED INTO DATA REG.
13122 053712  122766  000015  000002          CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
13123 053720  001003                          BNE     1$              ;;BRANCH IF NO
13124 053722  105037  053742                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
13125 053726  000406                          BR      $TYPEX          ;;EXIT
13126 053730  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
13127 053736  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
13128 053740  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
13129 053742  000000                  $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
13130 053744                          $TYPEX: POP     R0
13131 053746  005737  002516                  TST     CACHKN          ;IS THERE A CACHE?
13132 053752  001402                          BEQ     2$              ;BRANCH IF NOT
13133 053754                                  POP     CONTRL          ;POP CACHE STATUS
13134 053760                          2$:     POP     R1
13135 053762  000207                          RETURN
13136 053764                          SUPLIMIT:;!!!!!!!!!!!!THIS IS THE LIMIT ON SUPERVISOR MAPPED TO MUT SPACE
```

```
13814                                   .SBTTL  ERROR DATA SETUP
13815                           ;
13816                           ; USE THIS      IF THIS CONDITION DISCRIBES THE ERROR
13817                           ;
13818                           ; PERR01        TRAP
13819                           ;               BAD DATA IN R0 UNLESS ABORTED
13820                           ;                       THEN BAD DATA IS POINTED TO BY  (R4)
13821                           ;               GOOD DATA IN R5
13822                           ;
13823                           ; PERR02        TRAP
13824                           ;               BAD DATA IN R1 UNLESS ABORTED
13825                           ;                       THEN BAD DATA IS POINTED TO BY  (R4)
13826                           ;               GOOD DATA IN R2
13827                           ;
13828                           ; PERR03        TRAP
13829                           ;               BAD DATA IS POINTED TO BY  (R1)
13830                           ;               GOOD DATA IN R4
13831                           ;
13832                           ; PERR04        TRAP
13833                           ;               BAD DATA IN R4 UNLESS ABORTED
13834                           ;                       THEN BAD DATA IS POINTED TO BY  2(R0)
13835                           ;               GOOD DATA IN R2
13836                           ;
13837                           ; PERR05        JSR     PC
13838                           ;               BAD DATA IS POINTED TO BY  (R0)
13839                           ;               GOOD DATA IN R2
13840                           ;               RETURN AFTER SETTING UP GOOD,BAD,ADDRESS
13841                           ;
13842                           ; PERR06        JSR     PC
13843                           ;               BAD DATA IS POINTED TO BY -(R0)
13844                           ;               GOOD DATA IS ZERO
13845                           ;               RETURN AFTER SETTING UP GOOD,BAD,ADDRESS
13846                           ;
13847                           ; PERR07        TRAP
13848                           ;               BAD DATA IN R2 UNLESS ABORTED
13849                           ;                       THEN BAD DATA IS POINTED TO BY (R1)
13850                           ;               GOOD DATA IN DATBUF
13851                           ;
13852                           ; PERR10        TRAP
13853                           ;               BAD DATA IN R2 UNLESS ABORTED
13854                           ;                       THEN BAD DATA IS POINTED TO BY 2(R1)
13855                           ;               GOOD DATA IN DATBUF+2
13856                           ;
13857                           ; PERR11        TRAP
13858                           ;               BYTE TEST
13859                           ;               BAD DATA IN RIGHT BYTE OF R0 UNLESS ABORTED
13860                           ;                       THEN BAD DATA IS POINTED TO BY (R1)
13861                           ;               GOOD DATA IS A ZERO BYTE
13862                           ;
13863                           ; PERR12        TRAP
13864                           ;               BYTE TEST
13865                           ;               BAD DATA IN RIGHT BYTE OF R0 UNLESS ABORTED
13866                           ;                       THEN BAD DATA IS POINTED TO BY (R1)
13867                           ;               GOOD DATA IS A BYTE OF ONES
13868                           ;
13869                           ; PERR13        TRAP
13870                           ;               BAD DATA IN R0 UNLESS ABORTED
```

```
13871                   ;                       THEN BAD DATA IS POINTED TO BY (R1
13872                   ;               GOOD DATA IS ZERO
13873                   ;
13874                   ; PERR14        TRAP
13875                   ;               BAD DATA IN R0 UNLESS ABORTED
13876                   ;                       THEN BAD DATA IS POINTED TO BY (R1)
13877                   ;               GOOD DATA IS ONES
13878                   ;
13879                   ; PERR15        TRAP
13880                   ;               BAD DATA IN R0 UNLESS ABORTED
13881                   ;                       THEN BAD DATA IS POINTED TO BY (R1)
13882                   ;               GOOD DATA IN TSTDAT
13883                   ;
13884                   ; PERR16        TRAP
13885                   ;               BAD DATA IN R0 UNLESS ABORTED
13886                   ;                       THEN BAD DATA IS POINTED TO BY (R1)
13887                   ;               GOOD DATA IN TSTDAT+2
13888                   ;
13889                   ; PERR17        TRAP
13890                   ;               BAD DATA IN R0 UNLESS ABORTED
13891                   ;                       THEN BAD DATA IS POINTED TO BY (R1)
13892                   ;               GOOD DATA IN R2
13893                   ;
13894                   ; PERR20        TRAP
13895                   ;               BAD DATA IN R0 UNLESS ABORTED
13896                   ;                       THEN BAD DATA IS POINTED TO BY (R1)
13897                   ;               GOOD DATA IN R3
13898                   ;
13899                   ; PERR21        TRAP
13900                   ;               7 BIT BYTE TEST
13901                   ;               BAD DATA IN RIGHT BYTE OF R0 UNLESS ABORTED
13902                   ;                       THEN BAD DATA IS POINTED TO BY (R1)
13903                   ;               GOOD DATA IS A 7 BIT BYTE ON ONES
13904                   ;
13905                   ; PERR22        TRAP
13906                   ;               BAD DATA IN R2 UNLESS ABORTED
13907                   ;                       THEN BAD DATA IS POINTED TO BY (R1)
13908                   ;               GOOD DATA IN R0
13909                   ;
13910                   ; PERR23        TRAP
13911                   ;               BAD DATA IN R0 UNLESS ABORTED
13912                   ;                       THEN BAD DATA IS POINTED TO BY (R1)
13913                   ;               GOOD DATA IN R4
13914                   ;
13915                   ; PERR24        TRAP
13916                   ;               BAD DATA IN R0 UNLESS ABORTED
13917                   ;                       THEN BAD DATA IS POINTED TO BY (R2)
13918                   ;               GOOD DATA IN R3
13919                   ;
13920                   ; PERR25        TRAP
13921                   ;               BAD DATA POINTED TO BY -(R1)
13922                   ;               GOOD DATA IN R2 UNLESS LOC V177654 IS SET
13923                   ;                       THEN GOOD DATA IS IN R3
13924                   ;
13925                   ; PERR26        TRAP
13926                   ;               BAD DATA IS DOUBLE WORD POINTED TO BY R1 AND IN LOW 7 BITS OF R0
13927                   ;               GOOD DATA IS 000000,,100000,,100
```

```
13928                                   ;
13929                                   ;  PERR27        TRAP
13930                                   ;                BAD DATA IS DOUBLE WORD POINTED TO BY R1 AND IN LOW 7 BITS OF RO
13931                                   ;                GOOD DATA IS 000000,,000000,,077
13932                                   ;
13933                                   ;  PERR30        TRAP
13934                                   ;                BAD DATA IS POINTED TO BY  16(SP)
13935                                   ;                GOOD DATA IS IN R1
13936                                   ;
13937                                   ;  PERR31        TRAP
13938                                   ;                SPECIAL ECC FAILURE HANDLER
13939                                   ;
13940                                   ;  PERR32        TRAP
13941                                   ;                SPECIAL ECC FAILURE HANDLER
13942                                   ;
13943                                   ;  PERR33        TRAP
13944                                   ;                SPECIAL ECC FAILURE HANDLER
13945                                   ;
13946                                   ;  PERR34        TRAP
13947                                   ;                SPECIAL ECC FAILURE HANDLER
13948                                   ;
13949                                   ;  PERR35        TRAP
13950                                   ;                SPECIAL BRANCH GOBBLE FAILURE HANDLER
13951                                   ;
13952                                   ;       CALLING SEQUENCE FOR TRAP TYPES
13953                                   ;       BEQ     2$                ;NO - ERROR,BRANCH FOR CARD
13954                                   ;       PERRXX                    ;TRAP TO ERROR ROUTINE
13955                                   ;2$:    NEXT    INSTRUCTION       ;CONTINUE TESTING
```

```
13958 053764  010437  002032          $PER01: MOV     R4,ADDRESS
13959 053770  162737  000002  002032          SUB     #2,ADDRESS
13960 053776  010037  002050                  MOV     R0,BAD
13961 054002  010537  002042                  MOV     R5,GOOD
13962 054006  000137  054520                  JMP PERRAW
13963
13964 054012  010437  002032          $PER02: MOV     R4,ADDRESS
13965 054016  162737  000002  002032          SUB     #2,ADDRESS
13966 054024  010137  002050                  MOV     R1,BAD
13967 054030  010237  002042                  MOV     R2,GOOD
13968 054034  000137  054520                  JMP PERRAW
13969
13970 054040  010137  002032          $PER03: MOV     R1,ADDRESS
13971 054044  162737  000002  002032          SUB     #2,ADDRESS
13972 054052  010437  002042                  MOV     R4,GOOD
13973 054056  016137  177776  002050          MOV     -2(R1),BAD
13974 054064  000137  054520                  JMP PERRAW
13975
13976 054070  010037  002032          $PER04: MOV     R0,ADDRESS
13977 054074  162737  000002  002032          SUB     #2,ADDRESS
13978 054102  010437  002050                  MOV     R4,BAD
13979 054106  010237  002042                  MOV     R2,GOOD
13980 054112  000137  054520                  JMP     PERRAW
13981
13982 054116  010237  002042          PERR05: MOV     R2,GOOD
13983 054122  014037  002050          PERA05: MOV     -(R0),BAD
13984 054126  010037  002032                  MOV     R0,ADDRESS
13985 054132  062700  000002                  ADD     #2,R0           ;RESTORE R0
13986 054136  004737  040170                  CALL    BADSTACK
13987 054142  000207                          RETURN
13988
13989 054144  005037  002042          PERR06: CLR     GOOD
13990 054150  000764                          BR      PERA05
13991
13992 054152  010137  002032          $PER07: MOV     R1,ADDRESS
13993 054156  010237  002050                  MOV     R2,BAD
13994 054162  013737  002236  002042          MOV     DATBUF,GOOD
13995 054170  000137  054520                  JMP     PERRAW
13996
13997 054174                          $PER10: LET ADDRESS := R1 + #2
13998 054206                                  LET BAD := R2
13999 054212                                  LET GOOD := DATBUF+2
14000 054220  000137  054520                  JMP     PERRAW
14001
14002 054224                          $PER11: LET ADDRESS := R1
14003 054230                                  LET BAD := R0
14004 054234                                  LET GOOD := #0
14005 054240  000137  054572                  JMP     PERRAB
14006
14007 054244                          $PER12: LET ADDRESS := R1
14008 054250                                  LET BAD := R0
14009 054254                                  LET GOOD := #377
14010 054262  000137  054572                  JMP     PERRAB
```

```
14013 054266                        $PER13: LET ADDRESS := R1
14014 054272                                LET BAD := R0
14015 054276                                LET GOOD := 00
14016 054302  000137  054520                JMP     PERRAW
14017
14018 054306                        $PER14: LET ADDRESS := R1
14019 054312                                LET BAD := R0
14020 054316                                LET GOOD := ONES
14021 054324  000137  054520                JMP     PERRAW
14022
14023 054330                        $PER15: LET ADDRESS := R1
14024 054334                                LET BAD := R0
14025 054340                                LET GOOD := TSTDAT
14026 054346  000137  054520                JMP     PERRAW
14027
14028 054352                        $PER16: LET ADDRESS := R1
14029 054356                                LET BAD := R0
14030 054362                                LET GOOD := TSTDAT+2
14031 054370  000455                        BR      PERRAW
14032
14033 054372                        $PER17: LET ADDRESS := R1
14034 054376                                LET BAD := R0
14035 054402                                LET GOOD := R2
14036 054406  000444                        BR      PERRAW
14037
14038 054410                        $PER20: LET ADDRESS := R1
14039 054414                                LET BAD := R0
14040 054420                                LET GOOD := R3
14041 054424  000435                        BR      PERRAW
14042
14043 054426                        $PER21: LET ADDRESS := R1
14044 054432                                LET BAD := R0
14045 054436                                LET GOOD := #177
14046 054444  000477                        BR      PERRAW
14047
14048 054446                        $PER22: LET ADDRESS := R1
14049 054452                                LET BAD := R2
14050 054456                                LET GOOD := R0
14051 054462  000416                        BR      PERRAW
14052
14053 054464                        $PER23: LET ADDRESS := R1
14054 054470                                LET BAD := R0
14055 054474                                LET GOOD := R4
14056 054500  000407                        BR      PERRAW
14057
14058 054502                        $PER24: LET ADDRESS := R2
14059 054506                                LET BAD := R0
14060 054512                                LET GOOD := R3
14061 054516  000400                        BR      PERRAW
```

```
14063 054520

                        PERRAW: SUBTST  <<DATA WAS A WORD>>
                        ;*****************************************************************************
                        ;*SUBTEST       DATA WAS A WORD
                        ;*****************************************************************************
14064 054520  004737  054754           CALL    PERBNK
14065 054524                           IF ABORTFLAG IS TRUE THEN $CALL GETDATA
14066 054536                           IF BADPC EQ #0 THEN $CALL BADSTACK
14067 054550  004737  054730           CALL    PERXOR
14068 054554                           IF ABORTFLAG IS FALSE
14069 054562  104011                     ERROR .11
14070 054564                           ELSE
14071 054566  104012                     ERROR .12
14072 054570                           END ;OF IF ABORTFLAG
14073 054570  000002                     RTI
14074
14075 054572

                        PERRAB: SUBTST  <<DATA WAS A BYTE>>
                        ;*****************************************************************************
                        ;*SUBTEST       DATA WAS A BYTE
                        ;*****************************************************************************
14076 054572  004737  054754           CALL    PERBNK
14077 054576                           IF ABORTFLAG IS TRUE THEN $CALL GETDATA
14078 054610                           IF BADPC EQ #0 THEN $CALL BADSTACK
14079 054622  004737  054730           CALL    PERXOR
14080 054626                           IF ABORTFLAG IS FALSE
14081 054634  104014                     ERROR .14
14082 054636                           ELSE
14083 054640  104015                     ERROR .15
14084 054642                           END ;OF IF ABORTFLAG
14085 054642  000002                     RTI
```

```
14088 054644                                    PERRA7: SUBTST  <<DATA WAS A 7 BIT BYTE>>
                                                ;*****************************************************************
                                                ;*SUBTEST       DATA WAS A 7 BIT BYTE
                                                ;*****************************************************************
14089 054644                                            IF BADPC EQ #0 THEN $CALL BADSTACK
14090 054656  004737  054730                            CALL    PERXOR
14091 054662  004737  054754                            CALL    PERBAD
14092 054666  104022                                    ERROR   .22
14093 054670  000002                                    RTI
14094
14095 054672                                    $PER26: LET GOOD2 .= #100000
14096 054700                                            LET GOOD3 := #100
14097 054706  000137  051322                            JMP PERRA3
14098
14099 054712  005037  002044                    $PER27: CLR     GOOD2
14100 054716                                            LET GOOD3 := #077
14101 054724  000137  051322                            JMP PERRA3
14102
14103 054730                                    PERXOR: SUBTST  <<DETERMINE XOR OF GOOD & BAD>>
                                                ;*****************************************************************
                                                ;*SUBTEST       DETERMINE XOR OF GOOD & BAD
                                                ;*****************************************************************
14104 054730                                            PUSH    R0
14105 054732  013700  002042                            MOV     GOOD,R0
14106 054736  013737  002050  002056                    MOV     BAD,BADXOR
14107 054744  074037  002056                            XOR     R0,BADXOR
14108 054750                                            POP     R0
14109 054752  000207                                    RETURN
```

```
14112 054754                                    PERBNK: SUBTST<<LOG ERROR ON BAD BANK>>
                                                ;*****************************************************************
                                                ;*SUBTEST       LOG ERROR ON BAD BANK
                                                ;*****************************************************************
14113                                                   ;WHILE WE'RE HERE LET'S MARK THE BAD BANK IN THE CONFIGURATION TABLE
14114 054754                                            PUSH    R0,R1
14115 054760  013701  002100                      MOV     BANK,R1
14116 054764  006301                              ASL     R1
14117 054766  006301                              ASL     R1
14118 054770  052761  000001  002626              BIS     #BIT0,CONFIG(R1)
14119 054776  105261  002630                      INCB    CONFIG+2(R1)            ;BUMP BANK COUNTER
14120 055002  001002                              BNE     12$                     ;NO OVERFLOW - SKIP
14121 055004  105361  002630                      DECB    CONFIG+2(R1)            ;SET BACK TO 255.
14122 055010  126137  002630  002526  12$:       CMPB    CONFIG+2(R1),ERRMAX     ;IS IT PAST MAX?
14123 055016  101403                              BLOS    11$                     ;NO - SKIP
14124 055020                                      SET     TOOMANY                 ;YES
14125 055026                                    11$:    POP     R1,R0
14126 055032  000207                              RETURN
14127
14128 055034  010037  002050                    PERECC: MOV     R0,BAD
14129 055040                                            IF ADDRESS EQ TESTADD
14130 055050  013737  002242  002042                      MOV TSTDAT,GOOD
14131 055056                                            ELSE
14132 055060  013737  002244  002042                      MOV   TSTDAT+2,GOOD
14133 055066                                            END ;OF IF (R1)
14134 055066  004737  054730                            CALL    PERXOR
14135 055072                                            SET     HEADER
14136 055100  000207                                    RETURN
14137
14138 055102                                    $PER31: IF BADPC EQ #0 THEN $CALL BADSTACK
14139 055114  004737  055034                            CALL    PERECC
14140 055120                                            IF REALPAT EQ #11
14141 055130  104037                                      ERROR .37
14142 055132                                            END ;OF IF REALPAT
14143 055132                                            IF REALPAT EQ #14
14144 055142  104042                                      ERROR .42
14145 055144                                            END ;OF IF REALPAT
14146 055144                                            IF REALPAT EQ #15
14147 055154  104043                                      ERROR .43
14148 055156                                            END ;OF IF REALPAT
14149 055156                                            IF REALPAT EQ #16
14150 055166  104044                                      ERROR .44
14151 055170                                            END ;OF IF REALPAT
14152 055170                                            SET HEADER
14153 055176  000002                                    RTI
```

```
14156 055200                                $PER32: IF BADPC EQ #0 THEN $CALL BADSTACK
14157 055212  010137  002032                        MOV     R1,ADDRESS
14158 055216  010037  002050                        MOV     R0,BAD
14159 055222  010237  002042                        MOV     R2,GOOD
14160 055226                                        SET     HEADER
14161 055234  104040                                ERROR   .40
14162 055236                                        SET     HEADER
14163 055244  000002                                RTI
14164
14165 055246                                $PER33: IF BADPC EQ #0 THEN $CALL BADSTACK
14166 055260  010137  002032                        MOV     R1,ADDRESS
14167 055264  010037  002050                        MOV     R0,BAD
14168 055270  105037  002051                        CLRB    BAD+1
14169 055274  012737  000377  002042                MOV     #377,GOOD
14170 055302  004737  054730                        CALL    PERXOR
14171 055306                                        SET     HEADER
14172 055314  104041                                ERROR   .41
14173 055316                                        SET     HEADER
14174 055324  000002                                RTI
14175
14176 055326                                $PER34:IF BADPC EQ #0 THEN $CALL BADSTACK
14177 055340                                        IF #BIT15!BIT4 OFF.IN CSR
14178 055350  104016                                  ERROR .16                     ;NO SBE OR DBE
14179 055352                                        ELSE
14180 055354  104001                                  ERROR .1                      ;EXPECTED SBE SO DBE MUST HAVE GOTTEN SE
14181 055356                                        END ;OF IF #BIT15!BIT4
14182 055356  000002                                RTI
14183
14184                                               ;DURING BRANCH GOBBLE THE CONDITION CODES WERE WRONG
14185 055360  004737  054754                $PER35: CALL    PERBNK
14186 055364  004737  040170                        CALL    BADSTACK
14187 055370  013737  002030  002050                MOV     BADPSW,BAD
14188 055376  012737  000012  002042                MOV     #12,GOOD
14189 055404  104047                                ERROR   .47
14190 055406  062706  000004                        ADD     #4,SP                 ;FIX STACK FROM TRAP
14191 055412  000207                                RETURN                        ;ABORTING TEST
14192
14193 055414  010037  002042                $PER36: MOV     R0,GOOD
14194 055420  010137  002050                        MOV     R1,BAD
14195 055424                                        SET     HEADER
14196 055432  104023                                ERROR   .23
14197 055434                                        SET     HEADER
14198 055442  000002                                RTI
```

```
14201                                           .SBTTL  ROUTINE SCOPE HANDLER
14202                                   ;*******************************************************************************
14203                                   ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
14204                                   ;*AND LOAD THE DISPLAY DATA INTO THE DISPLAY REGISTER
14205                                   ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
14206                                   ;*SW14-1        LOOP ON TEST
14207                                   ;*SW9-1 LOOP ON ERROR
14208                                   ;*CALL
14209                                   ;*      SCOPE                   ;;SCOPE-IOT
14210 055444  005237  063106          $SCOPE: INC     $DEVCT          ;TELL APT WE ARE ALIVE
14211 055450                                  IF RESULT IS LT
14212 055452  005037  063106                    CLR   $DEVCT
14213 055456  105237  063110                    INCB  $UNIT
14214 055462                                  END ;OF IF RESULT
14215 055462  104410                          CKSWR                   ;;TEST  FOR CHANGE IN SOFT SWR
14216 055464  005737  006140                  TST     TRACE
14217 055470  001402                          BEQ     NOTRCE
14218 055472  004737  061402                  CALL    CONTT           ;TRACE
14219 055476                          NOTRCE:
14220 055476  005737  056560                  TST     CPERRF          ;IS THERE A CPU ERROR REGISTER?         ;R C
14221 055502  001410                          BEQ     SKJ             ;BRANCH IF NOT                          ;R C
14222 055504  013737  177766  056556          MOV     @#177766,CPSAVE ;GET CONTENTS OF ERROR REGISTER         ;R C
14223 055512  032737  000001  056556          BIT     #BIT0,CPSAVE    ;IS THE POWER FAIL MONITOR BIT SET?     ;R-C
14224 055520  001401                          BEQ     SKJ             ;BRANCH IF NOT                          ;R C
14225 055522  104177                          ERROR   +177            ;REPORT IF SO                           ;R C
14234 055524                          SKJ:    IF STOPOK IS TRUE AND #SW8 SET.IN @$WR                          ;R-C
14235 055542  005037  002372                    CLR   STOPOK
14236 055546  000137  045136                    JMP   EXIT
14237 055552                                  END ;OF IF STOPOK
14238 055552                                  IF NOSCOPE IS TRUE
14239 055560  000002                            RTI
14240 055562                                  END ;OF IF NOSCOPE
14241 055562                          1$:     IF #SW14 SET.IN @$WR THEN GOTO $OVER
14242                                 ;#####START OF CODE FOR THE XOR TESTER#####
14243 055572  000425                  $XTSTR: BR      2$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
14244                                                                 ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
14245 055574  013746  000004                  MOV     ERRVEC,-(SP)    ;;SAVE THE CONTENTS OF THE ERROR VECTOR
14246 055600  012737  055620  000004          MOV     #1$,ERRVEC      ;;SET FOR TIMEOUT
14247 055606  005737  177060                  TST     177060  ;;TIME OUT ON XOR?
14248 055612  012637  000004                  MOV     (SP)+,ERRVEC    ;;RESTORE THE ERROR VECTOR
14249 055616  000430                          BR      $SVLAD          ;;GO TO THE NEXT TEST
14250 055620  062706  000004          1$:     ADD     #4,SP           ;FIX STACK FROM TRAP
14251 055624  022737  000001  003720          CMP     #1,PROTYP       ;IS THIS AN 11/44?
14252 055632  001002                          BNE     6$              ;BRANCH IF NOT
14253 055634  005037  177766                  CLR     CPUERR          ;RESET CPU ERROR REGISTER
14254 055640  012637  000004          6$:     MOV     (SP)+,ERRVEC    ;;RESTORE THE ERROR VECTOR
14255 055644  000407                          BR      4$              ;;LOOP ON THE PRESENT TEST
14256 055646                          2$:;#####END OF CODE FOR THE XOR TESTER#####
14257 055646  105737  002012          3$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
14258 055652  001412                          BEQ     $SVLAD          ;;BR IF NO
14259 055654  032777  001000  124716          BIT     #SW9,@$WR       ;;LOOP ON ERROR?
14260 055662  001404                          BEQ     5$              ;;BR IF NO
14261 055664  013737  002566  002564  4$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
14262 055672  000410                          BR      $OVER
14263 055674  105037  002012          5$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
14264 055700  011637  002564          $SVLAD: MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
14265 055704  011637  002566                  MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
```

```
14266 055710  005037  002534                CLR     $ESCAPE          ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
14267 055714  004737  055726        $OVER:  CALL    GETDIS
14268 055720  013716  002564                MOV     $LPADR,(SP)      ;;FUDGE RETURN ADDRESS
14269 055724  000002                        RTI                      ;;FIXES PS
```

```
14271 055726                          GETDIS: SUBTST  <<SUBR  DISPLAY>>
                                      ;*************************************************************************
                                      ;*SUBTEST        SUBR    DISPLAY
                                      ;*************************************************************************
14272 055726  113737  002100  002011          MOVB    BANK,$BANK
14273 055734  113737  002262  002010          MOVB    REALPAT,$PATMAR
14274 055742                                  PUSH    R0
14275 055744  005737  002124                  TST     RLFLAG                  ;ARE WE RELOCATED?
14276 055750  001403                          BEQ     1$                      ;NO   SKIP
14277 055752  052737  100000  002010          BIS     #BIT15,$PATMAR          ;YES  SET MSB
14278 055760                          1$:
14282 055760  013777  002010  124614          MOV     $PATMAR,@DISPLAY
14283 055766  013737  002010  000174          MOV     $PATMAR,DISPREG         ;SOFTWARE DISPLAY REGISTER
14284 055774                                  POP     R0
14285 055776  000207                          RETURN
```

```
14288                                           .SBTTL  ROUTINE ERROR HANDLER
14289                                   ;******************************************************************************
14290                                   ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
14291                                   ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
14292                                   ;*AND GO TO $ERRTYP ON ERROR
14293                                   ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
14294                                   ;*SW15=1         HALT ON ERROR
14295                                   ;*SW13=1         INHIBIT ERROR TYPEOUTS
14296                                   ;*SW10=1         BELL ON ERROR
14297                                   ;*SW9=1 LOOP ON ERROR
14298                                   ;*CALL
14299                                   ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER
14300
14301                                           .ENABL  LSB
14302 056000  105037  056554      $ERROR: CLRB    IBSAVE                                          ;R C
14303 056004                              IF NOERROR IS FALSE
14304 056012  104410                        CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
14305 056014                              BACK:                                                   ;R-C
14306 056014  105237  002012      1$:         INCB    $ERFLG        ;;SET THE ERROR FLAG
14307 056020  001775                        BEQ     1$              ;;DON'T LET THE FLAG GO TO ZERO
14308 056022  004737  055726                CALL    GETDIS          ;SETUP DISPLAY STUFF
14309 056026  013737  002010  063102        MOV     $PATMAR,$TESTN  ;FOR APT
14310 056034  032777  002000  124536        BIT     #SW10,@SWR      ;;BELL ON ERROR?
14311 056042  001404                        BEQ     2$              ;;NO - SKIP
14312 056044                                TYPE    $BELL           ;;RING BELL
14313 056050                                TYPE    MSG014          ;CONTROL Z
14314 056054  005237  002572      2$:       INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
14315 056060                                IF RESULT IS MI
14316 056062  012737  077777  002572          MOV #77777,$ERTTL
14317 056070                                END ;OF IF RESULT
14318 056070                              END ;OF IF NOERROR
14319 056070  011637  002016              MOV     (SP),ERRPC      ;;GET ADDRESS OF ERROR INSTRUCTION
14320 056074  162737  000002  002016      SUB     #2,ERRPC
14321 056102  010637  002022              MOV     SP,ERRSP
14322 056106  016637  000002  002026      MOV     2(SP),ERRPSW
14323 056114  117737  123676  002013      MOVB    @ERRPC,$ITEMB   ;;STRIP AND SAVE THE ERROR ITEM CODE
14324
14325 056122  122737  000177  002013      CMPB    #177,$ITEMB     ;IS THIS THE POWER FAIL CALL?           ;R C
14326 056130  001431                      BEQ     1001$           ;BRANCH IF SO                           ;R-C
14327 056132  105737  056554              TSTB    IBSAVE          ;2ND ERROR CALL?                        ;R C
14328 056136  001024                      BNE     1000$           ;BRANCH IF SO                           ;R-C
14329 056140  005737  056560              TST     CPERRF          ;IS THERE A CPU ERROR REGISTER?         ;R C
14330 056144  001423                      BEQ     1001$           ;BRANCH IF NOT                          ;R C
14331 056146  013737  177766  056556      MOV     177766,CPSAVE   ;SAVE CONTENTS                          ;R-C
14332 056154  032737  000001  056556      BIT     #BIT0,CPSAVE    ;POWER MONITOR BIT SET?                 ;R C
14333 056162  001414                      BEQ     1001$           ;BRANCH IF NOT                          ;R-C
14334 056164  042737  000001  177766      BIC     #BIT0,177766    ;CLEAR THE BIT                          ;R-C
14335 056172  112737  002013  056554      MOVB    #$ITEMB,IBSAVE  ;MAKE IBSAVE NON ZERO FOR DUAL CALL     ;R-C
14336 056200  112737  000177  002013      MOVB    #177,$ITEMB     ;SET $ITEMB TO POWER FAIL POINTER       ;R C
14337 056206  000402                      BR      1001$                                                   ;R C
14338 056210  105037  056554      1000$:  CLRB    IBSAVE                                                  ;R C
14339 056214                      1001$:                                                                  ;R C
14340 056214                              IF NOERROR IS FALSE
14341 056222                                IF BADPC NE #0
14342 056230  013737  002020  002016            MOV BADPC,ERRPC
14343 056236  162737  000002  002016            SUB #2,ERRPC
14344 056244  013737  002024  002022            MOV BADSP,ERRSP
```

```
14345 056252  013737  002030  002026            MOV BADPSW,ERRPSW
14346 056260  005037  002020                    CLR BADPC
14347 056264                                   END ;IF
14348 056264  013737  002016  063100           MOV   ERRPC,$FATAL     ;FOR APT
14349 056272                                   IF @SW13 SET.IN @SWR
14350 056302  000412                             BR  3$
14351 056304                                   END ;OF IF @SW13
14357 056304                                   IF @SW5 SET.IN @SWR AND TOOMANY IS TRUE
14358 056322                                     GOTO 3$
14359 056324                                   END ;OF IF @SW5
14360 056324                                 END ;OF IF NOERROR
14361 056324  004737  056562                 CALL    $ERRTYP          ;;GO TO USER ERROR ROUTINE
```

```
14363 056330                                3$:     IF NOERROR IS FALSE
14364 056336  005777  124236                         TST   @SWR              ;;HALT ON ERROR
14365 056342  100002                                BPL   7$                ;;SKIP IF CONTINUE
14366 056344  000000                        $HALT:  HALT                    ;;HALT ON ERROR!
14367 056346  104410                                CKSWR                   ;;TEST FOR CHANGE IN SOFT SWR
14368 056350                                7$:     IF NOSCOPE IS FALSE AND @SW9 SET.IN @SWR
14369 056366  013716  002566                           MOV $LPERR,(SP)      ;;FUDGE RETURN FOR LOOPING
14370 056372                                        END ;OF IF NOSCOPE
14371 056372  005737  002334                         TST   $ESCAPE          ;;CHECK FOR AN ESCAPE ADDRESS
14372 056376  001402                                BEQ   9$                ;;BR IF NONE
14373 056400  013716  002334                        MOV   $ESCAPE,(SP)      ;;FUDGE RETURN ADDRESS FOR ESCAPE
14374 056404                                9$:     IF DETFLAG IS FALSE
14375 056412  022737  000001  003720                  CMP #1,PROTYP                 ;IS THIS AN 11/44?
14376 056420  001002                                  BNE 11$
14377 056422  005037  177766                         CLR CPUERR
14378 056426                                      11$: IF ACTFLAG IS TRUE OR APTFLAG IS TRUE OR FATAL$ IS TRUE
14379 056450  012737  000001  063076                    MOV     #1,$MSGTY       ;FOR APT
14380 056456  000137  045136                          JMP     EXIT
14381 056462                                        END ;OF IF ACTFLAG
14382 056462                                        IF XXDPCHAIN IS TRUE AND $ERTTL HI #20
14383 056500                                          TYPE    MSG066          ;ERROR COUNT EXCEEDED 20   ABORTING FOR xXDP CHAIN
14384 056504  013700  000042                          MOV     42,R0
14385 056510  005037  000042                          CLR     42
14386 056514  000137  014452                          JMP     $ZAP42
14387 056520                                        END ;OF IF XXDPCHAIN
14388 056520                                    END ;OF IF DETFLAG
14389 056520                                  ELSE
14390 056522                                    SET   HEADER
14391 056530                                  END ;OF IF NOERROR
14392 056530                                10$:  CLEAR   TOOMANY,NOERROR
14393 056540  105737  056554                   TSTB    IBSAVE                  ;POWER FAIL ERROR CALL? ;R-C
14394 056544  001402                           BEQ     213$                                    ;R C
14395 056546  000137  056014                   JMP     BACK                    ;JUMP IF SO     ;R C
14396 056552  000002                      213$: RTI                     ;;RETURN
14397 056554  000000                      IBSAVE: .WORD 0                                     ;R-C
14398 056556  000000                      CPSAVE: .WORD 0                                     ;R-C
14399 056560  000000                      CPERRF: .WORD 0                                     ;R-C
14400                                             .DSABL  LSB
```

```
14403                                   .SBTTL  ROUTINE ERROR MESSAGE TYPEOUT
14404
14405                           ;*******************************************************************************
14406                           ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
14407                           ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE  ($ERRTB),
14408                           ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
14409
14410 056562  104415          $ERRTYP:SAVREG
14411 056564                          TYPE    $CRLF           ;;"CARRIAGE RETURN" & 'LINE FEED
14412 056570  005000                  CLR     R0              ;;PICKUP THE ITEM INDEX
14413 056572  153700  002013          BISB    $ITEMB,R0
14414 056576  001004                  BNE     1$              ;;IF ITEM NUMBER IS ZERO, JUST
14415                                                         ;;TYPE THE PC OF THE ERROR
14416 056600                          TYPOCT  ERRPC,<ERROR ADDRESS>
14417 056606  000511                  BR      11$             ;;GET OUT
14418 056610  122700  000177  1$:     CMPB    #177,R0         ;POWER MONITOR CALL?                    ;R C
14419 056614  001003                  BNE     100$            ;BRANCH IF NOT                          ;R C
14420 056616  012700  057072          MOV     #PFECWS,R0      ;MOV ADDRESS OF PFE BIT ERROR TO R0     ;R C
14421 056622  000406                  BR      110$                                                    ;R C
14422 056624  005300  100$:   DEC     R0              ;;ADJUST THE INDEX SO THAT IT WILL      ;R C
14423 056626  006300                  ASL     R0              ;;      WORK FOR THE ERROR TABLE
14424 056630  006300                  ASL     R0
14425 056632  006300                  ASL     R0
14426 056634  062700  063520          ADD     #$ERRTB,R0      ;;FORM TABLE POINTER
14427 056640  012037  056676  110$:   MOV     (R0)+,3$        ;;PICKUP "ERROR MESSAGE' POINTER        ;R C
14428 056644  001417                  BEQ     4$              ;;SKIP TYPEOUT IF NO POINTER
14429 056646  005737  002402          TST     NOERROR         ;IS THIS REALLY AN ERROR?
14430 056652  001003                  BNE     12$             ;YES - SKIP
14431 056654  005737  002554          TST     HEADER          ;TYPE HEADER?
14432 056660  100011                  BPL     4$              ;NO - SKIP
14433 056662  005737  002062  12$:    TST     FATAL$          ;WAS IT A FATAL ERROR?
14434 056666  001402                  BEQ     2$              ;NO - SKIP
14435 056670                          TYPE    MSG067          ;FATAL
14436 056674                  2$:     TYPE                    ;;TYPE THE "ERROR MESSAGE"
14437 056676  000000          3$:     .WORD   0               ;;"ERROR MESSAGE" POINTER GOES HERE
14438 056700                          TYPE    $CRLF           ;;"CARRIAGE RETURN" & "LINE FEED"
14439 056704  012037  056730  4$:     MOV     (R0)+,5$        ;;PICKUP "DATA HEADER" POINTER
14440 056710  001412                  BEQ     6$              ;;SKIP TYPEOUT IF 0
14441 056712  005737  002402          TST     NOERROR         ;IS THIS REALLY AN ERROR?
14442 056716  001003                  BNE     13$             ;YES - SKIP
14443 056720  005737  002554          TST     HEADER          ;TYPE HEADER?
14444 056724  100004                  BPL     6$              ;NO - SKIP
14445 056726                  13$:    TYPE                    ;;TYPE THE "DATA HEADER"
14446 056730  000000          5$:     .WORD   0               ;;"DATA HEADER" POINTER GOES HERE
14447 056732                          TYPE    $CRLF           ;;"CARRIAGE RETURN" & "LINE FEED"
14448 056736  012001          6$:     MOV     (R0)+,R1        ;;PICKUP "DATA TABLE" POINTER
14449 056740  001427                  BEQ     10$             ;;BR IF NO DATA TO BE TYPED
14450 056742  012002                  MOV     (R0)+,R2        ;;PICKUP "DATA FORMAT" POINTER
```

