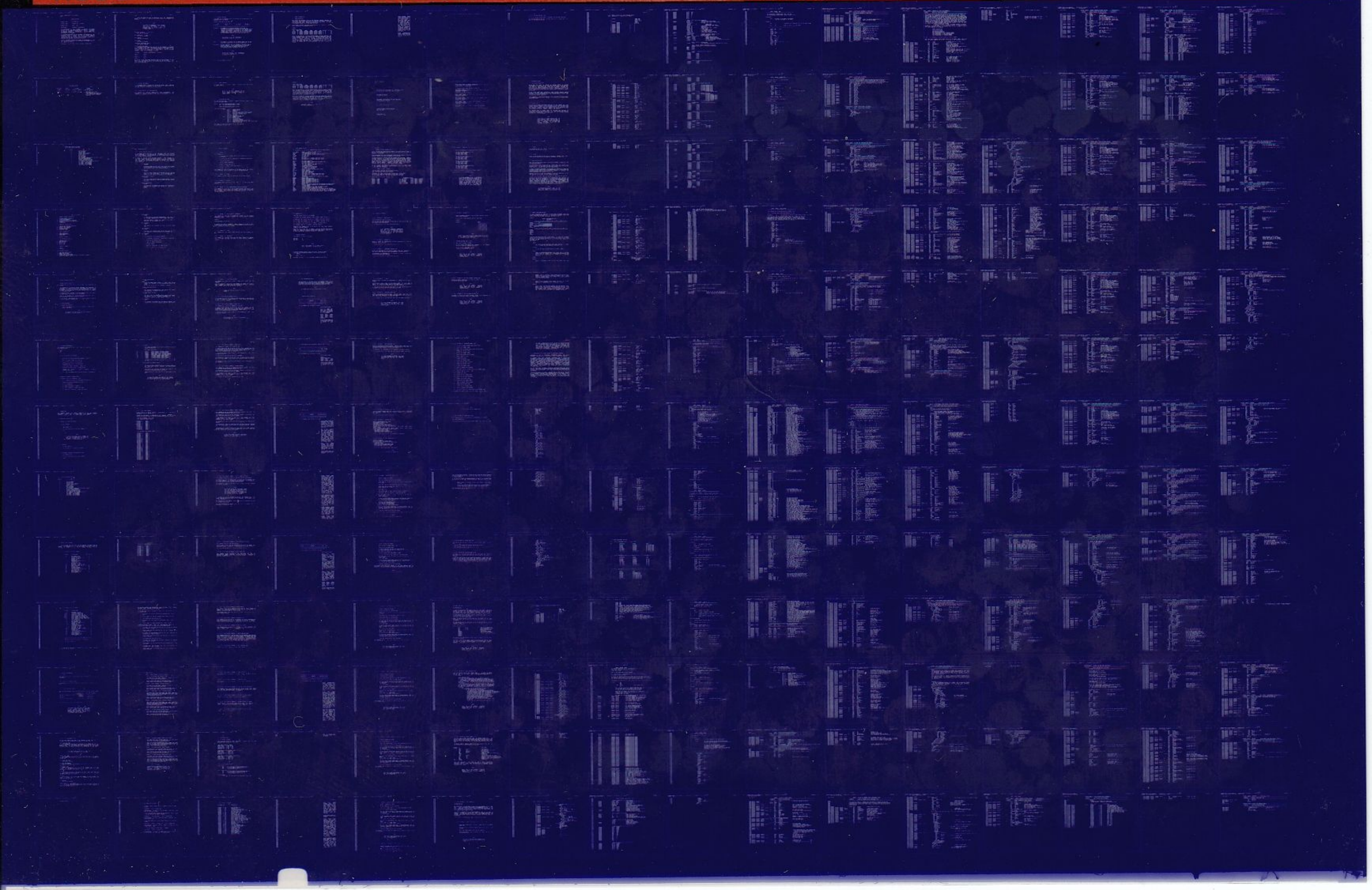
MS11-L*
MS11-M*

MS11L/M DIAG
CZMSDC0

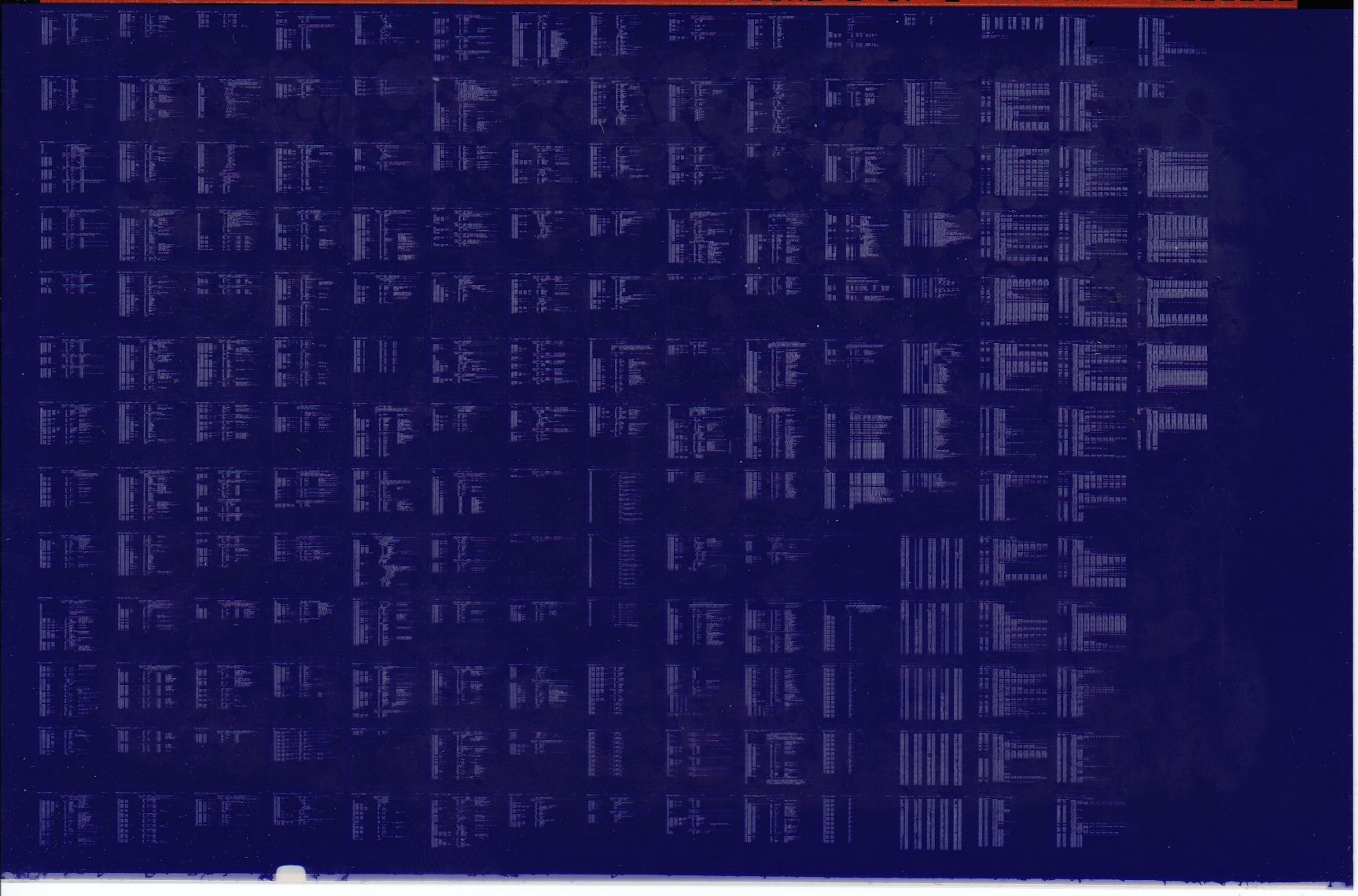
AH-F295C-MC
FICHE 1 OF 2



MS11-L*
MS11-M*

MS11L/M DIAG
CZMSDCO

AH-F295C-MC
FICHE 2 OF 2



```
                        .TITLE  CZMSDC0 MS11-L/M DIAGNOSTIC
                        .REM
```

IDENTIFICATION

PRODUCT CODE: AC-F294C-MC

PRODUCT NAME: CZMSDC0 MS11-L/M DIAG

PRODUCT DATE: APRIL, 1981

MAINTAINER: DIAGNOSTIC ENGINEERING

REVISION HISTORY

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

```
                        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 DIAGNOSTICS (WHERE APPLICABLE)

1.5 ASSUMPTIONS

THIS PROGRAM ASSUMES THE CORRECT OPERATION OF THE CPU, MEMORY MANAGEMENT, CACHE, AND THE UNIBUS MAP. THIS PROGRAM OCCUPIES (INITIALLY) BANK 0 (0-16K). THE XXDP LOADERS ARE IN BANK 1.

2.0 OPERATING INSTRUCTIONS

2.1 LOADING & STARTING PROCEDURES

2.1.1 QUICK STARTING -

1. LOAD ADDRESS 200
2. SET SWITCH REGISTER FOR OPTIONS (NORMALLY 0)
3. START

NOTE

IF ON AN 11/24 USING MS11-L MEMORY BE SURE THAT THE PERIPHERAL PAGE JUMPER IS IN PLACE; FAILURE TO DO SO SENDS THE DIAGNOSTIC TO NEVER-NEVER LAND.

2.1.2 STOPPING -

1. SET SW8, AND/OR
2. TYPE CONTROL "C" (REFERENCE SECTION 2.4.4.1).

2.1.3 RESTARTING (PRESERVE CONFIGURATION TABLE) -

1. LOAD ADDRESS 202
2. SET SWITCH REGISTER FOR OPTIONS (NORMALLY 0)
3. START

2.1.4 SWITCH REGISTER OPTIONS -

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

2.2 DEFAULT TEST SEQUENCE

THE FOLLOWING TWO LISTS GIVE THE TEST PROTOCOL FOR PARITY AND ECC MEMORY. TESTS MARKED WITH A "*" ARE NOT NORMALLY RUN EXCEPT UNDER ACT OR APT, OR THROUGH A FIELD SERVICE COMMAND (REFERENCE SECTION 2.4.4.8).

2.2.1 TEST PROTOCOL FOR PARITY MEMORY -

	PATTERN	PATTERN NAME	TIME (SEC/16K)
	34	SOFT ERROR TEST	<1
	6	INITIAL DATA TEST	<1
	17	HOLDING 1'S AND 0'S TEST	<1
	7	ADDRESS BIT TEST	<1
	1	ADDRESS TEST	<1
	2	COMPLEMENT ADDRESS TEST	<1
	3	3 XOR 9 TEST	1
	4	ROTATING 0'S TEST	1
	5	ROTATING 1'S TEST	1
	21	MARCHING 1'S AND 0'S TEST	1
	35	WORST CASE NOISE PARITY TEST	N/A
*	22	REFRESH TEST	10
*	23	SHIFTING DIAGONAL TEST	10
	26	RANDOM DATA TEST	<1
*	24	FAST GALLOPING PATTERN TEST	20
*	31	SOB-A-LONG TEST	3
*	32	WRITE RECOVERY TEST	<1
*	33	BRANCH GOBBLE TEST	35
	34	SOFT ERROR TEST	<1

2.2.2 TEST PROTOCOL FOR ECC MEMORY -

PATTERN	PATTERN NAME	TIME (SEC/16K)
5	ROTATING 1'S TEST	1
@ 25	INTERRUPT ENABLE TEST	<1
+@ 11	SINGLE BIT ERROR TEST	<2
+@ 12	WRITE BYTE CLEARS SBE TEST	<1
+@ 13	CREATE DOUBLE BIT ERROR TEST	1
%+@ 14	WRITE INHIBIT DATIP W/DBE TEST	1
+@ 15	WRITE INHIBIT OF BYTE W/DBE TEST	1
+@ 16	WRITE INHIBIT OF WORD W/DBE TEST	<1
34	SOFT ERROR TEST	<1
6	INITIAL DATA TEST	<1
10	BYTE ADDRESS TEST	<1
17	HOLDING 1'S AND 0'S TEST	<1
7	ADDRESS BIT TEST	<1
1	ADDRESS TEST	<1
2	COMPLEMENT ADDRESS TEST	<1
4	ROTATING 0'S TEST	1
5	ROTATING 1'S TEST	1
21	MARCHING 0'S AND 1'S TEST	1
@ 20	MARCHIN 0'S AND 1'S IN CB'S TEST	<1
* 22	REFRESH TEST	10
26	RANDOM DATA TEST	<1
* 24	FAST GALLOPING PATTERN TEST	20
* 31	SOB-A-LONG TEST	3
* 32	WRITE RECOVERY TEST	<1
* 33	BRANCH GOBBLE TEST	35
34	SOFT ERROR TEST	<1

@ - RUN ONLY ON THE FIRST PASS WHEN UNDER ACT OR APT

+ - RUN TWICE FOR EACH 16K BANK IF INTERLEAVED

% - RUN ONLY FOR MF11S-K

AT THE END OF EACH PASS THE PROGRAM WILL RUN CLEANUP PATTERNS #30, AND #27 FOR ALL BANKS.

2.3 SPECIAL ENVIRONMENTS

2.3.1 XXDP -

THE FIRST PASS WILL BE A QUICK VERIFY PASS IF AND ONLY IF IT IS IN CHAIN MODE.

2.3.2 ACT & APT AUTOMATIC MODE -

THE PROGRAM WILL NOT CREATE DOUBLE BIT ERRORS (DBE'S) AFTER THE 1ST PASS.

2.3.2.1 APT EXECUTION TIMES -

HERE ARE SOME MEASURED EXECUTION TIMES FOR AN 11/44 WITH CACHE UNDER APT

	1ST QV PASS	2ND PASS & ONWARD
128K MS11-M (NON-INTERLEAVED)	10 MIN 15 SEC	7 MIN 40 SEC
128K MS11-L	9 MIN 50 SEC	7 MIN 30 SEC
256K MS11-M (INTERLEAVED)	19 MIN 50 SEC	14 MIN 45 SEC

THE 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 #24 AND #33 FOR EXAMPLE) CAN BE EXTREMEMLY TIME CONSUMING.

2.3.2.2 APT ENVIRONMENT TABLE -

THE FOLLOWING TABLE GIVES SOME OF THE STANDARD SETTINGS FOR THE APT E-TABLE. THEY MAY BE MODIFIED AS NOTED AS THE USER SEES FIT.

FIRST PASS RUN TIME:
THIS PARAMETER SHOULD BE SET ACCORDING TO THE AMOUNT AND TYPE OF MEMORY TO BE TESTED. THE ABOVE TABLE (APT EXECUTION TIMES) GIVES SOME MEASURED TIMES. FOR ANY PATTERNS DELETED (THROUGH USE OF THE DEVICE DESCRIPTOR WORDS) REFERENCE SECTION 2.2 FOR INDIVIDUAL PATTERN TIMES.

NOTE

THE TIMES GIVEN IN SECTION 2.2 ARE FOR
16K CHUNKS OF MEMORY, NOT 128K BOARDS!

LONGEST TEST TIME:
THIS PARAMETER SHOULD BE SET TO THE EXECUTION TIME OF THE LONGEST PATTERN BEING RUN. FOR THE DEFAULT CASE THIS IS 35 SECONDS FOR PATTERN #33.

ADDITIONAL RUN TIME:
NOT USED BY PROGRAM.

SOFTWARE ENVIRONMENT:
FOR APT AUTO MODE THIS PARAMETER SHOULD BE SET TO A "1". FOR DUMP MODE SET THIS TO A "0".

ENVIRONMENT MODE:
WHEN THIS PARAMETER IS SET TO A "0" THE PROGRAM DOES IT'S OWN SIZING. IF THE USERS SETS BIT #7 HOWEVER, HE MUST SPECIFY THE TYPES AND AMOUNTS OF MEMORY TO BE TESTED.

SWITCH 1:
THE DEFAULT SETTING OF THIS SWITCH IS "101". APT USES THIS AS THE SWITCH REGISTER FOR THE PROGRAM. REFERENCE SECTION 2.4.1 FOR MORE INFORMATION ON SWITCH SETTINGS.

SWITCH 2:
THIS SWITCH, IF SET TO ANY NON-ZERO NUMBER, IS USED TO LIMIT THE AMOUNT OF PASSES APT WILL MAKE. THE PROGRAM WILL HANG AFTER THIS COUNT HAS BEEN REACHED.

CPU OPTIONS:
NOT USED BY PROGRAM.

MEMORY TYPE N (N=1 TO 4)
IF BIT #7 OF ENVIRONMENT MODE IS SET THESE FOUR WORDS ARE USED TO LOG THE DIFFERENT TYPES OF MEMORY TO BE TESTED. IF BIT #7 IS NOT SET THESE LOCATION ARE NOT USED.

```
    501                             MAXIMUM ADDRESS N       (N=1 TO 4)
```

THESE FOUR WORDS ARE USED IN CONJUNTION WITH THE CORRESPONDING MEMORY TYPE WORDS TO INDICATE THE HIGHEST ADDRESS THAT MEMORY TYPE OCCUPIES.

NOTE

THE ABOVE TWO PARAMETERS DO NOT ACTUALLY HAVE TO REPRESENT AN ACCURATE CONFIGURATION OF MEMORY. ALL THE PROGRAM LOOKS FOR IS AN ACCURATE TALLY OF MEMORY AMOUNT!

INTERRUPT VECTOR N (N=1 TO 2)
NOT USED BY PROGRAM.

BUS PRIORITY N (N=1 TO 2)
NOT USED BY PROGRAM.

BASE ADDRESS:
NOT USED BY PROGRAM.

DEVICE MAP:
NOT USED BY PROGRAM.

CONTROLLER DESCRIPTOR CODE N (N=1 TO 2)
NOT USED BY PROGRAM.

DEVICE DESCRIPTOR CODES:
THE DEVICE DESCRIPTOR CODES ARE USED BY THE PROGRAM TO DETERMINE WHICH PATTERNS IT WILL RUN. THE DEFAULT VALUES OF THESE WORDS ARE ALL "1"'S, INDICATING THAT ALL OF THE PATTERNS SHOWN IN SECTION 2.2 ARE EXECUTED (SAVE FOR EXCEPTIONS AS NOTED THERE). EACH SET OF WORDS CONTROLS A TABLE IN THE PROGRAM AS FOLLOWS:

DD WORDS	PROGRAM TABLE (SYMBOLIC LOCATION)
WORDS 0-1	MKCSRT
WORDS 2-3	MKPAT
WORDS 4-5	MJPAT

BIT #0 SET IN THE FIRST WORD INDICATES THAT THE FIRST PATTERN IN THE TABLE WILL BE EXECUTED, BIT #1 THE SECOND, BIT #2 THE THIRD,... BIT #0 OF THE SECOND WORD INDICATES THAT THE 17TH ENTRY IN THE TABLE WILL BE EXECUTED, AND SO ON.

2.3.3 NO SBE FREE BANKS -

IF THE PROGRAM CANNOT FIND ANY SBE (SINGLE BIT ERROR) FREE LOCATIONS (IN NON-PROTECTED ECC MEMORY) IT WILL PRINT OUT AN ERROR MESSAGE AND CONTINUE TESTING BY-PASSING THE ECC LOGIC TESTS.

2.3.4 MIXED PARITY & ECC CONFIGURATIONS -

THE PROGRAM WILL FUNCTION NORMALLY IN MIXED ENVIRONMENTS. THE SEQUENCE OF TESTING MAY SEEM STRANGE DUE TO THE RECURSIVE TEST MODE ALGORITHM (REFERENCE SECTIONS 2.4.1.1, 2.4.1.2, & 2.4.1.3).

2.4 PROGRAM OPTIONS

2.4.1 SWITCH REGISTER DETAILS -

IF A HARDWARE SWITCH REGISTER IS NOT AVAILABLE THEN THE SOFTWARE SWITCH REGISTER IS IN LOCATION 176. IF UNDER APT IF BIT7 IS SET IN THE E-TABLE SYMBOLIC LOCATION "$ENVM" THE APT SOFTWARE SWITCH REGISTER WILL BE USED (LOCATION $SWREG).

TO CHANGE THE SOFTWARE SWITCH REGISTER CONTENTS: TYPE "CONTROL G". THIS WILL CAUSE DISPLAY THE CURRENT VALUE OF THE SWR AND PROMPT FOR THE OCTAL INPUT OF THE NEW SWR VALUE FROM THE TERMINAL. THIS ROUTINE WILL IGNORE YOU (NOT RESPOND TO CONTROL "G") IF YOU HAVE A HARDWARE SWITCH REGISTER.

SW15 = HALT ON ERROR
(100000)

CONTINUING FROM THIS HALT WILL FIRST CHECK FOR A CHANGE IN THE SOFTWARE SWITCH REGISTER ("CONTROL G" IN THE TTY INPUT BUFFER) THEN IT WILL CONTINUE TESTING.

SW14 = LOOP ON TEST
(40000)

THIS WILL CAUSE LOOPING ON THE PRESENT TEST OR PATTERN (BACK TO LAST SCOPE TRAP). IF IN A PATTERN THEN THE LOOPING WILL BE FOR AN ENTIRE BANK OF 16K ADDRESSES.

SW13 = INHIBIT ERROR TYPEOUTS
(20000)

THIS WILL CAUSE RETURNS FROM THE ERROR ROUTINE WITHOUT THE TYPED MESSAGES. OTHER ON ERROR FUNCTIONS ARE NOT AFFECTED.

SW12 = INHIBIT RELOCATION
(10000)

THIS PREVENTS THE PROGRAM FROM MOVING AND CONSEQUENTLY PREVENTS THE PROGRAM FROM TESTING AT LEAST 32K OF MEMORY.

SW11 = QUICK VERIFY
(4000)

IF THIS SWITCH IS SELECTED APPROXIMATELY ONE 64TH OF THE POSSIBLE COMBINATIONS OF SBE'S & DBE'S ARE TESTED.

EACH PASS COMPLETE TYPEOUT WILL INDICATE THIS MODE BY PRECEDING THE PASS NUMBER WITH "QV".

SW10 = BELL ON ERROR
(2000)

THIS CAUSES A BELL (OR BEEP OR CLICK) ON EACH ERROR TRAP

SW9 = LOOP ON ERROR
(1000)

THIS WILL CAUSE LOOPING FROM FAILURE POINT BACK TO THE LAST CORRECTLY INITIALIZED AREA OF THE CURRENT TEST.

SW8 = HALT PROGRAM
(400)

THIS INITIATES THE FOLLOWING SEQUENCE:

1. IF PROGRAM IS RELOCATED IT MOVES BACK TO BANK ZERO.
2. FLUSH OUT ALL POSSIBLE DBE'S.
3. TURNS OFF MEMORY MANAGEMENT.
4. RESTORE LOADERS.
5. UNMAP THE UNIBUS MAP (IF THERE IS ONE).
6. HALT IF UNDER APT OR ACT BRANCH SEL.

```
                        SW7     = DETAILED ERROR REPORTS
                                  (200)

                                  AFTER ANY NORMAL ERROR  REPORT  IS  TYPED  THIS  OPTION
                                  CAUSES  THE  CONTENTS  OF THE FOLLOWING REGISTERS TO BE
                                  TYPED:
                                  R0, R1, R2, R3, R4, R5, SP, ''CONTROL'', ''CPUERR''

                        SW6     = INHIBIT CONFIGURATION MAP
                                  (100)

                                  THIS INHIBITS THE PRINTING OF A MAP SHOWING THE  MEMORY
                                  CONFIGURATION - REFERENCE SECTION 7.3

                        SW5     = LIMIT MAX ERRORS PER BANK
                                  (40)

                                  THIS WILL LIMIT THE NUMBER OF ERROR TYPEOUTS PER  BANK.
                                  THE  DEFAULT  IS  10.   DECIMAL,  HOWEVER  THIS  CAN BE
                                  CHANGED BY CHANGING LOCATION ''ERRMAX'' MANUALLY.

                        SW4     = FAT TERMINAL
                                  (20)

                                  THIS INFORMS THE PROGRAM THAT THE CONSOLE TERMINAL  HAS
                                  A WIDTH OF AT LEAST 132 COLUMNS (LA36 WITH WIDE PAPER).
```

SW3-1 = TEST MODE

TEST MODES DETERMINE THE RECURSION ALGORITHM TO BE USED DURING PATTERN TESTS.

		MODE NAME	DESCRIPTION
(0)	0	BAFPAF	BANKS FORWARD, PATTERNS FORWARD
(2)	1	BAFPAR	BANKS FORWARD, PATTERNS REVERSE
(4)	2	BAWPAF	BANKS WORST FIRST, PATTERNS FORWARD.
(6)	3	BAWPAR	BANKS WORST FIRST, PATTERNS REVERSE.
(10)	4	PAFBAF	PATTERNS FORWARD, BANKS FORWARD
(12)	5	PAFBAW	PATTERNS FORWARD, BANKS WORST FIRST
(14)	6	PARBAF	PATTERNS REVERSE, BANKS FORWARD
(16)	7	PARBAW	PATTERNS REVERSE, BANKS WORST FIRST.

FOR MORE DETAILS REFERENCE SECTION 2.4.1.1, 2.4.1.2 AND 2.4.1.3.

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

FOR MANUFACTURING PURPOSES THIS SWITCH SHOULD ALWAYS BE ON. FOR FIELD SERVICE PURPOSES THIS SWITCH SHOULD ALWAYS BE OFF.

THIS SWITCH WILL ALLOW ALL ECC SINGLE BIT ERRORS TO BE REPORTED BY DISABLING ERROR CORRECTION.

ERROR PRINTOUTS OF SBE'S ARE NOT DISTINGUISHABLE FROM DBE'S.

NOTE

IF DOUBLE BIT ERRORS ARE FOUND IN THE MEMORY, THIS SWITCH SHOULD BE SET TO MAKE SURE THAT NEW DATA CAN BE WRITTEN TO THE DBE LOCATIONS.

2.4.1.1 TEST MODE EXAMPLE -

EXAMPLE ANALYSIS OF MODE 5 'PAFBAW'. ASSUME BANKS 0 & 1 ARE MS11-L AND BANKS 2,3,4,& 5 ARE MS11-M.

ASSUME ALSO THAT BANK 3 IS KNOWN BAD BY THE PROGRAM VIA THE SIZING ROUTINE OR PREVIOUS RUNS THE TESTING SEQUENCE WOULD BE AS FOLLOWS:

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

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

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

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

```
794
```

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

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


                                                NOTE

                                THIS IS  AN  EXAMPLE  &  NOT  AN  ACTUAL
                                SEQUENCE.
```

THE PATTERN SEQUENCE WAS FORWAPD (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 W-RE PROTECTED OR PROGRAM SPACE (FROM 0 TO 200).

9. PARITY BANKS THAT ARE PRESUMED GOOD AND WERE PROGRAM SPACE (FROM 0 TO 200).

2.4.1.2 TEST MODE DETAILS -

MODE 0 = 'BAFPAF'' BANKS FORWARD, PATTERNS FORWARD

THIS IS THE DEFAULT AND SIMPLEST MODE.

THIS MODE TESTS EACH BANK COMPLETELY FROM 0 TO 200 EXCEPT THOSE REQUIRING RELOCATION*.

WHILE TESTING EACH BANK THE PATTERNS ARE RUN WITH THE SIMPLE ONES FIRST BUILDING TO THE MORE COMPLEX.

MODE 1 = 'BAFPAR'' = BANKS FORWARD, PATTERNS REVERSE

THIS MODE TESTS EACH BANK COMPLETELY FROM 0 TO 200 EXCEPT THOSE REQUIRING RELOCATION*.

WHILE TESTING EACH BANK THE PATTERNS ARE RUN WITH THE MOST COMPLEX ONES FIRST, WORKING TO THE SIMPLE ONES.

MODE 2 = 'BAWPAF'' = BANKS WORST FIRST, PATTERNS FORWARD

THIS MODE FIRST TESTS EACH KNOWN BAD BANK COMPLETELY FROM 0 TO 200 EXCEPT THOSE REQUIRING RELOCATION*, THEN PRESUMED GOOD BANKS ARE TESTED FROM 0 TO 200 EXCEPT THOSE REQUIRING RELOCATION*.

WHILE TESTING EACH BANK THE PATTERNS ARE RUN WITH THE SIMPLE ONES FIRST, BUILDING TO THE MORE COMPLEX.

MODE 3 = 'BAWPAR'' = BANKS WORST FIRST, PATTERNS REVERSE

THIS MODE FIRST TESTS EACH KNOWN BAD BANK COMPLETELY FROM 0 TO 200 EXCEPT THOSE REQUIRING RELOCATION*, THEN PRESUMED GOOD BANKS ARE TESTED FROM 0 TO 200 EXCEPT THOSE REQUIRING RELOCATION*.

WHILE TESTING EACH BANK THE PATTERNS ARE RUN WITH THE MOST COMPLEX ONES FIRST, WORKING TO THE SIMPLE ONES.

MODE 4 = 'PAFBAF'' = PATTERNS FORWARD, BANKS FORWARD

THIS MODE TESTS EACH PATTERN COMPLETELY WITH THE SIMPLE ONES FIRST, BUILDING TO THE MORE COMPLEX.

WHILE TESTING EACH PATTERN THE BANKS ARE RUN FROM 0 TO 200 EXCEPT THOSE REQUIRING RELOCATION*.

MODE 5 = 'PAFBAW' = PATTERNS FORWARD, BANKS WORST FIRST

THIS MODE TESTS EACH PATTERN COMPLETELY WITH THE SIMPLE ONES FIRST, BUILDING TO THE MORE COMPLEX.

WHILE TESTING EACH PATTERN FIRST EACH KNOWN BAD BANK FROM 0 TO 200 EXCEPT THOSE REQUIRING RELOCATION* IS RUN, THEN PRESUMED GOOD BANKS ARE RUN FROM 0 TO 200 EXCEPT THOSE REQUIRING RELOCATION*.

MODE 6 = 'PARBAF' = PATTERNS REVERSE, BANKS FORWARD

THIS MODE TESTS EACH PATTERN COMPLETELY WITH THE MOST COMPLEX ONES FIRST, WORKING TO THE SIMPLE ONES.

WHILE TESTING EACH PATTERN THE BANKS ARE RUN FROM 0 TO 200 EXCEPT THOSE REQUIRING RELOCATION*.

MODE 7 = 'PARBAW' = PATTERNS REVERSE, BANKS WORST FIRST

THIS MODE TESTS EACH PATTERN COMPLETELY WITH THE MOST COMPLEX ONES FIRST, WORKING TO THE SIMPLE ONES.

WHILE TESTING EACH PATTERN FIRST EACH KNOWN BAD BANK FROM 0 TO 200 EXCEPT THOSE THAT REQUIRE RELOCATION* IS RUN, THEN PRESUMED GOOD BANKS ARE RUN FROM 0 TO 200 EXCEPT THOSE REQUIRING RELOCATION*.

NOTE

* RELOCATION IS REQUIRED TO TEST THE BANK(S) IN PROGRAM SPACE AND ALSO TO TEST ANY ECC BANKS PROTECTED BY DIAGNOSTIC CHECKMODE WITH THE INHIBIT MODE POINTER OFF (ZERO)!

2.4.1.3 TEST MODE APPLICATIONS -

1. TO VERIFY CORRECT OPERATION OF THE MEMORY SYSTEM USE MODE 0 'BAFPAF''.

   ADVANTAGES: EASY TO UNDERSTAND.

   DISADVANTAGES: IN CASE OF A FAILING BANK, IT MAY TAKE A LONG TIME TO FIND THE FAILURE.

2. TO GET DETAILED ERROR INFORMATION ON KNOWN BAD BANKS (FOUND BY SIZING ROUTINE) USE MODE 2 'BAWPAF''.

   ADVANTAGES: SEEKS BAD BANKS. EASY TO UNDERSTAND.

   DISADVANTAGES: FAILURES OTHER THAN ZEROS & ONES MAY TAKE A LONG TIME TO FIND.

3. TO GET GOOD ERROR INFO ON ANY MEMORY PROBLEM FAST USE MODE 4 'PAFBAF''.

   ADVANTAGES: COVERS ALL BANKS FAST. EASY TO UNDERSTAND.

   DISADVANTAGES: FAILURES FROM ONLY COMPLEX PATTERNS MAY TAKE A LONG TIME TO FIND.

4. TO FIND ANY PROBLEM FAST USE MODE 7 'PARBAW''.

   ADVANTAGES: COVERS ALL BANKS FAST.

   DISADVANTAGES: DIFFICULT TO UNDERSTAND FAILURES REPORTED ARE NOT NECESSARILY THE MOST BASIC FAILURE MODES.

2.4.2 DISPLAY REGISTER -

A SOFTWARE DISPLAY REGISTER EXISTS IN LOCATION 174 IN ADDITION TO ANY HARDWARE DISPLAY EXISTENCE.

DISPLAY FIELDS ARE AS FOLLOWS:

15	14 13 12 11 10	9 8	7 6 5	4 3 2 1
RELOCATED	BANK #		NOT USED	PATTERN #

PATTERN # = THE NUMBER OF THE PATTERN PRESENTLY BEING RUN. ALL PATTERNS ARE DESCRIBED IN SECTION 6.2. ANY PATTERN CAN BE FOUND IN THE DIAGNOSTIC BY LOOKING UP THE SYMBOLIC TAGS 'MTOONN'' AND 'MTPONN'' - WHERE 'NN'' IS THE PATTERN NUMBER. MTOONN REFERS TO THE ROUTINE THAT SETS UP FOR THE TEST PATTERN WHEREAS MTPONN IS THE ACTUAL PATTERN ITSELF.

NOTE

THE PATTERN # IS NOT NECESSARILY AN INDICATION OF DEGREE OF DIFFICULTY.

BANK = THE NUMBER OF THE BANK (16K) OF MEMORY UNDER TEST (0-200). THESE BITS DIRECTLY MAP TO PHYSICAL ADDRESS BITS (21:15).

RELOCATED = THIS BIT INDICATES THAT THE PROGRAM IS RELOCATED AND NO LONGER IN BANK 0. IT WILL BE RELOCATED TO THE FIRST KNOWN GOOD NON-PROTECTED MEMORY BANK INDICATED ON THE CONFIGURATION MAP (REFERENCE SECTION 7.3).

NOTE

ANOTHER WAY TO OBTAIN THIS INFORMATION IS TO TYPE A CONTROL/T AT THE CONSOLE (REFERENCE SECTION 2.4.4.5).

2.4.3 SPECIAL MEMORY LOCATIONS -

2.4.3.1 CACHE CONSTANT -

THE CACHE CONSTANT IS LOCATED AT SYMBOLIC LOCATION ''CACHK'' AND IS USED TO ENABLE CACHE.

NOTE

BIT 0 IN THE CACHE CONSTANT HAS NO EFFECT SINCE IT IS UNCONDITIONALLY SET BY THE PROGRAM WHENEVER IT TRIES TO ENABLE CACHE.

2.4.3.2 CONFIGURATION TABLE

THE CONFIGURATION TABLE IS LOCATED AT SYMBOLIC LOCATION ''CONFIG'' AND HAS THE FOLLOWING FORMAT:

```
CONFIG:  FIRST 16K CONFIGURATION WORDS (2 EACH)
         2ND   16K CONFIGURATION WORDS (2 EACH)
         .....
         200TH 16K CONFIGURATION WORDS (2 EACH)
```

CONFIGURATION WORDS:

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

THIS TABLE IS USED AS THE SOURCE FOR THE CONFIGURATION MAP (REFERENCE. SECTION 7.3).

2.4.4 TERMINAL COMMANDS -

2.4.4.1 CONTROL ''C''

THIS COMMAND WILL:

1. IF SWITCH 8 (HALT PROGRAM) IN THE SWITCH REGISTER IS SET HALT THE PROGRAM.
2. IF SWITCH 8 IS NOT SET, UNRELOCATE IF PROGRAM WAS RELOCATED.
3. FLUSH OUT ANY DBE'S.
4. TURN OFF MEMORY MANAGEMENT.
5. ATTEMPT TO BOOT RK05 DRIVE 0.
6. FAILING 4, ATTEMPT TO BOOT RK04 DRIVE 1.
7. FAILING 5, GO TO 4.

THIS COMMAND WILL ONLY BE RECOGNIZED AT THE COMPLETION OF THE CURRENT TEST OR PATTERN, OR AT THE END OF A LINE OF AN ERROR MESSAGE.

2.4.4.2 CONTROL ''D'' (DEBUG)

THIS COMMAND TO ENTER A MODIFIED VERSION OF ODT HAS BEEN DELETED.

2.4.4.3 CONTROL ''E'' (PROCEED)

THIS COMMAND WOULD ALLOW YOU TO EXIT ODT. IT IS HAS ALSO BEEN DELETED.

2.4.4.4 CONTROL ''K'' (KILL ERROR PRINTOUT AND SKIP PATTERN)

THIS COMMAND WILL ALLOW YOU TO STOP AN ERROR PRINTOUT AND SKIP TO THE NEXT PATTERN. THIS IS HANDY, FOR EXAMPLE, WHEN YOU HAVE A WHOLE BANK FULL OF ERRORS, HAVE GOTTEN ENOUGH INFORMATION, AND WISH TO SKIP TO THE NEXT PATTERN.

2.4.4.5 CONTROL "T" (TELL ME WHAT'S HAPPENING)

THIS COMMAND WILL PRINT OUT THE INFORMATION ENCODED IN THE DISPLAY REGISTER. THIS IS MAINLY INTENDED FOR CPU'S WITHOUT A HARDWARE DISPLAY REGISTER.

EXAMPLE:

RELOCATED BANK= 23 PAT= 26

BY USE OF FIELD SERVICE COMMAND 17 "TRACE" CAN BE SET SO THAT IT WILL AUTOMATICALLY TYPE OUT THE BANK AND PATTERN NUMBERS AS EACH PATTERN IS RUN. (REFERENCE SECTION 2.4.4.8.18).

2.4.4.6 CONTROL "S" (STOP)

THIS COMMAND WILL STOP TYPEOUT (SOON) AND WILL WAIT FOR A CONTROL "Q".

2.4.4.7 CONTROL "Q" (QUINTINUE)

THIS COMMAND WILL CONTINUE TYPING THAT HAS BEEN STOPPED BY CONTROL "S". IF THERE HAS BEEN NO CONTROL "S" TYPED THEN THIS COMMAND IS IGNORED.

2.4.4.8 CONTROL "F" (FIELD SERVICE MODE)

THIS COMMAND WILL CAUSE YOU TO ENTER A MODE WHICH LOOKS FOR SUB COMMANDS.

WHEN THE PROGRAM IS LOOK.NG 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 TYPFOUT THE CONTENTS OF THE CSR.

IF THERE IS MORE THAN ONE CSR ON THE CPU (OR IF THE PROGRAM HAS NOT DETERMINED THE CSR STATUS YET), IT WILL ASK YOU "WHICH CSR(0-F)" TO WHICH YOU MUST RESPOND WITH AN HEXIDECIMAL NUMBER FROM 0 TO F. NOTE TYPING JUST CARRIAGE RETURN IS A DEFAULT 0.

IF THE CSR YOU SELECT CAUSES A TRAP TO 4 THE PROGRAM WILL TYPE "THIS CSR DOES NOT EXIST".

NOTE

CSR REFERENCES ARE DONE IN ACCORDANCE WITH SECTION 5.0.

2.4.4.8.3 FIELD SERVICE COMMAND 2 (LOAD CSR)

THIS COMMAND WILL ENABLE YOU TO LOAD THE CSR.

IF THERE IS MORE THAN ONE CSR ON THE CPU (OR IF THE PROGRAM HAS NOT YET DETERMINED THE CSR STATUS YET) IT WILL ASK YOU 'WHICH CSR(0-F)'' TO WHICH YOU MUST RESPOND WITH AN HEXIDECIMAL NUMBER FROM 0 TO F. NOTE TYPING JUST CARRIAGE RETURN IS A DEFAULT 0.

IF THE CSR YOU SELECT CAUSES A TRAP TO 4 THE PROGRAM WILL TYPE ''THIS CSR DOES NOT EXIST''.

THE CSR WILL BE READ AND DISPLAYED AS IN COMMAND 1.

THE PROGRAM WILL THEN ASK YOU FOR THE ''CSR?'' TO WHICH YOU MUST RESPOND WITH AN OCTAL NUMBER. NOTE TYPING JUST CARRIAGE RETURN IS A DEFAULT 0.

THE PROGRAM WILL THEN LOAD THE CSR AND READ IT AGAIN DISPLAYING ITS NEW CONTENTS.

2.4.4.8.4 FIELD SERVICE COMMAND 3 (EXAMINE MEMORY)

THIS COMMAND WILL ALLOW YOU TO EXAMINE ANY PHYSICAL ADDRESS AND DOES THE NECESSARY MEMORY MANAGEMENT MAPPING FOR YOU.

THE PROGRAM WILL ASK YOU FOR THE 'PHYSICAL ADDRESS (0-17757776)'' TO WHICH YOU MUST RESPOND WITH AN OCTAL NUMBER.

IF THE ADDRESS ACCESS CAUSES A TRAP TO 4 THE PROGRAM WILL TYPE ''TIMEOUT TRAP''. IF THE ADDRESS ACCESS CAUSES A TRAP TO 114 THE PROGRAM WILL TYPE 'PARITY ABORT''.

THE CONTENTS OF YOUR PHYSICAL ADDRESS WILL BE TYPED.

2.4.4.8.5 FIELD SERVICE COMMAND 4 (MODIFY MEMORY)

THIS COMMAND ALLOWS YOU TO MODIFY ANY PHYSICAL ADDRESS AND DOES THE NECESSARY MEMORY MANAGEMENT MAPPING FOR YOU.

THE PROGRAM WILL ASK YOU FOR THE "PHYSICAL ADDRESS (0-17757776)" TO WHICH YOU MUST RESPOND WITH AN OCTAL NUMBER.