```
14453 056744  112203                  7$:     MOVB    (R2)+,R3
14454 056746  006303                          ASL     R3              ;MAKE IT A WORD ADDRESS
14455 056750  004773  056756                    CALL    @8$(R3)
14456 056754  000412                          BR      9$
14457 056756  057202                  8$:     TAG70$
14458 056760  057212                          TAG71$
14459 056762  057222                          TAG72$
14460 056764  057272                          TAG73$
14461 056766  057332                          TAG74$
14462 056770  057344                          TAG75$
14463 056772  057356                          TAG76$
14464 056774  057422                          TAG77$
14465 056776  057430                          TAG78$
14466 057000  057510                          TAG79$
14471 057002  062701  000002          9$:     ADD     #2,R1           ;UPDATE DATA TABLE POINTER
14472 057006  005711                          TST     (R1)            ;;IS THERE ANOTHER NUMBER?
14473 057010  001403                          BEQ     10$             ;;BR IF NO
14474 057012                                  TYPE    MSG018          ;TYPE 2 SPACES
14475 057016  000752                          BR      7$              ;;LOOP
14476
14477 057020  005737  002106          10$:    TST     MUT             ;IS THERE A MEMORY UNDER TEST
14478 057024  001402                          BEQ     11$             ;NO - SKIP
14479 057026  005237  002554                    INC     HEADER          ;YES - BUMP HEADER FLAG
14480 057032  104416                  11$:    RESREG
14481 057034                                  IF #SW7 SET.IN @SWR AND DETFLAG IS FALSE AND NOERROR IS FALSE
14482 057060  004737  057532                      CALL  DETAIL
14483 057064                                  END ;OF IF #SW7
14484 057064                                  TYPE    MSG104          ;CONTROL Z
14485 057070  000207                          RETURN
14486                                         .EVEN                                   ;R C
14487 057072  057102  057136  057166  PFECWS: .WORD   PFECEM,PFECDH,PFECDT,PFECDF    ;R-C
      057100  057176
14488 057102     120     117     127  PFECEM: .ASCIZ  "POWER MONITOR BIT FOUND SET"  ;R-C
      057105     105     122     040
      057110     115     117     116
      057113     111     124     117
      057116     122     040     102
      057121     111     124     040
      057124     106     117     125
      057127     116     104     040
      057132     123     105     124
      057135     000
14489 057136     124     105     123  PFECDH: .ASCIZ  "TESTNO  ERR PC  CPUERR"       ;R-C
      057141     124     116     117
      057144     040     040     105
      057147     122     122     040
      057152     120     103     040
      057155     040     103     120
      057160     125     105     122
      057163     122     000
14490                                         .EVEN                                   ;R C
14491 057166  063102  002016  056556  PFECDT: .WORD   $TESTN,ERRPC,CPSAVE,0          ;R C
      057174  000000
14492 057176     000     000     000  PFECDF: .BYTE   0,0,0,0                         ;R C
      057201     000
14493
```

```
14496                                   ;*****************************************************************************
14497                                   ;*** OCTAL ***
14498                                   ;*****************************************************************************
14499 057202                            TAG708: TYPOCT  @(R1)           ;;TYPE AN OCTAL NUMBER
14500 057210  000207                            RETURN
14501
14502                                   ;*****************************************************************************
14503                                   ;*** DECIMAL ***
14504                                   ;*****************************************************************************
14505 057212                            TAG718: TYPDEC  @(R1)           ;;TYPE A DECIMAL NUMBER
14506 057220  000207                            RETURN
14507
14508                                   ;*****************************************************************************
14509                                   ;*** INTERLEAVE ***
14510                                   ;*****************************************************************************
14511 057222                            TAG728: PUSH    R1,R5
14512 057226  013701  002100                    MOV     BANK,R1
14513 057232  070127  000004                    MUL     #4,R1
14514 057236                                    SET     NOTAB                   ;INDICATE NO TABLE TO BE PRINTED  NOW
14515 057244                                    TYPE    MSG014
14516 057250  004737  037256                    CALL    TCFIG1
14517 057254  005037  002344                    CLR     NOTAB
14518 057260                                    POP     R5,R1
14519 057264                                    TYPE    MSG014                  ;1 SPACE
14520 057270  000207                            RETURN
14521
14522                                   ;*****************************************************************************
14523                                   ;*** CSR ***
14524                                   ;*****************************************************************************
14525 057272                            TAG738: PUSH    R1,R5
14526 057276  013701  002100                    MOV     BANK,R1
14527 057302  070127  000004                    MUL     #4,R1
14528 057306                                    SET     NOTAB
14529 057314  004737  037576                    CALL    TCFIG3
14530 057320  005037  002344                    CLR     NOTAB
14531 057324                                    POP     R5,R1
14532 057330  000207                            RETURN
14533
14534                                   ;*****************************************************************************
14535                                   ;*** PATTERN ***
14536                                   ;*****************************************************************************
14537 057332                            TAG748: TYPOCS  REALPAT,<TYPE (0-77)>,2,Z
14538 057342  000207                            RETURN
14539
14540                                   ;*****************************************************************************
14541                                   ;*** BANK ***
14542                                   ;*****************************************************************************
14543 057344                            TAG758: TYPOCS  BANK,<TYPE (0 167)>,3
14544 057354  000207                            RETURN
```

```
14546                                   ;*****************************************************************************
14547                                   ;*** MTYPE ***
14548                                   ;*****************************************************************************
14549 057356                            TAG768: PUSH    R1,R5
14550 057362  013701  002100                    MOV     BANK,R1
14551 057366  070127  000004                    MUL     #4,R1
14552 057372                                    SET     NOTAB
14553 057400                                    TYPE    MSG019
14554 057404  004737  037410                    CALL    TCFIG2
14555 057410  005037  002344                    CLR     NOTAB
14556 057414                                    POP     R5,R1
14557 057420  000207                            RETURN
14558
14559                                   ;*****************************************************************************
14560                                   ;*** UNKNOWN DATA ***
14561                                   ;*****************************************************************************
14562 057422                            TAG778: TYPE    MSG061
14563 057426  000207                            RETURN
14564
14565                                   ;*****************************************************************************
14566                                   ;*** PHYSICAL ADDRESS ***
14567                                   ;*****************************************************************************
14568 057430  013737  002032  002036    TAG788: MOV     ADDRESS,PHYADD
14569 057436  162737  060000  002036            SUB     #FIRST,PHYADD
14570 057444  013737  002100  002040            MOV     BANK,PHYADD+2
14571 057452  006237  002040                    ASR     PHYADD+2
14572 057456  103003                            BCC     1$
14573 057460  052737  100000  002036            BIS     #BIT15,PHYADD
14574 057466  012746  002036            1$:     MOV     #PHYADD,-(SP)           ;POINTER TO DOUBLE WORD ON STACK
14575 057472  004737  062756                    CALL    $DB2O                   ;CALL DOUBLE PRECISION CONVERSION ROUTINE
14576 057476  062706  000002                    ADD     #2,SP                   ;FIX STACK
14577 057502                                    TYPE    $OCT8
14578 057506  000207                            RETURN
14579
14580                                   ;*****************************************************************************
14581                                   ;*** OCTAL BYTE ***
14582                                   ;*****************************************************************************
14583 057510                            TAG798: TYPE    MSG018                  ;2 SPACES
14584 057514                                    TYPOCS  @(R1),<TYPE BYTE>,3,Z
14585 057524                                    TYPE    MSG014                  ;SPACE
14586 057530  00020?                            RETURN
```

```
14650 057532                          DETAIL: SUBTST  <<SUBR  DETAILED ERROR REPORT>>
                                      ;*****************************************************************
                                      ;*SUBTEST        SUBR    DETAILED ERROR REPORT
                                      ;*****************************************************************
14651 057532  005237  002214                INC     DETFLAG
14652 057536  022737  000003  002214        CMP     #3,DETFLAG
14653 057544  101473                        BLOS    4$
14654 057546  022737  000002  002214        CMP     #2,DETFLAG
14655 057554  001435                        BEQ     2$
14656 057556                                PUSH    HEADER,MUT
14657 057566                                SET     HEADER
14658 057574  005037  002106                CLR     MUT
14659 057600  010037  002174                MOV     R0,DETR0
14660 057604  012700  002176                MOV     #DETR1,R0
14661 057610  010120                        MOV     R1,(R0)+
14662 057612  010220                        MOV     R2,(R0)+
14663 057614  010320                        MOV     R3,(R0)+
14664 057616  010420                        MOV     R4,(R0)+
14665 057620  010520                        MOV     R5,(R0)+
14666 057622  013720  002022                MOV     ERRSP,(R0)+
14667 057626  013720  002026                MOV     ERRPSW,(R0)+
14668 057632  013700  002174                MOV     DETR0,R0
14669 057636                                SET     NOERROR
14670 057644  104013                        ERROR   +13
14671 057646  000423                        BR      1$
14672 057650                          2$:   PUSH    HEADER,MUT
14673 057660                                SET     HEADER
14674 057666  005037  002106                CLR     MUT
14675 057672                                SET     NOERROR
14676 057700  104031                        ERROR   +31
14677 057702  022737  000001  003720        CMP     #1,PROTYP               ;IS THIS AN 11/44?
14678 057710  001002                        BNE     1$
14679 057712  005037  177766                CLR     CPUERR
14680 057716                          1$:   POP     MUT,HEADER
14681                                       ;WARNING RECURSIVE
14682 057726  004737  057532                CALL    DETAIL
14683 057732  000207                        RETURN
```

```
14666                                           ;SIMULATE CONTROL "T"
14667 057734  004737  061402          41:       CALL    CONTT                   ;DISPLAY DISPLAY INFO
14668
14669                                           ;TYPE CONTENTS OF ALL CSR'S
14670 057740                                    PUSH    CSR,CSRNO,R1
14671 057752                                    TYPE    MSG058
14672 057756                                    TYPE    $CRLF
14673 057762  013701  002220                    MOV     TOTCSRS,R1
14674 057766                                    BEGIN DUMPCSRLOOP
14675 057766                                      FOR CSRNO := #0 TO #36 BY #2
14676 057772  006301                                ASL     R1
14677 057774                                        ON.ERROR
14678 057776  104426                                  READCSR
14679 060000                                          TYPOCT  CSR
14680 060006                                          TYPE    MSG018            ;2 SPACES
14681 060012                                        END ;OF ON.ERROR
14682 060012                                        IF R1 EQ #0 THEN LEAVE DUMPCSRLOOP
14683 060016                                      END ;OF FOR CSRNO
14684 060034                                    END DUMPCSRLOOP
14685 060034                                    POP     R1,CSRNO,CSR
14686
14687                                           ;TYPE STACKS
14688 060046                                    PUSH    R0,R1
14689 060052                                    TYPE    MSG088            ;KERNEL STACK
14690 060056  013701  002536                    MOV     KSTACK,R1
14691 060062  162701  000002                    SUB     #2,R1
14692 060066                                    FOR     R0 := SP TO R1 BY #2
14693 060070                                      TYPE  $CRLF
14694 060074                                      TYPOCT        R0
14695 060100                                      TYPE          MSG018            ;2 SPACES
14696 060104                                      TYPOCT        (R0)
14697 060110                                    END ;OF FOR R0
14698                                           ;SET PREVIOUS MODE TO SUPERVISOR
14699 060120  005737  002430                    TST     NOSUPER
14700 060124  001035                            BNE     DET1
14701 060126  042737  030000  177776            BIC     #BIT13!BIT12,PSW
14702 060134  052737  010000  177776            BIS     #BIT12,PSW
14703 060142  006506                            MFPI    SSP
14704 060144                                    POP     R1,R0
14705 060150                                    TYPE    MSG089            ;SUPERVISOR STACK
14706 060154                                    IF R0 LT #SUPSTK
14707 060162                                      FOR R0 := R0 TO #SUPSTK-2 BY #2
14708 060162                                        TYPE        $CRLF
14709 060166                                        TYPOCT      R0
14710 060172                                        TYPE        MSG018            ;2 SPACES
14711 060176                                        TYPOCT      (R0)
14712 060202                                      END ;OF FOR R0
14713 060214                                    ELSE
14714 060216                                      TYPE  MSG091            ;IS EMPTY
14715 060222                                    END ;OF IF R0
14716                                           ;SET PREVIOUS MODE TO USER
14717 060222  052737  030000  177776  DET1:     BIS     #BIT13!BIT12,PSW
14718 060230  006506                            MFPI    USP
14719 060232                                    POP     R0
14720 060234                                    TYPE    MSG090            ;USER STACK
14721 060240                                    IF R0 LT #USESTK
14722 060246                                      FOR R0 := R0 TO #USESTK-2 BY #2
```

```
14723 060246                                        TYPE        $CRLF
14724 060252                                        TYPOCT      R0
14725 060256                                        TYPE        MSG018          ;; SPACES
14726 060262                                        TYPOCT      (R0)
14727 060266                                      END ;OF FOR R0
14728 060300                                    ELSE
14729 060302                                      TYPE  MSG091          ;IS EMPTY
14730 060306                                    END ;OF IF R0
14731 060306                                    TYPE      $CRLF
14732 060312  005037  002214                    CLR       DETFLAG
14733 060316                                    POP       R0
14734 060320  000207                            RETURN
```

```
14772                                           .SBTTL  ROUTINE BINARY TO OCTAL (ASCII) AND TYPE
14773
14774                                   ;*****************************************************************************
14775                                   ;*THIS ROUTINE IS USED TO CHANGE A 16 BIT BINARY NUMBER TO A 6-DIGIT
14776                                   ;*OCTAL (ASCII) NUMBER AND TYPE IT.
14777                                   ;*$TYPOS - ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
14778                                   ;*CALL:
14779                                   ;*      MOV     NUM, (SP)       ;;NUMBER TO BE TYPED
14780                                   ;*      TYPOS                   ;;CALL FOR TYPEOUT
14781                                   ;*       BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
14782                                   ;*        BYTE  M               ;;M=1 OR 0
14783                                   ;*                                      ;;1=TYPE LEADING ZEROS
14784                                   ;*                                      ;;0=SUPPRESS LEADING ZEROS
14785                                   ;*
14786                                   ;*$TYPON-- -ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
14787                                   ;*$TYPOS OR $TYPOC
14788                                   ;*CALL:
14789                                   ;*      MOV     NUM, (SP)       ;;NUMBER TO BE TYPED
14790                                   ;*      TYPON                   ;;CALL FOR TYPEOUT
14791                                   ;*
14792                                   ;*$TYPOC   ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
14793                                   ;*CALL:
14794                                   ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
14795                                   ;*      TYPOC                   ;;CALL FOR TYPEOUT
14796
14797 060322  017646  000000           $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
14798 060326  116637  000001  060545           MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
14799 060334  112637  060547                   MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
14800 060340  062716  000002                   ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
14801 060344  000406                           BR      $TYPON
14802 060346  112737  000001  060545   $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
14803 060354  112737  000006  060547           MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
14804 060362  112737  000005  060544   $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
14805 060370  010346                           MOV     R3, (SP)        ;;SAVE R3
14806 060372  010446                           MOV     R4,-(SP)        ;;SAVE R4
14807 060374  010546                           MOV     R5,-(SP)        ;;SAVE R5
14808 060376  113704  060547                   MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
14809 060402  005404                           NEG     R4
14810 060404  062704  000006                   ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
14811 060410  110437  060546                   MOVB    R4,$OMODE       ;;SAVE IT FOR USE
14812 060414  113704  060545                   MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
14813 060420  016605  000012                   MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
14814 060424  005003                           CLR     R3              ;;CLEAR THE OUTPUT WORD
14815 060426  006105                   1$:     ROL     R5              ;;ROTATE MSB INTO "C"
14816 060430  000404                           BR      3$              ;;GO DO MSB
14817 060432  006105                   2$:     ROL     R5              ;;FORM THIS DIGIT
14818 060434  006105                           ROL     R5
14819 060436  006105                           ROL     R5
14820 060440  010503                           MOV     R5,R3
14821 060442  006103                   3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
14822 060444  105337  060546                   DECB    $OMODE          ;;TYPE THIS DIGIT?
14823 060450  100016                           BPL     6$              ;;BR IF NO
14824 060452  042703  177770                   BIC     #177770,R3      ;;GET RID OF JUNK
14825 060456  001002                           BNE     4$              ;;TEST FOR 0
14826 060460  005704                           TST     R4              ;;SUPPRESS THIS 0?
14827 060462  001403                           BEQ     5$              ;;BR IF YES
14828 060464  005204                   4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0 S
```

```
 14829 060466  052703  000060            BIS     # 0,R3          ;;MAKE THIS DIGIT ASCII
 14830 060472  052703  000040      5$:   BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
 14831 060476  110337  060542            MOVB    R3,8$           ;;SAVE FOR TYPING
 14832 060502                            TYPE    8$              ;;GO TYPE THIS DIGIT
 14833 060506  105337  060544      6$:   DECB    $OCNT           ;;COUNT BY 1
 14834 060512  003347                    BGT     2$              ;;BR IF MORE TO DO
 14835 060514  002402                    BLT     7$              ;;BR IF DONE
 14836 060516  005204                    INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
 14837 060520  000744                    BR      2$              ;;GO DO THE LAST DIGIT
 14838 060522  012605            7$:     MOV     (SP)+,R5        ;;RESTORE R5
 14839 060524  012604                    MOV     (SP)+,R4        ;;RESTORE R4
 14840 060526  012603                    MOV     (SP)+,R3        ;;RESTORE R3
 14841 060530  016666  000002  000004    MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
 14842 060536  012616                    MOV     (SP)+,(SP)
 14843 060540  000002                    RTI                     ;;RETURN
 14844 060542     000            8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
 14845 060543     000                    .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
 14846 060544     000            $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
 14847 060545     000            $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
 14848 060546  000000            $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
```

```
14850                                           .SBTTL  ROUTINE CONVERT BINARY TO DECIMAL AND TYPE
14851                                   ;******************************************************************************
14852                                   ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
14853                                   ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
14854                                   ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
14855                                   ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
14856                                   ;*REPLACED WITH SPACES.
14857                                   ;*CALL:
14858                                   ;*      MOV     NUM, (SP)       ;;PUT THE BINARY NUMBER ON THE STACK
14859                                   ;*      TYPDS                   ;;GO TO THE ROUTINE
14860 060550                           $TYPDS: PUSH    R0,R1,R2,R3,R5
14861 060562  012746  020200                   MOV     #20200, (SP)    ;;SET BLANK SWITCH AND SIGN
14862 060566  016605  000020                   MOV     20(SP),R5       ;;GET THE INPUT NUMBER
14863 060572  100004                           BPL     1$              ;;BR IF INPUT IS POS.
14864 060574  005405                           NEG     R5              ;;MAKE THE BINARY NUMBER POS.
14865 060576  112766  000055  000001           MOVB    #'- ,1(SP)      ;;MAKE THE ASCII NUMBER NEG.
14866 060604  005000                   1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
14867 060606  012703  060764                   MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
14868 060612  112723  000040                   MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
14869 060616  005002                   2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
14870 060620  016001  060754                   MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
14871 060624  160105                   3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
14872 060626  002402                           BLT     4$              ;;BR IF DONE
14873 060630  005202                           INC     R2              ;;INCREASE THE BCD DIGIT BY 1
14874 060632  000774                           BR      3$
14875 060634  060105                   4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
14876 060636  005702                           TST     R2              ;;CHECK IF BCD DIGIT=0
14877 060640  001002                           BNE     5$              ;;FALL THROUGH IF 0
14878 060642  105716                           TSTB    (SP)            ;;STILL DOING LEADING 0 S?
14879 060644  100407                           BMI     7$              ;;BR IF YES
14880 060646  106316                   5$:     ASLB    (SP)            ;;MSD?
14881 060650  103003                           BCC     6$              ;;BR IF NO
14882 060652  116663  000001  177777           MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
14883 060660  052702  000060           6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
14884 060664  052702  000040           7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
14885 060670  110223                           MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
14886 060672  005720                           TST     (R0)+           ;;JUST INCREMENTING
14887 060674  020027  000010                   CMP     R0,#10          ;;CHECK THE TABLE INDEX
14888 060700  002746                           BLT     2$              ;;GO DO THE NEXT DIGIT
14889 060702  003002                           BGT     8$              ;;GO TO EXIT
14890 060704  010502                           MOV     R5,R2           ;;GET THE LSD
14891 060706  000764                           BR      6$              ;;GO CHANGE TO ASCII
14892 060710  105726                   8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON ZERO?
14893 060712  100003                           BPL     9$              ;;BR IF NO
14894 060714  116663  177777  177776           MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
14895 060722  105013                   9$:     CLRB    (R3)            ;;SET THE TERMINATOR
14896 060724                                   POP     R5,R3,R2,R1,R0
14897 060736                                   TYPE    $DBLK           ;;NOW TYPE THE NUMBER
14898 060742  016666  000002  000004           MOV     2(SP),4(SP)     ;;ADJUST THE STACK
14899 060750  012616                           MOV     (SP)+,(SP)
14900 060752  000002                           RTI                     ;;RETURN TO USER
14901 060754  023420                   $DTBL:  10000.
14902 060756  001750                           1000.
14903 060760  000144                           100.
14904 060762  000012                           10.
14905 060764  000000  000000  000000   $DBLK:  .WORD   0,0,0,0
      060772  000000
```

```
14907                                   .SBTTL  ROUTINE TTY INPUT
14908                                   ;**************************************************************************
14909                                   ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
14910                                   ;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
14911                                   ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
14912                                   ;*WHEN OPERATING IN TTY FLAG MODE.
14913                                           .ENABLE LSB
14914 060774                            $CKSWR:
14920 060774  005737  053554                      TST     XOCHAR          ;;SOMETHING THERE?
14921 061000  001406                              BEQ     NOCH            ;; GO ON IF NOT
14922 061002  013746  053554                      MOV     XOCHAR, (SP)    ;; USE IT
14923 061006  005037  053554                      CLR     XOCHAR
14924 061012  000137  061034                      JMP     CONTS1
14925 061016  105777  121562             NOCH:    TSTB    @#TKS           ;;CHAR THERE?
14926 061022  100130                              BPL     12$             ;;IF NO, DON'T WAIT AROUND
14927 061024  117746  121556                      MOVB    @#TKB,-(SP)     ;;SAVE THE CHAR
14928 061030  042716  177600                      BIC     #^C177,(SP)     ;;STRIP-OFF THE ASCII
14929 061034  022716  000006             CONTS1:  CMP     #6,(SP)         ;IS IT CONTROL F?
14930 061040  001002                              BNE     1$              ;NO SKIP
14931 061042  004737  045406                      CALL    FIELDSERVICE
14932 061046  022716  000024             1$:      CMP     #24,(SP)        ;IS IT CONTROL T?
14933 061052  001002                              BNE     16$             ;NO   SKIP
14934 061054  004737  061402                      CALL    CONTT           ;YES - CALL CONTROL T ROUTINE
14935 061060  022716  000003             16$:     CMP     #3,(SP)         ;IS IT CONTROL C?
14936 061064  001454                              BEQ     5$              ;YES EXIT *****NOTE***** STACK IS SCREWED UP!
14937 061066  022716  000023             2$:      CMP     #23,(SP)        ;IS IT CONTROL S?
14938 061072  001002                              BNE     17$             ;NO - SKIP
14939 061074  004737  061456                      CALL    CONTS           ;YES - CALL CONTROL S ROUTINE
14940 061100  022716  000013             17$:     CMP     #13,(SP)        ;IS IT CONTROL K?
14941 061104  001005                              BNE     6$              ;NO - SKIP
14942 061106                                      TYPE    $CNTLK          ;TYPE A ^K
14943 061112  013706  002142                      MOV     CTLKVEC,SP      ;RESET KSP TO AFTER PATTERN EXEC ROUTINE
14944 061116  000207                              RETURN                  ;RETURN TO PATTERN EXEC ROUTINE
14945 061120  022737  000176  002600  6$:        CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED?
14946 061126  001067                              BNE     CKEND           ;;BRANCH IF NO
14947 061130  022716  000007                      CMP     #7,(SP)         ;;IS IT A CONTROL G?
14948 061134  001064                              BNE     CKEND           ;;NO, RETURN TO USER
14949 061136  005737  002060                      TST     $AUTO           ;ARE WE RUNNING IN AUTO-MODE?
14950 061142  001061                              BNE     CKEND           ;BRANCH IF YES
14951 061144                                      TYPE    $CNTLG          ;;ECHO THE CONTROL-G (^G)
14952 061150                            $GTSWR:   TYPE    $MSWR           ;;TYPE CURRENT CONTENTS
14953 061154                                      TYPOCT  @SWR            ;;OF THE SWR
14954 061162                                      TYPE    $MNEW           ;;PROMPT FOR NEW SWR
14955 061166  005046                      3$:     CLR     -(SP)           ;;CLEAR COUNTER
14956 061170  005046                              CLR     -(SP)           ;;THE NEW SWR
14957 061172  105777  121406             4$:      TSTB    @#TKS           ;;CHAR THERE?
14958 061176  100375                              BPL     4$              ;;IF NOT TRY AGAIN
14959 061200  117746  121402                      MOVB    @#TKB,-(SP)     ;;PICK UP CHAR
14960 061204  042716  177600                      BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
14961 061210  021627  000003                      CMP     (SP),#3         ;;IS IT A CONTROL-C?
14962 061214  001006                              BNE     7$              ;;BRANCH IF NOT
14963 061216                            5$:       TYPE    $CNTLC          ;;YES, ECHO CONTROL C (^C)
14964 061222  062706  000006                      ADD     #6,SP           ;;CLEAN UP STACK
14965 061226  000137  045030                      JMP     BOOT            ;;CONTROL-C RESTART
14966 061232  021627  000025             7$:      CMP     (SP),#25        ;;IS IT A CONTROL U?
14967 061236  001005                              BNE     9$              ;;BRANCH IF NOT
14968 061240                                      TYPE    $CNTLU          ;;YES, ECHO CONTROL-U (^U)
```

```
14969 061244  062706  000006          8$:     ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
14970 061250  000746                  BR      3$              ;;LET'S TRY IT AGAIN
14971 061252  021627  000015          9$:     CMP     (SP),#15        ;;IS IT A <CR>?
14972 061256  001016                  BNE     13$             ;;BRANCH IF NO
14973 061260  005766  000004                  TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
14974 061264  001403                  BEQ     10$             ;;BRANCH IF YES
14975 061266  016677  000002 121304          MOV     2(SP),@SWR      ;;SAVE NEW SWR
14976 061274  062706  000006          10$:    ADD     #6,SP           ;;CLEAR UP STACK
14977 061300                          TYPE    $CRLF           ;;ECHO <CR> AND <LF>
14978 061304  000002                  12$:    RTI                     ;;RETURN
14979 061306  062706  000002          CKEND:  ADD     #2,SP           ;FIX STACK
14980 061312  000002                  RTI                     ;RETURN
14981 061314  004737  053556          13$:    CALL    $TYPEC          ;;ECHO CHAR
14982 061320  021627  000060                  CMP     (SP),#60        ;;CHAR < 0?
14983 061324  002420                  BLT     15$             ;;BRANCH IF YES
14984 061326  021627  000067                  CMP     (SP),#67        ;;CHAR > 7?
14985 061332  003015                  BGT     15$             ;;BRANCH IF YES
14986 061334  042726  000060                  BIC     #60,(SP)+       ;;STRIP-OFF ASCII
14987 061340  005766  000002                  TST     2(SP)           ;;IS THIS THE FIRST CHAR
14988 061344  001403                  BEQ     14$             ;;BRANCH IF YES
14989 061346  006316                  ASL     (SP)            ;;NO, SHIFT PRESENT
14990 061350  006316                  ASL     (SP)            ;;    CHAR OVER TO MAKE
14991 061352  006316                  ASL     (SP)            ;;    ROOM FOR NEW ONE.
14992 061354  005266  000002          14$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
14993 061360  056616  177776                  BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
14994 061364  000702                  BR      4$              ;;GET THE NEXT ONE
14995 061366                          15$:    TYPE    $QUES           ;;TYPE ?<CR><LF>
14996 061372  000724                  BR      8$              ;;SIMULATE CONTROL U
14997 061374    136   113    015     $CNTLK: .ASCIZ  /^K/<15><12>    ;CONTROL K ASCII STRING
      061377    012   000
14998                                 .EVEN
14999                                 .DSABL  LSB
```

```
15002 061402                          CONTT:  SUBTST  <<CONTROL T>>
                                      ;*****************************************************************************
                                      ;*SUBTEST        CONTROL T
                                      ;*****************************************************************************
15003 061402                                  PUSH    R0
15004 061404                                  TYPE    $CRLF
15014 061410                                    IF RLFLAG IS TRUE
15015 061416                                      TYPE        MSG092          ;RELOCATED
15016 061422                                    END ;OF IF RLFLAG
15017 061422                                    TYPE  MSG093                    ;BANK=
15018 061426                                    TYPOCS        BANK,,3           ;TYPE 3 DIGITS
15019 061436                                    TYPE  MSG095                    ;PAT=
15020 061442                                    TYPOCS        REALPAT,,2        ;TYPE 2 DIGITS
15024 061452                                  POP     R0
15025 061454  000207                            RETURN
15026
15027 061456                          CONTS:  SUBTST  <<CONTROL S & CONTROL Q>>
                                      ;*****************************************************************************
                                      ;*SUBTEST        CONTROL S & CONTROL Q
                                      ;*****************************************************************************
15028 061456                                  POP     R0                ;GET RID OF RETURN ADDRESS FROM STACK
15029 061460  105777  121120          CONTS2: TSTB    @#TKS             ;WAIT FOR CHARACTER
15030 061464  100375                          BPL     CONTS2
15031 061466  117716  121114                  MOVB    @#TKB,(SP)        ;REPLACE OVER OLD CHARACTER ON STACK
15032 061472  042716  177600                  BIC     #^C177,(SP)       ;STRIP ALL BUT ASCII
15033 061476                                  IF (SP) EQ #21            ;IF IT IS A CONTROL Q
15034 061504  000137  061034                    JMP   CONTS1
15035 061510                                  ELSE
15036 061512  000762                            BR    CONTS2
15037 061514                                  END ;OF IF (SP)
```

```
15039                                   ;*************************************************************************
15040                                   ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
15041                                   ;*CALL:
15042                                   ;*      RDCHR                           ;;INPUT A SINGLE CHARACTER FROM THE TTY
15043                                   ;*      RETURN HERE                     ;;CHARACTER IS ON THE STACK
15044                                   ;*                                      ;;WITH PARITY BIT STRIPPED OFF
15045                                   ;
15046
15047 061514  011646                   $RDCHR: MOV     (SP),-(SP)              ;;PUSH DOWN THE PC
15048 061516  016666  000004  000002           MOV     4(SP),2(SP)             ;;SAVE THE PS
15049 061524  105777  121054           1$:     TSTB    @$TKS                   ;;WAIT FOR
15050 061530  100375                           BPL     1$                      ;;A CHARACTER
15051 061532  117766  121050  000004           MOVB    @$TKB,4(SP)             ;;READ THE TTY
15052 061540  042766  177600  000004           BIC     #^C<177>,4(SP)          ;;GET RID OF JUNK IF ANY
15053 061546  026627  000004  000023           CMP     4(SP),#23               ;;IS IT A CONTROL S?
15054 061554  001013                           BNE     3$                      ;;BRANCH IF NO
15055 061556  105777  121022           2$:     TSTB    @$TKS                   ;;WAIT FOR A CHARACTER
15056 061562  100375                           BPL     2$                      ;;LOOP UNTIL ITS THERE
15057 061564  117746  121016                   MOVB    @$TKB,-(SP)             ;;GET CHARACTER
15058 061570  042716  177600                   BIC     #^C177,(SP)             ;;MAKE IT 7-BIT ASCII
15059 061574  022627  000021                   CMP     (SP)+,#21               ;;IS IT A CONTROL-Q?
15060 061600  001366                           BNE     2$                      ;;IF NOT DISCARD IT
15061 061602  000750                           BR      1$                      ;;YES, RESUME
15062 061604  026627  000004  000021   3$:     CMP     4(SP),#21               ;;IS IT A RANDOM CONTROL Q?     ;R C
15063 061612  001744                           BEQ     1$                      ;;BRANCH BACK IF SO             ;R C
15064 061614  026627  000004  000140           CMP     4(SP),#140              ;;IS IT UPPER CASE?
15065 061622  002407                           BLT     4$                      ;;BRANCH IF YES
15066 061624  026627  000004  000175           CMP     4(SP),#175              ;;IS IT A SPECIAL CHAR?
15067 061632  003003                           BGT     4$                      ;;BRANCH IF YES
15068 061634  042766  000040  000004           BIC     #40,4(SP)               ;;MAKE IT UPPER CASE
15069 061642  000002                   4$:     RTI                             ;;GO BACK TO USER
15070                                   ;*************************************************************************
15071                                   ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
15072                                   ;*CALL:
15073                                   ;*      RDLIN                           ;;INPUT A STRING FROM THE TTY
15074                                   ;*      RETURN HERE                     ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
15075                                   ;*                                      ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
15076 061644  010346                   $RDLIN: MOV     R3,-(SP)                ;;SAVE R3
15077 061646  005046                           CLR     -(SP)                   ;;CLEAR THE RUBOUT KEY
15078 061650  012703  062142           1$:     MOV     #$TTYIN,R3              ;;GET ADDRESS
15079 061654  022703  062166           2$:     CMP     #$TTYIN+20.,R3          ;;BUFFER FULL?
15080 061660  101477                           BLOS    8$                      ;;BR IF YES
15081 061662  104411                           RDCHR                           ;;GO READ ONE CHARACTER FROM THE TTY
15082 061664  112613                           MOVB    (SP)+,(R3)              ;;GET CHARACTER
15083 061666  122713  000003                   CMPB    #3,(R3)                 ;;IS IT A CONTROL-C?
15084 061672  001016                           BNE     3$                      ;;BRANCH IF NO
15085 061674                                   TYPE    $CNTLC                  ;;TYPE A CONTROL-C (^C)
15086 061700  005726                           TST     (SP)+                   ;;CLEAN RUBOUT KEY OFF OF THE STACK
15087 061702  012603                           MOV     (SP)+,R3                ;;RESTORE R3
15088 061704  032777  000400  120666           BIT     #BIT8,@SWR              ;;IS THERE A HALT FLAG SET IN THE SWR?
15089 061712  001404                           BEQ     11$                     ;;BRANCH IF NOT TO BOOT ROUTINE
15090 061714  005037  002372                   CLR     STOPOK                  ;;GET READY TO HALT PROGRAM
15091 061720  000137  045136                   JMP     EXIT                    ;;GO HALT PROGRAM
15092 061724  000137  045030           11$:    JMP     BOOT                    ;;GOTO CONTROL-C RESTART
15093 061730  122713  000177           3$:     CMPB    #177,(R3)               ;;IS IT A RUBOUT
15094 061734  001022                           BNE     5$                      ;;BR IF NO
15095 061736  005716                           TST     (SP)                    ;;IS THIS THE FIRST RUBOUT?
```

```
15096 061740 001007                        BNE     4$              ;;BR IF NO
15097 061742 112737 000134 062140          MOVB    #'\,10$         ;;TYPE A BACK SLASH
15098 061750                               TYPE    10$
15099 061754 012716 177777                 MOV     #-1,(SP)        ;;SET THE RUBOUT KEY
15100 061760 005303                  4$:   DEC     R3              ;;BACKUP BY ONE
15101 061762 020327 062142                 CMP     R3,#$TTYIN      ;;STACK EMPTY?
15102 061766 103434                        BLO     8$              ;;BR IF YES
15103 061770 111337 062140                 MOVB    (R3),10$                ;;SETUP TO TYPEOUT THE DELETED CHAR.
15104 061774                               TYPE    10$             ;;GO TYPE
15105 062000 000725                        BR      2$              ;;GO READ ANOTHER CHAR.
15106 062002 005716                  5$:   TST     (SP)            ;;RUBOUT KEY SET?
15107 062004 001406                        BEQ     6$              ;;BR IF NO
15108 062006 112737 000134 062140          MOVB    #'\,10$         ;;TYPE A BACK SLASH
15109 062014                               TYPE    10$
15110 062020 005016                        CLR     (SP)            ;;CLEAR THE RUBOUT KEY
15111 062022 122713 000025           6$:   CMPB    #25,(R3)        ;;IS CHARACTER A CTRL U?
15112 062026 001003                        BNE     7$              ;;BR IF NO
15113 062030                               TYPE    $CNTLU          ;;TYPE A CONTROL "U'
15114 062034 000705                        BR      1$              ;;GO START OVER
15115 062036 122713 000022           7$:   CMPB    #22,(R3)        ;;IS CHARACTER A "^R ?
15116 062042 001011                        BNE     9$              ;;BRANCH IF NO
15117 062044 105013                        CLRB    (R3)            ;;CLEAR THE CHARACTER
15118 062046                               TYPE    $CRLF           ;;TYPE A "CR" & "LF"
15119 062052                               TYPE    $TTYIN          ;;TYPE THE INPUT STRING
15120 062056 000676                        BR      2$              ;;GO PICKUP ANOTHER CHACTER
15121 062060                         8$:   TYPE    $QUES           ;;TYPE A '?'
15122 062064 000671                        BR      1$              ;;CLEAR THE BUFFER AND LOOP
15123 062066 111337 062140           9$:   MOVB    (R3),10$                ;;ECHO THE CHARACTER
15124 062072                               TYPE    10$
15125 062076 122723 000015                 CMPB    #15,(R3)+       ;;CHECK FOR RETURN
15126 062102 001264                        BNE     2$              ;;LOOP IF NOT RETURN
15127 062104 105063 177777                 CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
15128 062110                               TYPE    $LF             ;;TYPE A LINE FEED
15129 062114 005726                        TST     (SP)+           ;;CLEAN RUBOUT KEY FROM THE STACK
15130 062116 012603                        MOV     (SP)+,R3        ;;RESTORE R3
15131 062120 011646                        MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
15132 062122 016666 000004 000002          MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
15133 062130 012766 062142 000004          MOV     #$TTYIN,4(SP)
15134 062136 000002                        RTI                     ;;RETURN
15135 062140    000                  10$:  .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
15136 062141    000                        .BYTE   0               ;;TERMINATOR
15137 062142 000024                $TTYIN: .REPT   20.             ;;RESERVE SIZE BYTES FOR TTY INPUT
15140 062166    136    103    015  $CNTLC: .ASCIZ  /^C/<15><12>    ;;CONTROL "C"
      062171    012    000
15141 062173    136    125    015  $CNTLU: .ASCIZ  /^U/<15><12>    ;;CONTROL "U"
      062176    012    000
15142 062200    136    107    015  $CNTLG: .ASCIZ  /^G/<15><12>    ;;CONTROL "G"
      062203    012    000
15143 062205    015    012    123  $MSWR:  .ASCIZ  <15><12>/SWR = /
      062210    127    122    040
      062213    075    040    000
15144 062216    040    040    116  $MNEW:  .ASCIZ  /  NEW = /
      062221    105    127    040
      062224    075    040    000
15145                                      .EVEN
```