IF THE ADDRESS ACCESS CAUSES A TRAP TO 4 THE PROGRAM WILL TYPE "TIMEOUT TRAP". IF THE ADDRESS ACCESS CAUSES A TRAP TO 114 THE PROGRAM WILL TYPE "PARITY ABORT".

THE PROGRAM WILL TYPE "OLD DATA WAS" AND THE CONTENTS OF YOUR PHYSICAL ADDRESS.

THE PROGRAM WILL THEN TYPE "INPUT NEW DATA" TO WHICH YOU MUST RESPOND WITH AN OCTAL NUMBER. NOTE TYPING JUST CARRIAGE RETURN IS A DEFAULT 0.

THE PROGRAM WILL ATTEMPT TO WRITE THIS NEW DATA INTO YOUR PHYSICAL ADDRESS AFTER WHICH IT WILL READ IT AGAIN AND TYPE "DATA IS NOW" AND THE NEW CONTENTS OF YOUR PHYSICAL ADDRESS.

NOTE

IF YOU CAN'T CHANGE THE DATA, THAT WOULD INDICATE THAT YOU HAVE A DOUBLE BIT ERROR IN THAT DOUBLE WORD PAIR.

2 4.4.8.6 FIELD SERVICE COMMAND 5 (SELECT BANK & PATTERN)

THIS COMMAND ALLOWS YOU TO RUN ANY BANK WITH ANY PATTERN FOREVER.

THE PROGRAM WILL ASK YOU "BANK(0-200)" TO WHICH YOU MUST RESPOND WITH AN OCTAL NUMBER. IF THE BANK IS NOT ACCESSIBLE. THE PROGRAM WILL TYPE "BANK NOT ACCESSIBLE" AND ASK QUESTION OVER.

THE PROGRAM WILL THEN ASK "PATTERN (0-37)" TO WHICH YOU MUST RESPOND WITH AN OCTAL NUMBER.

NOTE

ANY PATTERN CAN BE RUN INCLUDING THOSE THAT ARE NOT PART OF THE APT E-TABLE DEFAULTS (REFERENCE SECTION 6.2.1). IF YOU SELECT PATTERN 0, THE PROGRAM WILL ASK "PATTERN 0 DATA IS?" TO WHICH YOU MUST RESPOND WITH AN OCTAL NUMBER.

IF THE BANK YOU SELECTED REQUIRES RELOCATION THE PROGRAM WILL TYPE "BANK REQUIRES RELOCATION" AND EXIT THIS COMMAND. NOTE NORMALLY THIS IS TRUE FOR BANK 0.

THE PROGRAM WILL THEN ARM THE CONSOLE KEYBOARD FOR INTERRUPTS AND TYPE "TO ESCAPE TYPE ANY KEY!".

THE TEST PATTERN WILL BE ENTERED AND RUN UNTIL A CONSOLE KEY IS DEPRESSED TO ESCAPE THIS LOOP.

2.4.4.8.7 FIELD SERVICE COMMAND 6 (TYPE CONFIGURATION MAP)

THIS COMMAND TYPES THE CONFIGURATION MAP.

THIS IS USEFUL AFTER A LONG RUN (OVERNIGHT) TO SEE ALL THE BANKS THAT ARE MARKED AS BAD. (ESPECIALLY IF YOUR CONSOLE IS A VIDEO TERMINAL).

FOR A DETAILED EXPLANATION OF THE MAP REFERENCE SECTION 7.3.

2.4.4.8.8 FIELD SERVICE COMMAND 7 (SOB-A-LONG TEST)

THIS COMMAND ALLOWS EXECUTION OF THE SOB-A-LONG TEST ON ALL NON-PROTECTED BANKS REFERENCE SECTION 6.2.2.26. OPERATION IS IDENTICAL TO COMMAND 5 EXCEPT THAT NO PATTERN OR BANK IS ENTERED AND EACH PASS CAUSES A BELL.

2.4.4.8.9 FIELD SERVICE COMMAND 8 (ERROR SUMMARY)

THIS COMMAND TYPES OUT THE NUMBER OF PASSES AND THE TOTAL NUMBER OF ERRORS. IF THERE WERE ANY ERRORS IT WILL TYPE OUT THE BANKS AND THE NUMBER OF ERRORS PER BANK UP TO 255 DECIMAL.

THIS BECOMES USEFUL AFTER LONG RUNS (ALL NIGHT) ON SYSTEMS WITH A VIDEO 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   06
022132    35  060002  03500002  000000   000100   000100   0   M     1   06
022132    35  060006  03500006  000000   000100   000100   0   M     1   06
```

WHILE TESTING BANK 35, VIRTUAL ADDRESS 60000, PHYSICAL ADDRESS 3700000 WITH PATTERN 6 (INITIAL DATA TEST), THE GOOD DATA EXPECTED WAS 0 BUT THE DATA ACTUALLY READ (BAD) WAS 1, THE EXCLUSIVE OR AT GOOD & BAD YIELDS 1 WHICH INDICATES ONLY FAILING BIT(S) (BIT 0). IT IS AN MS11-M (ECC) MEMORY AND IT'S INTERLEAVED; SO SINCE ADDRESS BIT 1 WAS NOT ASSERTED, THE CSR IS LOCATED AT 172000.

WHILE ALSO IN BANK 35, VIRTUAL ADDRESSES 60002 AND 60006 WERE EXPECTED TO HAVE 0, BUT THE DATA READ WAS 100, THE EXCLUSIVE OR OF GOOD & BAD YIELDS 100 WHICH INDICATES ONE FAILING BIT (BIT 6). SINCE IT IS INTERLEAVED MS11-M MEMORY, AND ADDRESS BIT 1 IS ASSERTED, THE CSR IS LOCATED AT 172102 (CSR NUMBER 1 UNDER THE INT COLUMN)

NOTE

SUBSEQUENT ERRORS OF THE SAME TEST DO NOT TYPE A NEW HEADING.

3.2 ERROR ABBREVIATIONS

THE FOLLOWING IS A LIST OF ALL ABBREVIATIONS USED IN ERROR REPORTS.

# OF ERRORS	NUMBER OF ERRORS THAT WERE DETECTED.
1ST ADD	FIRST ADDRESS THAT FAILED.
ARRAY	THE ARRAY NUMBER THAT WAS LOCKED UP IN THE MS11-M CSR.
APT#	THE # OF CPU'S APT EXPECTS ON THE SYSTEM.
APTCORE	APT CORE SIZE.
APTMOS	APT MOS SIZE.
BAD	BAD DATA.
BAD-WD1	BAD WORD #1 OF A DOUBLE WORD DATA VALUE.
BAD-WD2	BAD WORD #2 OF A DOUBLE WORD DATA VALUE.
BAD-CHK	BAD CHECK CODE BITS.
BANK	THE BANK NUMBER. BANKS ARE 16K WORDS LONG.
BD-CC	BAD CHECK CODE BITS.
CHKBITS	THE 7 BIT VALUE OF THE CHECK CODE BITS.
CONTRL	THE CACHE CONTROL REGISTER.
CPUERR	CPU ERROR REGISTER.
CSR	CONTROL AND STATUS REGISTER.
CSRNO	CSR NUMBER (0-F HEXIDECIMAL).
DATARG	THE CACHE DATA REGISTER.
DBE	DOUBLE BIT ERROR (UNCORRECTABLE ERROR).
DEV ADD	DEVICE ADDRESS.
ECC	ERROR CORRECTABLE CODE.
GD-CC	GOOD CHECK CODE BITS.
GD-CHK	GOOD CHECK CODE BITS.
GD-WD1	GOOD WORD #1 OF A DOUBLE WORD DATA VALUE.
GD-WD2	GOOD WORD #2 OF A DOUBLE WORD DATA VALUE.
GOOD	GOOD DATA.
INT	INTERLEAVED (ADDRESS BIT 1 ASSERTED) CSR NUMBER.
LSIZE	MS11-L SIZE.
MEMERR	MEMORY ERROR REGISTER.
MMR0	MEMORY MANAGEMENT REGISTER #0.
MMR1	MEMORY MANAGEMENT REGISTER #1.
MMR2	MEMORY MANAGEMENT REGISTER #2.
MMR3	MEMORY MANAGEMENT REGISTER #3.
MSIZE	MS11-M SIZE.
MTYP	MEMORY TYPE (MS11-L,MS11-M,MF11S-K,BIPOLAR OR UNIBUS PARITY).
PADD	PHYSICAL ADDRESS (ASSERTED BY THE PROGRAM AFTER MAPPING).
PAT	PATTERN NUMBER.
PC	PROGRAM COUNTER AT THE TIME THE ERROR OCCURRED.
SBE	SINGLE BIT ERROR (CORRECTABLE ERROR).
VADD	VIRTUAL ADDRESS (ASSERTED BY THE PROGRAM BEFORE MAPPING).
WROTE1	THE DATA THAT WAS WRITTEN INTO THE 1ST HALF OF A DOUBLE WORD.
WROTE2	THE DATA THAT WAS WRITTEN INTO THE 2ND HALF OF A DOUBLE WORD.
XOR	EXCLUSIVE OR OF THE GOOD AND BAD DATA. SHOWS THE BAD BITS.

3.3 ERROR HALTS

THERE ARE SEVERAL HALTS IN THE PROGRAM.

ALL UNUSED TRAP VECTORS CONTAIN A TRAP CATCHER (.WORD .+2,HALT).

AN UNDEFINED TRAP INSTRUCTION HALTS AT SYMBOLIC LOCATION "$HALT2".

THE APT DOWN LOAD SEQUENCE WILL HALT AT SYMBOLIC LOCATION "APTHLT".

HALT ON ERROR OPTION (SW15 SET) AT SYMBOLIC LOCATION "$HALT".

HALT PROGRAM (SW8 SET) AT SYMBOLIC LOCATION "$EXHALT".

POWER FAIL WILL NORMALLY HALT AT THE END OF THE SHUT DOWN SEQUENCE (SYMBOLIC LOCATION "$DOWN").

POWER FAIL HAS A FATAL HALT AT SYMBOLIC LOCATION "$ILLUP" WHICH CAN BE CAUSED BY POWER UP OCCURRING BEFORE POWER DOWN SEQUENCE COMPLETED OR BY POWER DOWN BEFORE A POWER UP SEQUENCE IS COMPLETED.

4.0 PROGRESS REPORTS

PASS COMPLETE TYPEOUTS AS FOLLOWS:

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

NOTE

PASS 2 WAS FLAGGED AS A QUICK VERIFY PASS. (BECAUSE OF A CHANGE IN SW5)

TO OBTAIN PROGRESS REPORTS WHILE EXECUTING, TYPING A CONTROL "T" WILL PRINT OUT THE INFORMATION ENCODED IN THE DISPLAY REGISTER.

EXAMPLE:

BANK= 2 PAT= 34

REFERENCE SECTION 2.4.4.8.18 FOR MORE INFORMATION ON TRACING.

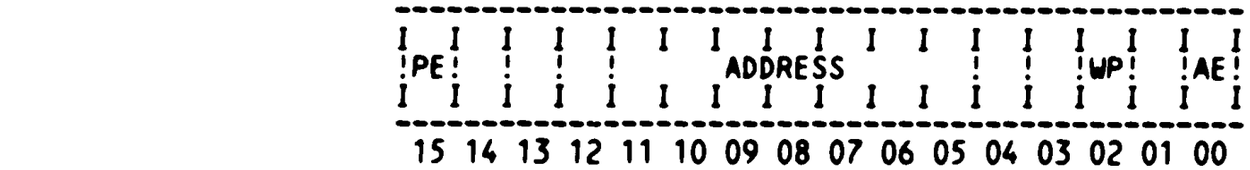
5.0 CSR INFORMATION TABLES

THE FOLLOWING IS A PICTURE VIEW OF THE CURRENT CONTROL STATUS REGISTERS WHICH CAN BE TESTED BY THIS PROGRAM. IT SHOWS BIT ASSIGNMENTS AND DEFINITIONS TO PROVIDE A HANDY REFERENCE, AND SHOWS THE SIMILARITIES AND DIFFERENCES BETWEEN EACH ONE:

NOTE

ALL UNUSED BITS IN EACH CSR ARE EQUAL TO ZERO.

5.1 CORE/MOS PARITY REGISTER



BIT ASSIGNMENTS ARE DEFINED AS FOLLOWS:

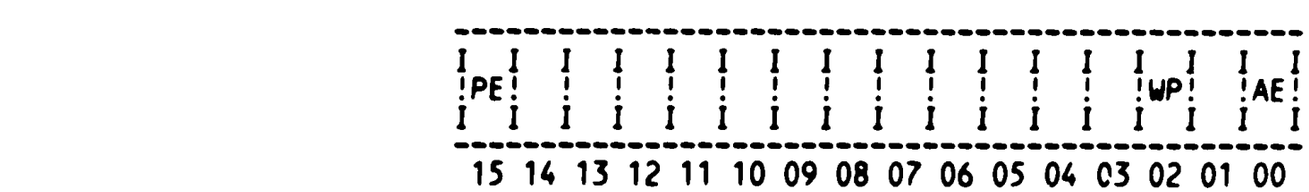
BIT15 PARITY ERROR

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)



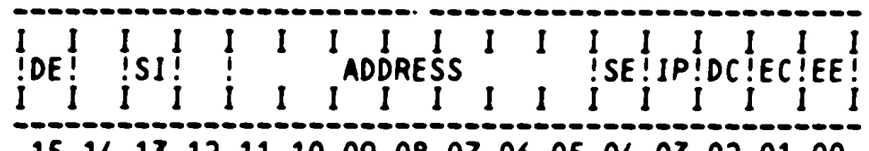
BIT ASSIGNMENTS ARE DEFINED AS FOLLOWS:

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 BITS READ WILL BE STORED IN BITS 11-5.

BIT 13 SET INHIBIT MODE WHEN THIS BIT IS SET TO A 1, IT ENABLES THE INH MODE POINTER TO INHIBIT EITHER THE FIRST OR SECOND 16K FROM EVER GOING INTO THE DIAG. CHECK OR ECC DISABLE MODE. WHEN THIS BIT IS SET TO ZERO, THE ENTIRE MEMORY OPERATES IN IN DIAGNOSTIC CHECK OR ECC DISABLE MODE.

BITS 11-5 ERROR ADDRESS WITH BIT02 CLEARED THEY CONTAIN THE HIGH ORDER ERROR ADDRESS (BITS 17-11); WITH BIT02 SET THEY CONTAIN THE CHECK BITS FOR ECC.

BIT04 SINGLE ERROR SET WHENEVER SINGLE ERROR OCCURS

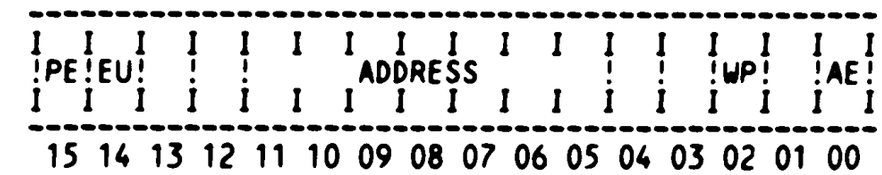
BIT03 INHIBIT MODE POINTER THE INHIBIT MODE POINTER WORKS IN CONJUNCTION WITH THE SET INHIBIT MODE BIT. WHEN BIT13 IS SET TO A 1, A 16K PORTION OF MEMORY IS INHIBITTED FROM OPERATING IN THE ECC DISABLE MODE OR DIAGNOSTIC CHECK MODE. THE INHIBIT MODE PCINTER INDICATES WHICH 16K IS BEING INHIBITED; E.G.-WHEN 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.

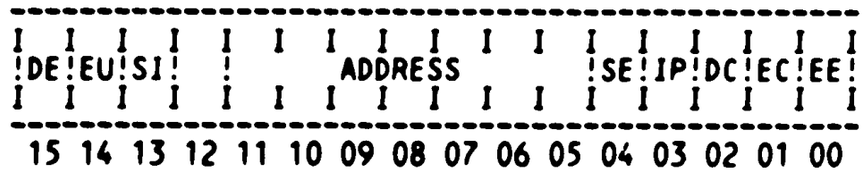
BITS 11-5 ERROR ADDRESS WITH BIT14 SET, THEY CONTAIN THE HIGH ORDER PARITY ERROR ADDRESS (BITS 21-18 OF ADDRESS); WITH BIT14 CLEARED, THEY CONTAIN THE LOW ORDER PARITY ERROR ADDRESS (BITS 17-11 OF ADDRESS).

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

1971 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

```
                                                    MODE  TO  OPERATE OVER
                                                    THE ENTIRE  MEMORY  ON
                                                    THE BOARD.
```

BITS 11-5 ERROR ADDRESS WITH BIT02 CLEARED AND BIT14 SET, THEY CONTAIN THE HIGH ORDER ERROR ADDRESS (BITS 21-18); WHEN BIT02 AND BIT14 ARE CLEARED, THEY CONTAIN THE LOW ORDER ERROR ADDRESS (BITS 17-11); WHEN BIT02 IS SET THEY CONTAINS CHECK BITS FOR ECC.

BIT04 SINGLE ERROR SET WHENEVER SINGLE ERROR OCCURS.

BIT03 INHIBIT MODE POINTER THE INHIBIT MODE POINTER WORKS IN CONJUNCTION WITH THE SET INHIBIT MODE BIT. WHEN BIT13 IS SET TO A 1, A 16K PORTION OF MEMORY IS INHIBITTED FROM OPERATING IN THE ECC DISABLE MODE OR DIAGNOSTIC CHECK MODE. THE INHIBIT MODE POINTER INDICATES WHICH 16K IS BEING INHIBITED; E.G.-IF BIT3 =1, THE SECOND 16K OF MEMORY IS INHIBITTED. WHEN BIT13 IS SET TO A 0, BIT3 BECOMES INOPERATIVE.

BIT02 DIAGNOSTIC CHECK MODE WHEN SET ENABLES READ-WRITE OF CHECK BITS(SEE BITS 11-5). IF A DBE OCCURS IN THIS MODE (WITH BIT1=0), BIT15 IS SET, BUT THE CHECK BITS READ ARE STORED IN BITS 11-5, NOT THE DBE ADDRESS BITS.

BIT01 DISABLE ERROR CORRECTION WHEN SET NO SINGLE ERROR CORRECTI-ON TAKES PLACE. A SINGLE BIT ERROR WILL SET BIT04 AND BIT15 AND ASSERT BUS PBL L IF BIT00 IS ASSERTED; A DOUBLE ERROR WILL SET SET BIT15 AND AS-SERT BUS PBL L IF BIT00 IS ASSERTED. THE ERROR ADDRESS IS STORED IN THE CSR, AND CORRECT CHECK BITS ARE GENERATED AND STORED ON A WRITE.

BIT00 UNCORRECTABLE ERROR ENABLE WHEN SET ENABLES TRAP TO VECTOR 114 ON UNCORRECTABLE ERROR.

```
2115
2116
2117
2118                    6.0  SUB-TEST SUMMARIES
2119
2120                    6.1  TESTS
2121
2122
2123
2124
2125                    TEST     1
2126
2127                             BIT TEST OF ALL CSR'S/MATCH ALL CSR'S WITH MEMORY
2128                             (CSR ACCESS MAY CAUSE WRONG TYPE OF TRAPS)
2129
2130                    TEST     2
2131
2132                             TEST BANK 0 ACCESSES
2133                             FAILURES ARE FATAL.
2134
2135                    TEST     3
2136
2137                             TEST BANKS 1-200 (OCTAL) FOR ZEROS AND ONES
2138                             ERRORS ARE NOT TYPED HERE - ONLY LOGGED IN
2139                             THE CONFIGURATION TABLE
2140
2141                    TEST     4
2142
2143                             ECC INHIBIT MODE POINTER TEST
2144
2145                    TEST     5
2146
2147                             DIAGNOSTIC MODE DISPATCH ROUTINE
2148                             THIS TEST RUNS ALL THE PATTERNS IN THE
2149                             MODE SELECTED.
2150
2151                    TEST     6
2152
2153                             UNIQUE BANK TEST
2154                             PATTERN 27 IS RUN
2155
2156
```

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 IS TO FORCE A THE PARITY BITS TO BECOME INVOLVED.

IT CAN EXECUTE OUT OF THE USER DATA PDR'S, THE USER INSTRUCTION PAR'S, THE KERNEL DATA PAR'S AND THE SUPERVISOR DATA PAR'S.

6.2.2.5 PATTERN 4 ROTATING ZEROS TEST

WRITES A BACKGROUND PATTERN OF ONES. ROTATES A ZERO CARRY BIT LEFT THRU EACH PAR OF BYTES (18 TIMES) AND THEN CHECKS THAT THE CARRY IS ZERO AND THE WORD (2 BYTES) IS STILL ALL ONES.

IT CAN EXECUTE OUT OF THE USER DATA PAR'S AND THE KERNEL DATA PAR'S.

NOTE

IT IS NOT UNCOMMON TO OBSERVE THE GOOD DATA EQUAL TO THE BAD DATA. THIS INDICATES THAT THE CARRY WAS NOT CLEAR AFTER 18 ROLB'S.

6.2.2.6 PATTERN 5 ROTATING ONES TEST

WRITES A BACKGROUND PATTERN OF ZEROS. ROTATES A ONE CARRY BIT LEFT THRU EACH PAIR OF BYTES (18 TIMES) AND THEN CHECKS THAT THE CARRY IS A ONE AND THE WORD (2 BYTES) IS STILL ALL ZEROS.

THIS PROVIDES THE COMPLEMENT OF THE COVERAGE OF PATTERN 4 IN DATA.

IT CAN EXECUTE OUT OF THE USER DATA PAR'S AND THE KERNEL DATA PAR'S.

NOTE

IT IS NOT UNCOMMON TO OBSERVE THE GOOD DATA EQUAL THE BAD DATA. THIS INDICATES THAT THE CARRY WAS NOT SET AFTER 18 ROLB'S.

6.2.2.7 PATTERN 6 INITIAL DATA TEST

WRITES & READS A DOUBLE WORD FIRST WITH ALL BITS 0 EXCEPT 1 (FOR EVERY BIT POSITION), SECOND WITH ALL BITS 1 EXCEPT 1 (FOR EVERY BIT POSITION).

THIS IS A VERY QUICK CHECK OF THE DATA PATHS.

6.2.2.8 PATTERN 7 ADDRESS BIT TEST

WRITES A BACKGROUND OF ALL ZEROS.
READ ADDRESS 1 FOR A 0 BYTE.
COMPLEMENT ADDRESS 1.
READ ADDRESS 1 FOR A NON 0 BYTE.
FOR EACH ADDRESS BIT POSITION FROM BIT 1:
VIRTUAL (2, 4, 10, 20, 40, 100, 200, 400, 1000, 2000, 4000, 10000, 60000, 20000)
PHYSICAL (60002, 60004, 60010, 60020, 60040, 60100, 60200, 60400, 61000, 62000, 64000, 70000, 140000, 100000)
  READ ADDRESS FOR A 0 WORD.
  COMPLEMENT ADDRESS CONTENTS.
  READ ADDRESS FOR A NON-ZERO WORD.

THIS IS A VERY QUICK CHECK OF THE ADDRESS BIT UNIQUENESS.

6.2.2.9 PATTERN 10 BYTE ADDRESSING TEST

WITH ECC DISABLED.
WRITES ALL ONES TO A DOUBLE WORD.
FOR EACH OF THE 4 BYTES IN THE DOUBLE WORD.
  CLEARS ONE BYTE.
  READS ALL 4 BYTES FROM DOUBLE WORD.
  CHECKS FOR ONLY PROPER BYTE CLEAR.
  ALL OTHER BYTES SET TO ALL ONES.

THIS IS ONLY DONE ON ONE DOUBLE WORD ADDRESS.

NOTE

THIS IS RUN FOR ECC MEMORY ONLY

6.2.2.10 PATTERN 11 SINGLE BIT ERROR TEST

1. CREATE A SINGLE BIT ERROR.
2. READ DATA UNCORRECTED (WITH ECC DISABLE).
3. CHECK THAT SBE AND DBE FLAGS ARE SET, AND THE ERROR ADDRESS IS LATCHED.
4. READ FIRST WORD OF DATA CORRECTED (WITH ECC ENABLED)
5. CHECK THAT THE CSR SINGLE BIT ERROR FLAG WAS SET, AND THE ERROR ADDRESS WAS LATCHED.
6. CLEAR SBE FLAG.
7. READ SECOND WORD OF DATA CORRECTED (WITH ECC ENABLED).
8. CHECK THAT THE CSR SINGLE BIT ERROR FLAG WAS SET.
9. DO (1-7) FOR A SINGLE BIT ERROR IN EACH OF 32 POSITIONS OF A DOUBLE WORD.
   I.E. (32 TIMES)
10. IF NOT IN QUICK VERIFY MODE THEN DO (1-8) FOR DATA CONSISTING OF 1 BIT SET IN EACH OF 32 POSITIONS OF A DOUBLE WORD.
    I.E. (32 X 32 = 1024 TIMES)
11. DO (1-9) FOR COMPLEMENTED DATA (1 BIT CLEAR IN EACH OF 32 POSITIONS OF A DOUBLE WORD).
    I.E. (1024 X 2 = 2048 TIMES)
    OR (32 X 2 = 64 TIMES (QUICK VERIFY))
12. DO (1-7) FOR A DOUBLE WORD EQUAL TO (000000,000000), AND ALL POSSIBLE SINGLE BIT ERROR COMBINATIONS FORCED INTO THE CHECK BITS (CSR BITS 5-11).
13. CLEAR ANY ERRORS OUT OF TEST LOCATIONS.

THIS INSURES THAT ALL SINGLE BIT ERRORS CAN BE CORRECTED AND DETECTED.

NOTE

THIS TEST IS RUN FOR ECC MEMORY ONLY

6.2.2.11 PATTERN 12 WRITE BYTE CLEARS SBE TEST

1. CREATE A SINGLE BIT ERROR.
2. WRITE A BYTE OF DOUBLE WORD TO ONES.
3. READ A BYTE OF DOUBLE WORD.
4. IF THIS IS MS11-M, THE SBE FLAG SHOULD BE SET.
   IF THIS IS MF11S-K THE SBE FLAG SHOULD BE SET IF THIS IS THE BYTE WITH THE ERROR.
5. THE BYTE SHOULD HAVE BEEN EQUAL TO ONES.
6. DO (1-5) FOR EACH OF THE 4 BYTES OF THE DOUBLE WORD
7. DO (1-6) FOR A SINGLE BIT ERROR IN EACH OF 32 POSITIONS OF A DOUBLE WORD
   I.E. (32 TIMES)
8. IF NOT IN QUICK VERIFY MODE THEN DO (1-7) FOR DATA CONSISTING OF 1 BIT SET IN EACH OF 32 POSITIONS OF A DOUBLE WORD.
   I.E. (32 X 32 = 1024 TIMES)
9. CLEAR ANY ERRORS OUT OF TEST LOCATIONS.

THIS INSURES THAT SINGLE BIT ERRORS IN THE DATA PORTION (NOT IN CHECKBITS) CAN BE CLEARED BY WRITING THE CORRESPONDING BYTE AND THAT WRITING ANY OTHER BYTE DOES NOT CHANGE THE EXISTING SINGLE BIT ERROR.

NOTE

THIS TEST IS RUN FOR ECC MEMORY ONLY.

6.2.2.12 PATTERN 13 CREATE DOUBLE BIT ERROR TEST

1. CREATE A DOUBLE BIT ERROR.
2. ACCESS THE DATA (TST INSTRUCTION).
3. CHECK THAT THE CSR DBE FLAG IS SET, AND THE ERROR ADDRESS IS LATCHED.
4. INITIALIZE CSR TO ALLOW PARITY TRAPS ON DBE'S.
5. ACCESS THE DATA (TST INSTRUCTION).
6. CHECK THAT A PARITY TRAP OCCURRED.
7. DO (1-6) FOR THE 2ND BIT OF EACH DOUBLE BIT ERROR IN EACH OF 32 POSITIONS OF A DOUBLE WORD LESS THE ONE POSITION OF THE 1ST BAD BIT.
   I.E. (31 TIMES)
8. IF NOT IN QUICK VERIFY MODE THEN DO (1-7) FOR THE 1ST BIT OF EACH OF DOUBLE BIT ERROR IN EACH OF 32 POSITIONS OF A DOUBLE WORD.
   I.E. (31 X 32 = 992 TIMES)
9. DO (1-8) FOR COMPLEMENTED DATA (ONES VERSUS ZEROS IN DOUBLE WORD)
   I.E. (992 X 2 = 1984 TIMES)
   OR (31 X 2 = 62 TIMES (QUICK VERIFY))
10. DO (1-6) FOR A DOUBLE WORD EQUAL TO (000000,000000), AND ALL POSSIBLE DOUBLE BIT ERROR COMBINATIONS FORCED INTO EACH OF THE CHECK BITS (CSR BITS 5-11).
11. CLEAR ANY ERRORS OUT OF TEST LOCATIONS.

THIS INSURES THAT ALL DOUBLE BIT ERRORS CAN BE CREATED AND DETECTED AND CAUSE TRAPS.

NOTE

THIS TEST IS ONLY RUN DURING THE FIRST (QV) PASS WHEN UNDER ACT OR APT, AND IS RUN FOR ECC MEMORY ONLY.

6.2.2.13 PATTERN 14 WRITE INHIBIT DURING DATIP WITH DBE TEST

1. CREATE A DOUBLE BIT ERROR.
2. DO ASRB ON TEST LOCATION.
3. CHECK THAT DOUBLE WORD IS STILL BAD (UNCHANGED-WITH DBE).
4. DO (2-3) ON ALL 4 BYTES OF DOUBLE WORD.
5. DO (1-4) FOR THE 2ND BIT OF EACH DOUBLE BIT ERROR IN EACH OF 32 POSITIONS OF A DOUBLE WORD LESS THE ONE POSITION OF THE 1ST BAD BIT.
   I.E. (31 TIMES)
6. IF NOT IN QUICK VERIFY MODE THEN DO (1-5) FOR THE 1ST BIT OF EACH DOUBLE BIT ERROR IN EACH OF 32 POSITIONS OF A DOUBLE WORD.
   I.E. (32 X 32 = 922 TIMES)
7. DO (1-6) FOR COMPLEMENTED DATA (ONES VERSUS ZEROS IN DOUBLE WORD).
   I.E. (922 X 2 = 1984 TIMES)
   OR (31 X 2 = 62 TIMES (QUICK VERIFY))
8. DO (1-4) FOR A DOUBLE WORD EQUAL TO (000000,000000), AND ALL POSSIBLE DOUBLE BIT ERROR COMBINATIONS FORCED INTO THE CHECK BITS(CSR BITS 5-11).
9. CLEAR ANY ERRORS OUT OF TEST LOCATIONS.

THIS INSURES THAT THE DOUBLE BIT ERROR CAN BE CLEARED BY A DATIP TO ANY AFFECTED BYTE.

NOTE

THIS TEST IS ONLY RUN DURING THE FIRST (QV) PASS WHEN UNDER ACT OR APT, AND IS RUN FOR MF11S-K ONLY.

6.2.2.14 PATTERN 15 WRITE INHIBIT OF BYTE WITH DBE

1. CREATE A DOUBLE BIT ERROR.
2. DO A MOVB IMMEDIATE TO TEST BYTE.
3. CHECK THAT DOUBLE WORD IS STILL BAD (UNCHANGED-WITH DBE).
4. DO (2-3) ON ALL 4 BYTES OF DOUBLE WORD.
5. DO (1-4) FOR THE 2ND BIT OF EACH DOUBLE BIT ERROR IN EACH OF 32 POSITIONS OF A DOUBLE WORD LESS THE ONE POSITION OF THE 1ST BAD BIT.
   I.E. (31 TIMES)
6. IF NOT IN QUICK VERIFY MODE THEN DO (1-5) FOR THE 1ST BIT OF EACH DOUBLE BIT ERROR IN EACH OF 32 POSITIONS OF A DOUBLE WORD.
   I.E. (31 X 32 = 922 TIMES)
7. DO (1-6) FOR COMPLEMENTED DATA (ONES VERSUS ZEROS IN DOUBLE WORD).
   I.E. (992 X 2 = 1984 TIMES)
   OR (31 X 2 = 62 TIMES (QUICK VERIFY))
8. DO (1-4) FOR A DOUBLE WORD EQUAL TO (000000,000000), AND ALL POSSIBLE DOUBLE BIT ERROR COMBINATIONS FORCED INTO THE CHECK BITS (CSR BITS 5-11).
9. CLEAR ANY ERRORS OUT OF TEST LOCATIONS.

THIS INSURES THAT NO DOUBLE BIT ERROR CAN BE CLEARED BY A MOVB TO ANY AFFECTED BYTE.

NOTE

THIS TEST IS ONLY RUN DURING THE FIRST (QV) PASS WHEN UNDER ACT OR APT, AND IS RUN FOR ECC MEMRY ONLY.

6.2.2.15 PATTERN 16 WRITE INHIBIT OF WORD WITH DBE TEST

1. CREATE A DOUBLE BIT ERROR.
2. DO MOV IMMEDIATE ON TEST LOCATION.
3. CHECK THAT DOUBLE WORD IS STILL BAD (UNCHANGED-WITH DBE).
4. DO (2-3) ON BOTH DOUBLE WORDS.
5. DO (1-4) FOR THE 2ND BIT OF EACH DOUBLE BIT ERROR IN EACH OF 32 POSITIONS OF A DOUBLE WORD LESS THE ONE POSITION OF THE 1ST BAD BIT.
   I.E. (31 TIMES)
6. IF NOT IN QUICK VERIFY MODE THEN DO (1-5) FOR THE 1ST BIT OF EACH DOUBLE BIT ERROR IN EACH OF 32 POSITIONS OF A DOUBLE WORD.
   I.E. (32 X 32 = 992 TIMES)
7. DO (1-6) FOR COMPLEMENTED DATA (ONES VERSUS ZEROS IN DOUBLE WORD).
   I.E. (992 X 2 = 1984 TIMES)
   OR (31 X 2 = 62 TIMES (QUICK VERIFY))
8. DO (1-4) FOR A DOUBLE WORD EQUAL TO (000000,000000), AND ALL POSSIBLE DOUBLE BIT ERROR COMBINATIONS FORCED INTO THE CHECK BITS (CSR BITS 5-11).
9. CLEAR ANY ERRORS OUT OF TEST LOCATIONS.

THIS INSURES THAT NO DOUBLE BIT ERROR CAN BE CLEARED BY A MOV TO ANY AFFECTED WORD.

NOTE

THIS TEST IS ONLY RUN DURING THE FIRST (QV) PASS WHEN UNDER ACT OR APT, AND IS RUN FOR ECC MEMORY ONLY.

6.2.2.16 PATTERN 17 HOLDING 1'S & 0'S TEST

1. WRITE A 16K BANK WITH ALTERNATING BYTES OF ZEROS & ONES WRITING A BYTE AT A TIME.
2. READ EACH WORD FOR CORRECT PATTERN.
3. DO (1-2) AGAIN FOR A COMPLEMENT PATTERN.

```
2669
2670                    THIS CHECKS THE MEMORY FOR THE CAPABILITY OF HOLDING 0'S & 1'S.
```

6.2.2.17 PATTERN 20 MARCHING 0'S & 1'S IN CHECK BITS TEST

1. WRITE DOUBLE WORDS OF 000000,,000000 WHICH CAUSES CHECK BITS TO EQUAL 077 WHILE ADDRESSING INCREMENTS.
   (WRITE UP/077 --> CHECK BITS)

2. IF IN QUICK VERIFY MODE THEN GO TO STEP (5).

3. READ DOUBLE WORDS & CHECK WHILE WRITING 000000,,100000 AND ADDRESSING DECREMENTS.
   (DOWN/077 --> 100)
   THIS FLIPS ALL THE CHECKBITS.

4. READ DOUBLE WORDS & CHECK WHILE WRITING ZEROS WHILE ADDRESSING INCREMENTS.
   (UP/100 --> 077)

5. READ DOUBLE WORDS & CHECK WHILE WRITING 000000,,100000 & ADDRESSING INCREMENTS.
   (UP/077 --> 100)

6. READ DOUBLE WORDS & CHECK WHILE WRITING ZEROS WHILE ADDRESSING DECREMENTS.
   (DOWN/100 --> 077)

7. READ DOUBLE WORDS & CHECK WHILE ADDRESSING INCREMENTS.
   (UP/077)

THIS CHECKS THE INTEGRITY OF THE MOS CHIPS THAT STORE THE CHECKBITS.

6.2.2.18 PATTERN 21 MARCHING 0'S & 1'S TEST

1. WRITE A BACKGROUND OF ALTERNATING BYTES OF ZEROS & ONES

2. FOR THE 16K BANK ADDRESSING DOWN
   (A) READ CHECK A WORD
   (B) BYTE SWAP A WORD
   (C) READ CHECK A WORD

3. FOR THE 16K BANK ADDRESSING UP
   (A) READ CHECK A WORD
   (B) BYTE SWAP A WORD
   (C) READ CHECK A WORD

4. FOR THE 16K BANK ADDRESSING UP
   (A) READ CHECK A WORD
   (B) BYTE SWAP A WORD
   (C) READ CHECK A WORD

5. FOR THE 16K BANK ADDRESSING DOWN
   (A) READ CHECK A WORD
   (B) BYTE SWAP A WORD
   (C) READ CHECK A WORD

THIS CHECKS THE INTEGRITY OF THE 32 BIT DOUBLE WORDS.

IT CAN EXECUTE OUT OF THE USER DATA PAR'S.

NOTE

IT IS NOT UNCOMMON TO SEE A MISLEADING ERROR TYPEOUT BECAUSE THE SECOND TEST IN EACH CASE IS BASED UPON A BYTESWAP OF THE FIRST TEST WHICH MAY OR MAY NOT HAVE FAILED. IF THE ERROR REPORT INDICATES ERRORS IN PAIRS WITH THE BAD BIT IN THE SECOND REPORT BEING THE SAME BIT POSITION RELATIVE TO A BYTE THEN YOU SHOULD IGNORE THE SECOND ERROR REPORT.

6.2.2.19 PATTERN 22 REFRESH TEST

1. WRITE A DIAGONAL PATTERN OF ONES ON EVERY KDIAG(TH) STRIPE & WRITE ZEROS ELSEWHERE.

   THIS PATTERN IS ON ADDRESSES NOT BIT POSITIONS.

   EXAMPLE:

ADDRESS	MSB'S
LSB'S	0 0 0 1 0 0 0 1
	0 0 1 0 0 0 1 0
	0 1 0 0 0 1 0 0
	1 0 0 0 1 0 0 0
	0 0 0 1 0 0 0 1
	0 0 1 0 0 0 1 0
	0 1 0 0 0 1 0 0
	1 0 0 0 1 0 0 0

NOTE

EXAMPLE USES KDIAG OF VALUE 4 MORE TYPICAL IS A VALUE OF 8. CONSULT THE SYMBOLIC DEFINITION OF 'KDIAG' IN THE PROGRAM LISTING TO BE SURE.

2. DISTURB EACH ROW FOR > 3.2MS

3. READ CHECK DIAGONAL PATTERN

4. DO (1-3) KDIAG TIMES MOVING THE PLACEMENT OF THE DIAGONAL STRIPE TO COVER ALL ADDRESS POSITIONS.

5. DO (1-4) FOR A COMPLEMENT PATTERN (ZEROS IN A BACKGROUND OF ONES)

NOTE

THIS TEST IS NOT NORMALLY EXECUTED EXCEPT UNDER APT OR ACT. IT MAY BE INVOKED VIA FIELD SERVICE COMMAND 13 (KAMIKAZE MODE).

6.2.2.20 PATTERN 23 SHIFTING DIAGONAL PATTERN TEST

SIMILAR IN OVERALL OPERATION TO PATTERN 22 EXCEPT IT DOES NOT DELAY FOR REFRESH AND DISTURB ROWS.

NOTE

THIS TEST IS NOT NORMALLY EXECUTED EXCEPT UNDER APT OR ACT. IT MAY BE INVOKED VIA FIELD SERVICE COMMAND 13 (KAMIKAZE MODE).

6.2.2.21 PATTERN 24 FAST GALLOPING PATTERN TEST

THIS DOES A CLASSICAL GALLOPING PATTERN EXCEPT THAT ADDRESSING IS INCREMENTED BY 400 OCTAL (EVERY 64TH DOUBLE WORD)

NOTE

THIS TEST IS NOT NORMALLY EXECUTED EXCEPT UNDER APT OR ACT. IT MAY BE INVOKED VIA FIELD SERVICE COMMAND 13 (KAMIKAZE MODE).

6.2.2.22 PATTERN 25 INTERRUPT ENABLE TEST

1. SET CSR TO ALLOW UNCORRECTABLE ERROR TRAPS.
2. ACCESS TEST DOUBLE WORDS.
3. CHECK THAT NO UNCORRECTABLE ERROR TRAP OCCURRED.
4. ENABLE CSR FOR SBE TRAPS.
5. ACCESS TEST DOUBLE WORDS.
6. CHECK THAT NO SBE TRAP OCCURRED.
7. WRITE A SBE IN 1 BYTE.
8. DISABLE CSR TRAPS.
9. ACCESS TEST DOUBLE WORDS.
10. CHECK THAT NO TRAPS OCCURRED.
11. ENABLE CSR FOR SBE TRAPS.
12. ACCESS TEST DOUBLE WORDS.
13. CHECK TO INSURE TRAP OCCURRED.
14. DO (7-13) FOR THE 3 OTHER BYTES IN THE DOUBLE WORD.
15. CREATE A DBE IN 1 BYTE.
16. DISABLE CSR TRAPS.
17. ACCESS THE TEST DOUBLE WORD.
18. CHECK THAT NO TRAPS OCCURRED.
19. ENABLE CSR FOR DBE TRAPS.
20. ACCESS THE TEST DOUBLE WORD.
21. CHECK TO INSURE TRAP OCCURRED.
22. ENABLE CSR FOR SBE TRAPS.
23. ACCESS THE TEST DOUBLE WORD.
24. CHECK TO INSURE TRAP OCCURRED.
25. DO (15-24) FOR THE 3 OTHER BYTES IN THE DOUBLE WORD.

THIS INSURES THAT SBE'S & DBE'S GIVE THE CORRECT TYPE OF TRAPS.

NOTE

THIS TEST IS RUN FOR ECC MEMORY ONLY.

6.2.2.23 PATTERN 26 RANDOM DATA TEST

WRITE RANDOM DATA IN A 16K BANK WHILE INCREMENTING THE ADDRESSES.

READ CHECK RANDOM DATA.

THIS ROUTINE REGENERATES THE SAME RANDOM NUMBERS BY USING THE SAME

SEED AS THE WRITE SEQUENCE. AFTER THE READ CHECK THE SEED IS UPDATED SO THAT THE NEXT USE OF THIS PATTERN WILL NOT INVOKE THE SAME SEQUENCE OF RANDOM NUMBERS.

IF YOU WISH TO CHANGE THE RANDOM SEQUENCE SO THAT IT IS DIFFERENT THAN ANY OTHER RUN IN THE SAME CONFIGURATION THEN THERE ARE 2 WAYS OF DOING SO.

1. MODIFY SYMBOLIC LOCATIONS ''SEEDHI'' AND ''SEEDLO'' TO ANY NUMBER YOU LIKE.

2. ENTER FIELD SERVICE MODE AND EXECUTE THIS PATTERN (COMMAND 5) ON SOME (ANY GOOD) BANK FOR A SHORT TIME (30 SEC OR SO).

THIS CAN EXECUTE OUT OF THE USER DATA PAR'S, THE KERNEL DATA PAR'S, AND THE SUPERVISOR DATA PAR'S.

6.2.2.24 PATTERN 27 UNIQUE BANK TEST

THIS PATTERN USES PATTERN 0 TO WRITE & READ THE BANK NUMBER IN EACH BANK.

IT DOES NOT TEST BANKS THAT REQUIRE RELOCATION TO TEST.

IT DOES NOT RUN AS PART OF ANY SCRIPT BUT RATHER IS ALWAYS RUN AFTER NORMAL PATTERN TESTS ARE COMPLETE.

6.2.2.25 PATTERN 30 FLUSH OUT DBE'S TEST

THIS READS EACH LOCATION THEN MOVES THE OLD VALUE BACK IN. THIS IS DONE WITH ECC DISABLED AND THEREFORE CORRECTS ANY DBE'S OR SBE'S (IF POSSIBLE).

IT DOES NOT RUN AS PART OF ANY SCRIPT BUT RATHER IS ALWAYS RUN JUST PRIOR TO THE END OF PASS CODE, AS PART OF A CONTROL "C" (BOOT) COMMAND, AS PART OF END OF PASS SHUTDOWN FOR ACT OR XXDP CHAIN MODE, AS PART OF HANGING SEQUENCE AFTER AN ERROR IF UNDER ACT OR APT, AND AS PART OF A SHUTDOWN SEQUENCE DIRECTED BY SWITCH 8 (HALT PROGRAM).

6.2.2.26 PATTERN 31 SOB-A-LONG TEST

RATIONALIZATION

IN ORDER TO CONCENTRATE THE MEMORY CYCLES OF A TEST INTO A PARTICULAR ADDRESS, WE MUST CUT THE OVERHEAD CYCLES TO A MINIMUM. FREQUENTLY, THE INSTRUCTION ITSELF MAY PROVIDE ADEQUATE DATA OR SET UP A BACKGROUND IN WHICH ANY COMPLEMENTED BIT MAY FIND IT HARD TO SURVIVE.

THE SOB INSTRUCTION IS THE ONLY PDP-11 INSTRUCTION THAT IS (1) A SINGLE OPERAND, (2) CAN BE REPEATEDLY EXECUTED AT THE SAME PC AND, (3) CAN ESCAPE THIS REPETITIOUS LOOP.

HENCE, IT CAN BE POSSIBLE TO SOB A MOS CELL TO DEATH (OR AT LEAST BRAIN WASH HIM), AND TO SOB A CORE INTO OVER-HEATING (OR AT LEAST WARM DISCOMFORT).

THE SOB ROUTINE WILL BE LOADED AND CALLED WITH R0 SET EQUAL TO THE SOB CONSTANT "SOBK", R1 SET EQUAL TO THE COMPLEMENT OF A "SOB R0,." INSTRUCTION "100776".

SIMPLIFIED SOB EXAMPLE:

```
1$:     SOB         R0,1$                   ;SOB TILL R0 UNDERFLOWS
        MOV         R1,1$                   ;WRITE COMPLEMENT OF SOB
        CMP         R1,1$                   ;READ & CHECK NOT SOB
        BEQ           2$                    ;SKIP IF OK
        SOBFAIL                             ;TRAP & REPORT ERROR
2$:     SOBMOV1                             ;CODE TO GET SELF MOVED
        SOBMOV2                             ;FORWARD 1 WORD AND RUN AGAIN
        SOBMOV3
        SOBMOV4
        SOBMOV...
```

THE VALUE OF THE SOB CONSTANT CAN BE FOUND AT SYMBOLIC LOCATION "SOBK" (TYPICAL 25 DECIMAL).

THIS TEST IS NOT IN THE NORMAL SCRIPT OF EXECUTION BUT MAY BE ADDED VIA THE APT E-TABLE, REFERENCE SYMBOLIC LOCATIONS 'MKPAT', 'MJPAT', '$DDW2-5". FIELD SERVICE MODE COMMAND 8 IS THE NORMAL METHOD OF RUNNING THIS PATTERN.

NOTE

THIS TEST IS NOT NORMALLY EXECUTED EXCEPT UNDER APT OR ACT. IT MAY BE INVOKED VIA FIELD SERVICE COMMAND 13 (KAMIKAZE MODE).

6.2.2.27 PATTERN 32 WRITE RECOVERY TEST

THIS TEST CAUSES A WRITE, READ, WRITE, READ, ... TO OCCUR IN MEMORY AND IF THE 1ST, 3RD, 5TH, ...READ IS BAD THE PROGRAM MAY BOMB OR IF THE 2ND, 4TH, 6TH, ... READ IS BAD THE PROGRAM WILL GRACEFULLY TYPE OUT THE ERROR.

WRITE RECOVERY TEST
THIS TEST DIFFERS FROM OTHER TESTS IN THAT IT CONSISTS
OF A SMALL TEST PROGRAM ACTUALLY RUNNING IN THE BANK UNDER TEST.
THE PROGRAM IS SELF MODIFYING AND MAY BE DIFFICULT TO DEBUG.
TO AID IN THE DEBUG, REMEMBER THAT THE BANK AND MARGIN ARE BEING
DISPLAYED. THIS WILL ALLOW THE USER TO AT LEAST SEE WHICH MEMORY
BANK FAILED.
THE TEST CONSISTS OF 1/2 OF THE BANK STORED WITH "MOV R2,-(PC)"
AND THE OTHER 1/2 CONTAINING "177667". "177667" IS THE COMPLEMENT
OF "JMP (R0)" INSTRUCTION. R2 CONTAINS "COM -(R1)" INSTRUCTION
ON ENTRY TO THE BANK AND R1 CONTAINS THE HIGHEST TEST ADDRESS IN
THAT BANK.
IF YOU UNDERSTAND THIS SO FAR THE REST IS EASY.
THE TEST EXECUTION IS AS FOLLOWS:
1. THE "MOV R2,-(PC)" INSTRUCTION EXECUTES STORING
THE CONTENTS OF R2 IN THE ADDRESS IT VACATED (DUE TO -(PC).
2. SINCE R2 CONTAINS A "COM -(R1)" INSTRUCTION IT COMPLEMENTS
THE HIGHEST ADDRESS UNDER TEST. THIS ADDRESS CONTAINED
"177667" SO AFTER THE COM -(R1) IT EQUALS 110
CLEVERLY THIS IS THE "JMP (R0)" INSTRUCTION.
3. THIS SEQUENCE CONTINUES UNTIL THE "MOV R2,-(PC)"
INSTRUCTIONS REACH THE MIDDLE OF THE TEST BANK.
THEN THE "JMP (R0)" INSTRUCTION IS MET
AND EXECUTED. R0 CONTAINED THE RETURN ADDRESS BACK
TO TEST 13.
4. THESE STEPS ARE REPEATED FOR EACH BANK UNDER TEST.

NOTE

THIS TEST IS NOT NORMALLY EXECUTED
EXCEPT UNDER APT OR ACT. IT MAY BE
INVOKED VIA FIELD SERVICE COMMAND 13
(KAMIKAZE MODE).

6.2.2.28 PATTERN 33 BRANCH GOBBLE TEST

THIS TEST LOADS A SMALL ROUTINE INTO THE MEMORY UNDER TEST. THE ROUTINE MOVES ITSELF ALONG IN MEMORY ONE WORD AFTER EACH 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 ALFHA 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 TC PRESERVE THE SOFTWARE STRUCTURE.

THIS PATTERN REPLACES ANY REAL PATTERNS WHEN THE APT E-TABLt DOES NOT SPECIFY A PATTERN TO BE RUN.

7.0 PROGRAM FEATURES

7.1 FAST DATA ACCESS RATES

ONE OF THE MAIN AREAS OF CONCERN IN TESTING MEMORY IN SYSTEMS ENVIRONMENTS IS SPEED. ONE OF THE PRIME REASONS THAT SYSTEM PROGRAMS LIKE RSTS,IAS AND MUMPS CAN CRASH DUE TO MEMORY FAILURES NOT DETECTABLE BY MEMORY DIAGNOSTICS (0-124K,0-2 MEG,ETC.) IS BECAUSE OF MULTIPLE NPR DEVICES CONTENDING FOR THE BUS. AFTER SOME DELAY A NPR DEVICE BECOMES BUS MASTER AND DOES SEVERAL MEMORY TRANSFERS AT MEMORY DATA RATES.

ON THE OTHER HAND MOST DIAGNOSTICS WHEN WRITING READING AND/OR CHECKING PATTERNS SPEND MOST OF THEIR TIME FETCHING INSTRUCTIONS AND OPERANDS OUT OF THEIR PROGRAM SPACE AND PROPORTIONALLY LITTLE TIME ACCESSING THE MEMORY UNDER TEST.

THIS DIAGNOSTIC'S ERROR DETECTING ABILITIES HAVE BEEN OPTIMIZED AROUND THE PRIMARY DESIGN CRITERIA OF SPEED. TO THIS END THE FOLLOWING STEPS HAVE BEEN TAKEN.

7.1.1 FAST CITY

UTILIZATION OF MEMORY MANAGEMENT REGISTERS AS NON MEMORY BUS, NON UNIBUS, BIPOLAR MEMORY. SINCE USER MODE IS ONLY USED FOR RELOCATION AND DATA SPACE IS NEVER USED, THEN SUBROUTINES CAN BE EXECUTED FROM THE UIPAR'S, UDPAR'S, KDPAR'S, SDPAR'S AND WITH SOME BIT PATTERN RESTRICTIONS THE UIPDR'S, UDPDR'S, KDPDR'S, AND SDPDR'S.

THE PROGRAM RUNS IN KERNEL MODE AND PATTERNS ARE EXECUTED IN SUPERVISOR MODE FOR MAPPING PURPOSES. ALL CORE PATTERNS AND SOME MOS PATTERNS ARE SUBROUTINES THAT ARE MOVED TO THIS BIPOLAR REGION REFERRED TO IN THE PROGRAM AS FAST CITY.

NOTE

18-BIT PDP-11'S CANNOT EXECUTE FROM THE PAR'S BECAUSE THEIR PAR'S ARE ONLY 12 BITS WIDE; THEY ALSO HAVE NO SUPERVISOR MODE. THEREFORE, ALL PATTERNS ARE EXECUTED IN MEMORY, USING USER MODE (REFERENCE SECTION 7.5).

7.1.2 SOB'S

UTILIZATION OF THE FULL PDP-11 INSTRUCTION SET TO SPEED PATTERN ALGORITHMS (PRINCIPALLY THE SOB).

7.1.3 CACHE

CACHE IS USED BETWEEN PATTERN TESTS TO DECREASE PROGRAM PASS TIMES. CACHE CAN BE DEFEATED BY THE OPERATOR (REFERENCE SECTION 2.4.3.1).

7.2 BANK ZERO TESTING

BANK ZERO HAS BEEN TRADITIONALLY NEGLECTED BY MEMORY DIAGNOSTICS FOR THE FOLLOWING REASON.

THE VECTOR SPACE EXISTS THERE AND ALL TRAPS MUST NOT ACCESS TEST PATTERN DATA. IF THE AREA IS TESTED THE DIAGNOSTIC MUST NOT USE ANY TRAPS, AND IT IS AGAINST THE RULES FOR POWER TO FAIL.

SYSTEMS WITH MEMORY MANAGEMENT CAN OVERCOME THIS BECAUSE ALL TRAPS ARE TO KERNEL VIRTUAL SPACE EVEN IF THE POWER SHOULD FAIL (CAUTION MUST BE OBSERVED BECAUSE POWER UP GOES TO PHYSICAL ADDRESS 24 (BECAUSE THE MEMORY MANAGEMENT UNIT COMES UP OFF)).

HOWEVER, CATCH 22 IS THAT THE DIAGNOSTIC IS NOT APT COMPATIBLE IN THIS MODE BECAUSE APT ACCESSES PHYSICAL MEMORY LOCATIONS.

THE PDP-11/44 CAN OVER COME THIS BECAUSE THE UNIBUS MAP CAN FOOL APT.

BECAUSE OF THE PREVIOUS ARGUMENTS THIS PROGRAM DOES NOT RELOCATE IN THE TRUE SINCE OF THE WORD (I.E. NO POSITION INDEPENDENT CODE WAS WRITTEN (AT LEAST NOT ON PURPOSE)), BUT RATHER THIS PROGRAM MOVES AND REMAPS (HEREAFTER REFERRED TO AS RELOCATES). THIS ENABLES THE COMPLETE TESTING OF BANK ZERO OR ANY OTHER PROGRAM SPACE OR PRIVILEGED SPACE EXACTLY AS ALL OTHER BANKS ARE TESTED. (THE CONDITIONAL TEST TO SEE IF A BANK IS PROTECTED IS COMPLEMENTED WHEN RELOCATED).

NOTE

THE PROGRAM WILL RELOCATE ONLY IN THE FIRST PASS UNDER APT; AFTER THIS, THE PROGRAM WILL REMAIN FIXED IN BANKS 0 AND 1.

7.3 MEMORY CONFIGURATION MAP

THIS MAP IS PRINTED OUT IMMEDIATELY AFTER SIZING THE MEMORY UNLESS SW6 IS SET (REFERENCE SECTION 2.4.1). IT CAN ALSO BE PRINTED AT ANY LATER TIME IN FIELD SERVICE MODE (REFERENCE SECTION 2.4.4.8.7)

EXAMPLE:

```
                        MEMORY CONFIGURATION MAP
                                16K BANKS
                  1         2         3         4         5         6         7
         01234567012345670123456701234567012345670123456701234567012345670123
ERRORS             XX
CPU MAP  11111111111111111111111111111111
INTRLV   ----------------3333333333333333
MEMTYPE  LLLLLLLLMMMMMMMMMMMMMMMMMMMMMMMM
CSR      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

```
3343                                    MEMORY  TYPE  M  (MS11-M);  WHILE BANKS 40-167 DO NOT EXIST.
3344                                    MEMORY TYPE K WOULD INDICATE MF11S-K, MEMORY  TYPE  P  WOULD
3345                                    INDICATE  UNIBUS  PARITY,  AND  MEMORY TYPE B WOULD INDICATE
3346                                    11/45-TYPE BIPOLAR MEMORY.
3347
3348                                CSR:
3349
3350                                    BANKS 0-7 ARE ASSIGNED TO CSR 172100, 10-17 TO  CSR  172102,
3351                                    AND 20-37 TO INTERLEAVED CSR'S 172104 AND 172106.
3352
3353                                PROTECT:
3354
3355                                    BANKS 0 AND 1 ARE PROTECTED BECAUSE THEY ARE PROGRAM  SPACE.
3356                                    BANK  0  AND 1 CAN ALSO BE PROTECTED BECAUSE THEY ARE IN THE
3357                                    BOTTOM 16K OF AN MS11-M CSR.  THE PROTECTION IS HIERARCHICAL
3358                                    AND  PROGRAM  SPACE  OVERSHADOWS MS11-M PROTECTION.  BANKS 0
3359                                    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 LNE IS VERY IMPORTANT. THE ELEMENTS SHOULD BE SEPARATED BY A COMMA OR A SPACE. TABS SHOULD NEVER BE USED FOR SPACING. FOR EXAMPLE: THE EXPRESSION A+B IS INTERPRETED DIFFERENT THAN A + B.

ALL THE CONDITIONAL STATEMENTS CAN BE WRITTEN AS MULTIPLE-LINE NESTED 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
                               BIT5_=40
                        A:     0
                        B:     0
                        C:     0
                        D:     0
                        E:     0
                        F:     0
                        G:     0
                        H:     0
                        I:     0
                        J:     0
                               .PAGE
                        ;LET EXAMPLES
                               LET R0 :_= A
                               LET B :_= C + D
                               LET E :_= F +  1
                               LET G :_= H +  2
                               LET J :_= J +  01
                               LET A :B_= B
                      ;IF EXAMPLES
                               IF A IS TRUE
                                 MOV    23,D
                               END ;OF IF A
                               IF B IS FALSE
                                 MOV    34,E
                               END ;OF IF B
                               IF A EQ B THEN LET C :_= D
                               IF A LT B
                                 MOV   C,D
                               ELSE
                                 MOV   E,D
                               END ;OF IF A
                               IF A EQ B AND C NE D
                                 MOV   F,G
                               END ;OF IF A
                               IF A EQ B OR C NE D
                                 MOV   F,G
                               END ;OF IF A
                               IFB A EQ B AND C EQ  1
                                 MOV   H,J
                               ELSE
                                 MOV   E,J
                               END ;OF IFB A
                               IFB A EQ B ANDB C EQ  1
                                 MOV   H,J
                               ELSE
                                 MOV   E,J
                               END ;OF IFB A
```

```
3460                                    IF RESULT IS EQ
3461                                      MOV   A,B
3462                                    END ;OF IF RESULT
3463                                    IF  BIT5 SET.IN A
3464                                      MOV   B,C
3465                                    END ;OF IF  BIT5
3466                                    IF  BIT5 OFF.IN A
3467                                      MOV   C,D
3468                                    END ;OF IF  BIT5
3469                                ;ON.ERROR IS LIKE AN IF STATEMENT ON THE C-BIT
3470                                ;ON.ERROR EXAMPLES
3471                                    ON.ERROR
3472                                      MOV   A,B
3473                                    ELSE
3474                                      MOV   C,B
3475                                    END ;OF ON.ERROR
3476                                    ON.NOERROR
```

```
3478
3479
3480
3481                                    MOV   C,B
3482                                  ELSE
3483                                    MOV   A,B
3484                                  END ;OF ON.NOERROR
3485                                  ON.ERROR THEN LET A :B_= B
3486                          ;FOR EXAMPLES
3487                                  FOR I :_=  -5 TO  23
3488                                    INC   A
3489                                  END ;OF FOR I
3490                                  FOR R0 :_=  0 TO  140 BY  4
3491                                    DEC   A(R0)
3492                                  END ;OF FOR R0
3493                                  FOR I :_=  133 DOWNTO  3 BY  2
3494                                    ADD   A,B
3495                                  END ;OF FOR I
3496                          ;BEGIN EXAMPLES
3497                                  BEGIN ALPHA
3498                                    FOR R0 :_=  0 TO  167
3499                                      MOVB        A(R0),B
3500                                      IF B LT  0 THEN LEAVE ALPHA
3501                                    END ;OF FOR R0
3502                                    FOR R0 :_=  400 TO  567
3503                                      IF B GE  0 THEN LEAVE ALPHA
3504                                    END ;OF FOR R0
3505                                  END ALPHA
3506                          ;$RETURN EXAMPLES
3507                                  $RETURN
3508                                  $RETURN ERROR
3509                                  $RETURN NOERROR
3510                          ;CASE EXAMPLES
3511                                  MOV     A,R0
3512                                  CASE R0
3513                                    A
3514                                    B
3515                                    C
3516                                    D
3517                                    E
3518                                    F
3519                                  END ;OF CASE R0
3520
3521                                  .END
```