```
ROUTINE READ AN OCTAL NUMBER FROM THE TTY

                                                .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
062230  011646                  $RDOCT: MOV     (SP), (SP)              ;;PROVIDE SPACE FOR THE
062232  016666  000004  000002          MOV     4(SP),2(SP)             ;;INPUT NUMBER
062240                                  PUSH    R0,R1,R2
062246  104412                  1$:     RDLIN                           ;;READ AN ASCIZ LINE
062250  012600                          MOV     (SP)+,R0                ;;GET ADDRESS OF 1ST CHARACTER
062252  010037  062356                  MOV     R0,5$                   ;;AND SAVE IT
062256  005001                          CLR     R1                      ;;CLEAR DATA WORD
062260  005002                          CLR     R2
062262  112046                  2$:     MOVB    (R0)+, (SP)             ;;PICKUP THIS CHARACTER
062264  001420                          BEQ     3$                      ;;IF ZERO GET OUT
062266  122716  000060                  CMPB    #'0,(SP)                ;;MAKE SURE THIS CHARACTER
062272  003026                          BGT     4$                      ;;IS AN OCTAL DIGIT
062274  122716  000067                  CMPB    #'7,(SP)
062300  002423                          BLT     4$
062302  006301                          ASL     R1                      ;;*2
062304  006102                          ROL     R2
062306  006301                          ASL     R1                      ;;*4
062310  006102                          ROL     R2
062312  006301                          ASL     R1                      ;;*8
062314  006102                          ROL     R2
062316  042716  177770                  BIC     #^C7,(SP)               ;;STRIP THE ASCII JUNK
062322  062601                          ADD     (SP)+,R1                ;;ADD IN THIS DIGIT
062324  000756                          BR      2$                      ;;LOOP
062326  005726                  3$:     TST     (SP)+                   ;;CLEAN TERMINATOR FROM STACK
062330  010166  000012                  MOV     R1,12(SP)               ;;SAVE THE RESULT
062334  010237  062376                  MOV     R2,$HIOCT
062340                                  POP     R2,R1,R0
062346  000002                          RTI                             ;;RETURN
062350  005726                  4$:     TST     (SP)+                   ;;CLEAN PARTIAL FROM STACK
062352  105010                          CLRB    (R0)                    ;;SET A TERMINATOR
062354                                  TYPE                            ;;TYPE UP THRU THE BAD CHAR.
062356  000000                  5$:     .WORD   0
062360                                  TYPE    MSG062                  ;INPUT MUST BE A
062364                                  TYPE    MSG063                  ;N OCTAL
062370                                  TYPE    MSG064                  ;NUMBER
062374  000724                          BR      1$                      ;;TRY AGAIN
062376  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
```

```
15204                                   ;+POSITIVE 32767 TO NEGATIVE 32768.
15205                                   ;+CALL:
15206                                   ;+      RDDEC                   ;;READ A DECIMAL NUMBER
15207                                   ;+      RETURN HERE             ;;NUMBER IS ON TOP OF THE STACK
15208                                   ;
15209
15210 062400  011646                   $RDDEC: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR
15211 062402  016666  000004  000002           MOV     4(SP),2(SP)     ;;THE INPUT NUMBER
15212 062410                                   PUSH    R0,R1,R2
15213 062416  104412                   1$:     RDLIN                   ;;READ AN ASCIZ LINE
15214 062420  012600                           MOV     (SP)+,R0        ;;ADDRESS OF 1ST CHAR.
15215 062422  010037  062546                   MOV     R0,6$           ;;SAVE INCASE OF BAD INPUT
15216 062426  005046                           CLR     -(SP)           ;;CLEAR DATA WORD
15217 062430  005002                           CLR     R2              ;;SIGN SET POSITIVE
15218 062432  122710  000055                   CMPB    #'-,(R0)        ;;SEE IF A MINUS SIGN WAS TYPED
15219 062436  001001                           BNE     2$              ;;BR IF NO MINUS SIGN
15220 062440  112002                           MOVB    (R0)+,R2        ;;SAVE FOR LATER USE
15221 062442  112001                   2$:     MOVB    (R0)+,R1        ;;PICKUP THIS CHARACTER
15222 062444  001424                           BEQ     3$              ;;GET OUT IF ZERO
15223 062446  122701  000060                   CMPB    #'0,R1          ;;MAKE SURE THIS CHARACTER
15224 062452  003032                           BGT     5$              ;;IS A DIGIT BETWEEN 0 & 9
15225 062454  122701  000071                   CMPB    #'9,R1
15226 062460  002427                           BLT     5$
15227 062462  032716  170000                   BIT     #^C7777,(SP)    ;;DON'T LET NUMBER GET TO BIG
15228 062466  001024                           BNE     5$              ;;BR IF NUMBER WOULD OVERFLOW
15229 062470  006316                           ASL     (SP)            ;;*2
15230 062472  011646                           MOV     (SP),-(SP)      ;;SAVE FOR LATER
15231 062474  006316                           ASL     (SP)            ;;*4
15232 062476  006316                           ASL     (SP)            ;;*8.
15233 062500  062616                           ADD     (SP)+,(SP)      ;;*10.
15234 062502  102416                           BVS     5$              ;;OVERFLOW ISN'T ALLOWED
15235 062504  162701  000060                   SUB     #'0,R1          ;;STRIP AWAY THE ASCII JUNK
15236 062510  060116                           ADD     R1,(SP)         ;;ADD IN THIS DIGIT
15237 062512  102412                           BVS     5$              ;;OVERFLOW ISN'T ALLOWED
15238 062514  000752                           BR      2$              ;;LOOP
15239 062516  005702                   3$:     TST     R2              ;;CHECK IF NUMBER IS NEG
15240 062520  001401                           BEQ     4$              ;;BR IF NO
15241 062522  005416                           NEG     (SP)            ;;YES--NEGATE THE NUMBER
15242 062524  012666  000012           4$:     MOV     (SP)+,12(SP)    ;;SAVE THE RESULT
15243 062530                                   POP     R2,R1,R0
15244 062536  000002                           RTI                     ;;RETURN
15245
15246 062540  005726                   5$:     TST     (SP)+           ;;CLEAN PARTIAL NUMBER FROM STACK
15247 062542  105010                           CLRB    (R0)            ;;SET A TERMINATOR
15248 062544                                   TYPE                    ;;TYPE THE INPUT UP TO BAD CHAR.
15249 062546  000000                   6$:     .WORD   0               ;;POINTER GOES HERE
15250 062550                                   TYPE    MSG062          ;INPUT MUSST BE A
15251 062554                                   TYPE    MSG065          ;DECIMAL
15252 062560                                   TYPE    MSG064          ;NUMBER
15253 062564  000714                           BR      1$              ;;TRY AGAIN
```

```
15255                                           .SBTTL  ROUTINE SAVE AND RESTORE R0 R5
15256
15257                                   ;*********************************************************************
15258                                   ;*SAVE R0-R5
15259                                   ;*CALL:
15260                                   ;*      SAVREG
15261                                   ;*UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
15262                                   ;*
15263                                   ;*TOP-  -(*16)
15264                                   ;*  *2-  (*18)
15265                                   ;*  *4-  R5
15266                                   ;*  *6---R4
15267                                   ;*  *8-  R3
15268                                   ;*  *10 - R2
15269                                   ;*  *12   R1
15270                                   ;*  *14---R0
15271
15272 062566                            $SAVREG:
15273 062566                                    PUSH    R0,R1,R2,R3,R4,R5
15274 062602  016646  000022                    MOV     22(SP),-(SP)    ;;SAVE PS OF MAIN FLOW
15275 062606  016646  000022                    MOV     22(SP),-(SP)    ;;SAVE PC OF MAIN FLOW
15276 062612  016646  000022                    MOV     22(SP),-(SP)    ;;SAVE PS OF CALL
15277 062616  016646  000022                    MOV     22(SP),-(SP)    ;;SAVE PC OF CALL
15278 062622  000002                            RTI
15279
15280                                   ;*RESTORE R0-R5
15281                                   ;*CALL:
15282                                   ;*      RESREG
15283 062624                            $RESREG:
15284 062624  012666  000022                    MOV     (SP)+,22(SP)    ;;RESTORE PC OF CALL
15285 062630  012666  000022                    MOV     (SP)+,22(SP)    ;;RESTORE PS OF CALL
15286 062634  012666  000022                    MOV     (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
15287 062640  012666  000022                    MOV     (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
15288 062644                                    POP     R5,R4,R3,R2,R1,R0
15289 062660  000002                            RTI
```

```
15291                                           .SBTTL  ROUTINE RANDOM NUMBER GENERATOR
15292
15293                                   ;*******************************************************************************
15294                                   ;*THIS ROUTINE IS A DOUBLE PRECISION PSEUDO RANDOM NUMBER GENERATOR
15295                                   ;*WITH A RANGE OF 0 TO 2**(-33) 1.
15296                                   ;*CALL:
15297                                   ;*      CALL    $RAND           ;;CALL THE ROUTINE
15298                                   ;*      RETURN                  ;;RETURN HERE THE RANDOM
15299                                   ;*                              ;;NUMBER WILL BE IN
15300                                   ;*                              ;;$HINUM,$LONUM
15301
15302 062662                           $RAND:  PUSH    R0,R1,R2
15303 062670  013700  002546                   MOV     SEEDLO,R0       ;SET R0 WITH LOW
15304 062674  013701  002544                   MOV     SEEDHI,R1       ;SET R1 WITH HIGH
15305 062700  012702  000007                   MOV     #7,R2           ;SET SHIFT COUNT
15306 062704  006300                   1$:     ASL     R0              ;;SHIFT R0 LEFT AND
15307 062706  006101                           ROL     R1              ;;ROTATE CARRY INTO R1 AND
15308 062710  077203                           SOB     R2,1$
15309 062712  063700  002546                   ADD     SEEDLO,R0       ;ADD NUMBER TO MAKE X 129
15310 062716  005501                           ADC     R1              ;;PROPOGATE CARRY
15311 062720  063701  002544                   ADD     SEEDHI,R1       ;ADD NUMBER TO MAKE X 129
15312 062724  062700  001057                   ADD     #1057,R0        ;;ADD LOW CONSTANT
15313 062730  005501                           ADC     R1              ;;PROPOGATE CARRY
15314 062732  062701  047401                   ADD     #47401,R1       ;;ADD HIGH CONSTANT
15315 062736  010037  002546                   MOV     R0,SEEDLO       ;SAVE R0
15316 062742  010137  002544                   MOV     R1,SEEDHI       ;SAVE R1
15317 062746                                   POP     R2,R1,R0
15318 062754  000207                           RETURN
```

```
15321                                   .SBTTL  ROUTINE DOUBLE LENGTH BINARY TO OCTAL ASCII CONVERT
15322                                   ;*********************************************************************
15323                                   ;*THIS ROUTINE WILL CONVERT A 32-BIT UNSIGNED BINARY NUMBER TO AN
15324                                   ;*UNSIGNED OCTAL ASCIZ NUMBER.
15325                                   ;*CALL
15326                                   ;*      MOV     #PNTR, (SP)     ;;POINTER TO LOW WORD OF BINARY NUMBER
15327                                   ;*      CALL    $D820           ;;CALL THE ROUTINE
15328                                   ;*      RETURN                  ;;THE ADDRESS OF THE FIRST ASCIZ CHAR. IS ON THE STACK
15329
15330
15331 062756  104415                   $D820:  SAVREG                  ;;SAVE ALL REGISTERS
15332 062760  016601  000002                   MOV     2(SP),R1        ;;PICKUP THE POINTER TO LOW WORD
15333 062764  012705  063075                   MOV     #$OCTVL+13.,R5  ;;POINTER TO DATA TABLE
15334 062770  012704  000014                   MOV     #12.,R4         ;;DO ELEVEN CHARACTERS
15335 062774  012703  177770                   MOV     #^C7,R3         ;;MASK
15336 063000  012100                           MOV     (R1)+,R0        ;;LOWER WORD
15337 063002  012101                           MOV     (R1)+,R1        ;;HIGH WORD
15338 063004  005002                           CLR     R2              ;;TERMINATOR
15339 063006  110245                   1$:     MOVB    R2,-(R5)        ;;PUT CHARACTER IN DATA TABLE
15340 063010  010002                           MOV     R0,R2           ;;GET THIS DIGIT
15341 063012  005304                           DEC     R4              ;;COUNT THIS CHARACTER
15342 063014  003007                           BGT     3$              ;;BR IF NOT THE LAST DIGIT
15343 063016  001405                           BEQ     2$              ;;BR IF IT IS THE LAST DIGIT
15344 063020  005205                           INC     R5              ;;ALL DIGITS DONE ADJUST POINTER FOR FIRST
15345 063022  010566  000002                   MOV     R5,2(SP)        ;;ASCIZ CHAR. & PUT IT ON THE STACK
15346 063026  104416                           RESREG                  ;;RESTORE ALL REGISTERS
15347 063030  000207                           RETURN                  ;;RETURN TO USER
15348 063032  006203                   2$:     ASR     R3              ;;POSITION THE MASK FOR THE LAST DIGIT
15349 063034  006001                   3$:     ROR     R1              ;;POSITION THE BINARY NUMBER FOR
15350 063036  006000                           ROR     R0              ;;      THE NEXT OCTAL DIGIT
15351 063040  006001                           ROR     R1
15352 063042  006000                           ROR     R0
15353 063044  006001                           ROR     R1
15354 063046  006000                           ROR     R0
15355 063050  040302                           BIC     R3,R2           ;;MASK OUT ALL JUNK
15356 063052  062702  000060                   ADD     #'0,R2          ;;MAKE THIS CHAR. ASCII
15357 063056  000753                           BR      1$              ;;GO PUT IT IN THE DATA TABLE
15358 063060  000016                   $OCTVL: .REPT   14.             ;;RESERVE DATA TABLE
15361         063064                   $OCT8=.$OCTVL+4                 ;POINTER TO 11 DIGIT NUMBER
```

```
15363                                           .SBTTL  TABLES
15364
15365                                           .SBTTL  APT MAILBOX ETABLE
15366 063076                            $MAIL:
15367 063076  000000                    $MSGTY: .WORD   0       ;;MESSAGE TYPE CODE
15368 063100  000000                    $FATAL: .WORD   0       ;;FATAL ERROR NUMBER (ERROR PC)
15369 063102  000000                    $TESTN: .WORD   0       ;;TEST PATTERN NUMBER
15370 063104  000000                    $PASS:  .WORD   0       ;;PASS COUNT
15371 063106  000000                    $DEVCT: .WORD   0       ;;DEVICE COUNT
15372 063110  000000                    $UNIT:  .WORD   0       ;;I/O UNIT NUMBER
15373 063112  000000                    $MSGAD: .WORD   0       ;;MESSAGE ADDRESS
15374 063114  000000                    $MSGLG: .WORD   0       ;;MESSAGE LENGTH
15375 063116                            $ETABLE:                ;;APT ENVIRONMENT TABLE
15376 063116      000                   $ENV:   .BYTE   0       ;;ENVIRONMENT BYTE      ;SET TO A 1 FOR APT AUTO MODE
15377                                   ;NOTE:  IF BIT #7 IS SET IN $ENVM THE TABLE BELOW (BEGINNING AT $MAMS1 AND
15378                                   ;       ENDING AT $MADR4) MUST BE FILLED IN TO INDICATE THE PROPER AMOUNT OF
15379                                   ;       EACH TYPE OF MEMORY.
15380 063117      000                   $ENVM:  .BYTE   0       ;ENVIRONMENT MODE
15381                                                           ;BIT7(200)=USE APT SIZE INFO ;BIT5(40)=NO CONSOLE
15382 063120  000101                    $SWREG: .WORD   101     ;;APT SWITCH REGISTER
15383 063122  000000                    $USWR:  .WORD   0       ;USED TO LIMIT THE NUMBER OF PASSES
15384 063124  000000                    $CPUOP: .WORD   0       ;;CPU TYPE,OPTIONS
1538?                                   ;*                      BITS 15-11=CPU TYPE
15386                                   ;*                              11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
15387                                   ;*                              11/70=06,PDQ=07,Q=10
15388                                   ;*                      BIT 10=REAL TIME CLOCK
15389                                   ;*                      BIT  9=FLOATING POINT PROCESSOR
15390                                   ;*                      BIT  8=MEMORY MANAGEMENT
15391 063126      001                   $MAMS1: .BYTE   1       ;;HIGH ADDRESS,M.S. BYTE ;DEFAULT = 64K
15392 063127      004                   $MTYP1: .BYTE   4       ;;MEM. TYPE,BLK#1
15393                                   ;*                      MEM.TYPE BYTE   -   (HIGH BYTE)
15394                                   ;*                              900 NSEC CORE=001
15395                                   ;*                              300 NSEC BIPOLAR=002
15396                                   ;*                              PARITY MOS=003
15397                                   ;*                              ERROR CORRECTING MOS=004
15398 063130  177776                    $MADR1: .WORD   177776  ;;HIGH ADDRESS,BLK#1
15399                                   ;*                        MEM.LAST ADDR.-3 BYTES,THIS WORD AND LOW OF "TYPE  ABOVE
15400 063132      000                   $MAMS2: .BYTE   0       ;;HIGH ADDRESS,M.S. BYTE
15401 063133      000                   $MTYP2: .BYTE   0       ;;MEM.TYPE,BLK#2
15402 063134  000000                    $MADR2: .WORD   0       ;;MEM.LAST ADDRESS,BLK#2
15403 063136      000                   $MAMS3: .BYTE   0       ;;HIGH ADDRESS,M.S.BYTE
15404 063137      000                   $MTYP3: .BYTE   0       ;;MEM.TYPE,BLK#3
15405 063140  000000                    $MADR3: .WORD   0       ;;MEM.LAST ADDRESS,BLK#3
15406 063142      000                   $MAMS4: .BYTE   0       ;;HIGH ADDRESS,M.S.BYTE
15407 063143      000                   $MTYP4: .BYTE   0       ;;MEM.TYPE,BLK#4
15408 063144  000000                    $MADR4: .WORD   0       ;;MEM.LAST ADDRESS,BLK#4
15409 063146  000000                    $VECT1: .WORD   0       ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
15410 063150  000000                    $VECT2: .WORD   0       ;;INTERRUPT VECTOR#2BUS PRIORITY#2
15411 063152  000000                    $BASE:  .WORD   0       ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
15412 063154  000000                    $DEVM:  .WORD   0       ;;DEVICE MAP
15413
15414 063156  000000                    $CDW1:  .WORD   0
15415 063160  000000                    $CDW2:  .WORD   0
```

```
15417                                   ;THE FOLLOWING LOCATIONS SPECIFY WHICH PATTERNS
15418                                   ;ARE TO BE RUN FOR PARTICULAR MEMORIES
15419                                   ;
15420                                   ;REFERENCE THE TABLE LISTED BELOW TO RELATE BITS TO PATTERNS.
15421                                   ;BIT0 SET WILL RUN THE FIRST ENTRY IN THE TABLE, BIT0 SET
15422                                   ;IN THE SECOND WORD WILL RUN THE 17TH ENTRY IN THE TABLE ...
15423                                   ;
15424                                   ;NOTE** NULL TESTS DO NOT TAKE ANY TIME
15425                                   ;                                       ;FIELD SERVICE VALUE
15426 063162  177777                   $DDW0:  .WORD   177777  ;ECC CSR TESTS           177777          TABLE - MKCSRT:
15427 063164  177777                   $DDW1:  .WORD   177777  ;ECC CSR TESTS           177777          TABLE - MKCSRT:
15428 063166  177777                   $DDW2:  .WORD   177777  ;ECC PATTERNS            103777          TABLE - MKPAT:
15429 063170  177777                   $DDW3:  .WORD   177777  ;ECC PATTERNS            177777          TABLE - MKPAT:
15430 063172  177777                   $DDW4:  .WORD   177777  ;PARITY PATTERNS         003777          TABLE - MJPAT:
15431 063174  177777                   $DDW5:  .WORD   177777  ;PARITY PATTERNS         177774          TABLE - MJPAT:
15435 063176                           $ETEND:
15436                                   ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT PDP11 DIAGNOSTIC
15437                                   ;INTERFACE SPEC.
15438
15439 063176                           $APTHD:
15440 063176  000000                   $HIBTS: .WORD   0               ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
15441 063200  063076                   $MBADR: .WORD   $MAIL           ;;ADDRESS OF APT MAILBOX (BITS 0-15)
15442 063202  000043                   $TSTM:  .WORD   35.             ;;RUN TIM OF LONGEST TEST
15443 063204  001274                   $PASTM: .WORD   700.            ;;RUN TIME IN SECS. OF 1ST PASS ON 128K (QUICK VERIFY)
15444 063206  000000                   $UNITM: .WORD   0.              ;;EXTRA RUN TIME OF A PASS FOR EACH ADDITIONAL 128K (QV)
15445 063210  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.

063212  010046                  $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
063214  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
063220  005740                          TST     -(R0)           ;;BACKUP BY 2
063222  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
063224  006300                          ASL     R0              ;;POSITION FOR INDEXING
063226  016000  063254                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
063232  000200                          RTS     R0              ;;GO TO ROUTINE


                                ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO

063234  011646                  $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
063236  016666  000004  000002          MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
063244  000002                          RTI                     ;;RESTORE THE PSW

063246                          $NOTRAP:TYPE    MSG006          ;UNDEFINED TRAP INSTRUCTION
063252  000000                  $HALT2: HALT
```

```
                                .SBTTL  TRAP TABLE

                        ;+THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
                        ;+BY THE "TRAP" INSTRUCTION.

                        ;       ROUTINE
                        ;        -----
063254  063234          $TRPAD: .WORD   $TRAP2
063256  053430                  $TYPE   ;CALL-TYPEIT    TRAP+1(104401)  TTY TYPEOUT ROUTINE
063260  060346                  $TYPOC  ;CALL-TYPOC     TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
063262  060322                  $TYPOS  ;CALL-TYPOS     TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
063264  063246                  $NOTRAP;$TYPON  ;CALL-TYPON     TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
063266  060550                  $TYPDS  ;CALL-TYPDS     TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
063270  063246                  $NOTRAP;$TYPBN  ;CALL-TYPBN     TRAP+6(104406)  TYPE BINARY (ASCII) NUMBER

063272  061150                  $GTSWR  ;CALL-GTSWR     TRAP+7(104407)  GET SOFT-SWR SETTING
063274  060774                  $CKSWR  ;CALL-CKSWR     TRAP+10(104410) TEST FOR CHANGE IN SOFT-SWR

063276  061514                  $RDCHR  ;CALL-RDCHR     TRAP+11(104411) TTY TYPEIN CHARACTER ROUTINE
063300  061644                  $RDLIN  ;CALL-RDLIN     TRAP+12(104412) TTY TYPEIN STRING ROUTINE
063302  062230                  $RDOCT  ;CALL-RDOCT     TRAP+13(104413) READ AN OCTAL NUMBER FROM TTY
063304  062400                  $RDDEC  ;CALL-RDDEC     TRAP+14(104414) READ A DECIMAL NUMBER FROM TTY

063306  062566                  $SAVREG ;CALL-SAVREG    TRAP+15(104415) SAVE R0-R5 ROUTINE
063310  062624                  $RESREG ;CALL-RESREG    TRAP+16(104406) RESTORE R0-R5 ROUTINE

063312  040220                  $KERNEL  ;CALL-KERNEL   TRAP+17(104417) ENTER KERNEL MODE
063314  040230                  $ENERGIZE;CALL-ENERGIZETRAP+20(104420) TURN ON MEMORY MANAGEMENT & TRAPS
063316  040240                  $DEENERGI;CALL-DEENERGITRAP+21(104421) TURN OFF MEMORY MENAGEMENT & TRAPS

063320  042436                  $KMAP    ;CALL-KMAP     TRAP+22(104422) MAP KERNEL 1 TO 1

063322  040250                  $CACHN  ;CALL-CACHON    TRAP+23(104423) TURN CACHE ON
063324  040274                  $CACHF  ;CALL-CACHOFF   TRAP+24(104424) TURN CACHE OFF

063326  040312                  $LOADC  ;CALL-LOADCSR   TRAP+25(104425) LOAD CORRECT CSR
063330  040406                  $READC  ;CALL-READCSR   TRAP+26(104426) READ CORRECT CSR

063332  053764                  $PER01  ;CALL-PERR01    TRAP+27(104427) PROGRAM DETECTED ERROR
063334  054012                  $PER02  ;CALL-PERR02    TRAP+30(104430) PROGRAM DETECTED ERROR
063336  054040                  $PER03  ;CALL-PERR03    TRAP+31(104431) PROGRAM DETECTED ERROR
063340  054070                  $PER04  ;CALL-PERR04    TRAP+32(104432) PROGRAM DETECTED ERROR
063342  054152                  $PER07  ;CALL-PERR07    TRAP+33(104433) PROGRAM DETECTED ERROR
063344  054174                  $PER10  ;CALL-PERR10    TRAP+34(104434) PROGRAM DETECTED ERROR
063346  054224                  $PER11  ;CAL -PERR11    TRAP+35(104435) PROGRAM DETECTED ERROR
063350  054244                  $PER12  ;CALL-PERR12    TRAP+36(104436) PROGRAM DETECTED ERROR
063352  054266                  $PER13  ;CALL-PERR13    TRAP+37(104437) PROGRAM DETECTED ERROR
063354  054306                  $PER14  ;CALL-PERR14    TRAP+40(104440) PROGRAM DETECTED ERROR
063356  054330                  $PER15  ;CALL-PERR15    TRAP+41(104441) PROGRAM DETECTED ERROR
063360  054352                  $PER16  ;CALL-PERR16    TRAP+42(104442) PROGRAM DETECTED ERROR
063362  054372                  $PER17  ;CALL-PERR17    TRAP+43(104443) PROGRAM DETECTED ERROR
063364  054410                  $PER20  ;CALL-PERR20    TRAP+44(104444) PROGRAM DETECTED ERROR
063366  054426                  $PER21  ;CALL-PERR21    TRAP+45(104445) PROGRAM DETECTED ERROR
063370  054446                  $PER22  ;CALL-PERR22    TRAP+46(104446) PROGRAM DETECTED ERROR
063372  054464                  $PER23  ;CALL-PERR23    TRAP+47(104447) PROGRAM DETECTED ERROR
063374  054502                  $PER24  ;CALL-PERR24    TRAP+50(104450) PROGRAM DETECTED ERROR
063376  051240                  $PER25  ;CALL-PERR25    TRAP+51(104451) PROGRAM DETECTED ERROR
```

```
  15531 063400  054672                $PER26  ;CALL-PERR26    TRAP-52(104452) PROGRAM DETECTED ERROR
  15532 063402  054712                $PER27  ;CALL-PERR27    TRAP-53(104453) PROGRAM DETECTED ERROR
  15533 063404  051466                $PER30  ;CALL-PERR30    TRAP-54(104454) PROGRAM DETECTED ERROR
  15534 063406  055102                $PER31  ;CALL-PERR31    TRAP-55(104455) PROGRAM DETECTED ERROR
  15535 063410  055200                $PER32  ;CALL-PERR32    TRAP-56(104456) PROGRAM DETECTED ERROR
  15536 063412  055246                $PER33  ;CALL-PERR33    TRAP-57(104457) PROGRAM DETECTED ERROR
  15537 063414  055326                $PER34  ;CALL-PERR34    TRAP-60(104460) PROGRAM DETECTED ERROR
  15538 063416  055360                $PER35  ;CALL-PERR35    TRAP-61(104461) PROGRAM DETECTED ERROR
  15539 063420  055414                $PER36  ;CALL-PERR36    TRAP-62(104462) PROGRAM DETECTED ERROR
  15540 063422  063246                $NOTRAP ;CALL-PERR37    TRAP-63(104463) PROGRAM DETECTED ERROR
  15541 063424  063246                $NOTRAP ;CALL-PERR40    TRAP-64(104464) PROGRAM DETECTED ERROR
  15542 063426  063246                $NOTRAP ;CALL-PERR41    TRAP-65(104465) PROGRAM DETECTED ERROR
  15543 063430  063246                $NOTRAP ;CALL-PERR42    TRAP-66(104466) PROGRAM DETECTED ERROR
  15544 063432  063246                $NOTRAP ;CALL-PERR43    TRAP-67(104467) PROGRAM DETECTED ERROR
  15545
  15546 063434  040632                $ECCDIS ;CALL-ECCDIS    TRAP-70(104470) DISABLE ECC ON ALL CSR S
  15547 063436  040646                $ECC1DIS;CALL-ECC1DIS   TRAP-71(104471) DISABLE ECC ON 1 SELECTED CSR
  15548 063440  040660                $ECCINIT;CALL-ECCINIT   TRAP-72(104472) INITIALIZE ALL MK11 CSR'S
  15549 063442  040674                $ECC1INIT;CALL-ECC1INIT TRAP-73(104473) INITIALIZE 1 SELECTED MK11 CSR
  15550 063444  040734                $CBCSR  ;CALL-CBCSR     TRAP-74(104474) WRITE GENERATED CHECKBITS IN ALL CSR'S
  15551 063446  040756                $CB1CSR ;CALL-CB1CSR    TRAP-75(104475) WRITE GENERATED CHECKBITS IN 1 SELECTED CSR
  15552 063450  040776                $WASSBE ;CALL-WASSBE    TRAP-76(104476) WAS THERE A SBE ON ANY CSR?
  15553 063452  041112                $WAS1SBE;CALL-WAS1SBE   TRAP-77(104477) WAS THERE A SBE ON 1 SELECTED CSR?
  15554 063454  041142                $WASDBE ;CALL-WASDBE    TRAP-100(104500)WAS THERE A DBE ON ANY CSR?
  15555 063456  041256                $WAS1DBE;CALL-WAS1DBE   TRAP-101(104501)WAS THERE A DBE ON 1 SELECTED CSR?
  15556 063460  041306                $CLRCSR ;CALL-CLRCSR    TRAP-102(104502)CLEAR ALL CSR'S
  15557 063462  041320                $CLR1CSR;CALL-CLR1CSR   TRAP-103(104503)CLEAR 1 SELECTED CSR
  15558 063464  041330                $CHKDIS ;CALL-CHKDIS    TRAP-104(104504)DISABLE ECC & WRITE CKBITS FROM ALL CSR'S
  15559 063466  041344                $CHK1DIS;CALL-CHK1DIS   TRAP-105(104505)DISABLE ECC & WRITE CKBITS FROM 1 CSR
  15560 063470  040706                $ENASBE ;CALL-ENASBE    TRAP-106(104506)ENABLE TRAPS ON SBE'S FROM ALL CSR S
  15561 063472  040722                $ENA1SBE;CALL-ENA1SBE   TRAP-107(104507)ENABLE TRAPS ON SBE'S FROM 1 SELECTED CSR
  15562 063474  040426                $TSTRD  ;CALL-TSTREAD   TRAP-110(104510)TEST LOC (R1) & TST FOR SBE (WITHOUT FETCHES
)
  15563 063476  041424                $INVALID;CALL-INVALID   TRAP-111(104511)INVALIDATE BACKGROUND PATTERN ON BANK
  15564 063500  041454                $ERRGEN ;CALL-ERRGEN    TRAP-114(104512)TEST ERROR ADDRESS
  15565 063502  063246                $NOTRAP
  15566 063504  063246                $NOTRAP
  15567 063506  063246                $NOTRAP
  15568 063510  063246                $NOTRAP
  15569 063512  063246                $NOTRAP
  15570 063514  063246                $NOTRAP
  15571 063516  063246                $NOTRAP
```

```
15574          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


063520                          $ERRTB: ;ERROR  1
063520  066011                         EM24
063522  070136                         DH13
063524  064370                         DT13
063526  064726                         DF11
                                        ;ERROR  2
063530  064765                         EM2
063532  067445                         DH1
063534  064220                         DT1
063536  064604                         DF2
                                        ;ERROR  3
063540  065023                         EM3
063542  067525                         DH3
063544  064232                         DT3
063546  064721                         DF9
                                        ;ERROR  4
063550  065055                         EM4
063552  067525                         DH3
063554  064242                         DT4
063556  064721                         DF9
                                        ;ERROR  5
063560  065123                         EM5
063562  067561                         DH5
063564  064252                         DT5
063566  064604                         DF2
                                        ;ERROR  6
063570  065200                         EM6
063572  067561                         DH5
063574  064252                         DT5
063576  064604                         DF2
                                        ;ERROR  7
063600  065225                         EM7
063602  067561                         DH5
063604  064252                         DT5
063606  064604                         DF2
                                        ;ERROR  10
063610  067343                         EM53
063612  070601                         DH25
063614  064546                         DT25
063616  064604                         DF2
```

```
15633                                           ;ERROR  11
15634 063620  065265                            EM11
15635 063622  067705                            DH7
15636 063624  064304                            DT7
15637 063626  064630                            DF3
15638                                           ;ERROR  12
15639 063630  065265                            EM11
15640 063632  067705                            DH7
15641 063634  064304                            DT7
15642 063636  064643                            DF4
15643                                           ;ERROR  13
15644 063640  065307                            EM12
15645 063642  070015                            DH10
15646 063644  064334                            DT10
15647 063646  064604                            DF2
15648                                           ;ERROR  14
15649 063650  065265                            EM11
15650 063652  067705                            DH7
15651 063654  064304                            DT7
15652 063656  064656                            DF5
15653                                           ;ERROR  15
15654 063660  065265                            EM11
15655 063662  067705                            DH7
15656 063664  064304                            DT7
15657 063666  064671                            DF6
15658                                           ;ERROR  16
15659 063670  065333                            EM13
15660 063672  070136                            DH13
15661 063674  064370                            DT13
15662 063676  064726                            DF11
15663                                           ;ERROR  17
15664 063700  065365                            EM14
15665 063702  070136                            DH13
15666 063704  064370                            DT13
15667 063706  064726                            DF11
15668                                           ;ERROR  20
15669 063710  065431                            EM15
15670 063712  070136                            DH13
15671 063714  064370                            DT13
15672 063716  064726                            DF11
15673                                           ;ERROR  21
15674 063720  067372                            EM55
15675 063722  070625                            DH26
15676 063724  064556                            DT26
15677 063726  064604                            DF2
15678                                           ;ERROR  22
15679 063730  065477                            EM17
15680 063732  067705                            DH7
15681 063734  064304                            DT7
15682 063736  064656                            DF5
15683                                           ;ERROR  23
15684 063740  067200                            EM50
15685 063742  070453                            DH23
15686 063744  064504                            DT23
15687 063746  064737                            DF13
```

```
15690                                   ;ERROR  24
15691 063750  065553                    EM19
15692 063752  070136                    DM13
15693 063754  064370                    DT13
15694 063756  064726                    DF11
15695                                   ;ERROR  25
15696 063760  065614                    EM20
15697 063762  070136                    DM13
15698 063764  064370                    DT13
15699 063766  064726                    DF11
15700                                   ;ERROR  26
15701 063770  000000                    0                       ;NO MESSAGE
15702 063772  070131                    DM12
15703 063774  064364                    DT12
15704 063776  064604                    DF2
15705                                   ;ERROR  27
15706 064000  065676                    EM21
15707 064002  070113                    DM11
15708 064004  064356                    DT11
15709 064006  064604                    DF2
15710                                   ;ERROR  30
15711 064010  065735                    EM22
15712 064012  070136                    DM13
15713 064014  064370                    DT13
15714 064016  064726                    DF11
15715                                   ;ERROR  31
15716 064020  000000                    0                       ;NO MESSAGE
15717 064022  070233                    DM14
15718 064024  064412                    DT14
15719 064026  064604                    DF2
15720                                   ;ERROR  32
15721 064030  065762                    EM23
15722 064032  067561                    DM5
15723 064034  064252                    DT5
15724 064036  064604                    DF2
15732                                   ;ERROR  33
15733 064040  066070                    EM25
15734 064042  070312                    DM15
15735 064044  064430                    DT16
15736 064046  064704                    DF7
15737                                   ;ERROR  34
15738 064050  066115                    EM26
15739 064052  070431                    DM16
15740 064054  064460                    DT17
15741 064056  064630                    DF3
```

```
15751                                   ;ERROR  55
15752 064060  067316                    EM52
15753 064062  070601                    DM25
15754 064064  064546                    DT25
15755 064066  064604                    DF2
15756                                   ;ERROR  56
15757 064070  066166                    EM27
15758 064072  070431                    DM16
15759 064074  064460                    DT17
15760 064076  064717                    DF8
15768                                   ;ERROR  57
15769 064100  066772                    EM35
15770 064102  067705                    DM7
15771 064104  064304                    DT7
15772 064106  064630                    DF3
15773                                   ;ERROR  60
15774 064110  066256                    EM29
15775 064112  067705                    DM7
15776 064114  064304                    DT7
15777 064116  064630                    DF3
15778                                   ;ERROR  61
15779 064120  066340                    EM30
15780 064122  067705                    DM7
15781 064124  064304                    DT7
15782 064126  064656                    DF5
15783                                   ;ERROR  62
15784 064130  066457                    EM31
15785 064132  067705                    DM7
15786 064134  064304                    DT7
15787 064136  064630                    DF3
15788                                   ;ERROR  63
15789 064140  066557                    EM32
15790 064142  067705                    DM7
15791 064144  064304                    DT7
15792 064146  064630                    DF3
15793                                   ;ERROR  64
15794 064150  066664                    EM33
15795 064152  067705                    DM7
15796 064154  064304                    DT7
15797 064156  064630                    DF3
15798                                   ;ERROR  65
15799 064160  067234                    EM51
15800 064162  070532                    DM24
15801 064164  064526                    DT24
15802 064166  064747                    DF14
15803                                   ;ERROR  66
15804 064170  067057                    EM36
15805 064172  067640                    DM6
15806 064174  064270                    DT6
15807 064176  064604                    DF2
```

```
                                ;ERROR  4?
064200  067126                  EM40
064202  067502                  DM2
064204  064466                  DT20
064206  064604                  DF2
                                ;ERROR  50
064210  067413                  EM56
064212  070643                  DM27
064214  064564                  DT27
064216  064756                  DF15
```

ERROR DATA TAGS (DT)