7.4.2 SAMPLE LISTING FILE (WITH NO EXPANDED MACROS) - -


```
                                                                .ENABL ABS
                                                                .ENABL AMA
                                                                .MCALL .SUPER
000000                                                          .SUPER
                                                                ;LST$$ =0
                    000040                                      BIT5_=40
000000  000000                                A:      0
000002  000000                                B:      0
000004  000000                                C:      0
000006  000000                                D:      0
000010  000000                                E:      0
000012  000000                                F:      0
000014  000000                                G:      0
000016  000000                                H:      0
000020  000000                                I:      0
000022  000000                                J:      0
```

```
3547
3548
3549
3550
3551
3552
3553                    18                                  ;LET EXAMPLES
3554                    19 000024                                   LET R0 :_= A
3555                    20 000030                                   LET B :_= C + D
3556                    21 000044                                   LET E :_= F +  1
3557                    22 000056                                   LET G :_= H +  2
3558                    23 000072                                   LET J :_= J +  01
3559                    24 000100                                   LET A :B_= B
3560                    25                                  ;IF EXAMPLES
3561                    26 000106                                   IF A IS TRUE
3562                    27 000114  012737  000023  000006          MOV    23,D
3563                    28 000122                                   END ;OF IF A
3564                    29 000122                                   IF B IS FALSE
3565                    30 000130  012737  000034  000010          MOV    34,E
3566                    31 000136                                   END ;OF IF B
3567                    32 000136                                   IF A EQ B THEN LET C :_= D
3568                    33 000154                                   IF A LT B
3569                    34 000164  013737  000004  000006          MOV   C,D
3570                    35 000172                                   ELSE
3571                    36 000174  013737  000010  000006          MOV   E,D
3572                    37 000202                                   END ;OF IF A
3573                    38 000202                                   IF A EQ B AND C NE D
3574                    39 000222  013737  000012  000014          MOV   F,G
3575                    40 000230                                   END ;OF IF A
3576                    41 000230                                   IF A EQ B OR C NE D
3577                    42 000250  013737  000012  000014          MOV   F,G
3578                    43 000256                                   END ;OF IF A
3579                    44 000256                                   IFB A EQ B AND C EQ  1
3580                    45 000276  013737  000016  000022          MOV   H,J
3581                    46 000304                                   ELSE
3582                    47 000306  013737  000010  000022          MOV   E,J
3583                    48 000314                                   END ;OF IFB A
3584                    49 000314                                   IFB A EQ B ANDB C EQ  1
3585                    50 000334  013737  000016  000022          MOV   H,J
3586                    51 000342                                   ELSE
3587                    52 000344  013737  000010  000022          MOV   E,J
3588                    53 000352                                   END ;OF IFB A
3589                    54 000352                                   IF RESULT IS EQ
3590                    55 000354  013737  000000  000002          MOV   A,B
3591                    56 000362                                   END ;OF IF RESULT
3592                    57 000362                                   IF  BIT5 SET.IN A
3593                    58 000372  013737  000002  000004          MOV   B,C
3594                    59 000400                                   END ;OF IF  BIT5
3595                    60 000400                                   IF  BIT5 OFF.IN A
3596                    61 000410  013737  000004  000006          MOV   C,D
3597                    62 000416                                   END ;OF IF  BIT5
3598                    63                          ;ON.ERROR IS LIKE AN IF STATEMENT ON THE C-BIT
3599                    64                          ;ON.ERROR EXAMPLES
3600                    65 000416                                   ON.ERROR
3601                    66 000420  013737  000000  000002          MOV   A,B
3602                    67 000426                                   ELSE
3603                    68 000430  013737  000004  000002          MOV   C,B
```

```
3604                    69 000436                       END ;OF ON.ERROR
3605                    70 000436                       ON.NOERROR
3606                    71 000440  013737  000004  000002    MOV   C,B
3607                    72 000446                       ELSE
3608                    73 000450  013737  000000  000002    MOV   A,B
3609                    74 000456                       END ;OF ON.NOERROR
```

```
                                                        ON.ERROR THEN LET A :B_= B
                                                ;FOR EXAMPLES
                                                        FOR I :_=  -5 TO  23
                                                          INC   A
                                                        END ;OF FOR I
                                                        FOR R0 :_=  0 TO  140 BY  4
                                                          DEC   A(R0)
                                                        END ;OF FOR R0
                                                        FOR I :_=  133 DOWNTO  3 BY  2
                                                          ADD   A,B
                                                        END ;OF FOR I
                                                ;BEGIN EXAMPLES
                                                        BEGIN ALPHA
                                                          FOR R0 :_=  0 TO  167
                                                            MOVB        A(R0),B
                                                            IF B LT  0 THEN LEAVE ALPHA
                                                          END ;OF FOR R0
                                                          FOR R0 :_=  400 TO  567
                                                            IF B GE  0 THEN LEAVE ALPHA
                                                          END ;OF FOR R0
                                                        END ALPHA
                                                ;$RETURN EXAMPLES
                                                        $RETURN
                                                        $RETURN ERROR
                                                        $RETURN NOERROR
                                                ;CASE EXAMPLES
                                                        MOV     A,R0
                                                        CASE R0
                                                          A
                                                          B
                                                          C
                                                          D
                                                          E
                                                          F
                                                        END ;OF CASE R0

                                                        .END
```

PAGE 81

7.4.3 SAMPLE LISTING FILE (WITH EXPANDED MACROS) - -

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

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

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

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

```
3736                                 000230                                  L5:
3737                              41 000230
3738                                 000230  023737  000000  000002          IF A EQ B OR C NE D
3739                                 000236  001404                          CMP A,B
3740                                 000240  023737  000004  000006          BEQ L6
3741                                 000246  001403                          CMP C,D
                                                                             BEQ L7
```

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

                    000400  032737  000040  000000          BIT  BIT5,A
                    000406  001003                          BNE L16
                 61 000410  013737  000004  000006                MOV   C,D

                 62 000416                                  END ;OF IF  BIT5
                    000416                                  L16:
                 63                                         ;ON.ERROR IS LIKE AN IF STATEMENT ON THE C-BIT
                 64                                         ;ON.ERROR EXAMPLES
                 65 000416                                          ON.ERROR
                    000416  103004                          BCC L17
```

```
3800                            66 000420  013737  000000  000002          MOV   A,B
3801                            67 000426                               ELSE
3802                               000426  000403                       BR L20
3803                               000430                       L17:
3804
3805                            68 000430  013737  000004  000002          MOV   C,B
3806
3807                            69 000436                               END ;OF ON.ERROR
3808                               000436                       L20:
3809                            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
```

```
3868                                    000604  005200                  INC R0
3869                                    000606  020027  000167          CMP R0, 167
3870                                    000612  003766                  BLE B4
3871                                    000614                  E4:
3872                                 92 000614                    FOR R0 := 400 TO 567
3873                                    000614  012700  000400          MOV  400,R0
```

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

7.5 MEMORY MANAGEMENT MAPPING

7.5.1 MEMORY MANAGEMENT MAPPING FOR THE 11/44 -

PAR	SUPERVISOR	KERNEL	USER
0	PROGRAM	PROGRAM	DST BK/FST MEM
1	PROGRAM	PROGRAM	SRC BK/FST MEM
2	PROGRAM	PROGRAM	SRC BK/FST MEM
3	TEST AREA	PROGRAM	SRC BK/FST MEM
4	TEST AREA	PROGRAM	DST BK/FST MEM
5	TEST AREA	PROGRAM	DST BK/FST MEM
6	TEST AREA	MAP TO CSR'S	DST BK/FST MEM
7	PERIF PAGE	PERIF PAGE	DST BK/FST MEM

7.5.2 MEMORY MANAGEMENT MAPPING FOR UNIBUS-11'S WITH SUPERVISOR MODE (EG 11/45) -

PAR	SUPERVISOR	KERNEL	USER
0	PROGRAM	PROGRAM	DST BK
1	PROGRAM	PROGRAM	SRC BK
2	PROGRAM	PROGRAM	SRC BK
3	TEST AREA	PROGRAM	SRC BK
4	TEST AREA	PROGRAM	DST BK
5	TEST AREA	PROGRAM	DST BK
6	TEST AREA	MAP TO CSR'S	DST BK
7	PERIF PAGE	PERIF PAGE	DST BK

7.5.3 MEMORY MANAGEMENT MAPPING FOR UNIBUS-11'S W/O SUPERVISOR MODE (EG 11/34) -

PAR	KERNEL	USER
0	PROGRAM	PROGRAM/DST BK
1	PROGRAM	PROGRAM/SRC BK
2	PROGRAM	PROGRAM/SRC BK
3	PROGRAM	TEST AREA/SRC BK
4	PROGRAM	TEST AREA/DST BK
5	PROGRAM	TEST AREA/DST BK
6	MAP TO CSR'S	TEST AREA/DST BK
7	PERIF PAGE	PERIF PAGE/DST BK

```
3968 000000                            .ENABL  ABS
3969                                   .ENABL  AMA
3970                                   .DSABL  GBL
3971                                   ;NOTE:  CZMSDC.SML IS THE SUPER.MAC SOURCE AND IS RELEASED WITH
3972                                   ;THIS PROGRAM.  ALL THESE .MCALL STATEMENTS REFERENCE THAT FILE.
3973                                   .MCALL  SMACIT,..PUSH,..POP,..TAG,..BRAN,..EMIT,..EMITN,..EMITL,..EMITR
3974                                   .MCALL  .IFOPR,.IS,.GENBR,..OPADD,..OPSUB,CLEAR,SET,CLEARB,SETB
3975                                   .MCALL  RNE,REQ,RLT,RGE,RGT,RLE,RPL,RMI,RHI,RLOS,RHIS,RLO,RCS,RCC
3976                                   .MCALL  IF,..OR,..IFARI,..LEAVE,..GOTO,OR,AND,THEN,ELSE,WHILE,CASE
3977                                   .MCALL  FOR,TO,DOWNTO,REPEAT,UNTIL,THRU,END,BEGIN
3978                                   .MCALL  $$END,LEAVE,JUMPTO,GOTO,PUSH,POP,LET
3979                                   .MCALL  .SIMPLE,..ARITH,ORB,ANDB,IFB,UNTILB,WHILEB,ON.ERROR,ON.NOERROR
3980                                   .MCALL  $CALL,$RETURN
3981
3982                                   .NLIST  TTM                     ;I WANT FAT PAPER!
3983                                   .LIST   MC,SYM                  ;LIST MACRO CALLS, SYMBOL TABLE
3984                                   .NLIST  MD,CND,ME               ;DON'T LIST MACRO DEFS & CONDITIONALS & EXPANSIONS
3985                                   ;LST$$= 0                       ;DEFINED TO LIST SUPERMAC EXPANSIONS
3986          163000                   $SWR=   163000                  ;USE THESE SYSMAC SWITCHES
3987          000001                   $TN=    1                       ;FIRST TEST NUMBER TO ONE(1)
3988 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
```

```
4048
4049          104425          LOADCSR=104425          ;LOAD CORRECT CSR
4050          104426          READCSR=104426          ;READ CORRECT CSR
4051
4052          104427          PERR01= 104427          ;PROGRAM DETECTED ERROR
4053          104430          PERR02= 104430          ;PROGRAM DETECTED ERROR
4054          104431          PERR03= 104431          ;PROGRAM DETECTED ERROR
4055          104432          PERR04= 104432          ;PROGRAM DETECTED ERROR
4056          104433          PERR07= 104433          ;PROGRAM DETECTED ERROR
4057          104434          PERR10= 104434          ;PROGRAM DETECTED ERROR
4058          104435          PERR11= 104435          ;PROGRAM DETECTED ERROR
4059          104436          PERR12= 104436          ;PROGRAM DETECTED ERROR
4060          104437          PERR13= 104437          ;PROGRAM DETECTED ERROR
4061          104440          PERR14= 104440          ;PROGRAM DETECTED ERROR
4062          104441          PERR15= 104441          ;PROGRAM DETECTED ERROR
4063          104442          PERR16= 104442          ;PROGRAM DETECTED ERROR
4064          104443          PERR17= 104443          ;PROGRAM DETECTED ERROR
4065          104444          PERR20= 104444          ;PROGRAM DETECTED ERROR
4066          104445          PERR21= 104445          ;PROGRAM DETECTED ERROR
4067          104446          PERR22= 104446          ;PROGRAM DETECTED ERROR
4068          104447          PERR23= 104447          ;PROGRAM DETECTED ERROR
4069          104450          PERR24= 104450          ;PROGRAM DETECTED ERROR
4070          104451          PERR25= 104451          ;PROGRAM DETECTED ERROR
4071          104452          PERR26= 104452          ;PROGRAM DETECTED ERROR
4072          104453          PERR27= 104453          ;PROGRAM DETECTED ERROR
4073          104454          PERR30= 104454          ;PROGRAM DETECTED ERROR
4074          104455          PERR31= 104455          ;PROGRAM DETECTED ERROR
4075          104456          PERR32= 104456          ;PROGRAM DETECTED ERROR
4076          104457          PERR33= 104457          ;PROGRAM DETECTED ERROR
4077          104460          PERR34= 104460          ;PROGRAM DETECTED ERROR
4078          104461          PERR35= 104461          ;PROGRAM DETECTED ERROR
4079          104462          PERR36= 104462          ;PROGRAM DETECTED ERROR
4080          104463          PERR37= 104463          ;PROGRAM DETECTED ERROR
4081          104464          PERR40= 104464          ;PROGRAM DETECTED ERROR
4082          104465          PERR41= 104465          ;PROGRAM DETECTED ERROR
4083          104466          PERR42= 104466          ;PROGRAM DETECTED ERROR
4084          104467          PERR43= 104467          ;PROGRAM DETECTED ERROR
4085
4086          104470          ECCDIS= 104470          ;DISABLE ECC ON ALL CSR'S
4087          104471          ECC1DIS=104471          ;DISABLE ECC ON 1 SELECTED CSR
4088          104472          ECCINIT=104472          ;INITIALIZE ALL ECC CSR'S
4089          104473          ECC1INIT=104473         ;INITIALIZE 1 SELECTED ECC CSR
4090          104474          CBCSR=  104474          ;WRITE GENERATED CHECKBITS IN ALL CSR'S
4091          104475          CB1CSR= 104475          ;WRITE GENERATED CHECKBITS IN 1 SELECTED CSR
4092          104476          WASSBE= 104476          ;WAS THERE A SBE ON ANY CSR?
4093          104477          WAS1SBE=104477          ;WAS THERE A SBE ON 1 SELECTED CSR?
4094          104500          WASDBE= 104500          ;WAS THERE A DBE ON ANY CSR?
4095          104501          WAS1DBE=104501          ;WAS THERE A DBE ON 1 SELECTED CSR?
4096          104502          CLRCSR= 104502          ;CLEAR ALL CSR'S
4097          104503          CLR1CSR=104503          ;CLEAR 1 SELECTED CSR
4098          104504          CHKDIS= 104504          ;DISABLE ECC & WRITE CHECKBITS FROM ALL CSR'S
4099          104505          CHK1DIS=104505          ;DISABLE ECC & WRITE CHECKBITS FROM 1 SELECTED CSR
4100          104506          ENASBE= 104506          ;ENABLE TRAPS ON SBE'S FROM ALL CSR'S
4101          104507          ENA1SBE=104507          ;ENABLE TRAPS ON SBE'S FROM 1 SELECTED CSR
4102          104510          TSTREAD=104510          ;TEST LOC (R1) & TST FOR SBE (WITHOUT FETCHES)
4103          104511          INVALID=104511          ;INVALIDATE BACKGROUND PATTERN ON 'BANK'
4104          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
```

```
4163          000200                BIT7=   200
4164          000100                BIT6=   100
4165          000040                BIT5=   40
4166          000020                BIT4=   20
4167          000010                BIT3=   10
4168          000004                BIT2=   4
4169          000002                BIT1=   2
4170          000001                BIT0=   1
4171
4172                                ;*BASIC "CPU" TRAP VECTOR ADDRESSES
4173          000004                ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
4174          000010                RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
4175                                ;TBITVEC=14             ;;"T" BIT
4176                                ;TRTVEC=        14              ;;TRACE TRAP
4177                                ;BPTVEC=        14              ;;BREAKPOINT TRAP (BPT)
4178          000020                IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
4179          000024                PWRVEC= 24              ;;POWER FAIL
4180          000030                EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
4181          000034                TRAPVEC=34              ;;"TRAP" TRAP
4182          000060                TKVEC=  60              ;;TTY KEYBOARD VECTOR
4183                                ;TPVEC= 64              ;;TTY PRINTER VECTOR
4184                                ;LKVEC= 100             ;;LINE CLOCK (KW11-L) VECTER
4185          000114                CACHVEC=114             ;;CACHE ERROR INTERRUPT VECTOR
4186          000114                PARVEC=CACHVEC
4187                                ;PIRQVEC=240            ;;PROGRAM INTERRUPT REQUEST VECTOR
4188          000250                MMVEC=  250             ;;MEMORY MANAGEMENT VECTOR
4189                                        .SBTTL  DEFINE  CACHE REGISTERS
4190                                ;MEMERR = 177744                ;;CACHE ERROR REGISTER
4191          177746                CONTRL = 177746                 ;;MEMORY CONTROL REGISTER
4192          177750                MAINT  = 177750                 ;;MEMORY MAINTENENCE REGISTER
4193                                ;HITMIS = 177752                ;;HIT MISS REGISTER "1" IMPLIES HIT IN CACHE
4194          177754                DATARG  = 177754                ;;DATA REGISTER
4195
4196                                        .SBTTL  DEFINE  CPU REGISTERS
4197          177766                CPUERR = 177766         ;;CPU ERROR REGISTER HOLDS CONDITION THAT CAUSED
4198
4199                                        .SBTTL  DEFINE  MEMORY MANAGEMENT REGISTERS
4200                                        ;*MEMORY MANAGEMENT STATUS REGISTER ADDRESSES
4201          177572                MMR0=   177572
4202          177574                MMR1=   177574
4203          177576                MMR2=   177576
4204          172516                MMR3=   172516
4205
4206                                        ;*USER "I" PAGE DESCRIPTOR REGISTERS
4207          177600                UIPDR0= 177600
4208                                ;UIPDR1=        177602
4209                                ;UIPDR2=        177604
4210                                ;UIPDR3=        177606
4211                                ;UIPDR4=        177610
4212                                ;UIPDR5=        177612
4213                                ;UIPDR6=        177614
4214                                ;UIPDR7=        177616
4215
4216                                        ;*USER "D" PAGE DESCRIPTOR REGISTORS
4217                                ;UDPDR0=        177620
4218                                ;UDPDR1=        177622
4219                                ;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
```

```
4277                                    ;*SUPERVISOR 'D' PAGE ADDRESS REGISTERS
4278          172260                SDPAR0= 172260
4279                                ;SDPAR1=        172262
4280                                ;SDPAR2=        172264
4281                                ;SDPAR3=        172266
4282                                ;SDPAR4=        172270
4283          172272                SDPAR5= 172272
4284          172274                SDPAR6= 172274
4285          172276                SDPAR7= 172276
4286
4287                                    ;*KERNEL 'I' PAGE DESCRIPTOR REGISTERS
4288          172300                KIPDR0= 172300
4289                                ;KIPDR1=        172302
4290                                ;KIPDR2=        172304
4291                                ;KIPDR3=        172306
4292                                ;KIPDR4=        172310
4293                                ;KIPDR5=        172312
4294                                ;KIPDR6=        172314
4295                                ;KIPDR7=        172316
4296
4297                                    ;*KERNEL 'D' PAGE DESCRIPTOR REGISTERS
4298                                ;KDPDR0=        172320
4299                                ;KDPDR1=        172322
4300                                ;KDPDR2=        172324
4301                                ;KDPDR3=        172326
4302                                ;KDPDR4=        172330
4303                                ;KDPDR5=        172332
4304                                ;KDPDR6=        172334
4305                                ;KDPDR7=        172336
4306
4307                                    ;*KERNEL 'I' PAGE ADDRESS REGISTERS
4308          172340                KIPAR0= 172340
4309                                ;KIPAR1=        172342
4310                                ;KIPAR2=        172344
4311                                ;KIPAR3=        172346
4312          172350                KIPAR4= 172350
4313          172352                KIPAR5= 172352
4314          172354                KIPAR6= 172354
4315                                ;KIPAR7=        172356
4316
4317                                    ;*KERNEL 'D' PAGE ADDRESS REGISTERS
4318          172360                KDPAR0= 172360
4319                                ;KDPAR1=        172362
4320                                ,KDPAR2=        172364
4321                                ;KDPAR3=        172366
4322                                ;KDPAR4=        172370
4323                                ;KDPAR5=        172372
4324          172374                KDPAR6= 172374
4325          172376                KDPAR7= 172376
4326
```

```
                                        .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
```