```
15878                                                   .SBTTL  ERROR DATA TAGS (DT)
15879 064220  002016  002032  002042  DT1:    .WORD   ERRPC,ADDRESS,GOOD,BAD,0
      064226  002050  000000
15883 064232  002016  002034  002070  DT3:    .WORD   ERRPC,PADDRESS,PARCNT,0
      064240  000000
15884 064242  002016  002032  002066  DT4:    .WORD   ERRPC,ADDRESS,MEMCNT,0
      064250  000000
15885 064252  002016  177572  177574  DT5:    .WORD   ERRPC,MMR0,MMR1,MMR2,MMR3,CPUERR,0
      064260  177576  172516  177766
      064266  000000
15886 064270  002016  002374  002352  DT6:    .WORD   ERRPC,APTPAR,LSIZE,APTECC,MSIZE,0
      064276  002376  002354  000000
15887 064304  002016  002172  002032  DT7:    .WORD   ERRPC,DUMMY,ADDRESS,DUMMY,GOOD,BAD,BADXOR
      064312  002172  002042  002050
      064320  002056
15888 064322  002172  002172  002172          .WORD   DUMMY,DUMMY,DUMMY,DUMMY,0
      064330  002172  000000
15889 064334  002174  002176  002200  DT10:   .WORD   DETR0,DETR1,DETR2,DETR3,DETR4,DETR5,DETSP,DETPSW,0
      064342  002202  002204  002206
      064350  002210  002212  000000
15890 064356  002016  002144  000000  DT11:   .WORD   ERRPC,CSR,0
15891 064364  002144  000000          DT12:   .WORD   CSR,0
15892 064370  002016  002172  002032  DT13:   .WORD   ERRPC,DUMMY,ADDRESS,DUMMY,TSTDAT,TSTDAT+2,CHECK,CSR,0
      064376  002172  002242  002244
      064404  002276  002164  000000
15893 064412  177746  177572  177574  DT14:   .WORD   CONTRL,MMR0,MMR1,MMR2,MMR3,CPUERR,0
      064420  177576  172516  177766
      064426  000000
15894 064430  002016  002172  002172  DT16:   .WORD   ERRPC,DUMMY,DUMMY,GOOD,GOOD2,GOOD3
      064436  002042  002044  002046
15895 064444  002050  002052  002054          .WORD   BAD,BAD2,BAD3,DUMMY,DUMMY,0
      064452  002172  002172  000000
15896 064460  002016  002172  000000  DT17:   .WORD   ERRPC,DUMMY,0
15901 064466  002016  002042  002050  DT20:   .WORD   ERRPC,GOOD,BAD,0
      064474  000000
15905 064476  002016  002172  000000  DT22:   .WORD   ERRPC,DUMMY,0
15906 064504  002016  002172  002042  DT23:   .WORD   ERRPC,DUMMY,GOOD,BAD,DUMMY,DUMMY,DUMMY,DUMMY,0
      064512  002050  002172  002172
      064520  002172  002172  000000
15907 064526  002016  002172  002144  DT24:   .WORD   ERRPC,DUMMY,CSR,DUMMY,DUMMY,DUMMY,DUMMY,0
      064534  002172  002172  002172
      064542  002172  000000
15908 064546  002016  002042  002144  DT25:   .WORD   ERRPC,GOOD,CSR,0
      064554  000000
15909 064556  002016  002050  000000  DT26:   .WORD   ERRPC,BAD,0
15910 064564  002016  002172  002032  DT27:   .WORD   ERRPC,DUMMY,ADDRESS,DUMMY,DUMMY,DUMMY,DUMMY,0
      064572  002172  002172  002172
      064600  002172  000000
```

```
                                                .SBTTL  ERROR DATA FORMATS (DF)
15917
15918 064604    000     000     000     DF2:    .BYTE   0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
      064607    000     000     000
      064612    000     000     000
      064615    000     000     000
      064620    000     000     000
      064623    000     000     000
      064626    000     000
15919 064630    000     005     000     DF3:    .BYTE   0,5,0,8.,0,0,0,3,6,2,4
      064633    010     000     000
      064636    000     003     006
      064641    002     004
15920 064643    000     005     000     DF4:    .BYTE   0,5,0,8.,0,8.,8.,3,6,2,4
      064646    010     000     010
      064651    010     003     006
      064654    002     004
15921 064656    000     005     000     DF5:    .BYTE   0,5,0,8.,9.,9.,9.,3,6,2,4
      064661    010     011     011
      064664    011     003     006
      064667    002     004
15922 064671    000     005     000     DF6:    .BYTE   0,5,0,8.,9.,8.,8.,3,6,2,4
      064674    010     011     010
      064677    010     003     006
      064702    002     004
15923 064704    000     005     010     DF7:    .BYTE   0,5,8.,0,0,9.,0,0,9.,2,4
      064707    000     000     011
      064712    000     000     011
      064715    002     004
15924 064717    000     005             DF8:    .BYTE   0,5
15925 064721    000     001     001     DF9:    .BYTE   0,1,1,1,1
      064724    001     001
15926 064726    000     005     000     DF11:    BYTE   0,5,0,8.,0,0,0,0,0
      064731    010     000     000
      064734    000     000     000
15927 064737    000     005     000     DF13:   .BYTE   0,5,0,0,3,6,2,4
      064742    000     003     006
      064745    002     004
15928 064747    000     005     000     DF14:   .BYTE   0,5,0,3,6,2,4
      064752    003     006     002
      064755    004
15929 064       000     005     000     DF15:   .BYTE   0,5,0,8.,,3,6,4
      0f        010     003     006
      064 ,04   004
```

```
15935                                           .SBTTL  ERROR MESSAGES (EM)
15944 064765    103     101     116  EM2:       .ASCIZ  /CAN'T SET 22 BIT MODE IN MMR3/
15945 065023    120     101     122  EM3:       .ASCIZ  /PARITY ERROR(S) IN BANK 0/
15946 065055    116     117     116  EM4:       .ASCIZ  /NON-EXISTANT MEMORY (HOLES) IN BANK 0/
15947 065123    111     114     114  EM5:       .ASCIZ  /ILLEGAL OR RESERVED INSTRUCTION (TRAP TO 10)
15948 065200    125     116     105  EM6:       .ASCIZ  /UNEXPECTED TRAP TO 4/
15949 065225    115     105     115  EM7:       .ASCIZ  /MEMORY MANAGEMENT (TRAP TO 250)/
15953
15954 065265    115     105     115  EM11:      .ASCIZ  /MEMORY DATA ERROR/
15955 065307    104     105     124  EM12:      .ASCIZ  /DETAILED ERROR DUMP/
15956 065333    115     111     123  EM13:      .ASCIZ  /MISSING EXPECTED SBE FLAG/
15957 065365    127     122     111  EM14:      .ASCIZ  /WRITE BYTE FAILED TO CLEAR SBE FLAG/
15958 065431    106     101     111  EM15:      .ASCIZ  /FAILED TO GET INTERRUPT WITH DBE FLAG/
15959 065477    115     105     115  EM17:      .ASCIZ  /MEMORY DATA ERROR IN CHECK BITS/
15960 065537    123     102     105  EM19:      .ASCIZ  /SBE OR DBE CAUSED PARITY TRAP WHEN INHIBITED/
15961 065614    123     102     105  EM20:      .ASCIZ  /SBE OR DBE DID NOT CAUSE PARITY TRAP WHEN ENABLED
15962 065676    123     102     105  EM21:      .ASCIZ  /SBE OR DBE ON MASTER TEST WORD/
15963 065735    115     111     123  EM22:      .ASCIZ  /MISSING EXPECTED DBE/
15964 065762    125     116     105  EM23:      .ASCIZ  /UNEXPECTED PARITY TRAP/
15965 066011    122     105     103  EM24:      .ASCIZ  /RECEIVED DBE FLAG WHEN EXPECTING ONLY SBE FLAG/
15966 066070    103     110     105  EM25:      .ASCIZ  /CHECK BIT DATA ERROR/
15967 066115    101     104     104  EM26:      .ASCIZ  /ADDRESS PARITY ERROR DID NOT CAUSE ABORT/
15968 066166    105     103     103  EM27:      .ASCIZ  /ECC INHIBIT MODE POINTER FAILURE   DID NOT PROTECT BANK/
15972 066256    103     117     122  EM29:      .ASCIZ  /CORRECTION FAILURE WITH ECC ENABLED ON FORCED SBE/
15973 066340    127     122     111  EM30:      .ASCII  /WRITE BYTE (MOVB) WITH ECC ENABLED FAILED TO CLEAR DATA AT/<CRLF>
15974 066433    106     117     122             .ASCIZ  /FORCED SBE LOCATION/
15975 066457    101     123     122  EM31:      .ASCIZ  /ASRB (R3)+ WITH ECC ENABLED CHANGED DATA AT FORCED DBE LOCATION/
15976 066557    115     117     126  EM32:      .ASCIZ  /MOVB #360,(R2)+ WITH ECC ENABLED CHANGED DATA AT FORCED DBE LOCATION/
15977 066664    115     117     126  EM33:      .ASCIZ  /MOV #177400,(R1) WITH ECC ENABLED CHANGED DATA AT FORCED DBE LOCATION
15978 066772    125     116     105  EM35:      .ASCIZ  /UNEXPECTED CORRECTION WITH ECC DISABLE ON FORCED SBE/
15979 067057    101     120     124  EM36:      .ASCIZ  /APT SIZE DISAGREES WITH PROGRAM SIZING/
15985 067126    102     122     101  EM40:       ASCIZ  /BRANCH GOBBLE FAILED CONDITION CODES TEST/
15994 067200    102     101     104  EM50:      .ASCIZ  /BAD ERROR ADDRESS GENERATED/
15995 067234    123     102     105  EM51:      .ASCIZ  /SBE & DBE FLAGS NOT SET ON FORCED UNCORRECTED SBE/
15996 067316    102     111     124  EM52:      .ASCIZ  /BIT SET ERROR IN CSR/
15997 067343    102     111     124  EM53:      .ASCIZ  /BIT CLEAR ERROR IN CSR/
15998 067372    111     114     114  EM55:      .ASCIZ  /ILLEGAL CSR TYPE/
15999 067413    102     101     104  EM56:      .ASCIZ  /BAD PARITY TRAP GENERATED/
```

ERROR DATA HEADERS (DH)

```
16005                                           .SBTTL  ERROR DATA HEADERS (DH)
16006 067445  040   040   120  DH1:    .ASCIZ  /  PC    DEV ADD   GOOD    BAD
16007 067502  040   040   120  DH2:    .ASCIZ  /  PC    GD-CC BD CC/
16008 067525  040   040   120  DH3:    .ASCIZ  /  PC    1ST ADD  # OF ERRORS/
16009 067561  040   040   120  DH5:    .ASCIZ  /  PC      MMR0     MMR1     MMR2     MMR3   CPUERR
16010 067640  040   040   120  DH6:    .ASCIZ  /  PC    APTPAR   LSIZE  APTECC  MSIZE/
16011 067705  040   040   120  DH7:    .ASCII  /  PC     BANK  VADD      PADD      GOOD/
16012 067751  040   040   040          .ASCIZ  /      BAD      COR  CSR  MTYP INT PAT/
16013 070015  040   040   122  DH10:   .ASCIZ  /  R0       R1       R2       R3       R4       R5       SP       PSW
16014 070113  040   040   120  DH11:   .ASCIZ  /  PC       CSR/
16015 070131  040   103   123  DH12:   .ASCIZ  / CSR/
16016 070136  040   040   120  DH13:   .ASCII  /  PC     BANK  VADD      PADD     WROTE1  WROTE2/
16017 070213  040   103   110          .ASCIZ  / CHKBITS     CSR/
16018 070233  103   117   116  DH14:   .ASCIZ  /CONTRL    MMR0     MMR1     MMR2     MMR3   CPUERR/
16019 070312  040   040   120  DH15:   .ASCII  /  PC     BANK     PADD   GD-WD1  GD WD2  GD-CHK/
16020 070367  040   102   101          .ASCIZ  / BAD-WD1 BAD-WD2  BAD-CHK INT PAT/
16021 070431  040   040   120  DH16:    ASCIZ  /  PC     BANK/
16026 070446  040   040   120  DH19:   .ASCIZ  /  PC/
16032 070453  040   040   120  DH23:   .ASCIZ  /  PC     BANK GD-ERR BAD-ERR CSR  MTYP INT  PAT/
16033 070532  040   040   120  DH24:   .ASCIZ  /  PC     BANK  (CSR) CSR  MTYP INT  PAT/
16034 070601  040   040   120  DH25:   .ASCIZ  /  PC  GD DAT  (CSR)/
16035 070625  040   040   120  DH26:   .ASCIZ  /  PC  BADCODE/
16036 070643  040   040   120  DH27:   .ASCIZ  /  PC     BANK  VADD      PADD   CSR  MTYP PAT/
```

```
16039                                           .SBTTL  MESSAGES
16040 070717  200  040  103  MSG000:  .ASCIZ  <CRLF>" CZMSDO   MS11L M MEMORY DIAGNOSTIC"
16041 070764  200  040  040  MSG001:  .ASCIZ  <CRLF>/                          MEMORY CONFIGURATION MAP/
16042 071046  200  040  040  MSG002:  .ASCIZ  <CRLF>/                              16K WORD BANKS/
16043 071123  200  040  040  MSG003:  .ASCII  <CRLF>/                  1       2       3/
16044 071165  040  040  040           .ASCIZ  /       4       5       6       7   /
16045 071230  200  040  040  MSG004:  .ASCII  <CRLF>/        01234567012345670123456701234567/
16046 071271  060  061  062           .ASCIZ  /0123456701234567012345670123456701234567/
16047 071336  200  105  122  MSG005:  .ASCIZ  <CRLF>/ERRORS  /
16048 071350  200  125  116  MSG006:  .ASCIZ  <CRLF>/UNDEFINED TRAP INSTRUCTION/<32>
16049 071405  200  111  116  MSG007:  .ASCIZ  <CRLF>/INTRLV  /                    ;INTERLEAVED CSR #
16050 071417  200  103  120  MSG008:  .ASCIZ  <CRLF>/CPU MAP /                    ;CPU ACCESSED BANK
16051 071431  200  115  105  MSG009:  .ASCIZ  <CRLF>/MEMTYPE /                    ;MEMORY TYPE
16052 071443  200  120  122  MSG010:  .ASCIZ  <CRLF>/PROTECT /                    ;MEMORY PROTECTED
16053 071455  040  040  040  MSG011:  .ASCIZ  /    0       1       2       3       4       5       6/
16054 071543  064  065  066  MSG012:  .ASCIZ  /4567012345670123456701234567012345670123456701234567
16055 071640  130  000       MSG013:  .ASCIZ  /X/
16056 071642  040  000       MSG014:  .ASCIZ  / /                ;SPACE
16057 071644  000  000       MSG015:  .BYTE   0,0                ;FOR SINGLE ASCII CHARACTERS & TERMINATOR
16058 071646  200  103  123  MSG016:  .ASCIZ  <CRLF>/CSR     /
16059 071660  040  040  040  MSG017:  .ASCIZ  /       /                    ;8 SPACES
16060 071671  040  040  000  MSG018:  .ASCIZ  /  /                        ;2 SPACES
16061 071674  040  040  040  MSG019:  .ASCIZ  /   /                       ;3 SPACES
16062 071700  200  106  123  MSG020:  .ASCIZ  <CRLF>/FS COMMAND MODE/
16063 071721  200  103  117  MSG021:  .ASCII  <CRLF>/COMMANDS AVAILABLE:/
16064 071745  200  060  040           .ASCII  <CRLF>/0 - EXIT/
16065 071756  200  061  040           .ASCII  <CRLF>/1 - READ CSR/
16066 071773  200  062  040           .ASCII  <CRLF>/2 - LOAD CSR/
16067 072010  200  063  040           .ASCII  <CRLF>/3 - EXAMINE MEMORY/
16068 072033  200  064  040           .ASCII  <CRLF>/4 - MODIFY MEMORY/
16069 072055  200  065  040           .ASCII  <CRLF>/5 - SELECT BANK & PATTERN/
16070 072107  200  066  040            ASCII  <CRLF>/6 - TYPE CONFIG MAP/
16071 072133  200  067  040           .ASCII  <CRLF>/7 - SOB-A-LONG TEST/
16072 072157  200  070  040           .ASCII  <CRLF>/8 - ERROR SUMMARY/
16073 072201  200  071  075           .ASCII  <CRLF>/9= REFRESH TEST/
16074 072221  200  061  060           .ASCII  <CRLF>/10= SET FILL COUNT/
16075 072244  200  061  061           .ASCII  <CRLF>/11= ENTER KAMIKAZE MODE/
16076 072274  200  061  062           .ASCII  <CRLF>/12= EXIT KAMIKAZE MODE/
16077 072323  200  061  063           .ASCII  <CRLF>/13= TURN CACHE OFF/
16078 072346  200  061  064           .ASCII  <CRLF>/14= TURN CACHE ON/
16083 072370  200  061  065           .ASCII  <CRLF>/15= TEST SELECTED BANKS/
16084 072420  200  061  066           .ASCII  <CRLF>/16= TEST ALL BANKS/
16085 072443  200  061  067           .ASCII  <CRLF>/17= ENABLE TRACE/
16086 072464  200  061  070           .ASCII  <CRLF>/18= DISABLE TRACE/
16087 072506  015  012  000           .BYTE   15,12,0
16088 072511  200  127  110  MSG022:  .ASCIZ  <CRLF>/WHICH CSR(0 F)? /
16089 072533  200  103  123  MSG023:  .ASCIZ  <CRLF>/CSR WORD? /
16090 072547  200  103  123  MSG025:  .ASCIZ  <CRLF>/CSR DOES NOT EXIST/
16091 072573  200  103  117  MSG026:  .ASCIZ  <CRLF>/COMMAND:/
16092 072605  200  117  114  MSG027:  .ASCIZ  <CRLF>/OLD CSR WAS/
16093 072622  200  103  123  MSG028:  .ASCIZ  <CRLF>/CSR IS NOW/
16094 072636  200  105  130  MSG029:  .ASCIZ  <CRLF>/EXAMINE MEMORY/
16095 072656  200  102  101  MSG030:  .ASCIZ  <CRLF>/BANK(0-177)? /
16096 072675  200  120  110  MSG031:  .ASCIZ  <CRLF>/PHYSICAL ADDRESS(0-17757776)? /
16097 072735  200  120  101  MSG032:  .ASCIZ  <CRLF>/PARITY ABORT/<32>
16098 072754  200  124  111  MSG033:  .ASCIZ  <CRLF>/TIMEOUT TRAP/<32>
16099 072773  200  102  131  MSGA34:  .ASCIZ  <CRLF>/BYPASSING ECC LOGIC TESTS ON BANK /
```

```
16100 073037  040  104  125  MSGB34: .ASCIZ  / DUE TO LACK OF SBE FREE LOCATIONS/
16101 073102  121  126  000  MSG035: .ASCIZ  /QV/
16102 073105  200  115  117  MSG036: .ASCIZ  <CRLF>/MODIFY MEMORY/
16103 073124  200  117  114  MSG037: .ASCIZ  <CRLF>/OLD DATA WAS /
16104 073143  200  104  101  MSG038: .ASCIZ  <CRLF>/DATA IS NOW /
16105 073161  200  111  116  MSG039: .ASCIZ  <CRLF>/INPUT NEW DATA? /
16106 073203  200  123  105  MSG040: .ASCIZ  <CRLF>/SELECT BANK & PATTERN TEST/
16107 073237  200  102  101  MSG041: .ASCIZ  <CRLF>/BANK NOT ACCESSABLE/
16108 073264  200  120  101  MSG042: .ASCIZ  <CRLF>/PATTERN(0-35)? /
16109 073305  200  120  101  MSG043: .ASCIZ  <CRLF>/PATTERN 0 DATA IS? /
16110 073332  200  124  117  MSG046: .ASCIZ  <CRLF>/TO ESCAPE TYPE ANY KEY/<CRLF><12><12>
16111 073365  200  124  105  MSG047: .ASCIZ  <CRLF>/TEST COMPLETE/
16112 073404  040  116  117  MSG048: .ASCIZ  / NOT AVAILABLE NOW   TRY LATER!/
16113 073444  200  102  101  MSG049: .ASCIZ  <CRLF>/BANK REQUIRES RELOCATION/
16114 073476  200  102  101  MSG050: .ASCII  <CRLF>/BATTERY BACKUP TEST/
16115 073522  200  127  122          .ASCIZ  <CRLF>/WRITING & CHECKING ADDRESS PATTERN AS BACKGROUND/
16116 073604  200  120  117  MSG051: .ASCIZ  <CRLF>/POWER RECOVERY/
16117 073624  200  122  105  MSG052: .ASCIZ  <CRLF>/REMOVE SYSTEM POWER FOR/
16118 073655  040  123  105  MSG053: .ASCIZ  / SECONDS MAX!/
16119 073673  200  116  117  MSG054: .ASCIZ  <CRLF>/NOW STARTING READ TEST OF MEMORY BANKS/
16120 073743  200  123  117  MSG055: .ASCIZ  <CRLF>/SOB-A-LONG TEST/
16121 073764  200  102  105  MSG056: .ASCIZ  <CRLF>/BELL = EACH PASS COMPLETE/
16122 074017  200  040  040  MSG058: .ASCIZ  <CRLF>/  CSR    CSR .../
16123 074041  077  077  077  MSG061: .ASCIZ  /??????/
16124 074050  111  116  120  MSG062: .ASCIZ  /INPUT MUST BE A/
16125 074070  116  040  117  MSG063: .ASCIZ  /N OCTAL /
16126 074101  116  125  115  MSG064: .ASCIZ  /NUMBER/<CRLF>
16127 074111  040  104  105  MSG065: .ASCIZ  / DECIMAL /
16128 074123  200  105  122  MSG066: .ASCIZ  <CRLF>/ERROR COUNT EXCEEDED 20   ABORTING FOR XXDP CHAIN/
16129 074206  106  101  124  MSG067: .ASCIZ  /FATAL /
16130 074215  113  040  127  MSG070: .ASCIZ  /K WORDS OF MEMORY TOTAL/<CRLF>
16131 074246  113  040  117  MSG071: .ASCIZ  /K OF BIPOLAR/<CRLF>
16132 074264  113  040  117  MSG072: .ASCIZ  /K OF MF11S-K/<CRLF>
16133 074302  200  122  105  MSG073: .ASCIZ  <CRLF>/REFRESH TEST/
16134 074320  200  122  105  MSG075: .ASCIZ  <CRLF>/RELOCATION NOT POSSIBLE/<32>
16135 074352  200  040  040  MSG076: .ASCIZ  <CRLF>/  BANK  ERRORS/<CRLF>
16136 074373  200  105  116  MSG077: .ASCIZ  <CRLF>/END PASS #/
16137 074407  040  105  122  MSG079: .ASCIZ  / ERROR(S) DETECTED/<CRLF>
16144 074433  200  106  111  MSG085: .ASCIZ  <CRLF>/FILL COUNT(OCTAL)? /
16152 074460  200  113  105  MSG088: .ASCIZ  <CRLF>/KERNEL STACK/
16153 074476  200  123  125  MSG089: .ASCIZ  <CRLF>/SUPERVISOR STACK/
16154 074520  200  125  123  MSG090: .ASCIZ  <CRLF>/USER STACK/
16155 074534  040  111  123  MSG091: .ASCIZ  / IS EMPTY/
16156 074546  122  105  114  MSG092: .ASCIZ  /RELOCATED  /
16157 074562  102  101  116  MSG093: .ASCIZ  /BANK=/
16158 074570  040  040  120  MSG095: .ASCIZ  /  PAT=/
16166 074577  200  105  116  MSG101: .ASCIZ  <CRLF>/ENTERING KAMIKAZE MODE/
16167 074627  200  114  105  MSG102: .ASCIZ  <CRLF>/LEAVING KAMIKAZE MODE/
16168 074656  200  114  105  MSG103: .ASCIZ  <CRLF>/LEAVING FS MODE/<CRLF>
16169 074700  032  000       MSG104: .BYTE   32,0                                    ;CONTROL Z
16170 074702  200  105  116  MSG105: .ASCIZ  <CRLF>/ENTER BANKS IN OCTAL - USE NUMBER OUTSIDE RANGE TO TERMINATE (177)
16171 075006  200  103  101  MSG106: .ASCIZ  <CRLF>/CACHE IS OFF/
16172 075024  200  103  101  MSG107: .ASCIZ  <CRLF>/CACHE IS ON (EXCEPT DURING ACTUAL PATTERNS)/
16177 075101  200  117  116  MSG110: .ASCIZ  <CRLF>/ONLY SELECTED BANKS WILL BE TESTED/
16178 075145  200  101  114  MSG111: .ASCIZ  <CRLF>/ALL BANKS WILL BE TESTED/
16179 075177  113  040  117  MSG112: .ASCIZ  /K OF MS11-L/<CRLF>
16180 075214  113  040  117  MSG113: .ASCIZ  /K OF MS11-M/<CRLF>
```

```
16181 075231  113  040  117  MSG114: .ASCIZ  /# OF UNIBUS PARITY <CRLF>
16182 075255  200  040  040  MSG116: .ASCIZ  <CRLF>"   11/34"
16183 075267  200  040  040  MSG117: .ASCIZ  <CRLF>"   11/44"
16184 075301  200  040  040  MSG118: .ASCIZ  <CRLF>"   11/60"
16185 075313  200  040  040  MSG119: .ASCIZ  <CRLF>/   NO/
16186 075322  040  103  101  MSG120: .ASCIZ  / CACHE AVAILABLE/
16187 075343  040  103  101  MSG121: .ASCIZ  / CACHE BYPASSED/
16188 075363  200  103  123  MSG122: .ASCII  <CRLF>/CSR NUMBER /
16189 075377  000               MSGA122:.BYTE   0
16190 075400  040  103  117          .ASCIZ  / CONTROLS TOO MANY BANKS/
16191 075431  200  120  122  MSG123: .ASCIZ  <CRLF>/PROGRAM RELOCATED   ECC TESTS INHIBITED/
16192 075502  200  116  125  MSG124: .ASCIZ  <CRLF>/NUMBER OF CSR'S IS WRONG IN BANK /
16193 075545  040  120  101  MSG125: .ASCIZ  / PASSES COMPLETED/
16194 075567  200  120  122  MSG126: .ASCIZ  <CRLF>/PROGRAM CSR COULD NOT BE DETERMINED/
16195 075634  200  124  122  MSG127: .ASCIZ  <CRLF>/TRACE ENABLED/
16196 075653  200  124  122  MSG128: .ASCIZ  <CRLF>/TRACE DISABLED/
16197                          .EVEN
16203 075674                   $$END
16204 075674               END:
16205 075674  004014             .PRINT  60000-SUPLIMIT  ;SUPERVISOR ADDRESSES LEFT
16217 075674  002104             .PRINT  100000-END      ;ADDRESSES LEFT IN 16K
16221         000200             .END START3
```

```
ABORTF  002140
ACFLAG  002114
ACTFLA  002322
ADDRES  002032
ANA2    007170
APTDOW  045252
APTECC  002376
APTFLA  002324
APTHAN  014556
APTHLT  045320
APTPAR  002374
APTSIZ  002416
BACK    056014
BACKGN  036532
BAD     002050
BADPC   002020
BADPSW  002030
BADSP   002024
BADSTA  040170
BADXOR  002056
BAD2    002052
BAD3    002054
BAFPAF  014712
BAFPAR  015020
BAKPAT  002574
BANK    002100
BANKIN  002102
BANKMO  043752
BANKOK  044752
BAUPAF  015126
BAUPAR  015256
BGTEST  036110
BIT0  = 000001
BIT1  = 000002
BIT10 = 002000
BIT11 = 004000
BIT12 = 010000
BIT13 = 020000
BIT14 = 040000
BIT15 = 100000
BIT2  = 000004
BIT3  = 000010
BIT4  = 000020
BIT5  = 000040
BIT6  = 000100
BIT7  = 000200
BIT8  = 000400
BIT9  = 001000
BLOCK1  045322
BLOCK2  045342
BLOCK3  045356
BMFLAG  002126
BOOT    045030
BOOT1   045074
BRGOBB  036112
BSIZE   002346
B0      004556
B1      011734
B10     014622
B11     016240
B12     016506
B13     016514
B14     016532
B15     016574
B16     016646
B17     022042
B2      012514
B20     022302
B21     024322
B22     024326
B23     024526
B24     024534
B25     025032
B26     034452
B27     034510
B3      012556
B30     034524
B31     034644
B32     035024
B33     035046
B34     036242
B35     041010
B36     041014
B37     041154
B4      013360
B40     041160
B41     041364
B42     041370
B43     042572
B44     042600
B45     042730
B46     042744
B47     050014
B5      013522
B50     050236
B51     050512
B52     051006
B53     051006
B54     051104
B55     052044
B56     052050
B57     052346
B6      013564
B60     052354
B61     057766
B62     057772
B63     060070
B64     060162
B65     060246
B7      014050
CACHKF  002522
CACHKN  002516
CACHOF- 104424
CACHON- 104423
CACHVE- 000114
CBCSR - 104474
CB1CSR- 104475
CHECK   002276
CHKDIS- 104504
CHKGEN  041724
CHKTAB  042032
CHK1DI- 104505
CKEND   061306
CKSWR - 104410
CLRCSR- 104502
CLREX   007140
CLRMEM  007040
CLR1CS- 104503
CMD16A  051072
CMD16L- 000052
CMD5B   047302
CMD5C   047552
CMD7B   050010
CMD7C   050064
CMD9B   050506
CMD9C   050562
CONFGE  002422
CONFIE  003632
CONFIG  002626
CONTFL  002216
CONTRL- 177746
CONTS   061456
CONTS1  061034
CONTS2  061460
CONTS3  053646
CONTT   061402
COUNT   002342
CPERRF  056560
CPSAVE  056556
CPUBIT  002104
CPUERR- 177766
CR    - 000015
CRLF  - 000200
CSR     002144
CSRADD- 172100
CSRCAS  017420
CSRFBA  002226
CSRFIR  002222
CSRHOL  002500
CSRINC  002302
CSRINF  002434
CSRINT  002232
CSRLAS  002224
CSRLBA  002230
CSRLOO  002304
CSRNO   002146
CSROUT  041356
CSRSTU- 000014
CTEST   006142
CTLKVE  002142
DATARG- 177754
DATBUF  002236
DBEMSK  002252
DDISP - 177570
DEENER- 104421
DETAIL  057532
DETFLA  002214
DETPSW  002212
DETR0   002174
DETR1   002176
DETR2   002200
DETR3   002202
DETR4   002204
DETR5   002206
DETSP   002210
DET1    060222
DF11    064726
DF13    064737
DF14    064747
DF15    064756
DF2     064604
DF3     064630
DF4     064643
DF5     064656
DF6     064671
DF7     064704
DF8     064717
DF9     064721
DH1     067445
DH10    070015
DH11    070113
DH12    070131
DH13    070136
DH14    070233
DH15    070312
DH16    070431
DH19    070446
DH2     067502
DH23    070453
DH24    070532
DH25    070601
DH26    070625
DH27    070643
DH3     067525
DH5     067561
DH6     067640
DH7     067705
DIAGFL  002002
DISPLA  002602
DISPRE- 000174
DISPTB  014240
DOBACK  014612
DSWR  - 177570
DT1     064220
DT10    064334
DT11    064356
DT12    064364
DT13    064370
DT14    064412
DT16    064430
DT17    064460
DT20    064466
DT22    064476
DT23    064504
DT24    064526
DT25    064546
DT26    064556
DT27    064564
DT3     064232
DT4     064242
DT5     064252
DT6     064270
DT7     064304
DUMMY   002172
DUMPCS- 000061
ECCDIS- 104470
ECCINI- 104472
ECC1DI- 104471
ECC1IN- 104473
EMTVEC- 000030
EM11    065265
EM12    065307
EM13    065333
EM14    065365
EM15    065431
EM17    065477
EM19    065537
EM2     064765
EM20    065614
EM21    065676
EM22    065735
EM23    065762
EM24    066011
EM25    066070
EM26    066115
EM27    066166
EM29    066256
EM3     065023
EM30    066340
EM31    066457
EM32    066557
EM33    066664
EM35    066772
EM36    067057
EM4     065055
EM40    067126
EM5     065123
EM50    067200
EM51    067234
EM52    067316
EM53    067343
EM55    067372
EM56    067413
EM6     065200
EM7     065225
```

SYMBOL TABLE

```
ENASBE= 104506
ENA1SB= 104507
END     075674
ENERGI= 104420
ENEXBK  044742
ERRADD  002432
ERRGEN= 104512
ERRMAX  002526
ERROR = 104000
ERRPC   002016
ERRPSW  002026
ERRSP   002022
ERRVEC= 000004
EUFLAG  002130
EVEN    002336
EXBANK  044302
EXCMD3  046372
EXCMD4  046712
EXIT    045136
EXIT2   045142
E0      004604
E1      012152
E10     014710
E11     016260
E12     017032
E13     017016
E14     016752
E15     016752
E16     016722
E17     022140
E2      012716
E20     022402
E21     024504
E22     024470
E23     024712
E24     024676
E25     025110
E26     035016
E27     035002
E3      012702
E30     034616
E31     034766
E32     035042
E33     035066
E34     036364
E35     041062
E36     041062
E37     041226
E4      013514
E40     041226
E41     041420
E42     041420
E43     042730
E44     042710
E45     043150
E46     043056
E47     050056
E5      013562
E50     050340
E51     050554
E52     051046
E53     051046
E54     051122
E55     052104
E56     052104
E57     052410
E6      013612
E60     052410
E61     060034
E62     060034
E63     060120
E64     060214
E65     060300
E7      014170
FASTCI= 177640
FATAL$  002062
FCMD10  050646
FCMD11  050674
FCMD12  050716
FCMD13  050736
FCMD14  050760
FCMD15  050776
FCMD16  051062
FCMD17  051124
FCMD18  051140
FIELDS  045406
FINDBA= 000045
FINT    006440
FIRST = 060000
FLIPLO  002560
FLIPWA  036402
FLUSH   014332
FSCMD0  045634
FSCMD1  045706
FSCMD2  046016
FSCMD3  046164
FSCMD4  046440
FSCMD5  046760
FSCMD6  047650
FSCMD7  047656
FSCMD8  050150
FSCMD9  050354
FSINFL  002414
FSPAT   047456
FSSTAC  002270
FS1     045472
FS7FLA  002420
FULLRE  002514
GBLENG= 000076
GETDAT  051534
GETDA1  051632
GETDIS  055726
GOOD    002042
GOOD2   002044
GOOD3   002046
GTSWR = 104407
HEADER  002554
HIPAT   045004
HOLDLO= 000011
HT    = 000011
I       002424
IBSAVE  056554
IIII  = 177777
ILLCSR  013260
IMPTES  012224
INCBNK  045014
INCPAT  044770
INCRPT  044770
INHBAN  002512
INHECC  002510
INTFLA  002134
INT64K  002136
INVALI= 104511
IOTVEC= 000020
JMPRL1  043302
KAMIKA  002004
KAMITE  026432
KDIAG = 000010
KDPAR0= 172360
KDPAR6= 172374
KDPAR7= 172376
KERNEL= 104417
KERSTK= 002000
KFLAG   002502
KIPAR0= 172340
KIPAR4= 172350
KIPAR5= 172352
KIPAR6= 172354
KIPDR0= 172300
KMAP  = 104422
KPFLAG  002112
KSIZE   002350
KSTACK  002536
LAST  = 157776
LASTBA  002530
LASTBL  002532
LASTER  002014
LBLS0 = 000455
LBLS1 = 000065
LBLS2 = 000447
LBLS3 = 000442
LBLS4 = 000315
LBLS5 = 000317
LBLS6 = 000016
LCSROU= 000041
LCSRRE= 000057
LCSRSA= 000055
LEGALC= 000003
LF    = 000012
LINK1   002474
LINK2   002476
LKS   = 177546
LOADBA  002404
LOADCS= 104425
LOADER= 000043
LOADHO  002540
LOOP    014212
LOWMAP  043706
LSIZE   002352
LWDBE = 000037
LWSBE = 000035
L0      004566
L1      004630
L10     005126
L100    014212
L101    014326
L102    014426
L103    014436
L104    014606
L105    014606
L106    014674
L107    015006
L11     005134
L110    015114
L111    015244
L112    015374
L113    015524
L114    015676
L115    016026
L116    016200
L117    016260
L12     005164
L120    016316
L121    016326
L122    016752
L123    016734
L124    017000
L125    017242
L126    017352
L127    017510
L13     005416
L130    017536
L131    017542
L132    017544
L133    017610
L134    017644
L135    017650
L136    017652
L137    020020
L14     005416
L140    020474
L141    021342
L142    021352
L143    021352
L144    021410
L145    021420
L146    021420
L147    021450
L15     010722
L150    021454
L151    021504
L152    021514
L153    021514
L154    021560
L155    021570
L156    021570
L157    021600
L16     010726
L160    021636
L161    021646
L162    021646
L163    021704
L164    021714
L165    021714
L166    021774
L167    022004
L17     010730
L170    022004
L171    022146
L172    022074
L173    022074
L174    022126
L175    022126
L176    022204
L177    022240
L2      004736
L20     011542
L200    022242
L201    022270
L202    022272
L203    022356
L204    022470
L205    022472
L206    023332
L207    023364
L21     011556
L210    023430
L211    023700
L212    023710
L213    023710
L214    024454
L215    024422
L216    024454
L217    024520
L22     011702
L220    024662
L221    024630
L222    024662
L223    025074
L224    025074
L225    025202
L226    025154
L227    025232
L23     011660
L230    025422
```

SYMBOL TABLE

```
L231    025754  L313    043046  L376    053704  L5      005054  MSG003  071123
L232    026244  L314    043046  L377    053706  L50     012630  MSG004  071230
L233    026302  L315    043046  L4      004776  L51     012640  MSG005  071336
L234    026402  L316    043026  L40     012216  L52     012660  MSG006  071350
L235    026454  L317    043046  L400    054536  L53     013504  MSG007  071405
L236    026462  L32     012020  L401    054550  L54     013460  MSG008  071417
L237    026466  L320    043114  L402    054566  L55     013434  MSG009  071431
L24     011672  L321    043114  L403    054570  L56     013426  MSG010  071443
L240    030304  L322    043144  L404    054610  L57     013432  MSG011  071455
L241    030302  L323    043264  L405    054622  L6      005134  MSG012  071543
L242    030304  L324    044444  L406    054640  L60     013456  MSG013  071640
L243    031732  L325    044670  L407    054642  L61     013452  MSG014  071642
L244    034474  L326    045052  L41     012420  L62     013456  MSG015  071644
L245    034504  L327    045156  L410    054656  L63     013504  MSG016  071646
L246    034616  L33     012136  L411    055060  L64     013504  MSG017  071660
L247    034546  L330    045162  L412    055066  L65     013504  MSG018  071671
L25     011706  L331    045170  L413    055114  L66     013600  MSG019  071674
L250    034560  L332    045204  L414    055132  L67     013650  MSG020  071700
L251    034576  L333    045216  L415    055144  L7      005134  MSG021  071721
L252    034604  L334    045430  L416    055156  L70     014022  MSG022  072511
L253    034630  L335    045440  L417    055170  L71     014212  MSG023  072533
L254    034766  L336    045574  L42     012420  L72     014160  MSG025  072547
L255    034666  L337    045634  L420    055212  L73     014076  MSG026  072573
L256    034700  L34     012070  L421    055260  L74     014100  MSG027  072605
L257    034732  L340    045640  L422    055340  L75     014140  MSG028  072622
L26     012136  L341    045670  L423    055354  L76     014154  MSG029  072636
L260    034716  L342    045702  L424    055356  L77     014210  MSG030  072656
L261    034730  L343    047070  L425    055462  MAINT = 177750  MSG031  072675
L262    034754  L344    047130  L426    055552  MAPHO = 170202  MSG032  072735
L263    034742  L345    047166  L427    055562  MAPLO = 170200  MSG033  072754
L264    034754  L346    047374  L43     012420  MAPL1 = 170204  MSG035  073102
L265    035066  L347    047410  L430    056070  MAPPER  042072  MSG036  073105
L266    036744  L35     012136  L431    056070  MBERR   013136  MSG037  073124
L267    037372  L350    047422  L432    056324  MEMDON  014260  MSG038  073143
L27     012136  L351    047726  L433    056264  MFPT = 000007  MSG039  073161
L270    037560  L352    047730  L434    056304  MJPAT   020044  MSG040  073203
L271    037662  L353    050042  L435    056324  MJTEST  017740  MSG041  073237
L272    040574  L354    050340  L436    056522  MKCONT  016306  MSG042  073264
L273    040602  L355    050324  L437    056372  MKCSRT  017430  MSG043  073305
L274    041040  L356    050274  L44     012424  MKFLAG  002116  MSG046  073332
L275    041040  L357    050424  L440    056520  MKLOOP  016470  MSG047  073365
L276    041102  L36     012210  L441    056450  MKPAT   017660  MSG048  073404
L277    041110  L360    050426  L442    056462  MKTEST  017520  MSG049  073444
L3      004760  L361    050540  L443    056520  MMR0 = 177572  MSG050  073476
L30     012136  L362    051030  L444    056530  MMR1 = 177574  MSG051  073604
L300    041204  L363    051032  L445    057064  MMR2 = 177576  MSG052  073624
L301    041204  L364    051276  L446    060012  MMR3 = 172516  MSG053  073655
L302    041246  L365    051312  L447    060216  MMTRAP  040144  MSG054  073673
L303    041254  L366    051316  L45     012702  MMVEC = 000250  MSG055  073743
L304    041376  L367    051334  L450    060222  MSEEDH  002550  MSG056  073764
L305    042544  L37     012214  L451    060302  MSEEDL  002552  MSG058  074017
L306    042560  L370    051462  L452    060306  MSGA12  075377  MSG061  074041
L307    042572  L371    051464  L453    061422  MSGA34  072773  MSG062  074050
L31     012014  L372    051530  L454    061512  MSGB34  073037  MSG063  074070
L310    042572  L373    051754  L455    061514  MSG000  070717  MSG064  074101
L311    042674  L374    052062  L46     012702  MSG001  070764  MSG065  074111
L312    042724  L375    052366  L47     012656  MSG002  071046  MSG066  074123
```

SYMBOL TABLE

MSG067 074206
MSG070 074215
MSG071 074246
MSG072 074264
MSG073 074302
MSG075 074320
MSG076 074352
MSG077 074373
MSG079 074407
MSG085 074433
MSG088 074460
MSG089 074476
MSG090 074520
MSG091 074534
MSG092 074546
MSG093 074562
MSG095 074570
MSG101 074577
MSG102 074627
MSG103 074656
MSG104 074700
MSG105 074702
MSG106 075006
MSG107 075024
MSG110 075101
MSG111 075145
MSG112 075177
MSG113 075214
MSG114 075231
MSG116 075255
MSG117 075267
MSG118 075301
MSG119 075313
MSG120 075322
MSG121 075343
MSG122 075363
MSG123 075431
MSG124 075502
MSG125 075545
MSG126 075567
MSG127 075634
MSG128 075653
MSIZE 002354
MTA030 024732
MTB020- 000017
MTEST 016212
MTLA11 030050
MTLB11 030062
MTLC11 030074
MTLD11 030170
MTLQ20 022034
MTPA03 027146
MTPA04 027304
MTPA20 034110
MTPA21 034274
MTPA24 035120
MTPA25 035530
MTPA26 035660
MTPB03 027206
MTPB04 027340
MTPB20 034140
MTPB21 034324
MTPB24 035160
MTPB25 035552
MTPB26 035674
MTPC03 027246
MTPC20 034170
MTPC21 034360
MTPC24 035174
MTPC25 035612
MTPC26 035730
MTPD03 027264
MTPD20 034220
MTPD21 034414
MTPD25 035456
MTPD26 035750
MTPE20 034250
MTPE25 035500
MTP000 027036
MTP001 027062
MTP002 027114
MTP005 027360
MTP006 027414
MTP007 027614
MTP010 027714
MTP011 030022
MTP012 030620
MTP013 031206
MTP014 031722
MTP015 032504
MTP016 033250
MTP017 034032
MTP020 034110
MTP022 034444
MTP025 035212
MTP030 035766
MTP031 035776
MTP032 036054
MTP033 036106
MTP034 036204
MTP035 036230
MTST3 011562
MTV020 022430
MT0000 020124
MT0001 020204
MT0002 020324
MT0003 020464
MT0004 020716
MT0005 021040
MT0006 021174
MT0007 021230
MT0010 021272
MT0011 021326
MT0012 021374
MT0013 021470
MT0014 021544
MT0015 021622
MT0016 021670
MT0017 021736
MT0020 021760
MT0021 023050
MT0022 023322
MT0023 023354
MT0024 023420
MT0025 023664
MT0026 023732
MT0027 024234
MT0030 024720
MT0031 025222
MT0032 025412
MT0033 025744
MT0034 026132
MT0035 026304
MT020Y 022616
MT020Z 022432
MT0999 026416
MT1 016266
MT2 016272
MUT 002106
NC 053706
NEMCNT 002066
NEMBAN 002272
NEMKER 044202
NEMLOA 044250
NOCH 061016
NOERRO 002402
NOFSMO 002400
NONEM 002076
NONEXI 040066
NOOJ 036744
NOPAR 002074
NORES 003642
NOSCOP 002412
NOSUPE 002430
NOTAB 002344
NOTRCE 055476
NO22BI 002426
NULLFL 002316
OLDCAC 002264
OLDCSR 002152
ONES 002556
PADDRE 002034
PAFBAF 015406
PAFBAM 015536
PARBAF 015710
PARBAM 016040
PARCNT 002070
PARITY 037762
PARTHE 002266
PARVEC- 000114
PASFLG 002260
PATERR 002072
PATPLU 004550
PATTER 002110
PCBUPP 002300
PCONF1 036632
PCONFS 037132
PCONF1 037042
PCONF2 037100
POP110 040156
PD1 051754
PERA05 054122
PERBNK 054754
PERECC 055034
PERRAB 054572
PERRAM 054520
PERRA3 051322
PERRA7 054644
PERR01- 104427
PERR02- 104430
PERR03- 104431
PERR04- 104432
PERR05 054116
PERR06 054144
PERR07- 104433
PERR10- 104434
PERR11- 104435
PERR12- 104436
PERR13- 104437
PERR14- 104440
PERR15- 104441
PERR16- 104442
PERR17- 104443
PERR20- 104444
PERR21- 104445
PERR22- 104446
PERR23- 104447
PERR24- 104450
PERR25- 104451
PERR26- 104452
PERR27- 104453
PERR30- 104454
PERR31- 104455
PERR32- 104456
PERR33- 104457
PERR34- 104460
PERR35- 104461
PERR36- 104462
PERR37- 104463
PERR40- 104464
PERR41- 104465
PERR42- 104466
PERR43- 104467
PERXOR 054730
PFECDF 057176
PFECDM 057136
PFECDT 057166
PFFCEM 057102
PFECMS 057076
PFLAG 002120
PGMCSR 002504
PHEBE 013140
PHYADD 002036
PROTYP 003720
PSIZE 002356
PSW - 177776
PWRVEC- 000024
QUICK 002410
QVFLAG 002320
RANODD 035710
RDCHR - 104411
RDDEC - 104414
RDLIN - 104412
RDOCT - 104413
READCS- 104426
READDN 002362
REALPA 002262
REFRES 035020
REFSUB 035070
REGCOP 036372
RELENT 043154
RELOCA 042530
RELOC1 043170
RESREG- 104416
RESTAR 002570
RESVEC- 000010
RES0 045702
RES1 045762
RES2 046130
RLFLAG 002124
RRFLAG 002122
RTNVAL- 000000
SAVCSR 002150
SAVREG- 104415
SBEMSK 002246
SBENT 017372
SBETES 017114
SCOPE - 000004
SDPAR0- 172260
SDPAR5- 172272
SDPAR6- 172274
SDPAR7- 172276
SEEDHI 002544
SEEDLO 002546
SELONL 002000
SETPAT 045004
SMADL1 011612
SMUTUP 045170
SIPAR0- 172240
SIPAR3- 172246
SIPAR5- 172252
SIPAR6- 172254
SIPDR0- 172200
SIZE - 040000
SKIPKA 002006

```
SKIP%K  002314
SKJ     055524
SKPERR  002064
SKUB    043144
SKUJ    013142
SOB%    002534
SOBLEN- 000056
SOFTPA  002562
SOURCE  002274
SPLTCS  002234
SSP     =000006
ST      = 177776
STACK   = 002000
START   003632
START1  000300
START2  000310
START3  000200
START27 024314
STOP%   002372
STRIPE  002340
SUBAAA  004606
SUBAAB  004736
SUBAAI  011606
SUBAAP  013322
SUBAAR  012466
SUBAAS  010540
SUCCES  002306
SUPD0A  002256
SUPD01  026466
SUPD02  026502
SUPD03  026644
SUPD04  026660
SUPDR0  002154
SUPDR1  002156
SUPDR2  002160
SUPDR3  002162
SUPDR4  002164
SUPDR5  002166
SUPDR6  002170
SUPLIM  053764
SUPST%- 000740
SWAPAT  002576
SWR     002600
SWREG   = 000176
SW0     = 000001
SW1     = 000002
SW10    = 002000
SW11    = 004000
SW12    = 010000
SW13    = 020000
SW14    = 040000
SW15    = 100000
SW2     = 000004
SW3     = 000010
SW4     = 000020
SW5     = 000040
SW6     = 000100
SW7     = 000200
SW8     = 000400
SW9     = 001000
SYSSIZ  003722
TAG2$   011202
TAG3$   011236
TAG4$   026562
TAG70$  057202
TAG71$  057212
TAG72$  057222
TAG73$  057272
TAG74$  057332
TAG75$  057344
TAG76$  057356
TAG77$  057422
TAG78$  057430
TAG79$  057510
TAG9$   011030
TBG4$   026740
TCFIG1  037256
TCFIG2  037410
TCFIG3  037576
TCONFI  037134
TEMP    002406
TEST    006044
TESTAD  002364
TESTMO  002524
TIME    002312
TIMEOU  040132
TKVEC   = 000060
TMFLAG  002132
TOOMAN  002360
TOTCSR  002220
TRACE   006140
TRAPVE- 000034
TSTBAN  011450
TSTDAT  002242
TSTRD1  040604
TSTREA- 104510
TST1    005436
TST2    010544
TST3    010730
TST4    011716
TST5    014212
TST6    014264
TYPDS   = 104405
TYPEIT- 104401
TYPOC   = 104402
TYPOS   = 104403
TYPS0   = 000000
TYPS1   = 000002
TYPS2   = 000000
TYPS3   = 000000
TYPS4   = 000000
TYPS5   = 000000
TYPS6   = 000002
T12A    033250
T12B    033272
UDPAR0- 177660
UDPAR7- 177676
UIPAR0- 177640
UIPAR1- 177642
UIPAR2- 177644
UIPAR3- 177646
UIPAR4- 177650
UIPAR5- 177652
UIPAR6- 177654
UIPDR0- 177600
UNITOP  002370
UNRELO  043420
UPPFLG  002261
USERMA  044120
USESTK- 000700
USP     =000006
WARN1   011116
WARN2   027160
WARN3   027174
WARN4   027220
WARN5   027234
WARN6   036614
WARN6A  036554
WARN6B  036606
WARN7   024272
WASDBE- 104500
WASSBE- 104476
WAS1DB- 104501
WAS1SB- 104477
WHICHC  051152
WOOPEN  053162
WOOPS   052614
WOOPSA  053212
WOOPUP  053000
WORST   002542
XOCHAR  053554
XXDPCH  002326
ZEROS   002310
$APTHD  063176
$AUTO   002060
$BANK   002011
$BASE   063152
$BELL   002615
$CACH%  040274
$CACHN  040250
$CBCSR  040734
$CB1CS  040756
$CDW1   063156
$CDW2   063160
$CHARC  053742
$CHKDI  041330
$CHK1D  041344
$CKSWR  060774
$CLRCS  041306
$CLR1C  041320
$CMTAG  002000
$CMTGE  002516
$CNTLC  062166
$CNTLG  062200
$CNTLK  061374
$CNTLU  062173
$CPUOP  063124
$CRLF   002622
$DBLK   060764
$DB20   062756
$DDW0   063162
$DDW1   063164
$DDW2   063166
$DDW3   063170
$DDW4   063172
$DDW5   063174
$DEENE  040240
$DEVCT  063106
$DEVM   063154
$DIDOO- 000000
$DOAGA  014606
$DOAGN  014502
$DOWN   052204
$DTBL   060754
$ECCDI  040632
$ECCIN  040660
$ECC1D  040646
$ECC1I  040674
$ENASB  040706
$ENA1S  040722
$ENDAD  014472
$ENERG  040230
$ENV    063116
$ENVM   063117
$EOP    014336
$ERFLG  002012
$ERRGE  041454
$ERROR  056000
$ERRTB  063520
$ERRTY  056562
$ERTTL  002572
$ESCAP  002334
$ETABL  063116
$ETEND  063176
$EXHAL  045162
$E$     = 000001
$FATAL  063100
$FILLC  002614
$FILLS  002331
$F$     = 000000
$GTSWR  061150
$HALT   056344
$HALT2  063252
$HIBTS  063176
$HIOCT  062376
$ILLUP  052606
$INVAL  041424
$ITEMB  002013
$I$     = 000001
$KERNE  040220
$KMAP   042436
$K$     = 000061
$L      = 000066
$LF     002623
$LL     = 000064
$LOADC  040312
$LPADR  002564
$LPERR  002566
$L$     = 000000
$MADR1  063130
$MADR2  063134
$MADR3  063140
$MADR4  063144
$MAIL   063076
$MAMS1  063126
$MAMS2  063132
$MAMS3  063136
$MAMS4  063142
$MBADR  063200
$MNEW   062216
$MSGAD  063112
$MSGLG  063114
$MSGTY  063076
$MSWR   062205
$MTYP1  063127
$MTYP2  063133
$MTYP3  063137
$MTYP4  063143
$NOTRA  063246
$NULL   002330
$NWTST- 000001
$OCNT   060544
$OCTVL  063060
$OCT8   = 063064
$OMODE  060546
$OVER   055714
$O$     = 000000
$PASS   063104
$PASTM  063204
$PATMA  002010
$PER01  053764
$PER02  054012
$PER03  054040
$PER04  054070
$PER07  054152
$PER10  054174
$PER11  054224
$PER12  054244
$PER13  054266
$PER14  054306
$PER15  054330
$PER16  054352
$PER17  054372
$PER20  054410
$PER21  054426
```

$PER22	054446	$QUE	062621	$SWR	= 163000	$TSTRD	040426	$VECT1	063146
$PER23	054464	$R	= 177777	$SWREG	063120	$TTYIN	062142	$VECT2	063150
$PER24	054502	$RAND	062662	$T	= 000456	$TYPDS	060550	$WASDB	041142
$PER25	051240	$RDCHR	061514	$TESTN	063102	$TYPE	053430	$WASSB	040776
$PER26	054672	$RDDEC	062400	$TKB	002606	$TYPEC	053556	$WAS1D	041256
$PER27	054712	$RDLIN	061644	$TKS	002604	$TYPEX	053744	$WAS1S	041112
$PER30	051466	$RDOCT	062230	$TN	= 000007	$TYPOC	060346	$XTSTR	055572
$PER31	055102	$READC	040406	$TPB	002612	$TYPON	060362	Y	= 000000
$PER32	055200	$RESRE	062624	$TPFLG	002332	$TYPOS	060322	$ZAP42	014452
$PER33	055246	$SAVRE	062566	$TPS	002610	$T1	= 000000	Z	= 000000
$PER34	055326	$SAVR6	052612	$TRAP	063212	$T2	= 000455	$$S	= 000000
$PER35	055360	$SCOPE	055444	$TRAP2	063234	$UNIT	063110	$$T	= 000441
$PER36	055414	$STN	= 000001	$TRPAD	063254	$UNITM	063206	$$TT	= 00044?
$PWRDN	051634	$SVLAD	055700	$TSTM	063202	$USWR	063122	$OFILL	060545
$PWRUP	052210	SV	= 000000						

. ABS. 075674 000
000000 001
ERRORS DETECTED: 0

VIRTUAL MEMORY USED: 26160 WORDS (103 PAGES)
DYNAMIC MEMORY: 20034 WORDS (77 PAGES)
ELAPSED TIME: 02:36:38
CZMSDO.BIN,CZMSDO.SEQ/CR/-SP/NL:TOC=CZMSDO.SML,CZMSDO.P11

SYMBOL CROSS REFERENCE
SYMBOL VALUE REFERENCES

```
ABORTF   002140  #139 5048  •322 10561  406 12694  408 12740  •408 12756  439 14065  439 14068  439 14077  439 14080
ACFLAG   002114  #139 5038   167 6302   169 6342   179 6835   188 7176   190 7192   192 7222   194 7252   196 7289
                  198 7330   200 7368   202 7413   204 7451   208 7545   236 8392   236 8421   238 8464   348 11105
                 •356 11378 •356 11393  356 11399  378 11943  382 12013
ACTFLA   002322  #139 5105  •153 5486   164 6163   170 6378   214 7730   214 7745   223 8003   223 8011   223 8025
                  223-8034   225 8045   225-8053   227 8068   227 8068   232 8320   246 8673   348 11099  362 11506
                  451 14378
ADDRE    002032  #139 5014  •164 6157  •266 9072  •266 9078  •266 9099  •266 9112  •268 9188  •268 9202  •270 9276
                 •274 9376  •274 9383  •276 9478  •276 9484  •278 9576  •278-9583  •302 10043 •302 10055 •312-10252
                 •312 10267 •324 10587 •406 12693 •406 12693 •406 12719 •408 12739 •408 12739  408-12742  408 12752
                 •436 13958 •436-13959 •436 13964 •436 13965 •436 13970 •436 13971 •436 13976 •436 13977 •436 13984
                 •436 13992 •436 13997 •436 13997 •436 14002 •436 14007 •438-14013 •438 14018 •438 14023 •438 14028
                 •438 14033 •438 14038 •438 14043 •438 14048 •438 14053 •438 14058  443-14129 •445 1415? •445 14166
                  458 14568  495 15879  495 15884  495 15887  495 15892  495 15910
ANAL     007170  #163 5094
APTDOW   045252   153 5480  #362 11540  362 11545 •362 11545
APTECC   002376  #139 5129  •183 7054   183 7059   495 15886
APTFLA   002324  #139 5106  •153 5479   183 7030   186 7158   223 8003   223 8011   223 8025   223 8034   225 8045
                  225-8053   232-8320   246 8673   348 11099  362 11506  451 14378
APTMAN   014556  #186-7160   186 7166
APTMLT   045320  #362-11547
APTPAR   002374  #139-5128  •183 7051   183 7059   495 15886
APTSIZ   002416  #139 5137  •153 5476   183 7030
BACK     056014  #450-14305   451 14395
BACKGN   036532   219-7940   219-7958   221 7986   227 8130   230 8216   232 8283   244 8657   244 8662  #314 10310
BAD      002050  #139 5020  •158-5692  •158-5700  •406 12696 •406-12722 •408 12742 •408-12754 •436-13960 •436 13966
                 •436 13973 •436-13978 •436-13983 •436 13993 •436 13998 •436-14003 •436-14008 •438-14014 •438 14019
                 •438-14024 •438 14029 •438-14034 •438-14039 •438 14044 •438 14049 •438 14054 •438 14059  441 14106
                 •443-14128 •445-14158 •445-14167 •445-14168 •445 14187 •445 14194  495-15879  495-15887  495 15895
                  495 15901  495-15906  495-15909
BADPC    002020  #139-5009  •324-10614  406-12707  439-14066  439 14078  441 14089  443-14138  445-14156  445 14165
                  445-14176  450 14341  450-14342 •450-14346
BADPSW   002030  #139-5013  •324-10615  445-14187  450-14345
BADSP    002024  #139 5011  •324 10612 •324-10613  450-14344
BADSTA   040170   322 10562   322-10570  324-10595  324-10599  324 10602 #324-10611  406-12707  436-13986  439 14066
                  439 14078  441-14089  443-14138  445 14156  445 14165  445 14176  445-14186
BADXOR   002056  #139-5023  •441 14106 •441-14107  495-15887
BAD2     002052  #139 5021  •406-12723  495-15895
BAD3     002054  #139 5022  •406-12725 •406-12726  495-15895
BAFPAF   014712   184 7081  #190-7188   190-7213
BAFPAR   015020   184-7082  #192-7218   192-7243
BAKPAT   002574  #141-5226  •147-5368   230-8215   232-8282
BANK     002100  #139 5032  •166-6178  •166-6190   166-6191   166-6193   166-6200  •166-6219  •167-6299  •167-6305
                 •167-6308  •167-6312   167-6314  •169-6338  •170-6375   170-6375  •170-6377   171-6387  •179-6810
                  179-6865  •179-6870   179-6870  •188-7174  •188-7182   188-7182  •190-7189  •192-7219  •194-7249
                 •196-7286  •198 7325  •200-7363  •202-7408  •204-7446   208-7514   208-7541   208-7578  •208-7593
                  210 7648   210 7673   227-8080   227-8087   227-8131   230-8226   230-8240   232-8293   234-8342
                  234-8355  •236-8390   236-8394  •236-8412   236-8412  •236-8419   236-8422  •236-8441   236-8441
                 •238-8461  •238 8470   238-8470  •238-8480   240-8498   242-8527   244-8591   248-8681   250-8714
                  339-10858  346 11062 •348 11103  348-11106 •348-11116  348-11116  350-11210 •350-11211 •350-11215
                 •350 11219  356 11381 •358 11470  358-11471   371-11691 •371-11696 •371-11700   371-11703   371-11712
                 •371 11727  373 11733 •373 11738 •373 11742   373-11745   373-11751 •373-11779   375-11785 •375 11790
```

SYMBOL CROSS REFERENCE CREF V01

SYMBOL	VALUE	REFERENCES								
		375-11791	375 11793	375 11812	375-11824	375 11840	375 11849	•375-11903	375 11904	378 11920
		•378-11941	•378 11947	378-11947	•378 11954	378 11955	380 11961	•380-11970	380-11971	380 11978
		•380-11984	380 11984	•380-11986	382-11991	•382-12011	•382-12017	382-12017	•382-12024	382-12025
		415-12953	•415 12954	415 12955	•415-12975	417-12994	•417-12995	417 12996	•417-13009	443-14115
		448-14272	457-14512	457-14526	457-14543	458 14550	458-14570	470-15018		
BANKIN	002102	#139-5033	•166-6196	166-6252	167 6301	169 6341	179 6812	208 7520	223 8015	227 8076
		244 8621	244 8647	340-10867	346 11040	348 11111	350 11213	350 11217	•356-11384	
BANKMC	043752	186 7157	348-11109	350 11208	#352-11261	362 11530				
BANKOK	044752	198 7328	200 7366	202-7411	204 7449	#358 11445				
BALPAF	015126	184 7083	#194 7248	194-7272	194 7280					
BALPAR	015256	184-7084	#196 7285	196 7309	196 7317					
BGTEST	036110	#310-10188	310-10209							
BIT0	• 000001	#111 4178	157-5652	157 5656	158 5682	158 5688	158-5698	163-6016	163-6037	163 6047
		163 6083	163-6091	166-6259	166-6278	312 10247	312-10262	318-10396	320 10478	320-10484
		326 10628	326-10632	326-10639	329-10700	329-10702	331-10723	331-10727	331-10731	331 10735
		333 10770	333-10772	333-10778	333-10781	335 10805	335 10807	335-10815	335-10817	348 11127
		348 11181	348-11190	350 11229	350-11232	356-11423	371-11706	373 11747	443 14118	447 14223
		450 14332	450-14334							
BIT1	• 000002	#111-4177	151-5449	157 5661	157-5667	158 5680	158 5696	159 5761	159-5774	163-5966
		163-5986	163 6091	179 6883	179 6909	179 6921	320 10438	320-10476	320-10505	331 10715
		331 10719	331 10731	331-10735	331-10740	331-10745	337-10831	337-10836	356-11388	
BIT10	• 002000	#111-4168	181 6959	312-10255						
BIT11	• 004000	#111 4167	163-6018	179-6850	272-9313	274 9421	276-9519	280-9621	356-11435	
BIT12	• 010000	#111 4166	155 5581	155-5582	155 5588	163-6014	179-6850	179-6889	179 6923	223 8016
		320 10436	348-11192	356-11432	375-11847	463 14701	463 14702	463-14717		
BIT13	• 020000	#111 4165	155-5581	155-5588	157-5658	157-5659	157-5662	157-5665	157-5668	171-6389
		171 6405	186-7119	244-8627	244 8641	266-9141	272-9310	274-9418	276 9516	280-9618
		328-10664	329 10690	339-10861	348-11113	348 11161	350-11218	350-11223	350-11241	350-11254
		360 11498	463 14701	463-14717						
BIT14	• 040000	#111-4164	157 5653	157-5654	163-6068	163-6070	248-8703	293-9888	293-9889	329-10709
		340 10893	340 10895	348 11167	350-11226	352-11285	352-11301	356-11429	385-12091	385 12107
		411 12806	411 12812	414 12909	414-12913	415 12956	417-12997			
BIT15	• 100000	#111 4163	157 5652	158-5714	158-5724	158 5733	160-5853	163-6031	163-6057	210-7642
		210 7660	210 7663	210-7667	228-8152	266-9083	329-10699	329-10709	335-10794	348 11112
		348 11128	348-11167	348-11176	348 11197	350-11214	350-11226	352-11285	352-11301	391-12126
		411-12806	414 12913	445-14177	448-14277	458-14573				
BIT2	• 000004	#111-4176	154 5515	154-5516	154-5517	158-5671	158-5679	158-5694	159-5763	163-5968
		163-6004	163-6093	312 10247	320-10472	322 10564	331-10740	331-10745	337-10831	337-10836
BIT3	• 000010	#111 4175	154-5515	154-5516	154-5519	157-5650	157-5664	158-5727	158-5736	
BIT4	• 000020	#111-4174	137-4968	145-5325	145-5326	157-5650	157-5667	158-5724	158-5733	163 6031
		163-6057	186-7153	210-7642	210-7667	266 9083	329-10699	333-10759	333-10779	445-14177
BIT5	• 000040	#111 4173	153-5472	163-5959	163-6022	163-6052	186-7153	348-11172	348-11180	350-11236
		350-11239	356-11438	360-11489	362-11534					
BIT6	• 000100	#111-4172	169-6358	316-10337	316-10374	320 10529	348-11131	356-11385	375-11836	375 11899
		378 11936	378-11952	382-12007	382-12022					
BIT7	• 000200	#111-4171	153-5475	154-5494	154-5495	167-6310	266-9132	272-9299	274-9407	276-9505
		280-9607	316 10336	329-10685	348-11127	348-11131	356-11417	375-11835	378-11935	382-12006
BIT8	• 000400	#111-4170	181-6960	181-6975	356-11408	356-11414	471-15088			
BIT9	• 001000	#111-4169	181-6961	181-6967	181-6974	348-11130	356-11411			
BLOCK1	045322	166-6187	217-7847	217 7869	217-7895	219-7919	219-7922	219-7948	219-7968	228-8162
		228 8171	228-8187	228 8196	228-8205	230 8244	230-8247	230-8251	230-8254	232-8303
		234 8359	234 8367	236 8385	238 8454	242 8556	244-8625	314-10323	329-10695	#364 11557

SYMBOL CROSS REFERENCE CREF V01

SYMBOL	VALUE	REFERENCES
BLOCK2	045342	#364 11562
BLOCK3	045356	219-7923 219 7924 219 7949 219 7969 232 8304 232 8305 234 8360 234 8362 240 8500 244 8594 348 11168 350 11227 352 11286 352 11302 #364 11566 415 12959 415 12962
BMFLAG	002126	#139-5043 194 7254 196 7291 200-7370 204 7453 348 11105 #356 11379 #356 11406 #356 11425 #356-11428
BOOT	045030	#360-11475 468-14965 471-15092
BOOT1	045074	360 11483 360 11488 #360-11490
BRGOBB	036112	#310-10189 310-10193 310-10227
BSIZE	002346	#139 5118 #181 6953 #181-6976 #181 6985 #181 6986 #181 6987 #181-6988 181 6989 181 7002 181 7004
CACHCF	002522	#141-5200 #145-5318 326 10646 367-11641
CACHCN	002516	#141-5199 #145-5320 154-5523 #154 5523 #154 5524 326 10636 326 10638 326 10643 365-11588 367-11639 367-11641 367-11644 384 12055 #384 12055 #384 12056 384-12061 #384-12061 410 12776 414 12933 419-13071 419 13131
CACHOF	• 104424	#110-4055 154-5522 159-5742 163-5955 164-6147 166-6209 166-6226 166-6243 166 6269 169-6336 208-7557 210-7626 248 8705 250 8742 292 9850 365-11592 384 12054 419 13075
CACHON	• 104423	#110-4054 147-5395 160 5861 163-6125 164-6131 164-6150 166 6217 166 6230 166-6251 166-6274 170-6383 208 7569 210 7678 210 7686 248 8707 250 8744 292 9828 384-12062 410-12779
CACHVE	• 000114	#111-4193 111 4194
CBCSR	• 104474	#110-4098
CB1CSR	• 104475	#110-4099 163-5976 266 9063 268-9173 270 9268 274 9367 276 9467 278-9566 302 10032
CHECK	002276	#139-5095 #163-5926 #266 9138 #266 9139 #272 9304 #272 9306 #272-9308 #274 9412 #274-9414 #274-9416 #276-9510 #276 9512 #276 9514 #280 9612 #280 9614 #280-9616 331-10739 331 10744 #342-10946 495-15892
CHKDIS	• 104504	#110-4106
CHKGEN	041724	266-9051 268-9164 270-9257 274 9355 276 9454 278 9554 300 9972 #342 10920
CHKTAB	042032	342-10930 #344 10951
CHKLDI	• 104505	#110-4107 171-6394 171-6406 171 6416 171 6426 406 12710
CKEND	061306	468-14946 468-14948 468-14950 #468 14979
CKSWR	• 104410	#110-4038 447-14215 450-14304 451 14367
CLRCSR	• 104502	#110-4104 159-5786 160-5854 #214-7733 236 8380 238 8473 375 11856 #378-11927 #382 11998
CLREX	007140	161-5873 #161-5889
CLRMEM	007040	159-5791 160 5862 #161-5873
CLR1CS	• 104503	#110 4105 163 5975 163-5984 163 6115 171 6398 171 6411 171-6421 171-6430 171 6444 210-7634 210 7658 210-7682 228 8208 266 9029 266 9090 266-9103 266-9149 268-9156 268-9176 268 9231 270-9237 270 9251 270 9270 272 9321 274-9349 274-9371 274-9429 276-9448 276-9469 276 9528 278 9570 280 9630 300 10007 312-10260 406-12716
CMD16A	051072	385-12080 #385-12103
CMD5B	047302	#375-11840 375-11865
CMD5C	047552	375 11817 375-11832 #375-11898
CMD7B	050010	#378 11941 378 11949
CMD7C	050064	378 11932 #378 11951
CMD9B	050506	#382-12011 382-12019
CMD9C	050562	382-12003 #382-12021
CONFGE	002422	#139-5143 #163-6039 #163 6085 #166 6260 #166-6279 #179-6866 #179-6948
CONFIE	003632	#143-5269 151 5445
CONFIG	002626	#143-5266 151-5443 #154 5494 #154-5495 163-5959 #163-6007 #163-6011 #163-6014 #163-6018 #163 6022 163-6023 #163 6023 163-6024 #163-6024 #163-6037 #163-6038 163-6040 #163-6040 163-6041 #163-6041 163 6050 #163-6050 163-6051 #163-6051 #163-6052 #163-6083 #163-6084 #164 6159 166 6201 #166 6259 #166 6261 #166-6278 #166-6282 #167-6310 169-6351 #169-6358 170 6363 #171 6447 #171 6448 179 6836 179 6844 #179-6850 #179-6851 179-6882 179 6889

SYMBOL CROSS REFERENCE
CREF ...

SYMBOL	VALUE	REFERENCES
		179-6909 179 6921 179 6923 181 6958 181 6959 181 6960 181 6961 181 6967 181 6974
		181 6975 181 6992 186 7118 208 7521 208 7532 223 8016 227-8108 244 8627 •244 8641
		244 8648 312 10255 318 10396 318 10410 320 10436 320 10438 320 10444 320 10465 320 10467
		320 10504 320 10525 320 10529 •339 10861 •348 11112 •348 11113 348 11127 348 11128 348 11129
		348 11130 348 11131 348 11131 •348 11161 348 11186 348 11192 348 11194 •350-11214 •350 11218
		•350 11223 350 11240 356 11385 356 11392 356 11401 356 11408 356 11411 356 11414 356-11417
		356 11423 356 11429 356-11432 356 11435 356 11438 375 11796 375 11826 375-11847 380 11973
		380-11979 •385 12091 •385 12107 •443 14118 •443 14119 •443 14121 443 14122
CONTF	002216	#139 5073 •184 7076 •190 7203 •192-7233 •194 7265 •196 7302 •198 7336 •200-7376 •202 7419
		•204-7459 206 7495 •208 7591
CONTRL	• 177746	#111-4199 145 5314 154 5505 •154 5515 •154 5516 154 5517 154 5519 •326 10638 •326 10639
		•326 10646 365-11590 •367 11646 410 12778 •414 12935 419 13073 •419 13133 495 15893
CONTS	061456	468 14939 #470 15027
CONTS1	061034	468 14924 #468 14929 470-15034
CONTS2	061460	#470-15029 470-15030 470-15036
CONTS3	053646	#419-13109 419-13110 419-13116
CONTT	061402	447-14218 463-14667 468 14934 #470 15002
COUNT	002342	#139-5116 •292-9829 292-9832 •292 9832 •292 9833 292 9834 •292 9841 •292-9848 293-9853
		•293-9853 •293-9854 293-9855 •293-9874
CPERRF	056560	•145-5354 •145-5356 447 14220 450-14329 #451 14399
CPSAVE	056556	•447-14222 447-14223 •450-14331 450-14332 #451-14398 455-14491
CPUBIT	002104	#139-5034 •151-5449 164-6159 166-6261 166-6282 181-6958 181-6992 320-10465 348-11129
		356 11392 375-11796
CPUERR	• 177766	#111-4205 •145-5357 •151-5461 •161-5891 •164-6160 •166-6220 •367-11658 •367-11662 •369 11674
		•369-11686 •371-11728 •373-11780 •447-14253 •451-14377 •461-14659 495-15885 495-15893
CR	• 000015	#111-4133 419-13122
CRLF	• 000200	#111-4134 419-13051 499-15973 503-16040 503-16041 503-16042 503-16043 503-16045 503-16047
		503-16048 503-16049 503-16050 503-16051 503-16052 503-16058 503-16062 503-16063 503-16064
		503-16065 503-16066 503-16067 503-16068 503-16069 503-16070 503-16071 503-16072 503-16073
		503-16074 503-16075 503-16076 503-16077 503-16078 503-16083 503-16084 503-16085 503-16086
		503-16088 503-16089 503-16090 503-16091 503-16092 503-16093 503-16094 503-16095 503-16096
		503-16097 503-16098 503-16099 503-16102 503-16103 503-16104 503-16105 503-16106 503-16107
		503-16108 503-16109 503-16110 503-16110 503-16111 503-16113 503-16114 503-16115 503-16116
		503-16117 503-16119 503-16120 503-16121 503-16122 503-16126 503-16128 503-16130 503-16131
		503-16132 503-16133 503-16134 503-16135 503-16135 503-16136 503-16137 503-16144 503-16152
		503-16153 503-16154 503-16166 503-16167 503-16168 503-16168 503-16170 503-16171 503-16172
		503-16177 503-16178 503-16179 503-16180 503-16181 503-16182 503-16183 503-16184 503-16185
		503-16188 503-16191 503-16192 503-16194 503-16195 503-16196
CSR	002144	#139-5050 •158-5722 •158-5731 163-5981 163-6057 •171-6389 •171-6405 •171-6415 •171-6425
		210-7642 210-7667 266-9083 266-9083 •312-10247 •312-10262 •328-10664 328-10665 •328-10673
		329-10699 329-10699 •329-10710 •331-10715 •331-10719 •331-10723 •331-10727 •331-10731 •331-10735
		•331-10739 •331-10740 •331-10744 •331-10745 333-10759 333-10779 335-10794 335-10813 •337-10822
		•337-10826 •337-10831 •337-10836 340-10884 •369-11677 •406-12709 406-12715 411-12823 •413-12890
		445-14177 463-14670 463-14679 •463-14685 495 15890 495-15891 495-15892 495-15907 495 15908
CSRADD	• 172100	#113-4410 157-5643 •159-5763 •159-5766 •159-5768 •159-5771 159-5773 •160-5811 160-5821
		•160-5838 160-5846 •163-5968 •163-5971 •163-6061 163-6063 •163-6068 163-6069 •163-6070
		•328-10665 328-10673 329-10679 •340-10893 340-10894 •340-10895
CSRCAS	017420	208-7566 #212-7692
CSRFBA	002226	#139-5078 •208-7515 208-7540 •208 7584
CSRFIR	002222	#139-5076 •208-7540 208-7546 208-7548 •208-7581 208-7581
CSRHOL	002500	#139-5152 •208-7525 •208 7535 208-7537
CSRINC	002302	#139 5097