```
4452                                    .SBTTL  MACRO   NEWTST
4453                                ;********************* NEWTST **********************
4454                                ;NEWTST IS USED AS THE FIRST INSTRUCTION OF A TEST.
4455                                ;IT WILL:
4456                                ;1)  GENERATE A TEST NUMBER FOR THE LABEL OF THIS TEST
4457                                ;2)  PUT STARS BEFORE AND AFTER A MESSAGE
4458                                ;ARGUMENTS
4459                                ;1)  ASCII  --  THIS IS THE MESSAGE THAT WILL APPEAR
4460                                ;               ON THE LISTING
4461                                ;2)  ICOUNT --  IF NON-BLANK AND BIT 11 OF $SWR = 1 IT WILL BE
4462                                ;               THE NUMBER OF ITERATIONS TO MAKE ON THIS TEST
4463                                ;3)  RETURN --  IF NON-BLANK WILL BE THE ADDRESS TO
4464                                ;               WHICH THE NEXT SCOPE STATEMENT WILL
4465                                ;               LOOP BACK TO.
4466                                ;4)  COMAND --  IF NON-BLANK WILL BE THE FIRST
4467                                ;               INSTRUCTION OF THE TEST
4468                                ;               IF BLANK SCOPE WILL BE THE
4469                                ;               FIRST INSTRUCTION
4470                                ;***************************************************
4471                                        .MACRO  NEWTST  ASCII,ICOUNT,RETURN,COMAND
4472                                        $STN=1
4473                                        $NWTST=0
4474                                        .NLIST  MC
4475                                        .IF B <COMAND>
4476                                        $$NEWTEST       \$TN,<ASCII>,SCOPE
4477                                        .IFF
4478                                        $$NEWTEST       \$TN,<ASCII>,<COMAND>
4479                                        .ENDC
4480                                        .NLIST
4481                                        .LIST   ME
4482                                        .LIST
4483                                        .IF NE 4000&$SWR
4484                                        .IF NB ICOUNT
4485                                        .IF LE <ICOUNT-1>
4486                                        MOV     #1,$TIMES       ;;DO 1 ITERATION
4487                                        .IFF
4488                                        MOV     #ICOUNT,$TIMES  ;;DO ICOUNT ITERATIONS
4489                                        .ENDC
4490                                        .ENDC
4491                                        .IF NB RETURN
4492                                        MOV     #RETURN,$LPADR  ;;SET SCOPE LOOP ADDRESS
4493                                        .ENDC
4494                                        .ENDC
4495                                        .NLIST
4496                                        .LIST   MC
4497                                        .LIST
4498                                        .NLIST  ME
4499                                        .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
```

```
                                                                    N 9
4752                                        TOHERE
4753                                        FROMHERE
4754                                        .DSABL      CRF
4755                                        .IIF DF LST$$ .NLIST  ME
4756                                        .ENABL      CRF
4757                                    .ENDC
4758                                .ENDM   BMOV
```

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

                .MACRO  MAP     BANK
                PUSH    R3
                .NLIST
                .DSABL  CRF
                .IIF DF LST$$ .LIST  ME
                .ENABL  CRF
                .LIST
                .IF B BANK
                MOV     #120.,R3
                .IFF
                MOV     BANK,R3
                .ENDC
                CALL    MAPPER
                .DSABL  CRF
                .IIF DF LST$$ .NLIST  ME
                .ENABL  CRF
                POP     R3
                .ENDM   MAP
```

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

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

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

                        .MACRO  USER
                        .NLIST
                        .DSABL  CRF
                        .IIF DF LST$$ .LIST  ME
                        .ENABL  CRF
                        .LIST
                        BIS     #BIT15!BIT14,PSW                ;GC TO USER MODE
                        .DSABL  CRF
                        .IIF DF LST$$ .NLIST  ME
                        .ENABL  CRF
                        .ENDM   USER
```

```
                        .SBTTL  MACRO   TESTAREA
                ;****************** TESTAREA ************************
                ;
                ; THIS MACRO SWITCHES TO THE SPECIFIED TEST MODE.
                ;
                ;ARGUEMENTS: NONE.
                ;
                ;*****************************************************

                        .MACRO  TESTAREA
                        .NLIST
                        .DSABL  CRF
                        .IIF DF LST$$ .LIST  ME
                        .ENABL  CRF
                        .LIST
                        BIS     TESTMODE,PSW                    ;GO TO SYSTEM TEST MODE
                        .DSABL  CRF
                        .IIF DF LST$$ .NLIST  ME
                        .ENABL  CRF
                        .ENDM   TESTAREA
```

```
                        .SBTTL  MACRO   SET4 & RES4
                ;****************** SET4 & RES4 *********************
                ;
                ; THESE MACROS SET & RESTORE VECTOR 4(TIMEOUT TRAP)
                ;
                ; IN IT'S RESTORED MODE TRAPS ARE REPORTED AS SUCH.
                ;
                ;ARGUEMENTS:    LOC     ;THE LOCATION TO VECTOR TO (ONLY USED IN ''SET4'' NOT ''RES4'')
                ;
                ;I USE THE SET4 AND RES4 MACROS AROUND CODE THAT I EXPECT TO TRAP TO 4
                ;LIKE LOOKING FOR ALL POSSIBLE CSR'S AND ETC.  WHENEVER CODE IS NOT
                ;SURROUNDED BY SET4 AND RES4 THEN ANY TRAPS TO 4 WILL CAUSE AN ERROR
                ;PRINTOUT THAT SAYS ''UNEXPECTED TRAP TO 4'' AND ALL THE ASSOCIATED REGISTER JUNK
                ;*****************************************************

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

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

```
4910                                    .SBTTL  MACRO   DLEFT
4911                                ;****************** DLEFT ****************************
4912                                ;
4913                                ; THIS MACRO DOES A DOUBLE WORD LEFT SHIFT
4914                                ;
4915                                ;ARGUEMENTS:    LOC     ;THE LOCATION TO BE SHIFTED LEFT (CARRY TO LOC+2)
4916                                ;
4917                                ;*****************************************************
4918
4919                                        .MACRO  DLEFT   ARG
4920                                        .NLIST
4921                                        .DSABL  CRF
4922                                        .IIF DF LST$$ .LIST  ME
4923                                        .ENABL  CRF
4924                                        .LIST
4925                                        ROL     ARG
4926                                        ROL     ARG+2
4927                                        .DSABL  CRF
4928                                        .IIF DF LST$$ .NLIST  ME
4929                                        .ENABL  CRF
4930                                        .ENDM   DLEFT
4931                                        .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  014470                  $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
                000052                  .=52
000052  000020                  .WORD   BIT4            ;;2)SET LOC.52 TO INDICATE MEMORY SIZE DEPENDANT
                                        .SBTTL  APT11 HOOKS
                000024                  .=24    ;;SET POWER FAIL TO POINT TO START OF PROGRAM
000024  000200                  200     ;;FOR APT START UP
                000042                  .=42
000042  002000                  STACK                   ;SO RT11 CAN START WITH RUN COMMAND
                000044                  .=44    ;;POINT TO APT INDIRECT ADDRESS PNTR.
000044  063126                  $APTHDR ;;POINT TO APT HEADER BLOCK
                000200                  .=200
000200  000437                  START3: BR      START1          ;"NORMAL" START
000202  000442                          BR      START2          ;RESTART (SAVE ERROR ACCOUNTING)
                000300                  .=300
000300  005037  002566          START1: CLR     RESTART
000304  000137  003630                  JMP     START
000310                          START2: SET     RESTART
000316  000137  003630                  JMP     START
                002000                  .=STACK
```

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

VARIABLES INITIALIZED TO ZERO

```
5043 002146  000000                CSRNO:  0                       ;CSR ADDRESS NUMBER (4 LSB'S)
5044 002150  000000                OLDCSR: .WORD   0               ;OLD CSR NUMBER(USED IN INH PTR TEST)
5045                                       ;THESE LOCATIONS STORE GPR'S DURING SUPERVISOR TESTS
5046 002152  000000                SUPDR0: 0
5047 002154  000000                SUPDR1: 0
5048 002156  000000                SUPDR2: 0
5049 002160  000000                SUPDR3: 0
5050 002162  000000                SUPDR4: 0
5051 002164  000000                SUPDR5: 0
5052 002166  000000                SUPDR6: 0
5053 002170  000000                DUMMY:  0                       ;DUMMY LOCATION FOR ADDRESS PASSING
5054                               ;THESE LOCATIONS STORE GPR'S & PSW DURING DETAILED ERROR PRINTOUTS
5055 002172  000000                DETR0:  0
5056 002174  000000                DETR1:  0
5057 002176  000000                DETR2:  0
5058 002200  000000                DETR3:  0
5059 002202  000000                DETR4:  0
5060 002204  000000                DETR5:  0
5061 002206  000000                DETSP:  0
5062 002210  000000                DETPSW: 0
5063 002212  000000                DETFLAG:0                       ;DETAILED REPORT FLAG
5064 002214  000000                CONTFLAG:0                      ;CSR'S HAVE BEEN TESTED FLAG
5065 002216  000000                TOTCSRS:.WORD   0               ;1 BIT PER EXISTING CSR, EG-
5066                                                              ;CSR 0 REPRESENTED BY BIT 15, ETC.
5067 002220  000000                CSRFIRST:.WORD  0               ;FIRST ADDRESS UNDER CONTROL OF THIS CSR
5068 002222  000000                CSRLAST: .WORD  0               ;
5069 002224  000000                CSRFBANK:.WORD  0               ;
5070 002226  000000                CSRLBANK:.WORD  0               ;
5071 002230  000000                CSRINT:  .WORD  0               ;
5072 002232  000000                SPLTCSR: .WORD  0               ;
5073 002234  000000  000000        DATBUF: .WORD   0,0             ;TWO WORD DATA BUFFER
5074 002240  000000  000000        TSTDAT: .WORD   0,0             ;TWO WORD TEST DATA
5075 002244  000000  000000        SBEMSK: .WORD   0,0             ;TWO WORD SINGLE BIT ERROR MASK
5076 002250  000000  000000        DBEMSK: .WORD   0,0             ;TWO WORD DOUBLE BIT ERROR MASK
5077 002254  000000                SUPDOADD:.WORD  0               ;ADDRESS OF SUBROUTINE TO EXECUTE IN SUPERVISOR MODE
5078 002256     000                PASFLG: .BYTE   0               ;LOCAL LOOP PASS CONTROL
5079 002257     000                UPPFLG: .BYTE   0               ;LOCAL LOOP PASS CONTROL
5080 002260  000000                REALPAT:.WORD   0               ;REAL PATTERN UNDER TEST
5081 002262  000000                OLDCACHE:.WORD  0               ;BACKED UP VALUE OF CACHE CONTROL REGISTER
5082 002264  000000                PARTHERE:.WORD  0               ;PARITY TRAPS SOMETIMES GO TO ADDRESS STORED HERE
5083 002266  000000                FSSTACK:.WORD   0               ;STACK SAVED HERE IF IN FIELD SERVICE MODE
5084 002270  000000                NEWBANK:.WORD   0               ;USED FOR RELOCATION TO A NEW BANK
5085 002272  000000                SOURCE: .WORD   0               ;SOURCE OF DATA WORDS FOR CHECKBIT GENERATION SUBROUTINE
5086 002274  000000                CHECK:  .WORD   0               ;CHECKBITS TO BE LOADED INTO CSR
5087 002276  000000                PCBUMP: .WORD   0               ;VALUE TO BUMP THE PC BY TO RECOVER AFTER A PARITY TRAP
5088 002300  000000                CSRINC: .WORD   0               ;VALUE TO INCREMENT ADDRESS BY TO REMAIN IN THE SAME CSR
5089 002302  000000                CSRLOOP:.WORD   0               ;LOOP CONTROL FOR CSR TESTING
5090 002304  000000                SUCCESS:.WORD   0               ;FLAG SET BY SUCCESSFULL TASK OR SUBROUTINE
5091 002306  000000                ZEROS:  .WORD   0               ;FOR AID IN "MOV" INSTRUCTIONS
5092 002310  000000                TIME:   .WORD   0               ;SECONDS THAT BATTERIES SHOULD LAST
5093 002312  000000                SKIPMK: .WORD   0               ;FLAG TO SKIP MKCONTROL SUBROUTINE
5094 002314  000000                NULLFLAG:.WORD  0               ;SET WHEN RUNNING NULL PATTERNS
5095 002316  000000                QVFLAG: 0                       ;FLAGS QUICK VERIFY PASS UNDER APT, ACT, OR XXDP CHAIN MODE
5096 002320  000000                ACTFLAG:0                       ;FLAGS ACT AUTOMATIC MODE PROGRAMMING RULES
5097 002322  000000                APTFLAG:0                       ;FLAGS APT AUTOMATIC MODE PROGRAMMING RULES
5098 002324  000000                XXDPCHAIN:0                     ;FLAGS XXDP CHAIN MODE PROGRAMMING RULES
5099                               ;NOTE: THESE TWO BYTES MUST STAY TOGETHER
```

```
VARIABLES       INITIALIZED TO ZERO

5100 002326  000                $NULL:  .BYTE   0              ;;CONTAINS NULL CHARACTER FOR FILLS
5101 002327  000                $FILLS: .BYTE   0              ;CONTAINS # OF FILL CHARACTERS
5102 002330  000                $TPFLG: .BYTE   0              ;;"TERMINAL NOT AVAILABLE" FLAG
5103                                    .EVEN
5104 002332  000000             $ESCAPE:0                      ;;ESCAPE ON ERROR ADDRESS
5105 002334  000000             EVEN:   0                      ;USED FOR ALTERNATE DATA PATTERNS
5106 002336  000000             STRIPES:0                      ;COUNTS DIAGONAL STRIPES
5107 002340  000000             COUNT:  0                      ;BACKED UP COPY OF STRIPES
5108 002342  000000             NOTAB:  0                      ;NO TABLE BEING PRINTED - NOW
5109 002344  000000             BSIZE:  0                      ;SIZE OF 11/45 MOS MEMORY IN K WORDS
5110 002346  000000             KSIZE:  0                      ;SIZE OF MF11S-K MEMORY IN K WORDS
5111 002350  000000             LSIZE:  0                      ;SIZE OF MS11-L MEMORY IN K WORDS
5112 002352  000000             MSIZE:  0                      ;SIZE OF MS11-M MEMORY IN K WORDS
5113 002354  000000             PSIZE:  0                      ;SIZE OF UNIBUS PARITY MEMORY IN K WORDS
5114 002356  000000             TOOMANY:0                      ;FLAGS WHEN TOO MANY ERRORS HAVE BEEN PRINTED FOR A BANK
5115 002360  000000             READONLY:0                     ;FLAG TO PATTERNS TO READ ONLY
5116 002362  000000  000000      TESTADD:0,0                    ;THE ADDRESS TO RUN CSR TESTS ON
5117 002366  000000             UNITOP: 0                      ;HIGHEST ACCESSABLE BANK OF MEMORY THRU UNIBUS MAP
5118 002370  000000             STOPOK: 0                      ;FLAG TO ALLOW STOPPING WITH SWITCH REGISTER
5119 002372  000000             APTPAR: .WORD   0              ;AMOUNT OF PARITY MEMORY ACCORDING TO APT
5120 002374  000000             APTECC: .WORD   0              ;AMOUNT OF ECC MEMORY ACCORDING TO APT
5121 002376  000000             NOFSMODE:0                     ;FLAG TO DISABLE FIELD SERVICE MODE
5122 002400  000000             NOERROR:0                      ;"THIS IS NOT AN ERROR" FLAG
5123 002402  000000             LOADBANK:0                     ;BANK LOADERS ARE RELOCATED TO
5124 002404  000000             TEMP:   0                      ;USED FOR JUNK
5125 002406  000000             QUICK:  0                      ;QUICK STOP FLAG FOR APT POWER FAIL
5126 002410  000000             NOSCOPE:0                      ;"NO SCOPE LOOP ALLOWED" FLAG
5127 002412  000000             FSINFLAG:0                     ;"FIELD SERVICE - NO INTERNAL INTERLEAVE" FLAG
5128 002414  000000             APTSIZE:0                      ;APT SIZING INFO FLAG
5129 002416  000000             FS7FLAG:0                      ;TRUE WHEN IN FIELD SERVICE COMMAND 7
5134 002420  000000             CONFGERROR:0                   ;CONFIGURATION ERROR FLAG
5135 002422  000000             I:      0                      ;USED FOR GENERAL PURPOSE INDEXING
5136 002424  000000             NO22BIT:0                      ;NO 22-BIT MODE FLAG
5137 002426  000000             NOSUPER:0                      ;NO SUPERVISOR MODE FLAG
5138 002430  000000             ERRADD: .WORD   0              ;HOLDS THE CSR'S ERROR ADDRESS
5139 002432  000000  000000  000000  CSRINFO:0,0,0,0,0,0,0,0   ;USED TO STORE INFORMATION ABOUT THE 16
     002440  000000  000000  000000
     002446  000000  000000
5140 002452  000000  000000  000000          0,0,0,0,0,0,0,0   ;POSSIBLE CSR'S
     002460  000000  000000  000000
     002466  000000  000000
5141 002472  000000             LINK1:  0                      ;USED TO HOLD LINKS TO PATTERNS WHICH
5142 002474  000000             LINK2:  0                      ;CAN EXECUTE IN THE PAR/PDR'S OR NOT
5143 002476  000000             CSRHOLD:0                      ;USED TO STORE CSR VALUES FOR CSR TESTS
5144 002500  000000             KFLAG:  0                      ;USED TO FLAG MF11S-K MEMORY TO TESTS
5145 002502  000000  000000      PGMCSR: .WORD   0,0            ;POINTS TO PROGRAM CSR
5146 002506  000000             INHECC: .WORD   0              ;FLAGS INHIBIT ECC TESTS ON RELOCATION
5147 002510  000000             INHBANK:.WORD   0              ;
5148 002512  000000             FULLREL:.WORD   0              ;
5186 002514                     $CMTGE: ;*END OF COMMON TAGS
```