```
SYMBOL  VALUE          REFERENCE
CSRINF   002434        #139 5148  •157 5649  •157 5650  •157 5656  •157 5661  •158 5679   158 5680   158 5682  •158 5688
                         158 5690   158 5692   158 5694   158-5696   158 5698   158 5700  •158 5702  •158 5727  •158 5736
                         159 5761   159 5774   163 5966   163 5986   163 6008   163 6091   163 6095   179 6804  •179 6839
                        •179 6859  •179 6860   179 6876   179-6878
CSRINT   002232        #139-5080  •208 7517  •208 7531   208-7583
CSRLAS   002224        #139-5077  •208 7546  •208 7547   208-7573
CSRLBA   002230        #139 5079  •208 7516  •208-7529   208 7581
CSRLOC   002304        #139 5098  •208 7519  •208 7530   208-7587
CSRNC    002146        #139 5051  •159 5749  •159-5755  •160 5805  •160 5818  •160 5832  •160 5843  •163 5947  •163 5954
                         163-6060  •169-6354   169 6355   169 6356  •170 6366  •208 7537  •208 7538  •227-8120  •244 8652
                         244-8653   328-10654  328 10660  328 10662  328-10672  329 10680  329 10686  329 10688 •333-10755
                        •333 10765  333-10765 •335 10790 •335 10800  335 10800 •339 10845 •339-10851  339 10851  340-10892
                         365-11591 •367 11638  375-11823 •375 11830  375 11855 •375 11857 •375-11909 •391 12128 •391 12142
                         410 12782 •411-12819 •411 12826  411-12826 •413 12887 •413-12894  413 12894 •414 12929  463 14670
                        •463 14675 •463 14683  463 14683 •463-14685
CSROUT   041356          331 10716  331-10724  331 10732  331 10741  337 10823  337 10832 #339 10841
CTEST    006142          158 5716  #159-5742
CTLKVE   002142        #139 5049  •208-7564  •208 7565  •214 7742  •214 7743  •216-7797  •216 7798   468 14943
DATARG  • 177754       #111-4202   154-5509
DATBUF   002236        #139-5082  •260-8896  •260 8897   260 8898   260 8899   260 8901   260 8906   260 8910  •260-8912
                        •260-8912  •260-8915  •260-8916   260 8917   260 8918   260-8920   260-8925   260-8929  •260-8931
                        •260-8931  •266-9036  •266-9037   266-9042   266-9043   266 9124  •266-9126  •266 9126  •268-9157
                        •268-9158   268-9161   268-9162   268-9221  •268 9223  •268-9223  •270-9239  •270-9240   270-9245
                         270-9246  •274-9337  •274-9338   274-9343   274 9344  •276-9436  •276-9437   276-9442   276-9443
                        •278-9537  •278-9538   278-9543   278-9544   436-13994  436-13999
DBEMSK   002252        #139-5085  •270-9243  •270-9244   270-9252   270-9254   270-9262   270-9264   272-9284  •272-9286
                        •272-9286  •272-9303   272-9307  •272 9309   272-9310  •272-9315  •272-9316  •274-9341  •274 9342
                         274-9350   274-9352   274-9360   274-9362   274 9392  •274-9394  •274-9394  •274 9411   274-9415
                        •274-9417   274-9418  •274-9423  •274-9424  •276-9440  •276 9441   276-9449   276-9451   276 9459
                         276-9461   276-9491  •276-9493  •276-9493  •276 9509   276 9513  •276 9515   276-9516  •276 9521
                        •276-9522  •278-9541  •278-9542   278-9549   278-9551   278 9559   278-9561   280-9593  •280-9595
                        •280-9595  •280-9611   280-9615  •280 9617   280 9618  •280 9623  •280-9624
DDISP   • 177570       #111-4127   141-5231   151-5456
DEENER  • 104421       #110-4051   155-5576   360 11486  362 11531
DETAIL   057532          455-14482 #461-14630  461-14662
DETFLA   002214        #139-5072   451-14374  455 14481 •461 14631  461 14632  461 14634 •463 14732
DETPSW   002212        #139-5071   495-15889
DETR0    002174        #139-5064  •461 14639  461 14648  495-15889
DETR1    002176        #139-5065   461-14640  495 15889
DETR2    002200        #139-5066   495-15889
DETR3    002202        #139-5067   495-15889
DETR4    002204        #139-5068   495-15889
DETR5    002206        #139-5069   495-15889
DETSP    002210        #139-5070   495-15889
DET1     060222          463-14700 #463 14717
DF11     064726          485-15595  487-15662  487 15667  487-15672  489-15694  489 15699  489-15714 #497-15926
DF13     064737          487 15687 #497 15927
DF14     064747          491 15802 #497-15928
DF15     064756          493 15868 #497-15929
DF2      064604          485 15600  485-15615  485 15620  485-15625  485-15630  487-15647  487-15677  489-15704  489 15709
                         489 15719  489 15724  491 15755  491-15807  493-15826 #497-15918
DF3      064630          487 15637  489 15741  491 15772  491 15777  491 15787  491-15792  491 15797 #497-15919
```

SYMBOL CROSS REFERENCE CREF V01

SYMBOL	VALUE	REFERENCES
DF4	064643	487 15642 #497 15920
DF5	064656	487 15652 487 15682 491 15782 #497 15921
DF6	064671	487 15657 #497 15922
DF7	064704	489 15735 #497 15923
DF8	064717	491 15760 #497 15924
DF9	064721	485-15605 485 15610 #497 15925
DH1	067445	485 15598 #501 16006
DH10	070015	487 15645 #501 16013
DH11	070115	489 15707 #501 16014
DH12	070131	489-15702 #501-16015
DH13	070136	485 15593 487 15660 487 15665 487 15670 489 15692 489 15697 489 15712 #501 16016
DH14	070233	489-15717 #501 16018
DH15	070312	489 15734 #501 16019
DH16	070431	489 15739 491 15758 #501 16021
DH19	070446	#501 16026
DH2	067502	493 15824 #501 16007
DH23	070453	487 15685 #501-16032
DH24	070532	491 15800 #501 16033
DH25	070601	485 15628 491 15753 #501 16034
DH26	070625	487-15675 #501-16035
DH27	070643	493 15866 #501 16036
DH3	067525	485-15603 485-15608 #501 16008
DH5	067561	485 15613 485-15618 485-15623 489 15722 #501 16009
DH6	067640	491 15805 #501 16010
DH7	067705	487-15635 487-15640 487 15650 487-15655 487 15680 491 15770 491-15775 491 15780 491 15785 491 15790 491 15795 #501 16011
DIAGFL	002002	#139 4999 *231-8272 *231-8274 292 9846
DISPLA	002602	#141-5231 *151-5456 *151-5463 414-12927 448 14282
DISPRE *	000174	#113 4406 151-5463 *414-12928 *448-14283
DISPTB	014240	184-7079 #184-7081
DOBACK	014612	184-7090 184-7101 #188 7172
DSWR *	177570	#111-4126 141-5230 151-5455
DT1	064220	485-15599 #495-15879
DT10	064334	487 15646 #495-15889
DT11	064356	489 15708 #495 15890
DT12	064364	489-15703 #495-15891
DT13	064370	485-15594 487 15661 487-15666 487 15671 489-15693 489 15698 489-15713 #495-15892
DT14	064412	*145-5321 *145 5333 *145 5338 489-15718 #495 15893
DT16	064430	489 15735 #495-15894
DT17	064460	489 15740 491-15759 #495-15896
DT20	064466	493 15825 #495-15901
DT22	064476	#495-15905
DT23	064504	487 15686 #495 15906
DT24	064526	491 15801 #495 15907
DT25	064546	485-15629 491 15754 #495 15908
DT26	064556	487 15676 #495 15909
DT27	064564	493 15867 #495 15910
DT3	064232	485 15604 #495 15883
DT4	064242	485 15609 #495 15884
DT5	064252	*145 5332 *145 5337 485 15614 485-15619 485 15624 489 15723 #495-15885
DT6	064270	491 15806 #495 15886
DT7	064304	487 15636 487 15641 487 15651 487 15656 487 15681 491 15771 491 15776 491 15781 491 15786

SYMBOL CROSS REFERENCE

SYMBOL	VALUE	REFERENCES									
		491 15791	491 15796	#495 15887							
DUMMY	002172	#139-5062	495 15887	495 15887	495-15888	495 15888	495 15888	495 15888	495 15892	495 15892	
		495-15894	495-15894	495 15895	495-15895	495 15896	495-15905	495 15906	495 15906	495 15906	
		495 15906	495 15906	495 15907	495 15907	495 15907	495 15907	495 15907	495 15910	495 15910	
		495 15910	495-15910	495 15910							
ECCDIS	• 104470	#110-4094	214 7731	238-8459	375-11853	378 11925	382 11996				
ECCINI	• 104472	#110-4096	164-6132	#164 6166	#170-6381	208-7590	#214 7748	#227 8124	238 8476	238 8483	
		238 8487	360-11482	362-11523	#406-12733						
ECC1DI	• 104471	#110 4095	171-6440	210-7627	210-7650	266-9061	266-9068	266 9145	268 9171	268 9226	
		270 9266	272-9318	274-9365	274-9426	276 9464	276 9524	278 9564	280 9624	300 10012	
		302-10020	302-10030								
ECC1IN	• 104473	#110-4097	210-7677	210-7685	300-9977	302 10023					
EMTVEC	• 000030	#111-4188	#147-5377	#147 5378							
EM11	065265	487 15634	487-15639	487-15649	487 15654	#499 15954					
EM12	065307	487-15644	#499-15955								
EM13	065333	487-15659	#499-15956								
EM14	065365	487-15664	#499-15957								
EM15	065431	487-15669	#499-15958								
EM17	065477	487-15679	#499-15959								
EM19	065537	489-15691	#499-15960								
EM2	064765	485-15597	#499-15944								
EM20	065614	489-15696	#499-15961								
EM21	065676	489-15706	#499-15962								
EM22	065735	489-15711	#499-15963								
EM23	065762	489-15721	#499-15964								
EM24	066011	485-15592	#499-15965								
EM25	066070	489-15733	#499-15966								
EM26	066115	489-15738	#499-15967								
EM27	066166	491-15757	#499-15968								
EM29	066256	491-15774	#499-15972								
EM3	065023	485-15602	#499-15945								
EM30	066340	491-15779	#499-15973								
EM31	066457	491-15784	#499-15975								
EM32	066557	491-15789	#499-15976								
EM33	066664	491-15794	#499-15977								
EM35	066772	491-15769	#499-15978								
EM36	067057	491 15804	#499-15979								
EM4	065055	485-15607	#499-15946								
EM40	067126	493 15823	#499-15985								
EM5	065123	485-15612	#499-15947								
EM50	067200	487-15684	#499-15994								
EM51	067234	491-15799	#499-15995								
EM52	067316	491-15752	#499-15996								
EM53	067343	485-15627	#499-15997								
EM55	067372	487-15674	#499 15998								
EM56	067413	493 15865	#499-15999								
EM6	065200	485-15617	#499-15948								
EM7	065225	485 15622	#499-15949								
ENASBE	• 104506	#110-4108	164-6164	170-6379	214-7746	227-8122	406-12731				
ENA1SB	• 104507	#110-4109	300 9981	300 10014	302 10025						
END	075674	#503-16204	503-16216	503 16217							
ENERGI	• 104420	#110 4050	155 5615	186 7154							

SYMBOL CROSS REFERENCE CREF V01

SYMBOL VALUE REFERENCES

ENEXBK 044742 356-11407 356-11433 356 11439 #356 11441
ERRADD 002432 #139-5147 210-7645 210 7670 *340 10914
ERRGEN . 104512 #110-4112 210-7644 210 7669 266 9089 266 9115 272-9281 312 10258
ERRMAX 002526 #141 5202 443-14122
ERROR . 104000 #111-4121 158-5693 158 5701 158-5726 158-5735 164 6154 164 6158 171 6449 183 7060
266-9084 268-9189 270-9277 300 9986 302 10044 302 10056 312 10253 312 10268 322 10571
324-10596 324-10600 324-10603 367 11657 369 11673 369 11682 406-12728 439 14069 #439 14071
439 14081 #439-14083 441-14092 443-14141 443 14144 443 14147 443 14150 445 14161 445 14172
445 14178 #445-14180 445-14189 445-14196 447-14225 461-14650 461-14656
ERRPC 002016 #139-5008 *450-14319 *450-14320 450 14323 *450-14342 *450 14343 450-14348 453-14416 455 14491
495-15879 495 15883 495-15884 495 15885 495-15886 495 15887 495 15890 495-15892 495 15894
495-15896 495-15901 495-15905 495 15906 495 15907 495 15908 495-15909 495 15910
ERRPSW 002026 #139 5012 *450-14322 *450-14345 461-14647
ERRSP 002022 #139-5010 *450-14321 *450-14344 461-14646
ERRVEC . 000004 #111 4181 *147-5387 *147-5388 447-14245 *447-14246 *447 14248 *447 14254
EUFLAG 002130 #139 5044 219-7901 340-10874 340-10889 *356-11377 *356 11410
EVEN 002336 #139-5114 *292-9817 292 9818 *293-9880 293-9880
EXBANK 044302 167-6300 169-6340 179 6811 188-7175 190-7191 192-7221 194 7251 196 7288 198-7327
200-7365 202-7410 204-7448 236-8391 236-8420 238 8462 238 8481 348-11104 350-11212
350-11216 #356-11354 375-11814 375-11905 378 11942 378 11956 382-12012 382-12026
EXCMD3 046372 371-11717 371-11720 371-11724 #371-11726
EXCMD4 046712 373-11760 373-11764 #373-11778
EXIT 045136 #362-11504 447-14236 451-14380 471 15091
EXIT2 045142 #362-11506
FASTCI . 177640 #111-4235 166-6215 166-6249 248-8706 329 10696 364-11558
FATALS 002062 #139-5025 *164-6154 *164-6158 *300-9986 *322 10571 *324-10596 *324-10600 *324-10603 451 14378
453-14433
FCMD10 050646 365-11612 #384-12031
FCMD11 050674 365-11613 #384-12041
FCMD12 050716 365-11614 #384-12046
FCMD13 050736 365-11615 #384-12052
FCMD14 050760 365-11616 #384-12059
FCMD15 050776 365-11621 #385-12078
FCMD16 051062 365-11622 #385-12098
FCMD17 051124 365-11623 #387-12112
FCMD18 051140 365-11624 #389-12118
FIELDS 045406 #365-11579 468-14931
FINT 006440 159-5790 #160-5797
FIRST . 060000 #113-4413 166-6204 166-6237 166-6267 169-6339 208-7515 216-7790 216-7791 217-7839
217-7852 217-7876 219-7905 219-7941 219-7959 221-7987 227-8075 228-8179 228-8180
230-8221 232-8287 234-8336 236-8395 236-8424 238-8466 240-8500 240-8505 240-8506
242-8532 242-8562 242-8569 244-8594 244-8597 244-8598 244-8599 244-8600 244-8617
282-9641 292-9830 292-9849 293-9885 293-9887 293-9888 312-10246 314-10313 *362-11543
*362-11544 *362-11545 371-11705 373-11746 375-11845 375-11846 415-12957 415-12958 415 12959
*415-12960 *415-12961 415-12962 415-12964 *417-12998 *417-12999 417-13004 458-14569
FLIPLO 002560 #141-5215 *147-5362 219-7931 *313-10285 *313-10286 313-10287 313-10289 313-10291
FLIPWA 036402 219-7904 #313-10283
FLUSH 014332 #184-7106
FSCMD0 045604 #365-11602 #367-11629
FSCMD1 045706 365-11603 #367-11651
FSCMD2 046016 365-11604 #369-11666
FSCMD3 046164 365 11605 #371 11690

SYMBOL CROSS REFERENCE CREF V01

SYMBOL	VALUE	REFERENCES
FSCMD4	046440	365-11606 #373 11732
FSCMD5	046760	365-11607 #375 11784
FSCMD6	047650	365-11608 #376-11913
FSCMD7	047656	365 11609 #378 11919
FSCMD8	050150	365 11610 #380-11960
FSCMD9	050354	365-11611 #382-11990
FSINFL	002414	#139-5136 #186 7117
FSPAT	047456	375-11863 #375-11867
FSSTAC	002270	#139-5092 #367-11653 367-11661 #369 11668 369 11685 #375-11786 375 11898 #378-11921 378-11951 #382-11992 382-12021
FS1	045472	#365-11594 365-11600 365 11626
FS7FLA	002420	#139-5138 236-8414
FULLRE	002514	#139-5157 #238-8449 #238-8477 #238 8488 348-11143
GBLENG	• 000076	244-8594 244-8597 #310-10228
GETDAT	051534	#408-12747 439-14065 439-14077
GETDA1	051632	#408-12749 408-12757 #408-12760
GETDIS	055726	248-8682 250-8715 447-14267 #448-14271 450-14308
GOOD	002042	#139-5017 #158-5723 #158-5732 #406-12700 #406-12702 #406-12720 #408-12738 #436-13961 #436-13967 #436-13972 #436-13979 #436-13982 #436-13989 #436-13994 #436-13999 #436-14004 #436-14009 #438 14015 #438-14020 #438-14025 #438-14030 #438-14035 #438-14040 #438-14045 #438-14050 #438-14055 #438 14060 441-14105 #443-14130 #443-14132 #445-14159 #445-14169 #445-14188 #445-14193 495-15879 495 15887 495-15894 495-15901 495-15906 495-15908
GOOD2	002044	#139-5018 #441-14095 #441-14099 495-15894
GOOD3	002046	#139-5019 #441-14096 #441-14100 495-15894
GTSWR	• 104407	#110-4037 155-5599
HEADER	002554	#141-5213 #147-5363 #184-7096 #184-7098 #188-7177 #188-7180 #206-7489 #206-7504 #208-7560 #266-9098 #266-9101 #266-9111 #266-9114 #268-9187 #268-9190 #268-9201 #268-9204 #270-9275 #270-9278 #302-10045 #302-10057 #375-11839 #378-11939 #382-12009 #443-14135 #443-14152 #445-14160 #445-14162 #445-14171 #445-14173 #445-14195 #445-14197 #451-14390 453-14431 453-14443 #455-14479 461-14636 #461-14637 461-14652 #461-14653 #461-14660
HIPAT	045004	202-7406 204-7444 #358-11464
HT	• 000011	#111-4131 419-13048
I	002424	#139-5144 #149-5428 #149-5434 149-5434 #163-5962 #163-5964 163-6004 163-6012 163-6016 163-6043 163-6047 163-6095 163-6100 163-6102 163-6106 #181-6983 #181-6989 181-7000 #181-7000 181-7022 #208-7559 208-7562 #208-7568 208-7568 #236-8389 236-8399 236-8405 #236-8413 236-8413 #236-8418 236-8428 236-8434 #236-8442 236-8442
IBSAVE	056554	#450-14302 450-14327 #450-14335 #450-14338 451-14393 #451-14397
ILLCSR	013260	179-6880 179-6897 #179-6940
IMPTES	012224	169-6360 170-6368 #171-6386
INCBNK	045014	190-7204 192-7234 194-7266 196-7303 198-7337 200-7377 202-7420 204-7460 #358-11468
INCPAT	044770	190-7200 194-7262 #358-11456
INCRPT	044770	198-7340 200-7380 #358-11455
INHBAN	002512	#139-5156 #348-11137 348-11145
INHECC	002510	#139-5155 208-7513 328-10655 348-11135 #348-11136 348-11144 #348-11152 #348-11160 #350-11220
INTFLA	002134	#139-5046 167-6302 169-6345 170-6362 179-6843 208-7526 227-8077 228-8174 340-10903 #356-11380 #356-11434
INT64K	002136	#139-5047 167-6303 340-10906 346-11041 #356-11380 #356-11437
INVALI	• 104511	#110-4111 208-7542 214-7740 216-7795 236-8393 373-11752 375-11813 378-11944 382-12014
IOTVEC	• 000020	#111-4186 #147-5375 #147-5376
JMPRL1	043302	348-11171 #348-11181
KAMIKA	002004	#139-5000 246-8673 365-11591 #365-11593 #367-11636 #384-12043 #384-12048
KAMITE	026432	231-8260 231-8268 232-8278 240-8493 242-8522 244-8586 #246-8672

SYMBOL CROSS REFERENCE CREF V01

SYMBOL	VALUE	REFERENCES
KDIAG	• 000010	#292-9816 292-9832 293-9853 293-9879
KDPAR0	• 172360	#111-4326 219-7923 219-7925 219-7949 219-7950 219 7969 219-7970 232 8305 232 8308 234-8360 234-8363 234-8368
KDPAR6	• 172374	#111-4332 *219-7926 *232 8309
KDPAR7	• 172376	#111-4333 *219-7951 *219-7971 *234-8361
KERNEL	• 104417	#110-4048 166-6216 166-6229 166-6250 166 6273 171-6453 210 7676 210 7684 240 8501 242-8557 244-8595 248-8708 250 8745 348-11169 350 11228 352-11287 352 11303 362 11546 371-11715 371-11726 373-11756 373-11774 373-11778 406-12697 406-12713 406 12724 408 12743 408-12755 411-12808 411-12814 414-12911 414-12915 415-12974 417-13008
KERSTK	• 002000	#111-4118
KFLAG	002502	#139-5153 179-6854 223-8037 266-9081 268-9181 274-9329 *356-11377 *356-11416
KIPAR0	• 172340	#111-4316 346-11001 346-11077 354-11311 354-11331 415-12963 417-12981
KIPAR4	• 172350	#111-4320 *159-5744 *160-5800 *160-5815 *160-5855 *160 5859 *161-5876 *161-5885 *163-5942 *163-5950 163-6028 163-6078 *163-6117 163-6118 163-6120 354-11345
KIPAR5	• 172352	#111 4321 *163-5943 *163-5951 163-6081 *163-6098 *163-6104 *163-6110 *163-6118
KIPAR6	• 172354	#111-4322
KIPDR0	• 172300	#111-4296 346-11079 354-11313 410-12797 413-12873 413-12897 417-12982
KMAP	• 104422	#110-4052 155-5610
KPFLAG	002112	#139-5037 *356-11377 *356-11387
KSIZE	002350	#139-5119 *181-6953 *181-6964 181-7006 181 7008
KSTACK	002536	#141-5206 144-5279 163-5952 186-7150 463-14690
LAST	• 157776	#113-4414 208-7516 217-7874 227-8093 227-8094 227-8102 230-8219 232-8288 240-8507 242-8533 242-8561 244-8601 282-9643 292-9831 293-9852 293-9889 312-10270 371-11708 373-11748
LASTBA	002530	#141-5203 *144-5289 *145-5336 163-5935 163-5937 166-6191 167-6314 170-6375 179-6870 *179-6915 181-6954 188-7182 232-8299 236-8412 236-8419 238-8470 316-10342 316-10351 346-11058 346-11065 348-11116 348-11123 358-11471 371-11703 373-11745 375-11791 378-11947 380-11984 382-12017 385-12103
LASTBL	002532	#141-5204 *163-5933 *163-5940 163-6120
LASTER	002014	#139-5007 *186-7123
LF	• 000012	#111-4132 419-13126
LINK1	002474	#139-5150 *166-6184 *166-6188 166-6203 166-6242 *236-8383 *236-8387 236-8403 236-8432 *238-8455 *238-8457 238-8467 *240-8507 *240-8515 *242-8531 *242-8544 *244-8601 *244-8609 306-10129 308-10176 310-10219
LINK2	002476	#139-5151 *166-6185 *166-6189 166-6228 *242-8534 *242-8545 242-8563 242-8570
LKS	• 177546	#111-4128
LOADBA	002404	#139-5132 *348-11107 350-11206 350-11211
LOADCS	• 104425	#110-4057 312-10248 312-10263 331-10720 331-10728 331-10736 331-10746 337-10827 337-10837 339-10848 369-11678 413-12891
LOADHO	002540	#141-5207 186-7155 348-11108 350-11207 350-11215 362-11529
LOOP	014212	#184-7075 186-7169
LOWMAP	043706	348-11179 350-11238 #350-11248 413-12870
LSIZE	002352	#139-5120 *181-6953 *181-6968 181-7010 181-7012 183-7059 495-15886
MAINT	• 177750	#111-4200 145-5316 154-5507
MAPHO	• 170202	#113-4341 *348-11178 *350-11237 411-12841 *413-12869
MAPLO	• 170200	#113-4340 *348-11176 *350-11237 350-11250 411-12841 *413-12869
MAPL1	• 170204	#113-4342 350-11251
MAPPER	042072	155-5614 166-6200 170-6376 171-6387 208-7541 240-8498 242-8527 244-8591 248-8681 250-8714 #346-11000 362-11541 371-11712 373-11751 375-11812 375-11849 375-11904 378-11955 382-12025 415-12955 417-12996
MBERR	013136	*179-6875 *179-6891 179-6895 *179-6899 *179-6904 #179-6917
MEMDON	014260	184-7080 #184-7090

```
SYMBOL CROSS REFERENCE
SYMBOL   VALUE      REFERENCE
MFPT     * 000007   #111 4135    145 5342
MJPAT      020044    149 5419    216 7794    216 7794    216 7799   #216 7807
MJTEST     017740    206 7502   #216 7784
MKCONT     016506    206 7497   #208 7508
MKCSRT     017430    149 5405   #212 7698
MKFLAG     002116   #139 5039    169 6343    179 6840    206 7492    238 8463   *356 11377  *356 11413   358 11451
MKLOOP     016470   #208 7537    208 7589
MKPAT      017660    149 5412    214 7739    214 7739    214 7744   #214 7756
MKTEST     017520    188 7178    206 7500   #214 7726
MMRC     * 177572   #111 4209   *326 10628  *326 10632  *348-11181  *348 11190  *350 11229  *350 11232   411 12829  *413 12882
                     415-12973  *417 12992   495 15885   495 15893
MMR1     * 177574   #111 4210    411 12829  *413 12882   415-12972  *417 12991   495-15885   495-15893
MMR2     * 177576   #111 4211    411 12829  *413 12882   415 12971  *417 12990   495 15885   495-15893
MMR3     * 172516   #111 4212    145-5323   *145 5324   *145 5325    145 5326   *186 7153   *348-11172  *348 11180  *350 11236
                    *350 11239  *360 11489  *362 11534   411 12832  *413 12881   415 12970  *417-12989   495 15885   495 15893
MMTRAP     040144    147 5389   #324 10599
MMVEC    * 000250   #111-4196   *147-5389   *147 5390
MPT      * ******    113-4416    137-4979    137 4987    139-5139    139 5158    141-5221    144 5273    144 5280    144-5302
                     147 5370    147-5391    154-5529    155 5594    155-5602    155-5607    155-5611    172-6456    173-6499
                     179-6807    179 6813    179-6905    183-7062    184-7102    214-7727    216-7785    274 9331    308 10140
                     318-10417   324-10605   346-11087   356-11389   356 11395   362 11520   362-11535   365 11617   384 12065
                     391-12144   410-12768   413-12855   415-12942   419-13076   419 13119   420 13138   447 14226   448 14279
                     450 14352   455-14467   458-14587   464-14736   468 14915   470-15005   470-15021   478-15432   489-15725
                     489-15742   491-15761   493-15810   493 15827   493 15869   495 15880   495 15897   495 15902   495 15911
                     497-15930   499-15941   499-15950   499-15969   499-15980   499 15986   499 16000   501 16022   501 16027
                     503 16079   503-16138   503-16145   503 16159   503 16173   503-16209
MSEEDH     002550   #141 5211   *147-5364    147-5366
MSEEDL     002552   #141-5212   *147-5365    147 5367
MSGA12     075377   *179-6946   #503-16189
MSGA34     072773    208-7577   #503-16099
MSGB34     073037    208-7579   #503-16100
MSG000     070717    154 5497   #503-16040
MSG001     070764    316 10339  #503-16041
MSG002     071046    316 10340  #503-16042
MSG003     071123    316-10341  #503 16043
MSG004     071230    316-10349   316-10369  #503 16045
MSG005     071336    318-10395  #503-16047
MSG006     071350    479 15470  #503-16048
MSG007     071405    320-10434  #503-16049
MSG008     071417    318-10409  #503-16050
MSG009     071431    320-10464  #503-16051
MSG010     071443    320 10524  #503-16052
MSG011     071455    316-10355   316-10368  #503-16053
MSG012     071543    316-10358   316-10370  #503-16054
MSG013     071640    318-10398  #503-16055
MSG014     071642    316-10367   318-10400   320-10534   450-14313   457-14515   457-14519   458-14585  #503-16056
MSG015     071644   *318-10423   318 10424  *320-10440  *320 10442  *320-10452   320-10453  *320 10474  *320-10480  *320 10482
                    *320-10486  *320-10488  *320 10490   320-10491  *320 10503  *320-10513   320 10514  *320 10527  *320 10531
                     320-10532  #503 16057
MSG016     071646    320-10502  #503-16058
MSG017     071660    316-10554   316 10357  #503 16059
MSGCLR     071671    455 14474   458 14583   463 14680   463 14695   463 14710   463 14725  #503-16060
```

SYMBOL CROSS REFERENCE CREF V01

SYMBOL	VALUE	REFERENCES
MSG019	071674	458 14553 #503 16061
MSG020	071700	365-11581 #503 16062
MSG021	071721	365 11599 #503 16063
MSG022	072511	391 12131 #503 16088
MSG023	072533	369-11675 #503 16089
MSG025	072547	367-11660 369 11684 #503 16090
MSG026	072573	365 11594 #503 16091
MSG027	072605	369-11671 #503-16092
MSG028	072622	369-11680 #503-16093
MSG029	072636	371-11693 #503-16094
MSG030	072656	375-11788 385-12083 #503 16095
MSG031	072675	371-11694 373-11736 #503-16096
MSG032	072735	371-11719 373-11759 #503-16097
MSG033	072754	371-11725 373-11763 #503-16098
MSG035	073102	186-7128 #503-16101
MSG036	073105	373-11735 #503-16102
MSG037	073124	373-11766 #503-16103
MSG038	073143	373-11775 #503-16104
MSG039	073161	373-11768 #503-16105
MSG040	073203	375-11787 #503-16106
MSG041	073237	375-11797 #503-16107
MSG042	073264	375-11801 #503-16108
MSG043	073305	375-11806 #503-16109
MSG046	073332	375-11819 378-11931 382 12002 #503 16110
MSG047	073365	#503-16111
MSG048	073404	365-11584 #503-16112
MSG049	073444	375 11816 #503-16113
MSG050	073476	#503-16114
MSG051	073604	414-12931 #503 16116
MSG052	073624	#503-16117
MSG053	073655	#503-16118
MSG054	073673	#503-16119
MSG055	073743	378-11922 #503-16120
MSG056	073764	378-11929 382-12000 #503-16121
MSG058	074017	463-14671 #503-16122
MSG061	074041	458-14562 #503-16123
MSG062	074050	472-15191 472-15250 #503-16124
MSG063	074070	472-15192 #503-16125
MSG064	074101	472-15193 472-15252 #503-16126
MSG065	074111	472-15251 #503-16127
MSG066	074123	451-14383 #503-16128
MSG067	074206	453-14435 #503-16129
MSG070	074215	181 7023 #503-16130
MSG071	074246	181-7005 #503-16131
MSG072	074264	181-7009 #503-16132
MSG073	074302	382-11993 #503-16133
MSG075	074320	348-11118 348-11156 #503-16134
MSG076	074352	380-11975 #503-16135
MSG077	074373	186-7126 #503-16136
MSG079	074407	380-11967 #503-16137
MSG085	074433	384-12033 #503-16144
MSG088	074460	463 14689 #503 16152

SYMBOL CROSS REFERENCE CREF V01

SYMBOL	VALUE	REFERENCES							
MSG089	074476	463 14705	#503 16153						
MSG090	074520	463-14720	#503 16154						
MSG091	074534	463 14714	463 14729	#503 16155					
MSG092	074546	470 15015	#503 16156						
MSG093	074562	470-15017	#503-16157						
MSG095	074570	470 15019	#503-16158						
MSG101	074577	384 12042	#503-16166						
MSG102	074627	384-12047	#503 16167						
MSG103	074656	367 11630	#503-16168						
MSG104	074700	455 14484	#503-16169						
MSG105	074702	385-12079	#503-16170						
MSG106	075006	384-12053	#503-16171						
MSG107	075024	384 12060	#503-16172						
MSG110	075101	385 12094	#503 16177						
MSG111	075145	385-12099	#503-16178						
MSG112	075177	181-7013	#503-16179						
MSG113	075214	181-7017	#503-16180						
MSG114	075231	181-7021	#503-16181						
MSG116	075255	154-5512	#503-16182						
MSG117	075267	154-5510	#503-16183						
MSG118	075301	154-5514	#503-16184						
MSG119	075313	154-5526	#503-16185						
MSG120	075322	154-5527	#503-16186						
MSG121	075343	154-5521	#503-16187						
MSG122	075363	179-6947	#503-16188						
MSG123	075431	348-11148	#503-16191						
MSG124	075502	179-6864	#503-16192						
MSG125	075545	380-11965	#503-16193						
MSG126	075567	160-5863	#503-16194						
MSG127	075634	387-12113	#503-16195						
MSG128	075653	389-12119	#503-16196						
MSIZE	002354	#139-5121	*181-6953	*181-6970	181-7014	181-7016	183-7059	495-15886	
MTA030	024732	#238-8450	238-8482						
MTEST	016212	190-7198	192-7228	194-7260	196 7297	198 7334	200-7374	202-7417	204 7457 #206 7488
MTLA11	030050	266-9031	#266-9036	266-9131					
MTLB11	030062	#266-9039	266-9127						
MTLC11	030074	#266-9042	266-9121						
MTLD11	030170	#266-9059	266-9143						
MTLQ20	022054	#227-8077	227-8144						
MTPA03	027146	219-7913	219-7919	#254-8798					
MTPA04	027304	219-7945	219-7948	219-7966	#258-8857	258-8877			
MTPA20	034110	228-8158	228-8159	228-8162	#284-9672	284-9682			
MTPA21	034274	230-8230	230-8244	#288-9741					
MTPA24	035120	232-8303	#296-9906	298-9965					
MTPA25	035530	300-9976	300-10011	302-10019	#302-10030				
MTPA26	035660	234-8346	234-8347	234-8359	#304-10062	304-10105			
MTPB03	027206	219-7916	219-7922	#254-8821	256-8854				
MTPB04	027340	219-7949	*219-7964	*219-7966	219-7969	258-8861	#258-8870	258-8883	
MTPB20	034140	228-8167	228-8168	228-8171	#284-9686	284-9696			
MTPB21	034324	230-8232	230-8247	#288-9758					
MTPB24	035160	232-8297	232-8304	296-9947	#298-9950				
MTPB25	035552	300 10013	302-10021	#302 10037					

SYMBOL CROSS REFERENCE CREF V01

SYMBOL	VALUE	REFERENCES								
MTP026	035674	234-8350	234 8351	234 8367	#304 10069					
MTPC03	027246	219-7923	254 8823	#256-8839	256 8853					
MTPC20	034170	228 8183	228 8184	228 8187	#284 9700	284 9710				
MTPC21	034360	230 8235	230 8251	#288 9775						
MTPC24	035174	232-8305	298-9956	#298 9958						
MTPC25	035612	300-10015	302 10024	302-10026	#302 10049					
MTPC26	035730	234 8360	304 10063	304-10072	#304 10087					
MTP003	027264	219-7924	256-8845	#256 8847						
MTP020	034220	228-8192	228-8193	228-8196	#286-9715	286-9725				
MTP021	034414	230-8237	230-8254	#290-9793						
MTP025	035456	300-9988	300-9990	300 9993	300-9995	#300 10011				
MTP026	035750	#234-8346	#234-8350	234-8362	#304-10100					
MTPE20	034250	228-8201	228-8202	228-8205	#286-9729	286 9737				
MTPE25	035500	300-9999	300-10001	300-10004	300-10006	#302 10019				
MTP000	027036	217-7844	217-7847	#252-8761	252-8763	#314-10317	314-10318	#314-10320	314 10323	
MTP001	027062	217-7866	217-7869	#252-8772						
MTP002	027114	217-7892	217-7895	#252-8784						
MTP005	027360	219-7963	219-7964	219 7968	#258-8879					
MTP006	027414	221-7980	#260-8893							
MTP007	027614	221-7990	#262-8936							
MTP010	027714	221-7997	#264-8976							
MTP011	030022	223-8007	#266-9012							
MTP012	030620	223-8021	#268-9153							
MTP013	031206	223-8029	#270-9235							
MTP014	031722	223-8039	#274-9325							
MTP015	032504	223 8049	#276-9433							
MTP016	033250	225-8057	#278-9532							
MTP017	034032	225-8062	#282-9634							
MTP020	034110	#284 9667								
MTP022	034444	231-8263	231-8271	#292-9811						
MTP025	035212	232 8324	#300-9968							
MTP030	035766	238-8454	238-8458	#306-10108						
MTP031	035776	240-8500	#306-10114	306-10136						
MTP032	036054	242 8556	242-8575	#308-10165						
MTP033	036106	244-8594	#310-10185	310-10228						
MTP034	036204	236-8385	236-8402	236-8408	236 8471	236-8437	244-8625	244 8631	244 8637	#311-10231
		311-10233								
MTP035	036230	244-8658	#312-10243							
MTST3	011562	166-6184	166-6185	166-6187	166-6213	166 6247	#166-6285	166-6286		
MTV020	022430	#227-8127	227-8131	#227-8139	227 8139	#227 8147				
MT0000	020124	#217-7837	375-11867							
MT0001	020204	214-7760	216-7812	#217-7850	375-11868					
MT0002	020324	214-7761	216-7813	#217-7872	375-11869					
MT0003	020464	216-7814	#219-7900	375-11870						
MT0004	020716	214-7762	216-7815	#219-7936	375-11871					
MT0005	021040	214-7763	216-7816	#219-7954	375-11872					
MT0006	021174	212-7698	216-7809	#221-7976	375-11873					
MT0007	021230	214-7759	216-7811	#221-7983	375-11874					
MT0010	021272	212-7699	#221-7993	375-11875						
MT0011	021326	212-7701	#223-8002	375-11876						
MT0012	021374	212-7702	#223-8010	375-11877						
MT0013	021470	212 7703	#223-8024	375 11878						

SYMBOL CROSS REFERENCE CREF V01

SYMBOL	VALUE	REFERENCES								
MT0014	021544	212-7704	#223-8033	375-11879						
MT0015	021622	212-7705	#225-8044	375-11880						
MT0016	021670	212-7706	#225-8052	375-11881						
MT0017	021736	214-7758	216-7810	#225-8060	375-11882					
MT0020	021760	214-7765	#227-8067	375-11883						
MT0021	023050	214-7764	216-7817	#230-8212	375-11884					
MT0022	023322	214-7766	216-7819	#231-8259	375-11885	382-12015				
MT0023	023354	216-7820	#231-8267	375-11886						
MT0024	023420	214-7768	216-7822	#232-8277	375-11887					
MT0025	023664	212-7700	#232-8314	375-11888						
MT0026	023732	214-7767	216-7821	#234-8329	375-11889					
MT0027	024234	184-7097	#236-8376	375-11890						
MT0030	024720	184-7107	#238-8447	360-11485	362-11526	375-11891				
MT0031	025222	214-7769	216-7823	#240-8492	375-11892	378-11945				
MT0032	025412	214-7770	216-7824	#242-8521	375-11893					
MT0033	025744	214-7771	216-7825	#244-8585	375-11894					
MT0034	026132	214-7739	214-7757	214-7772	216-7794	216-7808	216-7826	#244-8615	375-11895	
MT0035	026304	216-7818	#244-8645	375-11896						
MT020Y	022616	228-8154	228-8170	#228-8174						
MT020Z	022432	227-8136	#228-8149							
MT0999	026416	149-5431	212-7707	212-7708	212-7709	212-7710	212-7711	212-7712	212-7713	212-7714
		212-7715	212-7716	212-7717	212-7718	212-7719	212-7720	212-7721	214-7739	214-7774
		214-7775	214-7776	214-7777	214-7778	214-7779	214-7780	214-7781	216-7794	216-7828
		216-7829	216-7830	216-7831	216-7832	#246-8667				
MT1	016266	206-7493	#206-7502							
MT2	016272	206-7501	#206-7503							
MUT	002106	#139-5035	*184-7096	*184-7099	*188-7177	*188-7179	*206-7490	*206-7503	*375-11839	*378-11939
		*382-12009	455-14477	461-14636	*461-14638	461-14652	*461-14654	*461-14660		
NC	053706	419-13104	419-13108	419-13114	#419-13118					
NEMCNT	002066	#139-5027	*164-6142	164-6155	*166-6199	166-6223	166-6235	166-6257	166-6281	*324-10584
		324-10585	*324-10589	495-15884						
NEWBAN	002272	#139-5093	238-8480	*348-11164	348-11173	348-11183	*350-11224	354-11319	354-11332	
NEWKER	044202	348-11182	350-11230	#354-11329						
NEWLOA	044250	348-11110	350-11209	#354-11342						
NOCH	061016	468-14921	#468-14925							
NOERRO	002402	#139-5131	*367-11656	*369-11672	*369-11681	450-14303	450-14340	451-14363	*451-14392	453-14429
		453-14441	455-14481	*461-14649	*461-14655					
NOFSMO	002400	#139-5130	*236-8388	*236-8415	*236-8443	*238-8460	*238-8477	*238-8488	365-11583	
NONEM	002076	#139-5031	*164-6143	*166-6180	*166-6266	*166-6275	324-10582			
NONEXI	040066	164-6144	166-6181	#324-10582						
NOOJ	036744	316-10343	#316-10346							
NOPAR	002074	#139-5030	*144-5294	*144-5300	*157-5644	*161-5875	*164-6141	*166-6179	*166-6221	*208-7558
		*214-7735	*214-7750	*216-7788	*216-7800	*223-8030	*227-8072	*227-8145	*238-8451	*240-8497
		*242-8526	*244-8590	*300-9973	*300-10008	*312-10244	*312-10272	322-10555	322-10559	322-10566
		371-11691	*371-11692	*371-11727	373-11733	*373-11734	*373-11779	*375-11859	*375-11864	378-11920
		*378-11954	382-11991	*382-12024						
NORES	003642	144-5277	#144-5279							
NOSCOP	002412	#139-5135	*230-8213	*230-8256	*232-8280	*232-8317	*238-8460	*238-8477	*238-8488	*240-8495
		*240-8517	*242-8524	*242-8579	*244-8588	*244-8612	447-14238	451-14368		
NOSUPE	002430	#139-5146	*145-5331	*145-5350	155-5577	217-7854	217-7878	240-8508	242-8535	244-8602
		250-8737	340-10868	346-11004	346-11021	346-11046	346-11067	410-12789	411-12810	411-12830
		413-12879	414-12906	415-12968	417-12987	463-14699				

SYMBOL CROSS REFERENCE
CREF V01

SYMBOL	VALUE	REFERENCES
NOTAB	002344	#139-5117 320 10454 320 10492 320-10515 *457 14514 *457 14517 *457 14528 *457 14530 *458 14552 *458-14555
NOTRCE	055476	447-14217 #447-14219
NO22BI	002426	#139 5145 *145 5335 163 5931 163-6066 186 7151 227 8079 227-8086 227-8131 340 10887 346-11060 360-11487 362 11532
NULLFL	002316	#139-5103 *246-8669
OLDCAC	002264	#139-5090
OLDCSR	002152	#139-5053 *169-6337 169-6355 *169-6356
ONES	002556	#141-5214 166-6241 171-6407 171-6409 210 7653 210-7655 219-7939 292 9820 292 9822 313-10294 313-10299 438-14020
PADDRE	002034	#139-5015 495-15883
PAFBAF	015406	184-7085 #198-7322 198 7347 198-7355
PAFBAW	015536	184-7086 #200-7360 200-7387 200 7392 200-7400
PARBAF	015710	184-7087 #202-7405 202-7430 202-7438
PARBAW	016040	184-7088 #204-7443 204-7470 204-7475 204-7483
PARCNT	002070	#139-5028 *164-6140 164-6152 *166-6198 166-6233 166 6255 166-6277 *322-10557 495 15883
PARITY	037762	147-5383 #322-10555
PARTHE	002266	#139-5091 *228-8159 *228-8163 *228-8168 *228-8172 *228-8184 *228-8188 *228-8193 *228-8197 *228-8202 *228-8206 *300-9975 *302-10037 *302-10049 *302-10052 *312-10249 *312-10264 322-10568 371-11691 *371-11710 *371-11727 373-11733 *373-11749 *373-11779
PARVEC	000114	#111-4194 *147-5383 *147-5384
PASFLG	002260	#139-5087 *206-7491 *208-7561 *238-8448 238-8471 *238-8472 266-9044 266-9116 266-9128 *266-9130 *266-9132 270-9247 272-9282 272 9294 *272-9296 *272-9299 274-9345 274-9390 274-9402 *274-9404 *274-9407 276-9444 276-9489 276-9501 *276-9503 *276-9505 278 9545 280-9591 280-9603 *280-9605 *280-9607 *375-11844
PATERR	002072	#139-5029 *166-6197 166-6231 166-6253 *166-6291
PATPLU	004550	149-5407 149-5410 149-5414 149-5417 149-5421 149-5424 #149-5427
PATTER	002110	#139-5036 *188-7173 *190-7196 *192-7230 *194-7258 *196-7299 *198-7323 *200 7361 *202-7423 *204-7463 214-7737 216-7792 *358-11459 358-11460 *358-11465 375-11785 *375-11803 375 11804 375-11805 375-11861 *375-11903 378-11920 *378-11954 382-11991 *382-12024
PCBUMP	002300	#139-5096 *214-7736 *216-7789 *219-7903 *219-7938 *219 7956 *221-7978 *221-7995 *234-8331 322-10563 375-11785 *375-11860 *375-11902
PCONF1	036632	181-7025 #316-10330 376-11914
PCONFS	037132	*316-10332 316-10373 #316-10379
PCONF1	037042	316-10345 #316-10364
PCONF2	037100	316-10333 316-10352 316-10362 #316-10373
PDP110	040156	147-5385 #324-10602
PD1	051754	410-12790 #410-12797
PERA05	054122	#436-13983 436-13990
PERBMK	054754	312-10254 312-10269 406-12727 439-14064 439-14076 441-14091 #443-14112 445-14185
PERECC	055034	#443-14128 443-14139
PERRAB	054572	436-14005 436-14010 #439-14075
PERRAW	054520	406-12704 408-12745 436-13962 436-13968 436-13974 436-13980 436-13995 436-14000 438-14016 438-14021 438-14026 438-14031 438-14036 438-14041 438-14051 438-14056 438-14061 #439-14063
PERRA3	051322	#406-12706 441-14097 441-14101
PERRA7	054644	438-14046 #441-14088
PERR01	104427	#110-4060 252-8780 308-10180
PERR02	104430	#110 4061 252-8768 252-8793 308 10174 311 10238
PERR03	104431	#110-4062 256-8842 256-8850
PERR04	104432	#110-4063 258-8866 258-8888
PERR05	054116	#436-13982
PERR06	054144	#436 13989

SYMBOL CROSS REFERENCE
CREF V01

SYMBOL	VALUE	REFERENCES							
PERR07	• 104433	#110 4064	260 8903	260 8922					
PERR10	• 104434	#110-4065	260 8908	260 8927					
PERR11	• 104435	#110 4066	262 8944	264 8994					
PERR12	• 104436	#110 4067	262 8950	264 9001					
PERR13	• 104437	#110 4068	262 8958						
PERR14	• 104440	#110-4069	262 8964						
PERR15	• 104441	#110-4070							
PERR16	• 104442	#110-4071							
PERR17	• 104443	#110 4072	288-9746	288-9768	288-9779	290 9803	293 9858	293 9862	
PERR20	• 104444	#110 4073	288-9752	288 9762	288 9785	290 9797	293-9867	293 9871	
PERR21	• 104445	#110-4074							
PERR22	• 104446	#110-4075	282-9656						
PERR23	• 104447	#110 4076	296 9935	298 9964					
PERR24	• 104450	#110-4077	296-9940						
PERR25	• 104451	#110-4078	304-10075	304 10080					
PERR26	• 104452	#110 4079	284-9690	286-9721					
PERR27	• 104453	#110-4080	284-9678	284 9704	286-9733				
PERR30	• 104454	#110-4081	306-10121						
PERR31	• 104455	#110-4082	266-9073	266 9079	274-9377	274 9384	276 9479	276 9485	278-9577	278-9584
PERR32	• 104456	#110-4083	266-9094	266-9107					
PERR33	• 104457	#110-4084	268-9195						
PERR34	• 104460	#110-4085	266-9100	266-9113	268-9203				
PERR35	• 104461	#110-4086	310-10197	310-10204					
PERR36	• 104462	#110 4087	340 10913						
PERR37	• 104463	#110-4088							
PERR40	• 104464	#110-4089							
PERR41	• 104465	#110-4090							
PERR42	• 104466	#110-4091							
PERR43	• 104467	#110-4092							
PERXOR	054730	439-14067	439-14079	441-14090	#441-14103	443-14134	445-14170		
PFECDF	057176	455-14487	#455-14492						
PFECDH	057136	455-14487	#455-14489						
PFECDT	057166	455-14487	#455-14491						
PFECEM	057102	455-14487	#455-14488						
PFECWS	057072	453-14420	#455-14487						
PFLAG	002120	#139-5040	169-6357	348-11105	•356-11379	•356-11419			
PGMCSR	002504	#139-5154	•159-5743	•159-5784	159-5789	•160-5827	•160-5852	•160-5853	•160-5864	244-8653
		328-10656	328-10658	328-10662	329-10682	329-10684	329-10688	348-11165	•348-11165	•348-11191
		•348-11198	350-11231	•350-11231						
PHEBE	013140	#179-6918	•179-6920	•179-6925	179-6928	179-6930	179-6932	179-6934	•179-6936	•179-6938
		266-9030	266-9087							
PHYADD	002036	#139-5016	•458-14568	•458-14569	•458-14570	•458-14571	•458-14573	458-14574		
PROTYP	003720	#145-5312	•145-5347	145-5348	145-5357	151-5461	161-5891	164-6160	166-6182	166-6211
		166-6220	166-6245	217-7842	217-7864	217-7890	219-7910	219-7943	219-7961	228-8156
		228-8165	228-8181	228-8190	228-8199	230-8222	230-8224	232-8289	232-8291	234-8338
		234-8340	236-8381	236-8400	236-8406	236-8429	236-8435	238-8452	242-8553	242-8573
		244-8623	244-8629	244-8635	314-10315	329-10691	348-11170	350-11234	367-11658	367-11662
		369-11674	369-11686	371-11728	373-11780	411-12839	413-12867	447-14251	451-14375	461-14657
PSIZE	002356	#139-5122	•181-6953	•181-6962	181-7018	181-7020				
PSW	• 177776	#111-4123	•155-5581	•155-5582	•155-5588	•166-6210	•166-6227	•166-6244	•166-6270	•171-6390
		•210-7625	•240-8499	•242-8547	•244-8593	•248-8703	•250-8736	•316-10336	•329-10707	•329-10709
		•348-11167	•350-11226	•352-11285	•352-11301	•362-11542	•371-11713	•373-11753	•373-11771	•375 11835

SYMBOL CROSS REFERENCE

SYMBOL	VALUE	REFERENCES
		•378 11935 •382 12006 •406 12695 •406-12711 •406 12721 •408 12741 •408 12753 •411 12806 •411-12812 •414-12909 •414 12913 •415 12956 •417 12997 •463 14701 •463 14702 •463-14717
PWRVEC	• 000024	#111 4187 •147-5381 •147 5382 •153 5480 •410 12780 •410 12781 •411 12847 •413 12858 •414-12940 415 12957 415 12958 •415 12960 •415 12961 •417 12998 •417-12999
QUICK	002410	#139-5134 •153-5479 362 11525
QVFLAG	002520	#139-5104 •153 5479 •153-5484 186 7127 •186 7129 228 8153 244 8660 266 9123 268-9220 272-9289 274 9397 276 9496 280 9598
RANODC	035710	•234-8349 •234 8366 #304 10077 •304-10081
RDCHR	• 104411	#110-4040 471 15081
RDDEC	• 104414	#110-4043 365-11595
RDLIN	• 104412	#110-4041 391-12132 472 15162 472 15213
RDOCT	• 104413	#110-4042 369 11676 371 11695 373 11737 373 11769 375-11789 375 11802 375-11807 384-12034 385 12084
READCS	• 104426	#110 4058 163 5979 266 9082 300-9985 302 10042 302-10054 312-10257 333-10758 333-10777 335-10793 335-10812 367-11655 369-11670 369-11679 406-12714 411-12822 463-14678
READON	002362	#139-5124
REALPA	002262	#139-5089 •217-7838 •217 7851 •217-7873 •219-7902 •219-7937 •219 7955 •221-7977 •221-7984 •221-7994 •223-8006 •223-8014 •223-8028 •223-8038 •225-8048 •225 8056 •225-8061 •227-8071 •230-8214 •231-8262 •231-8270 •232-8281 •232-8323 •234-8330 •236-8379 •238-8450 •240-8496 •242-8525 •244-8589 •244-8616 •244-8646 •246-8668 348-11146 348-11154 443 14140 443-14143 443-14146 443-14149 448-14273 457-14537 470-15020
REFRES	035020	292-9846 #293-9883
REFSUB	035070	293-9886 293-9890 #293-9894
REGCOP	036372	217-7841 217-7863 #313-10277
RELENT	043154	348-11147 348-11149 #348-11161
RELOCA	042530	190-7210 192-7240 194-7277 196-7314 198-7352 200-7397 202-7435 204-7480 238-8474 #348-11097
RELOC1	043170	#348-11164
RESREG	• 104416	#110-4046 219-7915 219-7929 232-8313 234-8352 234 8369 314-10321 314-10326 352 11268 352-11305 365-11585 367-11648 455-14480 476-15346
RESTAR	002570	•137-4983 •137-4985 #141-5219 151-5442
RESVEC	• 000010	#111-4182 •147-5385 •147-5386
RES0	045702	367-11640 367-11645 #367-11648
RES1	045762	367-11654 #367-11660
RES2	046130	369-11669 #369 11684
RLFLAG	002124	#139-5042 190-7207 192-7237 194-7274 196-7311 198-7349 200-7394 202-7432 204-7477 •348-11199 •350-11233 356-11420 360-11484 362-11524 365-11583 410-12795 448-14275 470-15014
RRFLAG	002122	#139-5041 188-7176 190-7194 192-7224 194 7256 196-7293 198-7332 200-7372 202-7415 204-7455 208-7545 236-8392 236-8421 238-8464 •356-11379 •356-11419 •356-11422 •356-11430 375-11815 378-11943 382-12013
SAVCSR	002150	#139-5052 •375-11823 375-11909
SAVREG	• 104415	#110-4045 219-7912 219-7920 232-8296 232-8302 234-8345 234-8358 314-10312 352-11266 352-11269 365-11580 453-14410 476-15331
SBEMSK	002246	#139-5084 •266-9039 •266-9040 266-9055 266-9057 266-9118 •266-9120 •266-9120 •268-9159 •268-9160 268-9165 268-9167 268-9178 268-9215 •268-9217 •268-9217 •270-9241 •270-9242 270-9252 270-9254 270-9258 270-9260 272-9290 •272-9292 •272-9292 •272-9302 272-9305 •272-9312 272-9313 272-9315 •274-9339 •274-9340 274-9350 274-9352 274-9356 274-9358 274-9398 •274-9400 •274-9400 •274-9410 274-9413 •274-9420 274-9421 274-9423 •276-9438 •276-9439 276-9449 276-9451 276-9455 276-9457 276-9497 •276-9499 •276-9499 •276-9508 276-9511 •276-9518 276-9519 276-9521 •278-9539 •278-9540 278-9549 278-9551 278-9555 278-9557 280-9599 •280-9601 •280-9601 •280-9610 280-9613 •280-9620 280-9621 280-9623
SBEMT	017372	210-7630 210 7632 210 7637 210-7639 210-7648 210-7654 210-7656 210 7662 210 7664

SYMBOL CROSS REFERENCE

SYMBOL	VALUE	REFERENCES								
		210-7673	#210 7682							
SBETES	017114	208-7555	#210 7597							
SCOPE	• 000004	#111-4122	157-5618	164-6133	166 6170	169-6318	184-7075	184 7092	248 8709	250 8?4•
SDPAR0	• 172260	#111 4286	219 7924	219 7926	232-8304	232 8306	232 8307	234 8362		
SDPAR5	• 172272	#111-4291	•219-7928	•232-8308						
SDPAR6	• 172274	#111-4292	•234-8364							
SDPAR7	• 172276	#111-4293	•219 7927							
SEEDHI	002544	#141-5209	•147-5366	234-8333	•234-8372	474 15304	474 15311	•474 15316		
SEEDLO	002546	#141-5210	•147-5367	234-8332	•234-8371	474 15303	474-15309	•474-15315		
SELONL	002000	#139-4998	184-7095	208-7512	356 11429	•385 12095	•385-12100			
SETPAT	045004	192-7226	196-7295	#358-11463						
SHADL1	011612	#167-6300	167-6315							
SHUTUP	045170	186-7138	362-11505	#362-11513						
SIPAR0	• 172240	#111 4276	346 11002							
SIPAR3	• 172246	#111-4279	266-9032	•266-9033	340-10870	346-11020				
SIPAR5	• 172252	#111-4281	217-7856	217-7880	240 8510	242 8537	244 8604	266-9033	•266 9034	346 11050
		346-11071								
SIPAR6	• 172254	#111-4282	217-7856	217-7880	240-8510	242-8537	244-8604	•346 11071		
SIPDR0	• 172200	#111-4256	346-11003	410-12799	413-12899					
SIZE	• 040000	#113-4415	164-6146	166-6206	166-6239	217 7840	217-7853	217-7875	219 7942	219 7960
		234-8337	236-8397	236-8426	238-8465	242-8530	242 8567	244-8618	314-10314	348 11168
		350-11227	352-11286	352-11302						
SKIPKA	002006	#139-5001	367-11632	•367-11634	•384-12043	•384 12049				
SKIPMM	002314	#139 5102	169-6344	206-7496	227-8098	•356 11380	•356 11440			
SKJ	055524	447-14221	447-14224	#447-14234						
SKPERR	002064	#139-5026	•210-7643	•210-7668	340-10911	•340-10915				
SKUB	043144	348-11155	#348-11158							
SKUJ	013142	179-6916	#179-6919							
SOBK	002534	#141-5205	240-8502							
SOBLEN	• 000056	240-8500	240-8505	#306-10136						
SOFTPA	002562	#141-5216	244-8619							
SOURCE	002274	#139-5094	•266-9050	•268-9163	•270-9256	•274-9354	•276-9453	•278-9553	•300-9971	342 10931
SPLTCS	002234	#139-5081	•169-6347	•170-6369	•208-7518	•208-7528	208-7550	•208-7582	•208-7585	•208 7592
		•227-8104	•227-8137	•227-8142	346-11028	346-11033	•375-11843	•375-11846		
SSP	-#000006	#111-4140	•155-5585	•248-8704	•250-8741	411-12813	•414-12910	463-14703		
ST	• 177776	#483-15574								
STACK	• 002000	#111-4117	111-4118	137-4973	137-4991	141-5206	153-5483	186-7134		
START	003632	137-4984	137-4986	#144-5272	362-11510					
START1	000300	137-4977	#137-4983							
START2	000310	137-4978	#137-4985							
START3	000200	#137-4977	503-16221							
STAR27	024314	236-8384	#236-8389							
STOPOK	002372	#139-5127	•177-6799	447-14234	•447-14235	•471-15090				
STRIPE	002340	#139-5115	•292-9825	292 9829	292-9848	•293-9879	293-9879			
SUBAAA	004606	149-5425	#151-5438							
SUBAAB	004736	#153-5468								
SUBAAI	011606	166-6222	#167-6295							
SUBAAP	013322	179-6939	#181-6952							
SUBAAR	012466	170-6384	#177-6799							
SUBAAS	010540	163-5953	#164-6131							
SUCCES	002306	#139-5099	•208-7544	•208 7570	208-7576	•380-11969	380-11974	•380-11976		
SUPDOA	002256	#139 5086	•217 7844	•217 7866	•217-7892	•219 7913	•219-7916	•219-7945	•219-7963	•221 7980

```
SYMBOL  VALUE        REFERENCES
                     •221 7990   •221 7997   •223 8007   •223-8021   •223-8029   •223-8039   •225 8049   •225 8057   •225 806.
                     •228-8158   •228 8167   •228 8183   •228 8192   •228-8201   •230-8230   •230 8232   •230 8235   •230 823.
                     •231 8263   •231-8271   •232 8297   •232-8306   •232-8324   •234 8347   •234-8351   •236 8386   •236-8402
                     •236-8408   •236-8431   •236 8437   •238-8458   •240 8506   •242 8562   •242 8575   •244 8598   •244 8626
                     •244-8631   •244-8637   •244 8658    250-8743   •314-10318
SUPD01    026466      217-7848    217-7870    217-7896    219 7921    228 8164    228 8189    230 8245    234 8365    234 8370
                      236-8387    238-8455    244-8640   #248 8680    314 10325
SUPD02    026502      219-7930    219-7952    219 7972    228-8173    228 8198    228-8207    230 8248    230 8252    230 8255
                      242-8578   #248-8682
SUPD03    026644      217-7845    217-7867    217-7893    219-7914    221 7981    221-7991    221-7998    223 8008    223 8022
                      223 8031    223-8040    225 8050    225-8058    225-8063    228-8160    228-8185    230-8231    231 8264
                      231-8273    232-8325    234-8348    234-8353    236-8383    236 8409    236 8438    238 8457    244-8632
                      244 8638    244 8659   #250-8714    314 10319
SUPD04    026660      219 7917    219 7946    219 7965    228 8169    228-8194    228 8203    230-8233    230 8236    230 8238
                      232-8310    232 8316    240-8516    242-8566    242 8576    244 8611    244-8663   #250 8715
SUPDR0    002154     #139-5055   •248 8684    248-8692   •250 8717    250 8725
SUPDR1    002156     #139-5056    248-8685    250 8718
SUPDR2    002160     #139-5057
SUPDR3    002162     #139-5058
SUPDR4    002164     #139-5059
SUPDR5    002166     #139-5060
SUPDR6    002170     #139-5061    248-8695    250-8728
SUPLIM    053764     #419-13136   503-16204   503-16205
SUPSTK  = 000740     #111-4119    155-5584    248-8704    250-8741    463 14706   463 14712
SWAPAT    002576     #141-5227   •147-5369
SWR       002600     #141-5230   •151-5455    151-5457   •151-5462   •153 5481    155-5598    164-6163    170-6378    181-7024
                      184-7077    186-7127    214-7730    214-7745    227-8121    228-8153    266-9122    268-9219    272-9288
                      274-9396    276-9495    280-9597    316-10345   348-11098   348-11117   348-11153   375 11852   378-11924
                      382-11995   406-12730   411-12843   413-12865   447-14234   447-14241   447-14259   450-14310   450 14349
                      450-14357   451-14364   451-14368   455-14481   468-14945   468-14953   468 14975   471-15088
SWREG   = 000176     #113-4407    151-5462    155-5598    468-14945
SW0     = 000001     #111-4160    164-6163    170-6378    214-7730    214-7745    227-8121    375 11852   378-11924   382 11995
                      406-12730
SW1     = 000002     #111-4159
SW10    = 002000     #111-4150    450-14310
SW11    = 004000     #111-4149    186-7127    228-8153    266-9122    268-9219    272 9288    274-9396    276 9495    280-9597
SW12    = 010000     #111-4148    348-11098
SW13    = 020000     #111-4147    348-11117   348-11153   450-14349
SW14    = 040000     #111-4146    447-14241
SW15    = 100000     #111-4145
SW2     = 000004     #111-4158
SW3     = 000010     #111-4157
SW4     = 000020     #111-4156    316-10345
SW5     = 000040     #111-4155    450-14357
SW6     = 000100     #111-4154    181-7024
SW7     = 000200     #111-4153    455-14481
SW8     = 000400     #111-4152    447-14234
SW9     = 001000     #111-4151    447-14259   451-14368
SYSSIZ    003722      145-5311   #145-5313
TAG2$     011202      166-6192   #166-6219
TAG3$     011236      166-6218   #166-6223
TAG4$     026562      248-8693   #248-8695
```

SYMBOL CROSS REFERENCE CREF V01

SYMBOL	VALUE	REFERENCES
TAG70$	057202	455 14457 #457 14499
TAG71$	057212	455-14458 #457 14505
TAG72$	057222	455-14459 #457-14511
TAG73$	057272	455-14460 #457 14525
TAG74$	057332	455-14461 #457-14537
TAG75$	057344	455-14462 #457-14543
TAG76$	057356	455-14463 #458-14549
TAG77$	057422	455-14464 #458-14562
TAG78$	057430	455-14465 #458-14568
TAG79$	057510	455-14466 #458-14583
TAG9$	011030	166-6186 #166-6190 166 6225 166 6262 166-6284
TBG4$	026740	250-8726 #250-8728
TCFIG1	037256	#320-10436 320-10456 457-14516
TCFIG2	037410	#320-10465 320-10494 458-14554
TCFIG3	037576	#320-10503 320-10517 457-14529
TCONFI	037134	316-10350 316-10361 316-10371 #318-10395
TEMP	002406	#139-5133 *163-5983 163-6031 163-6033 *183-7031 183 7048 *183 7049 *380-11962 *380 11963 380-11964
TEST	006044	157-5651 157-5663 157-5666 158-5670 158-5676 #158-5718
TESTAD	002364	#139-5125 159-5745 *159-5746 *159-5747 163-5927 *163-5957 *163-5958 *163-6097 *163-6108 *163-6109 *208-7548 208-7549 *208-7549 *208-7552 *208-7554 *208-7573 208-7573 210-7623 210-7624 *216-7790 *216-7791 221-7979 221-7996 *227-8081 *227-8088 *227-8093 *227-8096 *227 8101 *227-8103 *227-8110 *227-8111 *227-8132 *227-8134 228-8150 228-8151 266-9059 266-9060 266-9146 266 9147 268-9169 268-9210 268-9227 270-9238 272-9319 274-9364 274-9370 274-9427 276-9463 276-9470 276-9474 276-9525 278-9563 278-9572 280 9627 300-9974 375-11785 *375-11845 *375-11846 *375-11850 *375-11902 443-14129
TESTMO	002524	#141-5201 *145-5330 *145-5351 166-6210 166-6227 166-6244 166-6270 171-6390 210 7625 240-8499 242-8547 244-8593 250-8736 329-10707 362-11542 371-11713 373-11753 373-11771 406-12695 406-12711 406-12721 408-12741 408-12753
TIME	002312	#139-5101
TIMEOU	040132	145-5357 147-5387 151-5461 161-5891 164-6160 166-6220 #324-10595 367-11658 367-11662 369-11674 369-11686 371-11728 373-11780
TKVEC	000060	#111-4190 316-10331 316-10331 *316-10333 *316-10334 *316-10376 *316-10376 375-11785 375-11785 *375-11832 *375-11833 *375-11900 *375-11900 378-11920 378-11920 *378-11932 *378-11933 *378-11954 *378-11954 382-11991 382-11991 *382-12003 *382-12004 *382-12024 *382-12024
TMFLAG	002132	#139-5045 *198-7343 *200-7383 *202-7426 *204-7466 358 11449
TOOMAN	002360	#139-5123 *443-14124 450-14357 *451-14392
TOTCSR	002220	#139-5074 *158-5715 159-5750 160-5803 160-5830 163-5928 333-10752 335-10787 339-10843 391-12125 411-12817 413-12884 463-14673
TRACE	006140	#158-5738 *161-5874 *161-5880 161-5882 *161-5886 *161-5890 *387-12114 *389-12120 447-14216
TRAPVE	000034	#111-4189 *147-5379 *147-5380
TSTBAN	011450	166-6202 #166-6265
TSTDAT	002242	#139-5083 *266-9042 *266-9043 *266-9046 *266-9047 266-9048 266 9049 266-9050 *266-9056 *266-9058 266-9062 266-9064 266-9070 266-9076 *266-9135 *266-9136 *268-9161 *268-9162 268-9163 *268-9166 *268-9168 268-9172 268-9175 *270-9245 *270-9246 *270-9249 *270-9250 270-9256 *270-9259 *270-9261 *270-9263 *270-9265 270-9267 270-9269 *272-9300 *272-9301 *274-9343 *274-9344 *274-9347 *274-9348 274-9354 *274-9357 *274-9359 *274-9361 *274-9363 274-9366 274-9368 274-9374 274-9381 *274-9408 *274-9409 *276-9442 *276-9443 *276-9446 *276-9447 276-9453 *276-9456 *276-9458 *276-9460 *276-9462 276-9465 276 9468 276-9476 276-9482 *276-9506 *276-9507 *278-9543 *278-9544 *278-9547 *278-9548 278-9553 *278-9556 *278-9558 *278-9560 *278-9562 278-9565 278-9567 278-9574 278-9581 *280 9608 *280 9609 *300 9969 *300-9970 300 9971 *300-9987 *300 9989 *300-9991 *300-9992 *300-9994 *300-9997

SYMBOL CROSS REFERENCE CREF V01

SYMBOL	VALUE	REFERENCES								
		•300-9998	•300 10000	•300-10002	•300 10003	•300 10005	302 10031	302-10033	438 14025	438 14030
		443-14130	443 14132	495 15892	495 15892					
TSTRD1	040604	329 10693	329-10695	•329-10706						
TSTREA •	104510	•110-4110	210 7641	210 7666	266 9096	266 9109	268-9185	268 9197		
TST1	005436	•157 5618								
TST2	010544	•164-6133								
TST3	010730	•166-6170								
TST4	011716	•169-6318								
TST5	014212	•184-7075								
TST6	014264	•184-7092								
TYPDS •	104405	•110-4034	181-7004	181 7008	181 7012	181-7016	181-7020	181-7022	186 7131	380 11964
		380-11966	380-11981	457 14505						
TYPEIT •	104401	•110-4030	154-5497	154-5510	154-5512	154 5514	154-5521	154-5526	154-5527	160-5863
		179-6864	179-6947	181-7001	181-7005	181 7009	181-7013	181-7017	181-7021	181-7023
		186-7126	186-7128	208-7577	208-7579	316-10339	316-10340	316-10341	316-10349	316-10353
		316-10354	316-10355	316-10356	316 10357	316-10358	316-10367	316-10368	316-10369	316 10370
		318-10395	318-10398	318 10400	318-10409	318-10424	320-10434	320-10453	320-10464	320-10491
		320-10502	320-10514	320 10524	320-10532	320-10534	348-11118	348-11148	348-11156	365-11581
		365-11584	365-11594	365-11599	367-11630	367-11660	369-11671	369-11675	369-11680	369-11684
		371-11693	371-11694	371-11719	371-11723	373-11735	373-11736	373-11759	373-11763	373-11766
		373-11768	373-11775	375-11787	375-11788	375-11797	375-11801	375-11806	375-11816	375-11819
		378-11922	378-11929	378-11931	378-11948	380-11965	380-11967	380-11975	380-11982	382-11993
		382-12000	382-12002	382-12018	384-12033	384-12042	384-12047	384-12053	384-12060	385-12079
		385-12083	385-12094	385-12099	387-12113	389-12119	391-12131	414-12931	419-13054	450-14312
		450-14313	451-14383	453-14411	453-14435	453-14436	453-14438	453-14445	453-14447	455-14474
		455-14484	457-14515	457-14519	458-14553	458-14562	458-14577	458-14583	458-14585	463-14671
		463-14672	463-14680	463-14689	463-14693	463-14695	463-14705	463-14708	463-14710	463-14714
		463-14720	463-14723	463-14725	463-14729	463-14731	466-14832	467-14897	468-14942	468-14951
		468-14952	468-14954	468-14963	468-14968	468-14977	468-14995	470-15004	470-15015	470-15017
		470-15019	471-15085	471-15098	471-15104	471-15109	471-15113	471-15118	471-15119	471-15121
		471-15124	471-15128	472-15189	472-15191	472-15192	472-15193	472-15248	472-15250	472-15251
		472-15252	479-15470							
TYPOC •	104402	•110-4031	453-14416	457-14499	463-14679	463-14694	463-14696	463-14709	463-14711	463-14724
		463-14726	468-14953							
TYPOS •	104403	•110-4032	179-6865	208-7578	371-11716	373-11767	373 11776	380-11978	457-14537	457-14543
		458-14584	470-15018	470 15020						
T12A	033250	•278-9537	280-9606							
T12B	033272	•278-9541	280-9602							
UDPAR0 •	177660	•111-4246	232-8309							
UDPAR7 •	177676	•111-4253	•232-8307							
UIPAR0 •	177640	111-4235	•111-4236	346-11006	354-11310					
UIPAR1 •	177642	•111-4237	•219-7925	•234-8363	•234-8368					
UIPAR2 •	177644	•111-4238	166-6188	219-7927	234-8364	•314-10324				
UIPAR3 •	177646	•111-4239	166-6189	236-8386	244-8626	340-10872	346-11023			
UIPAR4 •	177650	•111-4240	228-8172	228-8188	228-8206	352-11272	352-11289			
UIPAR5 •	177652	•111-4241	217-7859	217-7883	•219-7950	•219-7970	240-8513	242 8540	244-8607	346-11048
		346-11069								
UIPAR6 •	177654	•111-4242	217-7859	217-7883	219-7951	219-7971	228-8163	228-8197	240-8513	242-8540
		244-8607	•346-11069							
UIPDR0 •	177600	•111-4215	346-11007	354-11312	410-12798	413-12898				
UNITOP	002370	•139-5126	•181-6993	•181-6996	•181-6997	•181-6998	•181-6999	181 7000	181-7000	
UNRELO	043420	190 7214	192-7244	194 7281	196-7318	198-7356	200-7401	202-7439	204-7484	238-8484

SYMBOL CROSS REFERENCE CREF V01

SYMBOL	VALUE	REFERENCES
		#350 11203 360 11484 362 11524
UPPFLG	002261	#139-5088 *274 9369 274 9386 *274-9388 *278-9568 280 9587 *280 9589
USERMA	044120	348-11166 350 11225 352-11267 #354-11309
USESTK	* 000700	#111 4120 155 5590 250 8739 463-14721 463 14727
USP	*#000006	#111 4141 *155-5591 *250 8739 411 12807 *414 12914 463 14718
WARN1	011116	#166-6203
WARN2	027160	#254-8810 *313 10303
WARN3	027174	#254-8815 *313 10304
WARN4	027220	#254-8827 *313 10305
WARN5	027234	#254-8832 *313 10306
WARN6	036614	#314-10324
WARN6A	036554	#314-10317
WARN6B	036606	314-10316 #314-10323
WARN7	024272	#236-8386
WASDBE	* 104500	#110-4102
WASSBE	* 104476	#110-4100
WAS1DB	* 104501	#110-4103 171-6400 171 6413 171-6423 171-6433 270-9273
WAS1SB	* 104477	#110-4101
WHICHC	051152	367-11652 369-11667 #391-12124
WOOPEN	053162	415-12959 415-12962 415 12964 417-12980 417-13003 #417-13011 417-13014
WOOPS	052614	410-12795 #415-12952
WOOPSA	053212	*415-12957 *415-12958 415-12959 417-12998 417-12999 417-13002 #417-13014
WOOPUP	053000	415-12959 415-12959 415-12960 415-12962 415-12962 415-12962 #417-12979 417-13003 417-13004 417-13014
WORST	002542	#141-5208 *147-5361 *194-7269 *196-7306 *200-7389 *204-7472 356-11426
XOCHAR	053554	#419-13068 *419-13102 *419-13105 *419-13106 419-13107 *419 13111 *419-13112 419-13113 468-14920 468-14922 *468-14923
XXDPCH	002326	#139-5107 *153-5488 451-14382
ZEROS	002310	#139-5100 145-5321 292-9819 292-9823
$APTHD	063176	137-4975 #478-15439
$AUTO	002060	#139-5024 *153-5479 *153-5484 155-5597 468-14949
$BANK	002011	#139-5004 *448-14272
$BASE	063152	#477-15411
$BELL	002615	#141-5237 378-11948 382-12018 450 14312
$CACHF	040274	#326-10643 481-15507
$CACHN	040250	#326-10636 481-15506
$CBCSR	040734	#331-10739 481-15550
$CB1CS	040756	#331-10744 481-15551
$CDW1	063156	#477-15414
$CDW2	063160	#477-15415
$CHARC	053742	*419-13056 *419-13066 *419-13124 #419-13129
$CHKDI	041330	#337-10831 481-15558
$CHK1D	041344	#337-10836 481-15559
$CKSWR	060774	#468-14914 481-15490
$CLRCS	041306	#337-10822 481-15556
$CLR1C	041320	#337-10826 481-15557
$CMTAG	002000	#139-4997 144-5285
$CMTGE	002516	#139-5195 144-5287
$CNTLC	062166	468-14963 471-15085 #471-15140
$CNTLG	062200	468-14951 #471-15142
$CNTLK	061374	468-14942 #468-14997
$CNTLU	062173	468-14968 471-15113 #471 15141

SYMBOL CROSS REFERENCE

SYMBOL	VALUE	REFERENCE
$CPUDF	063124	#477 15384
$CRLF	002622	#141-5239 181 7001 316 10353 316 10356 380 11982 419 13055 453 14411 453 14438 453 14447 463 14672 463 14693 463 14708 463 14725 463 14751 468 14[illegible] 470 15004 471 15118
$DBLK	060764	46[illegible] 14867 46[illegible] 14897 #46[illegible] 14905
$DB20	062756	458 14575 #476 15331
$DDW0	063162	149 5403 #478 15426
$DDW1	063164	#478 1542[illegible]
$DDW2	063166	#478-15428
$DDW3	063170	#478 15429
$DDW4	063172	#478 15430
$DDW5	063174	#478-15431
$DEENE	040240	#326-10632 481 15502
$DEVC[illegible]	063106	#186 7162 #447 14210 #447 14212 #47[illegible] 153[illegible]1
$DEVM	063154	#477 15412
$DOAGA	014606	186 7133 186 7135 #186 7169
$DOAGN	014502	#186 7145
$DOWN	052204	#411-12848 411 12849
$DTBL	060754	467 14870 #467 14901
$ECCDI	040632	#331-10715 481-15546
$ECCIN	040660	#331-10723 481-15548
$ECC1D	040646	#331-10719 481-15547
$ECC1I	040674	#331-10727 481-15549
$ENASB	040706	#331 10731 481-15560
$ENA1S	040722	#331 10735 481-15561
$ENDAD	014472	137-4966 153-5485 154-5496 #186 7141
$ENERG	040230	#326 10628 481 15501
$ENV	063116	144-5276 153-5478 #477-15376
$ENVM	063117	153-5472 153-5475 #477 15380
$EOP	014336	#186 7117
$ERFLG	002012	#139 5005 447 14257 #447 14263 #450-14306
$ERRGE	041454	#340-10865 481 15564
$ERROR	056000	147 5377 #450-14302
$ERRTB	063520	453 14426 #485 15591
$ERRTV	056562	450-14361 #453 14410
$ERTTL	002572	#141 5220 186 7123 380 11966 380 11968 #450 14314 #450 14316 451 14382
$ESCAP	002334	#139-5113 #447 14266 451 14371 451 14373
$ETABL	063116	#477-15375
$ETEND	063176	#478-15435 478 15445
$EXMAL	045162	#362 11509
$FATAL	063100	#450 14348 #477 15368
$FILLC	002614	#141-5236 419 13059
$FILLS	002331	#139-5110 #384-12037
$GTSWR	061150	#468-14952 481 15489
$HALT	056344	#451 14366
$HALT2	063252	#479-15471
$HIBTS	063176	#478-15440
$HIOCT	062376	371-11696 373 11738 #472 15184 #472-15195
$ILLUP	052606	410-12780 413-12858 #414 12937 414-12938
$INVAL	041424	#339-10856 481-15563
$ITEMB	002013	#139-5006 #450-14323 450 14325 450 14335 #450 14336 453 14413
$KERNE	040220	#326 10624 481-15500
$KMAP	042436	#346 11075 481 15504

SYMBOL CROSS REFERENCE CREF V01

SYMBOL	VALUE	REFERENCES
$LF	002623	#141-5240 471 15128
$LOADC	040312	#328-10655 481 15509
$LPADR	002564	#141-5217 248-8683 •248 8693 248 8694 •248 8710 250-8716 •250-8726 250 8727 •250-8747 •447 14261 •447 14264 447 14268
$LPERR	002566	#141-5218 248-8683 •248 8694 •248 8710 250-8716 •250 8727 •250-8747 447 14261 •447-14265 451-14369
$MADR1	063130	#477-15398
$MADR2	063134	#477-15402
$MADR3	063140	#477 15405
$MADR4	063144	#477-15408
$MAIL	063076	#477-15366 478-15441 478 15445
$MAMS1	063126	183-7032 #477-15391
$MAMS2	063132	#477-15400
$MAMS3	063136	#477 15403
$MAMS4	063142	#477-15406
$MBADR	063200	#478-15441
$MNEW	062216	468-14954 #471 15144
$MSGAD	063112	#477-15373
$MSGLG	063114	#477-15374
$MSGTY	063076	•451-14379 #477-15367
$MSWR	062205	468-14952 #471-15143
$MTYP1	063127	#477-15392
$MTYP2	063133	#477-15401
$MTYP3	063137	#477-15404
$MTYP4	063143	#477-15407
$NOTRA	063246	#479-15470 481-15485 481 15487 481-15540 481-15541 481 15542 481 15543 481-15544 481 15565 481-15566 481-15567 481-15568 481-15569 481-15570 481-15571
$NULL	002330	#139-5109 419-13061
$NWTST	• 000001	#157-5618 157-5618 #157-5618 #164-6133 164-6133 #164-6133 #166-6170 166-6170 #166 6170 #169-6318 169-6318 #169-6318 #184-7075 184-7075 #184-7075 #184 7092 184 7092 #184-7092
$OCNT	060544	•466-14804 •466-14833 #466 14846
$OCTVL	063060	476-15333 #476-15358 476-15361
$OCTB	• 063064	458-14577 #476-15361
$OMODE	060546	•466-14799 •466-14803 466-14808 •466-14811 •466 14822 #466-14848
$OVER	055714	447-14241 447-14262 #447-14267
$PASS	063104	•153-5471 •186-7124 •186-7125 186-7131 186-7159 •186-7165 223-8004 223-8012 223-8026 223-8035 225-8046 225-8054 227-8069 232-8321 348 11100 380-11962 #477 15370
$PASTM	063204	#478-15443
$PATMA	002010	#139-5003 414-12927 414-12928 •448-14273 •448-14277 448-14282 448-14283 450-14309
$PER01	053764	#436-13958 481-15512
$PER02	054012	#436-13964 481-15513
$PER03	054040	#436-13970 481-15514
$PER04	054070	#436-13976 481-15515
$PER0`	054152	#436-13992 481-15516
$PER10	054174	#436-13997 481-15517
$PER11	054224	#436-14002 481-15518
$PER12	054244	#436-14007 481-15519
$PER13	054266	#438-14013 481-15520
$PER14	054306	#438-14018 481-15521
$PER15	054330	#438-14023 481-15522
$PER16	054352	#438-14028 481-15523
$PER17	054372	#438-14033 481-15524

SYMBOL CROSS REFERENCE CREF V01

SYMBOL	VALUE	REFERENCES
$PER20	054410	#438-14038 481 15525
$PER21	054426	#438-14043 481 15526
$PER22	054446	#438-14048 481-15527
$PER23	054464	#438-14053 481-15528
$PER24	054502	#438-14058 481-15529
$PER25	051240	#406-12693 481-15530
$PER26	054672	#441-14095 481-15531
$PER27	054712	#441-14099 481-15532
$PER30	051466	#408-12738 481-15533
$PER31	055102	#443-14138 481-15534
$PER32	055200	#445-14156 481-15535
$PER33	055246	#445-14165 481-15536
$PER34	055326	#445-14176 481-15537
$PER35	055360	#445-14185 481-15538
$PER36	055414	#445-14193 481-15539
$PWRDN	051634	147-5381 #410-12767 414-12930
$PWRUP	052210	411-12847 #413-12854 417 13010
$QUES	002621	#141-5238 468-14995 471 15121
$RAND	062662	#474-15302
$RDCHR	061514	#471-15047 481-15492
$RDDEC	062400	#472-15210 481-15495
$RDLIN	061644	#471-15076 481-15493
$RDOCT	062230	#472-15159 481-15494
$READC	040406	#328-10671 481-15510
$RESRE	062624	#473-15283 481-15498
$SAVRE	062566	#473-15272 481-15497
$SAVR6	052612	#411-12845 413-12860 #413-12861 #413 12862 #414-12939 417-12993
$SCOPE	055444	147-5375 #447-14210
$STN -	000001	#157-5618 #164-6133 #166-6170 #169-6318 #184-7075 #184-7092
$SVLAD	055700	447-14249 447-14258 #447-14264
$SWR -	163000	#108-3994 157-5618 164-6133 166-6170 169-6318 184-7075 184-7092
$SWREG	063120	153-5481 #477-15382
$TESTN	063102	#450-14309 455-14491 #477-15369
$TKB	002606	#141-5233 316-10335 316-10375 375-11834 375-11901 378-11934 378-11953 382-12005 382 12023 419-13105 419 13111 468-14927 468-14959 470-15031 471-15051 471-15057
$TKS	002604	#141-5232 316-10337 316-10374 375-11836 375-11899 378-11936 378-11952 382-12007 382-12022 419-13103 419-13109 468-14925 468-14957 470-15029 471-15049 471-15055
$TN -	000007	#108-3995 157-5618 157-5618 #157-5618 164-6133 164-6133 #164-6133 166-6170 166-6170 #166-6170 169-6318 169-6318 #169-6318 184-7075 184-7075 #184-7075 184 7092 184 7092 #184-7092
$TPB	002612	#141-5235 419-13118
$TPFLG	002332	#139-5111 #153-5473 419-13038
$TPS	002610	#141-5234 419-13100
$TRAP	063212	147-5379 #479-15455
$TRAP2	063234	#479-15466 481-15481
$TRPAD	063254	479-15460 #481-15481
$TSTM	063202	#478-15442
$TSTRD	040426	#329-10678 481-15562
$TTYIN	062142	471-15078 471-15079 471-15101 471-15119 471-15133 #471-15137
$TYPDS	060550	#467-14860 481-15486
$TYPE	053430	#419-13038 481-15482
$TYPEC	053556	419-13058 419-13065 #419 13069 468-14981

SYMBOL CROSS REFERENCE CREF V01

SYMBOL	VALUE	REFERENCES
$TYPEX	053744	419-13125 419 13127 #419 13130
$TYPOC	060346	#466-14802 481-15483
$TYPON	060362	466-14801 #466 14804
$TYPOS	060322	#466-14797 481 15484
$UNIT	063110	*186-7163 *447-14213 #477 15372
$UNITM	063206	#478-15444
$USWR	063122	186 7159 #477 15383
$VECT1	063146	#477 15409
$VECT2	063150	#477-15410
$WASOB	041142	#335-10786 481-15554
$WASSB	040776	#333-10751 481 15552
$WAS1D	041256	#335-10812 481-15555
$WAS1S	041112	#333-10777 481-15553
$XTSTR	055572	#447-14243
$ZAP42	014452	#186-7134 451-14386
$OFILL	060545	*466-14798 *466-14802 466 14812 #466 14847

MACRO CROSS REFERENCE
CREF V0.

MACRO NAME	REFERENCES
AND	#6 4500 #108 3984
ANDB	#18 600 #108 3987
BEGIN	#12 6300 #108 3985 179 6842 206 7494 208-7543 227 8078 333 10754 335 10789 339 10844 348 11102 348-11122 385 12081 411 12818 413 12886 463-14674
BMOV	#126-4725 #166-6187 #217 7847 #217 7869 #217-7895 #219-7919 #219 7922 #219-7923 #219 7924 #219 7948 #219-7949 #219-7968 #219 7969 #228-8162 #228-8171 #228 8187 #228 8196 #228 8205 #230-8244 #230-8247 #230 8251 #230 8254 #232-8303 #232-8304 #232-8305 #234 8359 #234-8360 #234-8362 #234-8367 #236-8385 #238 8454 #240-8500 #242-8556 #244 8594 #244 8625 #314-10323 #329 10695 #348-11168 #350 11227 #352 11286 #352 11302 #415-12959 #415-12962
CASE	#9 100 #108-3984 212-7693 365 11601
CLEAR	#19-100 #108-3982 #181-6953 #210 7628 #210-7635 #210-7683 #238-8477 #238-8488 #266-9148 #272-9320 #274-9428 #276-9526 #313-10296 #316-10348 #316 10366 #337-10822 #337-10826 #348-11160 #350-11220 #350 11237 #356-11377 #356 11379 #356-11380 #451 14392
CLEARB	#19-1300 #108-3982
DLEFT	#135-4927 #260-8912 #260-8931 #266-9120 #266-9126 #268-9217 #268-9223 #272 9286 #272-9292 #274 9394 #274-9400 #276 9493 #276 9499 #280-9595 #280-9601
DOWNTO	#10-3800 #108-3985
ELSE	#7-100 #108-3984 #153-5482 #153-5487 #164-6165 #167-6306 #167-6311 #169-6348 #170-6380 #179-6861 #181-6963 #181-6966 #181-6969 #181-6973 #183-7039 #214-7732 #214-7747 #223-8018 #227-8097 #227 8115 #227-8123 #244-8633 #246-8675 #292-9821 #292-9837 #293-9864 #329-10701 #333-10771 #335-10806 #362 11508 #367-11635 #367-11643 #375-11854 #378-11926 #382-11997 #406-12701 #406-12732 #419-13115 #439-14070 #439-14082 #443-14131 #445-14179 #451-14389 #463-14713 #463-14728 #470-15035
END	#12-100 #108-3985 #149-5432 #149-5434 #151-5447 #151-5464 #153-5474 #153-5477 #153-5489 #153-5490 #153-5491 #154-5498 #155-5600 #155-5601 #164-6167 #166-6280 #166-6283 #167-6309 #167-6313 #169-6350 #169-6359 #170-6370 #170-6371 #170-6372 #170-6373 #170-6374 #170-6375 #170-6382 #171-6434 #171-6435 #171-6436 #171-6439 #179-6853 #179-6856 #179-6863 #179-6867 #179-6868 #179-6869 #179-6870 #181-6965 #181-6971 #181-6972 #181-6977 #181-6978 #181-6979 #181-6980 #181-6981 #181-6990 #181-6994 #181-6995 #181-7026 #183-7041 #183-7052 #183-7055 #183-7057 #183-7058 #183-7061 #183-7073 #184-7100 #186-7130 #186-7167 #186-7168 #188-7181 #188-7182 #206-7498 #206-7499 #208-7568 #208-7572 #208-7573 #208-7574 #208-7575 #208-7580 #208-7581 #208-7583 #210-7649 #210-7674 #212-7722 #214-7734 #214-7741 #214-7749 #216-7796 #223-8005 #223-8013 #223-8020 #223-8027 #223-8036 #225-8047 #225-8055 #227-8070 #227-8084 #227-8085 #227-8091 #227-8092 #227-8095 #227-8107 #227-8117 #227-8125 #227-8135 #227-8139 #228-8155 #232-8322 #236-8404 #236-8410 #236-8411 #236-8412 #236-8413 #236-8417 #236-8433 #236-8439 #236-8440 #236-8441 #236-8442 #238-8468 #238-8469 #238-8470 #238-8479 #238-8486 #244-8642 #246-8677 #266-9085 #266-9086 #292-9824 #292-9840 #292-9843 #293-9859 #293-9863 #293-9868 #293-9872 #293-9873 #293-9876 #293-9879 #293-9880 #293-9887 #293-9892 #312-10270 #329-10703 #333-10762 #333-10763 #333-10765 #333-10766 #333-10773 #335-10797 #335-10798 #335-10800 #335-10801 #335-10808 #339-10849 #339-10851 #339-10852 #348-11101 #348-11115 #348-11116 #348-11119 #348-11121 #348-11134 #348-11138 #348-11139 #348-11140 #348-11141 #348-11142 #348-11150 #348-11151 #348-11157 #348-11159 #348-11177 #356-11394 #356-11431 #362-11511 #362-11527 #365-11587 #365-11625 #367-11637 #367-11647 #375-11799 #375-11809 #375-11818 #375-11851 #375-11858 #378-11928 #378-11946 #378-11947 #380-11977 #380-11983 #380-11984 #380-11985 #382-11999 #382-12016 #382-12017 #385-12088 #385-12092 #385-12093 #385-12108 #406-12698 #406-12703 #406-12734 #408-12744 #411-12824 #411-12826 #411-12827 #413-12892 #413-12894 #413-12895 #419-13117 #439-14072 #439-14084 #443-14133 #443-14142 #443-14145 #443 14148 #443-14151 #445-14181 #447-14214 #447-14237 #447-14240 #450-14317 #450-14318 #450-14347 #450-14351 #450-14359 #450-14360 #451-14370 #451-14381 #451-14387 #451-14388 #451-14391 #455-14483 #463-14681 #463-14683 #463 14684 #463 14697 #463-14712 #463-14715 #463 14727 #463-14730 #470-15016 #470-15037
ENDPRO	#18-5000
FATAL	#115-4429 #164-6154 #164-6158 #300-9986 #322-10571 #324-10596 #324-10600 #324-10603
FI	#18 3600
FOR	#10-100 #108-3985 149-5428 169-6338 179-6810 181-6957 181-6984 181-6991 183-7033 188-7174 208-7539 208-7540 208-7548 208-7559 227-8127 236-8389 236-8390 236-8418 236-8419 238 8461 292 9817 292 9825 293 9885 312-10246 #333 10755 #335-10790 #339-10845 #348-11103 348 11126 378 11941

MACRO CROSS REFERENCE CREF V01

MACRO NAME	REFERENCES									
	380-11970	382 12011	385 12106	#411 12819	#413-12887	#463-14675	463 14692	463 14707	463 14722	
GOTO	#13-2000	#108-3986	#228-8154	#375-11798	#375-11817	#378-11949	#382 12019	#450 14358		
IF	#4 100	#108-3984	151-5442	151-5457	#153-5483	153-5485	154 5496	155 5597	155 5598	164 6163
	166-6277	166-6281	167-6302	167-6303	169-6342	169-6343	169 6344	169 6345	169-6355	169 6357
	170-6362	170-6378	179-6835	179-6840	#179-6843	179-6849	179-6854	#181-6958	181-6959	181-6960
	181-6961	#181-6967	#181-6974	181-6975	#181-6992	181-7000	181 7024	183-7030	183-7037	183-7059
	184-7095	186-7127	186-7158	186-7159	188-7176	#206 7495	206-7496	208-7512	#208-7513	208-7545
	208-7576	210-7642	210-7648	210 7667	210-7673	214-7730	214-7739	214-7745	216-7794	219 7901
	223 8003	223-8004	223-8011	223-8012	223-8016	223-8025	223-8026	223-8034	223 8035	223 8037
	225-8045	225-8046	225-8053	225-8054	227-8068	227-8069	227-8077	#227-8079	227-8080	227 8086
	227-8087	227-8109	227-8121	227-8131	228-8153	232-8320	232-8321	236-8392	236-8399	236-8405
	236-8414	236-8421	236-8428	236-8434	238-8463	238-8464	238-8471	244-8627	244-8660	246-8673
	266-9081	266-9083	266-9122	266-9123	268-9219	268-9220	272-9288	272-9289	274-9329	274-9396
	274-9397	276-9495	276-9496	280-9597	280-9598	292-9818	#292-9832	#292-9833	#292-9834	292-9846
	#293-9853	#293-9854	#293-9855	293-9857	293-9861	293-9866	293-9870	316-10345	320-10454	320-10492
	320-10515	328-10655	329-10699	333-10759	333-10764	335-10794	335-10799	339-10850	348-11098	#348-11099
	348-11100	348-11105	348-11117	#348-11127	348-11128	348-11129	348-11130	348-11131	348-11135	348-11143
	348-11144	348-11153	356-11392	356-11429	360-11484	362-11506	362-11524	#362-11525	365-11583	367-11632
	367-11641	373-11745	373-11747	373-11748	375-11791	375-11796	375-11804	375-11805	375-11815	375-11847
	375-11852	378-11924	378-11943	380-11968	380-11974	382-11995	382-12013	385-12086	406-12694	406-12699
	406-12707	406-12730	408-12740	410-12795	411-12825	413-12893	419-13113	439-14065	#439-14066	439-14068
	439-14077	#439-14078	439-14080	441-14089	443-14129	443-14138	443-14140	443-14143	443-14146	443-14149
	445-14156	445-14165	445-14176	#445-14177	447-14211	447-14234	447-14238	447-14241	450-14303	450-14315
	450-14340	450-14341	450-14349	450-14357	451-14363	451-14368	451-14374	451-14378	451-14382	455-14481
	463-14682	463-14706	463-14721	470-15014	470-15033					
IFB	#18-1100	#108-3987	#153-5472	#153-5475	#153-5478	#183 7034	#183-7050	#183 7053	#380-11973	
JUMPTO	#13-1500	#108-3986								
LEAVE	#13-100	#108-3986	#179-6852	#179-6855	#179-6862	#208-7571	#227-8083	#227-8090	#333-10761	#335-10796
	#348-11114	#348-11133	#385-12087							
LET	#16-100	#108-3986	#181-6962	#181-6964	#181-6968	#181-6970	#181-6976	#181-6993	#208-7562	#236-8394
	#236-8422	#292-9819	#292-9820	#292-9822	#292-9823	#292-9829	#292-9830	#292-9835	#292-9836	#292-9838
	#292-9839	#292-9841	#292-9842	#292-9848	#292-9849	#293-9856	#293-9860	#293-9865	#293-9869	#293-9874
	#293-9875	#293-9888	#293-9891	#348-11164	#350-11224	#406-12693	#406-12696	#406-12700	#406-12702	#406-12719
	#408-12738	#408-12739	#408-12742	#436-13997	#436-13998	#436-13999	#436-14002	#436-14003	#436-14004	#436-14007
	#436-14008	#436-14009	#438-14013	#438-14014	#438-14015	#438-14018	#438-14019	#438-14020	#438-14023	#438-14024
	#438-14025	#438-14028	#438-14029	#438-14030	#438-14033	#438-14034	#438-14035	#438-14038	#438-14039	#438-14040
	#438-14043	#438-14044	#438-14045	#438-14048	#438-14049	#438-14050	#438-14053	#438-14054	#438-14055	#438-14058
	#438-14059	#438-14060	#441-14095	#441-14096	#441-14100					
MAP	#128-4788	155-5614	166-6200	170-6376	171-6387	#208-7541	240-8498	242-8527	244 8591	248-8681
	250-8714	362-11541	371-11712	373-11751	375-11812	375 11849	375-11904	378-11955	382-12025	415-12955
	417-12996									
NEWTST	#117-4479	157-5618	164-6133	166-6170	169-6318	[illegible]-7075	184-7092			
ON.ERR	#18-3100	#108-3987	#171-6437	#190-7211	#192-7241	#194-7278	#196-7315	#198-7353	#200-7398	#202-7436
	#204-7481	#231-8261	#231-8269	#232-8279	#238-8475	#240-8494	#242-8523	#244-8587	#333-10757	#333-10769
	#335-10792	#335-10804	#339-10847	#348-11175	#411-12835	#413-12889	#463-14677			
ON.NOE	#18-2600	#108-3987	149-5430	171-6404	171-6414	171-6424	208-7556			
OR	#6-100	#108-3984								
ORB	#18-100	#108-3987								
POP	#14 900	#108-3986	155-5614	163-5985	163-6074	166-6200	166-6276	170-6376	171-6387	171-6450
	171-6452	179-6903	186-7139	208-7541	208-7567	208-7595	210-7679	210-7687	227-8138	227-8140
	240-8498	242-8527	244-8591	248-8681	248 8710	250-8714	250-8747	313-10307	316-10376	328-10666
	328-10674	329-10698	333 10768	335-10803	339-10853	339-10862	340 10900	340 10916	342-10947	346-11072

MACRO CROSS REFERENCE CREF V01

MACRO NAME	REFERENCES									
	346-11093	350 11219	350 11245	350-11257	354-11339	354-11350	356 11441	362 11541	364 11573	365 11596
	#367-11636	367 11638	367-11646	369-11677	371-11697	371-11712	371 11727	373 11739	373-11751	373-11770
	373 11779	375 11790	375-11803	375 11808	375-11812	375 11849	375-11857	375 11900	375-11902	375-11903
	375-11904	378-11954	378-11955	380 11986	382-12024	382-12025	384 12035	384-12038	385-12085	391-12133
	406-12729	408-12758	413-12865	413-12869	413-12875	413 12881	413-12882	413-12890	414-12908	414 12912
	414-12919	414 12924	414 12929	414-12935	415-12955	415-12975	417 12996	417-13009	419-13130	419-13133
	419-13134	441 14108	443-14125	448-14284	457-14518	457-14531	458-14556	461-14660	463 14685	463-14704
	463-14719	463 14733	467-14896	470 15024	470-15028	472-15185	472 15243	473-15288	474-15317	
PROC1	#18-4100									
P SM	#14-100	#108-3986	#155-5584	#155-5590	#155-5614	#163-5965	#163-6062	#166-6200	#166-6265	#170-6376
	#171-6387	#171-6391	#171-6393	#179-6871	#186-7137	#208-7514	#208-7541	#208-7563	#210-7622	#227-8126
	#227-8128	#240-8498	#242-8527	#244-8591	#248-8681	#248-8683	#250-8714	#250-8716	#313-10284	#316-10331
	#328-10653	#328-10671	#329-10678	#333-10751	#335-10786	#339-10842	#339-10857	#340-10866	#340-10891	#342-10928
	#346-11000	#346-11075	#350-11205	#350-11210	#350-11249	#354-11330	#354-11344	#356-11376	#362-11541	#364-11557
	#364-11562	#364-11566	#365-11590	#365-11591	#371-11691	#371-11712	#373-11733	#373-11751	#375-11785	#375-11812
	#375-11849	#375-11855	#375-11904	#378-11920	#378-11955	#380-11961	#382-11991	#382-12025	#384-12032	#406-12708
	#408-12748	#410-12778	#410-12782	#410 12786	#410-12793	#411-12809	#411-12815	#411-12823	#411-12829	#411-12832
	#411-12836	#411-12841	#411 12843	#415-12953	#415-12955	#417-12994	#417-12996	#419 13069	#419 13073	#419 13074
	#441-14104	#443-14114	#448 14274	#457-14511	#457-14525	#458-14549	#461 14636	#461-14652	#463 14670	#463 14688
	#467-14860	#470-15003	#472 15161	#472-15212	#473-15273	#474 15302				
RCC	#3-4000	#108-3983								
RCS	#3-3700	#108-3983								
REPEAT	#11-100	#108-3985	#385-12082							
REQ	#3-400	#108-3983								
RES4	#133-4900	145-5357	151-5461	161 5891	164-6160	166-6220	367 11658	367 11662	369 11674	369-11686
	371-11728	373-11780								
RGE	#3-1000	#108-3983								
RGT	#3-1300	#108-3983								
RMI	#3-2500	#108-3983								
RMIS	#3-3100	#108-3983								
RLE	#3-1600	#108-3983								
RLO	#3-3400	#108-3983								
RLOS	#3-2800	#108-3983								
RLT	#3-700	#108-3983								
RMI	#3-2200	#108-3983								
RNE	#3-100	#108-3983								
RPL	#3-1900	#108-3983								
SET	#19-700	#108-3982	137-4985	147-5361	147-5363	153-5473	153-5476	153-5479	153-5484	153-5486
	#153-5488	163-6039	163-6085	166-6260	166-6279	177-6799	179-6866	179-6948	184-7096	184-7098
	188-7177	188-7180	206-7489	206-7490	206-7504	208-7560	208-7570	208-7591	210-7643	210-7668
	230-8213	231-8272	232-8280	236-8388	238-8449	238-8460	238-8472	240-8495	242-8524	244-8588
	246-8669	266-9098	266-9101	266-9111	266-9114	268-9187	268-9190	268-9201	268-9204	270-9275
	270-9278	302-10045	302-10057	322-10561	333-10760	335-10795	348-11136	348-11199	356-11378	356-11387
	356-11406	356-11410	356-11413	356-11416	356-11419	356-11425	356-11430	356-11434	356-11437	356-11440
	365-11593	367-11656	369-11672	369-11681	375-11839	378-11939	380-11976	382-12009	384-12043	384-12049
	385-12095	443-14124	443-14135	443-14152	445-14160	445-14162	445-14171	445-14173	445-14195	445-14197
	#451-14390	457-14514	457-14528	458-14552	461-14637	461-14649	461-14653	461-14655		
SET8	#19-1900	#108-3982								
SET4	#133-4888	#145-5313	#145-5315	#145-5322	#145-5341	#145-5353	#151-5454	#154-5504	#154-5506	#154-5508
	#157-5645	#160-5797	#160-5813	#160-5828	#160-5856	#161-5873	#163-5925	#164-6144	#166-6181	#360-11483
	#367-11654	#369-11669	#371-11711	#373-11750						
SMACIT	#1 300	#108-3981	108-3996							

MACRO CROSS REFERENCE
CREF V01

MACRO NAME	REFERENCES									
SUBTST	#119 4544	#144 5272	#144 5290	#145 5310	#147 5360	#147 5374	#149 5398	#149 5427	#151 5438	#151 5450
	#153-5468	#154 5493	#154-5500	#155 5575	#155-5593	#155 5606	#161 5866	#163 5894	#164-6162	#167 6295
	#179 6802	#181-6952	#183-7029	#184 7106	#188 7172	#190 7188	#192 7218	#194 7248	#196-7285	#198 7322
	#200-7360	#202 7405	#204 7443	#206 7488	#208-7508	#210 7597	#212 7692	#214-7726	#216 7784	#217-7837
	#217-7850	#217-7872	#219-7900	#219 7936	#219-7954	#221 7976	#221-7983	#221-7993	#223-8002	#223-8010
	#223-8024	#223-8033	#225 8044	#225-8052	#225-8060	#227-8067	#230 8212	#231-8259	#231-8267	#232-8277
	#232-8319	#234 8329	#236-8376	#238-8447	#240-8492	#242-8521	#244-8585	#244-8615	#244-8645	#246-8667
	#246-8672	#248-8680	#252-8761	#252-8772	#252 8784	#254-8798	#254-8821	#256-8839	#256-8847	#258-8857
	#258-8870	#258-8879	#260 8893	#262-8936	#264 8976	#266-9012	#268-9153	#270-9235	#274-9325	#276 9433
	#278-9532	#282-9634	#284-9667	#288-9741	#292-9811	#293 9883	#296-9906	#298-9950	#298-9958	#300-9968
	#304-10062	#304-10069	#304-10087	#304-10100	#306-10108	#306-10114	#308-10165	#310-10185	#311 10231	#312-10243
	#313-10277	#313-10283	#314-10310	#316-10330	#318-10382	#324 10611	#339-10841	#339-10856	#340-10865	#342-10920
	#346-10989	#348-11097	#350-11203	#350-11248	#352-11261	#354-11309	#354-11329	#354-11342	#356-11354	#358-11445
	#358-11456	#358-11464	#358-11468	#360-11475	#362-11504	#362-11513	#362-11540	#364-11550	#365-11579	#367-11629
	#367-11651	#369-11666	#371-11690	#373-11732	#375-11784	#376-11913	#378-11919	#380-11960	#382-11990	#384-12031
	#384-12041	#384-12046	#384-12052	#384-12059	#385-12078	#385-12098	#387-12112	#389-12118	#391-12124	#406-12706
	#408-12747	#415-12952	#417-12979	#439 14063	#439-14075	#441-14088	#441-14103	#443-14112	#448-14271	#461 14630
	#470-15002	#470-15027								
SUPERV	#130-4817	#248-8703	#411-12812	#414-12909	#415-12956	#417-12997				
TESTAR	#131-4860	166-6210	166-6227	166-6244	166-6270	171-6390	210-7625	240-8499	242-8547	244-8593
	250-8736	329-10707	362-11542	371-11713	373-11753	373-11771	406-12695	406-12711	406-12721	408 12741
	408-12753									
THEN	#6-6900	#108-3984								
THRU	#11-4400	#108-3985								
TO	#10-2900	#108-3985								
TYPDEC	#125-4689	181-7004	181-7008	181-7012	181-7016	181-7020	181-7022	186-7131	380-11964	380-11966
	380-11981	457-14505								
TYPE	#115-4443	#154-5497	#154-5510	#154-5512	#154-5514	#154-5521	#154-5526	#154-5527	#160-5863	#179-6864
	#179-6947	#181-7001	#181-7005	#181-7009	#181-7013	#181-7017	#181-7021	#181-7023	#186-7126	#186-7128
	#208-7577	#208-7579	#316-10339	#316-10340	#316-10341	#316-10349	#316-10353	#316-10354	#316-10355	#316-10356
	#316-10357	#316-10358	#316-10367	#316-10368	#316-10369	#316-10370	#318-10395	#318-10398	#318-10400	#318-10409
	#318-10424	#320-10434	#320-10453	#320-10464	#320-10491	#320-10502	#320-10514	#320-10524	#320-10532	#320-10534
	#348-11118	#348-11148	#348-11156	#365-11581	#365-11584	#365-11594	#365-11599	#367-11630	#367-11660	#369-11671
	#369-11675	#369-11680	#369-11684	#371-11693	#371-11694	#371-11719	#371-11723	#373-11735	#373-11736	#373-11759
	#373-11763	#373-11766	#373-11768	#373-11775	#375-11787	#375-11788	#375-11797	#375-11801	#375-11806	#375-11816
	#375-11819	#378-11922	#378-11929	#378-11931	#378-11948	#380-11965	#380-11967	#380-11975	#380-11982	#382-11993
	#382-12000	#382-12002	#382-12018	#384-12033	#384-12042	#384-12047	#384-12053	#384-12060	#385-12079	#385-12083
	#385-12094	#385-12099	#387-12113	#389-12119	#391-12131	#414-12931	#419-13054	#450-14312	#450-14313	#451-14383
	#453-14411	#453-14435	#453-14436	#453-14438	#453-14445	#453-14447	#455-14474	#455-14484	#457-14515	#457-14519
	#458-14553	#458-14562	#458-14577	#458-14583	#458-14585	#463-14671	#463-14672	#463-14680	#463-14689	#463-14693
	#463-14695	#463-14705	#463-14708	#463-14710	#463-14714	#463-14720	#463-14723	#463-14725	#463-14729	#463-14731
	#466-14832	#467-14897	#468-14942	#468-14951	#468-14952	#468-14954	#468-14963	#468-14968	#468-14977	#468-14995
	#470-15004	#470-15015	#470-15017	#470-15019	#471-15085	#471-15098	#471-15104	#471-15109	#471-15113	#471-15118
	#471-15119	#471-15121	#471-15124	#471-15128	#472-15189	#472-15191	#472-15192	#472-15193	#472-15248	#472 15250
	#472-15251	#472-15252	#479-15470							
TYPOCS	#123-4636	179-6865	208-7578	371-11716	373-11767	373 11776	380-11978	457-14537	457-14543	458-14584
	470-15018	470-15020								
TYPOCT	#121-4590	#453-14416	#457-14499	#463-14679	#463-14694	#463-14696	#463 14709	#463-14711	#463 14724	#463 14726
	#468-14953									
UNTIL	#11-2900	#108-3985								
UNTILB	#18-1600	#108-3987								
USER	#130-4838	348-11167	350 11226	352-11285	352 11301	411 12806	414-12913			

```
MACRO CROSS REFERENCE                                         CREF    V01
MACRO NAME     REFERENCE
WHILE          #8 100      #108 3984   292 9831    293 9852    293 9889
WHILEB         #18 2100    #108 3987
$CALL          #15-100     #108-3988
$RETUR         #15 3800    #108 3988   210 7680    210 7688    246 8674    #246 8676   348 11120   348 11158   348 11200
$SUBTS         #119-4551   144-5272    144 5290    145 5310    147 5360    147 5374    149 5398    149 5427    151 5438    151 5450
                153-5468   154-5493    154 5500    155-5575    155-5593    155 5606    161 5866    163 5894    164 6162    167-6295
                179-6802   181 6952    183-7029    184-7106    188 7172    190 7188    192-7218    194 7248    196-7285    198-7322
                200-7360   202-7405    204-7443    206 7488    208 7508    210-7597    212 7692    214 7726    216 7784    217 7837
                217 7850   217-7872    219-7900    219-7936    219 7954    221 7976    221 7983    221 7993    223-8002    223 8010
                223-8024   223-8033    225-8044    225-8052    225-8060    227-8067    230-8212    231-8259    231 8267    232 8277
                232-8319   234-8329    236-8376    238-8447    240-8492    242-8521    244-8585    244-8615    244-8645    246-8667
                246-8672   248-8680    252-8761    252-8772    252-8784    254-8798    254-8821    256-8839    256-8847    258-8857
                258-8870   258-8879    260-8893    262-8936    264-8976    266 9012    268-9153    270-9235    274-9325    276-9433
                278-9532   282-9634    284-9667    288 9741    292-9811    293 9883    296 9906    298-9950    298-9958    300-9968
                304-10062  304-10069   304-10087   304-10100   306-10108   306 10114   308-10165   310-10185   311 10231   312-10243
                313-10277  313-10283   314-10310   316-10330   318-10382   324-10611   339-10841   339-10856   340-10865   342-10920
                346-10989  348 11097   350-11203   350-11248   352-11261   354-11309   354-11329   354 11342   356-11354   358-11445
                358-11456  358-11464   358 11468   360-11475   362-11504   362-11513   362-11540   364-11550   365 11579   367-11629
                367-11651  369-11666   371-11690   373 11732   375-11784   376 11913   378 11919   380-11960   382-11990   384 12031
                384-12041  384-12046   384-12052   384-12059   385-12078   385 12098   387 12112   389 12118   391 12124   406 12706
                408-12747  415-12952   417-12979   439-14063   439-14075   441 14088   441-14103   443 14112   448 14271   461 14630
                470-15002  470-15027
$$END          #12-7400    #108-3986   503-16203
$$NEWT         #119-4511   157-5618    164 6133    166-6170    169 6318    184 7075    184 7092
.ARITH         #17-7700    #108-3987
.EMIT          #1-4600     #108-3981
.EMITL         #1-6800     #108-3981
.EMITN         #1-5800     #108-3981
.EMITR         #1 7300     #108-3981
.GENBR         #2-5100     #108-3982
.GOTO          #5-4500     #108-3984
.IFARI         #5-1500     #108-3984
.IFOPR         #2 100      #108-3982
.IS            #2-3300     #108-3982
.LEAVE         #5-3100     #108-3984
.OPADD         #2 7800     #108-3982
.OPSUB         #2 9300     #108-3982
.OR            #5 100      #108-3984
.SIMPL         #17 100     #108-3987
.SUPER         #19-2500
 .BRAN         #1 3500     #108-3981
 .POP          #1 2200     #108-3981
 .PUSH         #1-1500     #108-3981
..TAG          #1 3200     #108-3981
```