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

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

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

CONFIGURATION TABLE

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

```
5262                                            .SBTTL  ********** MAIN **********
5263 003630                                START:  SUBTST  <<INITIALIZE VARIABLES TO ZERO>>
                                                ;*************************************************************************
                                                ;*SUBTEST        INITIALIZE VARIABLES TO ZERO
                                                ;*************************************************************************
5267 003630  105737  063046                            TSTB    $ENV
5268 003634  001001                                     BNE     NORES
5269 003636  000005                                     RESET
5270 003640  013706  002534                NORES:       MOV     KSTACK,SP       ;;SETUP THE STACK POINTER
5276 003644  012700  002000                             MOV     #$CMTAG,R0      ;;FIRST LOCATION TO BE CLEARED
5277 003650  005020                        1$:          CLR     (R0)+           ;;CLEAR MEMORY LOCATION
5278 003652  022700  002514                             CMP     #$CMTGE,R0      ;;DONE?
5279 003656  001374                                     BNE     1$              ;LOOP BACK IF NO
5280 003660  012737  000167  002526                     MOV     #167,LASTBANK   ;RESTORE LASTBANK (THIS MUST BE DONE PRIOR TO SYSTEM SIZING)
5281 003666                                             SUBTST  <<CLEAR NON-PROGRAM SPACE>>
                                                ;*************************************************************************
                                                ;*SUBTEST        CLEAR NON-PROGRAM SPACE
                                                ;*************************************************************************
5282                                                    ;THIS ATTEMPS TO GET RID OF ANY PARITY ERRORS BY WRITING INTO
5283                                                    ;EVERY LOCATION THAT IS NOT LOADED INTO BY THE PROGRAM OR ALLOCATED
5284                                                    ;TO THE XXDP LOADERS
5285 003666  012737  000001  002074                     MOV     #1,NOPAR                ;PARITY ACTION = COUNT & IGNORE
5286 003674  005000                                     CLR     R0
5287 003676  000241                        2$:          CLC
5288 003700  005520                                     ADC     (R0)+
5289 003702  020027  160000                             CMP     R0,#160000
5290 003706  103773                                     BLO     2$
5291 003710  005037  002074                             CLR     NOPAR                   ;RESTORE DEFAULT PARITY ACTION
5292
```

```
5301 003714                                     SUBTST  <<TYPE OF SYSTEM SIZER>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       TYPE OF SYSTEM SIZER
                                        ;*****************************************************************************
5302 003714  000401                             BR      SYSSIZ                  ;SKIP OVER VARIABLE LOCATION
5303 003716  000000                     PROTYP: .WORD   0
5304 003720                             SYSSIZ: SET4    #4$
5305 003726  005737  177746                     TST     CONTRL                  ;SEE IF CACHE REGISTER RESPONDS
5306 003732                                     SET4    #9$                     ;YES - DO WE HAVE 11/44 TYPE CACHE
5307 003740  005737  177750                     TST     MAINT                   ;OR 11/60 TYPE CACHE?
5308 003744  000411                             BR      5$                      ;BRANCH IF 11/44 TYPE CACHE
5309 003746  012737  000014  002520      9$:    MOV     #14,CACHKF              ;TURN OFF CONSTANT FOR 11/60 CACHE
5310 003754  000405                             BR      5$
5311 003756  005037  002514              4$:    CLR     CACHKN                  ;NO CACHE ON SYSTEM
5312 003762  012737  002306  064342             MOV     #ZEROS,DT14             ;DO NOT PRINT CONTRL ERROR MESSAGES
5313 003770                             5$:     SET4    #6$
5314 003776  005737  172516                     TST     MMR3                    ;DO WE HAVE AN MMR3?
5315 004002  005037  172516                     CLR     MMR3                    ;YES WE DO
5316 004006  052737  000020  172516             BIS     #BIT4,MMR3              ;SEE IF THERE IS 22-BIT MODE
5317 004014  032737  000020  172516             BIT     #BIT4,MMR3
5318 004022  001024                             BNE     10$                     ;BRANCH IF 22-BIT RELOCATION
5319 004024  000411                             BR      7$                      ;BRANCH IF MMR3 BUT NO 22-BIT RELOC.
5320                                    ;* 11/34 TYPE MACHINES ENTER HERE
5321 004026  012737  140000  002522      6$:    MOV     #140000,TESTMODE        ;MAKE TEST MODE USER
5322 004034  005237  002426                     INC     NOSUPER                 ;NO SUPERVISOR MODE
5323 004040  005037  064212                     CLR     DT5+10
5324 004044  005037  064352                     CLR     DT14+10
5325                                    ;* 11/45 TYPE MACHINES ENTER HERE
5326 004050  005237  002424              7$:    INC     NO22BIT                 ;NO 22 BIT MODE
5327 004054  012737  000007  002526             MOV     #7,LASTBANK             ;124K MEMORY MAX. MEMORY SIZE
5328 004062  005037  064214                     CLR     DT5+12                  ;DO NOT TRY TO PRINT ERROR REGISTER
5329 004066  005037  064354                     CLR     DT14+12                 ;ERROR MESSAGES, BECAUSE THERE IS
5330                                                                            ;IS NO ERROR REGISTER!
5331 004072  000417                             BR      8$
5332 004074                             10$:    SET4    #8$
5333 004102  000007                             MFPT                            ;TYPE OF PROCESSOR TEST: THIS INSTRUCTION
5334                                                                            ; (AVAILABLE ON NEWER PROCESSORS ONLY) PLACES
5335                                                                            ; A CODE IN THE LOWER BYTE OF R0 THAT
5336                                                                            ; INDICATES THE  PROCESSOR TYPE. 1=11/44
5337                                                                            ; 3=11/24
5338 004104  110037  003716                     MOVB    R0,PROTYP               ;MOV THE CODE TO PROTYP
5339 004110  022737  000003  003716             CMP     #3,PROTYP               ;IS THIS AN 11/24?
5340 004116  001005                             BNE     8$                      ;BRANCH IF NOT - WE HAVE AN 11/44
5341 004120  005237  002426                     INC     NOSUPER                 ;NO SUPERVISOR MODE
5342 004124  012737  140000  002522             MOV     #140000,TESTMODE        ;MAKE TEST MODE USER
5343
5344 004132                             8$:     SET4    #11$                    ;TRAPS GO TO 11$                    ;R-C
5345 004140  005037  056510                     CLR     CPERRF                  ;CLEAR THE FLAG                     ;R-C
5346 004144  005737  177766                     TST     @#177766                ;IS THERE A CPU ERROR REGISTER?     ;R-C
5347 004150  012737  177777  056510             MOV     #-1,CPERRF              ;YES - TRAPPED                      ;R-C
5348 004156                             11$:    RES4                                                                ;R-C
```

```
                                        SUBTST  <<INITIALIZE VARIABLES TO NON ZERO>>
                                ;*******************************************************************************
                                ;*SUBTEST        INITIALIZE VARIABLES TO NON ZERO
                                ;*******************************************************************************
                                        SET     WORS1
                                        MOV     #3,FLIPLOC
                                        SET     HEADER
                                        MOV     #176543,MSEEDH
                                        MOV     #123456,MSEEDL
                                        MOV     MSEEDH,SEEDHI   ;PRIME THE RANDOM NUMBER GENERATOR
                                        MOV     MSEEDL,SEEDLO   ;BOTH HIGH AND LOW WORDS
                                        MOV     #377,BAKPAT
                                        MOV     #177400,SWAPAT
                                        SUBTST  <<INITIALIZE VECTORS>>
                                ;*******************************************************************************
                                ;*SUBTEST        INITIALIZE VECTORS
                                ;*******************************************************************************
                                        MOV     #$SCOPE,IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
                                        MOV     #340,IOTVEC+2   ;;LEVEL 7
                                        MOV     #$ERROR,EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
                                        MOV     #340,EMTVEC+2   ;;LEVEL 7
                                        MOV     #$TRAP,TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
                                        MOV     #340,TRAPVEC+2;LEVEL 7
                                        MOV     #$PWRDN,PWRVEC ;;POWER FAILURE VECTOR
                                        MOV     #340,PWRVEC+2   ;;LEVEL 7
                                        MOV     #PARITY,PARVEC;GET READY FOR PARITY ERRORS
                                        MOV     #340,PARVEC+2
                                        MOV     #PDP1105,RESVEC;RESERVED INSTRUCTION TRAP
                                        MOV     #340,RESVEC+2
                                        MOV     #TIMEOUT,ERRVEC;SETUP TIMEOUT ERRORS
                                        MOV     #340,ERRVEC+2   ;SET PRIORITY OF ERROR TRAPS
                                        MOV     #MMTRAP,MMVEC   ;VECTOR FOR MEMORY MANAGEMENT
                                        MOV     #340,MMVEC+2
                                        CACHON                  ;TURN CACHE ON
```

```
5389 004430                             SUBTST  <<INITIALIZE PATTERNS>>
                                ;*******************************************************************************
                                ;*SUBTEST        INITIALIZE PATTERNS
                                ;*******************************************************************************
5390                                    ;THE APT E-TABLE DETERMINES WHICH PATTERNS ARE GOING TO BE RUN.
5391                                    ;EACH BIT SET REPRESENTS A PATTERN TABLE ENTRY THAT IS TO BE LEFT
5392                                    ;ALONE (TO BE RUN).  EACH BIT CLEARED REPRESENTS A PATTERN TABLE ENTRY
5393                                    ;THAT IS TO BE OVERLAYED WITH THE ADDRESS OF A NULL PATTERN.
5394 004430  012700  063112            MOV     #$DDW0,R0
5395 004434  012001                    MOV     (R0)+,R1
5396 004436  012703  017426            MOV     #MKCSRT,R3
5397 004442  012702  000020            MOV     #16.,R2
5398 004446  004737  004546            CALL    PATPLUG
5399 004452  012001                    MOV     (R0)+,R1
5400 004454  012702  000010            MOV     #8.,R2
5401 004460  004737  004546            CALL    PATPLUG
5402 004464  012001                    MOV     (R0)+,R1
5403 004466  012703  017656            MOV     #MKPAT,R3
5404 004472  012702  000020            MOV     #16.,R2
5405 004476  004737  004546            CALL    PATPLUG
5406 004502  012001                    MOV     (R0)+,R1
5407 004504  012702  000010            MOV     #8.,R2
5408 004510  004737  004546            CALL    PATPLUG
5409 004514  012001                    MOV     (R0)+,R1
5410 004516  012703  020042            MOV     #MJPAT,R3
5411 004522  012702  000020            MOV     #16.,R2
5412 004526  004737  004546            CALL    PATPLUG
5413 004532  012001                    MOV     (R0)+,R1
5414 004534  012702  000010            MOV     #8.,R2
5415 004540  004737  004546            CALL    PATPLUG
5416 004544  000417                    BR      SUBAAA
5417
5418 004546                     PATPLUG:SUBTST  <<SUBR  PLUG IN NULL PATTERNS>>
                                ;*******************************************************************************
                                ;*SUBTEST        SUBR    PLUG IN NULL PATTERNS
                                ;*******************************************************************************
5419 004546                             FOR I := #1 TO R2
5420 004554  006001                       ROR   R1
5421 004556                               ON.NOERROR            ;IF CARRY CLEAR
5422 004560  012713  026414                MOV #MT0999,(R3)
5423 004564                               END ;OF ON.ERROR
5424 004564  062703  000002               ADD #2,R3
5425 004570                             END ;OF FOR
5426 004602  000207                     RETURN
```

```
5429 004604                          SUBAAA: SUBTST  <<CLEAR THE CONFIGURATION TABLE>>
                                     ;*****************************************************************************
                                     ;*SUBTEST       CLEAR THE CONFIGURATION TABLE
                                     ;*****************************************************************************
5430                                         ;THIS ZEROS (UNLESS WE STARTED AT ADDRESS 202) THE CONFIG TABLE
5431                                         ;WHICH IS FULLY DISCRIBED AT LOCATION "CONFIG".
5432                                         .ENABLE LSB
5433 004604                                  IF RESTART IS FALSE
5434 004612  012700  002624                     MOV   #CONFIG,R0
5435 004616  005020                    1$:       CLR   (R0)+
5436 004620  022700  003630                     CMP   #CONFIEND,R0
5437 004624  001374                             BNE   1$
5438 004626                                  END ;OF IF RESTART
5439                                         .DSABL  LSB
5440 004626  012737  000002  002104          MOV     #BIT1,CPUBIT            ;SET ID BIT
5441 004634                                  SUBTST  <<SIZE FOR A HARDWARE SWITCH REGISTER>>
                                     ;*****************************************************************************
                                     ;*SUBTEST       SIZE FOR A HARDWARE SWITCH REGISTER
                                     ;*****************************************************************************
5442                                         ;;IF NOT FOUND OR IT IS
5443                                         ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
5444                                         .ENABL  LSB
5445 004634                                  SET4    #3$                 ;TRAPS TO 4 GOTO 3$
5446 004642  012737  177570  002576          MOV     #DSWR,SWR           ;;SETUP FOR A HARDWARE SWITCH REGISTER
5447 004650  012737  177570  002600          MOV     #DDISP,DISPLAY      ;;AND A HARDWARE DISPLAY REGISTER
5448 004656                                  IF #-1 EQ @SWR              ;IF NO TRAP FROM REFERENCE TO @SWR AND @SWR = #-1
5449 004666  000403                             BR    2$                  ;;BRANCH IF NO TIMEOUT
5450 004670  012716  004676          3$:       MOV   #2$,(SP)            ;;SET UP FOR TRAP RETURN
5451 004674  000002                             RTI
5452 004676                          2$:        RES4                      ;RESET TRAPS TO 4 TO DEFAULT
5453 004720  012737  000176  002576             MOV   #SWREG,SWR          ;;POINT TO SOFTWARE SWR
5454 004726  012737  000174  002600             MOV   #DISPREG,DISPLAY
5455 004734                                  END ;OF IF #-1
5456                                         .DSABL  LSB
```

```
5459 004734                          SUBAAB: SUBTST  <<SETUP ACT, APT, & XXDP>>
                                     ;*****************************************************************************
                                     ;*SUBTEST       SETUP ACT, APT, & XXDP
                                     ;*****************************************************************************
5460                                         ;THIS SETS UP A BUNCH OF FLAGS TO TELL THE PROGRAM EVERYTHING
5461                                         ;IT CARES TO KNOW ABOUT APT, ACT, & XXDP.
5462 004734  005037  063034                  CLR     $PASS           ;CLEAR PASS COUNT
5463 004740                                  IFB #BIT5 SET.IN $ENVM
5464 004750                                    SET   $TPFLG          ;INDICATE NO TERMINAL
5465 004756                                  END ;OF IFB #BIT5
5466 004756                                  IFB #BIT7 SET.IN $ENVM
5467 004766                                    SET   APTSIZE
5468 004774                                  END ;OF IFB #BIT7
5469 004774                                  IFB $ENV EQ #1
5470 005004                                    SET   APTFLAG,QVFLAG,$AUTO,QUICK
5471 005034  012737  045250  000024            MOV   #APTDOWN,PWRVEC
5472 005042  012737  063050  002576            MOV   #$SWREG,SWR     ;USE APT SWR
5473 005050                                  ELSE
5474 005052                                    IF 42 NE #STACK AND 42 NE #0
5475 005070                                      SET QVFLAG,$AUTO
5476 005104                                      IF 42 EQ #$ENDAD
5477 005114                                        SET       ACTFLAG
5478 005122                                      ELSE
5479 005124                                        SET       XXDPCHAIN
5480 005132                                      END ;OF IF 42
5481 005132                                    END ;OF IF 42
5482 005132                                  END ;OF IFB $ENV
```

```
                                        SUBTST  <<PROTECT PROGRAM & LOADERS>>
                                ;*******************************************************************************
                                ;*SUBTEST        PROTECT PROGRAM & LOADERS
                                ;*******************************************************************************
005132  052737  000200  002624            BIS     #BIT7,CONFIG            ;PROTECT PROGRAM SPACE (BANK 0)
005140  052737  000200  002630            BIS     #BIT7,CONFIG+4          ;PROTECT LOADER SPACE (BANK 1)
005146                                    IF #$ENDAD NE 42                ;NOT ACT-11?
005156                                      TYPE  MSG000                  ;TYPE PROGRAM TITLE
005162                                    END ;OF IF #$ENDAD

005162                                  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.
005162                                  SET4    #3$
005170  005737  177746                  TST     CONTRL                  ;IS THERE A CONTROL REGISTER?
005174                                  SET4    #2$
005202  005737  177750                  TST     MAINT                   ;IS THERE A MAINTENANCE REGISTER?
005206                                  SET4    #1$
005214  005737  177754                  TST     DATARG                  ;IS THERE A DATA REGISTER?
005220                                  TYPE    MSG117                  ; 11/44
005224  000405                          BR      4$
005226                          1$:     TYPE    MSG116                  ; 11/34
005232  000402                          BR      4$
005234                          2$:     TYPE    MSG118                  ; 11/60
005240  052737  000014  177746    4$.   BIS     #BIT2!BIT3,CONTRL        ;SET CACHE DISABLE BITS
005246  042737  000014  177746          BIC     #BIT2!BIT3,CONTRL        ;CLEAR CACHE DISABLE BITS
005254  032737  000004  177746          BIT     #BIT2,CONTRL             ;IS THE BIT SET?
005262  001004                          BNE     7$                       ;BRANCH IF THE BIT IS SET
005264  032737  000010  177746          BIT     #BIT3,CONTRL             ;IS THE BIT SET?
005272  001413                          BEQ     6$                       ;BRANCH IF THE BIT IS SET
005274                          7$:     TYPE    MSG121                   ; CACHE BYPASSED
005300  104424                          CACHOFF
005302  013737  002514  002516          MOV     CACHKN,CACHKN+2          ;SAVE INFO ABOUT CACHE
005310  005037  002514                  CLR     CACHKN                   ;CACHE CANNOT BE USED - IT'S BYPASSED
005314  000404                          BR      8$
005316                          3$:     TYPE    MSG119                   ; NO
005322                          6$:     TYPE    MSG120                   ; CACHE AVAILABLE
```

```
                                        SUBTST  <<SETUP USER & SUPERVISOR STACK>>
                                ;*******************************************************************************
                                ;*SUBTEST       SETUP USER & SUPERVISOR STACK
                                ;*******************************************************************************
005326  104421                  8$:     DEENERGIZE              ;TURN OFF MEMORY MANAGEMENT
005330  005737  002426                  TST     NOSUPER         ;IS THERE A SUPERVISOR MODE?
005334  001011                          BNE     5$              ;NO-SKIP SUPERVISOR SETUP.

                                        ;SET PREVIOUS MODE TO SUPERVISOR
005336  042737  030000  177776          BIC     #BIT13!BIT12,PSW
005344  052737  010000  177776          BIS     #BIT12,PSW

005352                                  PUSH    #SUPSTK
005356  006606                          MTPI    SSP

                                        ;SET PREVIOUS MODE TO USER
005360  052737  030000  177776  5$:     BIS     #BIT13!BIT12,PSW

005366                                  PUSH    #USESTK
005372  006606                          MTPI    USP

005374                                  SUBTST  <<GET SOFTWARE SWITCH REGISTER IF NECESSARY>>
                                ;*******************************************************************************
                                ;*SUBTEST       GET SOFTWARE SWITCH REGISTER IF NECESSARY
                                ;*******************************************************************************
005374                                  IF $AUTO IS FALSE                       ;IF NOT(APT OR ACT)
005402                                    IF SWR EQ #SWREG              ;IF SOFTWARE SWITCH REG SELECTED
005412  104407                              GTSWR                               ;;GET SOFT-SWR SETTINGS
005414                                    END ;OF IF SWR
005414                                  END ;OF IF $AUTO

005414                                  SUBTST  <<GET MEMORY MANAGEMENT READY>>
                                ;*******************************************************************************
                                ;*SUBTEST       GET MEMORY MANAGEMENT READY
                                ;*******************************************************************************
005414  104422                          KMAP                    ;MAP KERNEL SPACE 1 TO 1
005416                                  MAP                     ;MAP SUPERVISOR SPACE (TEST AREA) 1 TO 1
005432  104420                          ENERGIZE                ;TURN ON MEMORY MANAGEMENT
```

```
                                        NEWTST  <<BIT TEST OF ALL CSR'S>>
                                ;*******************************************************************************
                                ;*TEST 1         BIT TEST OF ALL CSR'S
                                ;*******************************************************************************
 005434  000004                 TST1:   SCOPE
                                ;* THE FIRST PART OF THE CONFIGURATION ANALYSIS DOES THE FOLLOWING:
                                ;*      1) FINDS WHICH CSR'S RESPOND, AND PUTS THEM INTO THE CSR INFORMATION
                                ;*              TABLE, AND STORES ANOTHER BIT FOR "TOTCSRS".
                                ;*      2) TESTS THE CSR BITS COMMON TO ALL CSR'S.
                                ;*      3) FIGURES OUT IF THERE IS AN EUB BIT, AN ECC BIT, AND THE EXISTANCE
                                ;*              OF THE ERROR ADDRESS BITS, AND MARKS THIS IN THE CSR
                                ;*              INFORMATION TABLE.
                                ;*      4) TESTS THE BITS PARTICULAR TO THAT TYPE OF CSR.
                                ;*      5) IF ANY BITS TEST BAD IN THE CSR UNDER TEST, THE CSR OK BIT IN THE
                                ;*              CSR INFORMATION TABLE IS CLEARED.
                                ;* THE INFORMATION BITS ONE THROUGH THREE FORM A CODE WHICH GIVES THE TYPE
                                ;* OF CSR:
                                ;*                      ERR. ADDR.   PARITY  NOT EUB
                                ;*      TYPE              BIT2        BIT1    BIT0      CODE TOTALS
                                ;* UNIBUS PARITY          1           1       1           7
                                ;*     MS11-L             1           1       0           6
                                ;*     MF11S-K            1           0       1           5
                                ;*     MS11-M             1           0       0           4
                                ;* 11/45 BIPOLAR          0           1       1           3
                                ;*
                                ;* THIS MEMORY CODE WILL BE USED IN THE SECOND PART OF THIS ANALYSIS
                                ;*
 005436  005005                         CLR     R5              ;R5 IS THE TOTAL CSR NUMBER
 005440  005000                         CLR     R0              ;R0 IS A TABLE INDEX
 005442  012703  172100                 MOV     #CSRADD,R3      ;R3 HAS THE CSR ADDRESS
 005446  012737  000001  002074         MOV     #1,NOPAR        ;IGNORE PARITY ERRORS
 005454                                 SET4    #2$
 005462  005713                 1$:     TST     (R3)    ;DOES THE CSR RESPOND?
 005464  052705  000001                 BIS     #1,R5
 005470  005004                         CLR     R4              ;CLEAR THE LAST CSR INDICATOR
 005472  052760  000007  002432         BIS     #7,CSRINFO(R0)  ;SET ALL THE MEMORY INFO BITS
 005500  052760  000030  002432         BIS     #BIT4!BIT3,CSRINFO(R0)  ;YES-MARK IT IN CSR INFORMATION TABLE
 005506  004537  006042                 JSR     R5,TEST         ;TEST BIT 0 AND 15
 005512  100001                         .WORD   BIT15!BIT0
 005514  012713  040000                 MOV     #BIT14,(R3)     ;IS THERE A BIT 14 RESPONDING
 005520  032713  040000                 BIT     #BIT14,(R3)     ;(IT'S THE EUB BIT)
 005524  001403                         BEQ     3$              ;BRANCH IF NO EUB BIT
 005526  042760  000001  002432         BIC     #BIT0,CSRINFO(R0) ;CLEAR EUB INFO IN THE CSR TABLE
 005534  005013                 3$:     CLR     (R3)            ;CLEAR THE CSR UNDER TEST
 005536  012713  020000                 MOV     #BIT13,(R3)     ;DOES BIT 13 RESPOND
 005542  032713  020000                 BIT     #BIT13,(R3)     ;(TO TEST FOR ECC CSR)
 005546  001417                         BEQ     4$              ;BRANCH IF NOT ECC CSR
 005550  042760  000002  002432         BIC     #BIT1,CSRINFO(R0) ;CLEAR PARITY INFO IN THE CSR TABLE
 005556  012713  020000                 MOV     #BIT13,(R3)     ;SET THE INHIBIT MODE POINTER TO 1ST 16K
 005562  004537  006042                 JSR     R5,TEST         ;TEST BIT 3
 005566  000010                         .WORD   BIT3
 005570  012713  020000                 MOV     #BIT13,(R3)
 005574  004537  006042                 JSR     R5,TEST         ;TEST BIT 1 AND 4
 005600  000022                         .WORD   BIT4!BIT1
 005602  012713  020000                 MOV     #BIT13,(R3)
```

```
5661 005606  004537  006042          4$:   JSR     R5,TEST          ;TEST BIT 2
5662 005612  000004                        .WORD   BIT2
5663 005614  005013                        CLR     (R3)
5664 005616  052713  007740                BIS     #7740,(R3)       ;ARE THERE ERROR ADDRESS BITS?
5665 005622  032713  007740                BIT     #7740,(R3)
5666 005626  001404                        BEQ     5$               ;BRANCH IF NO ERROR ADDR. BITS.
5667 005630  004537  006042                JSR     R5,TEST          ;TEST BITS 5->11
5668 005634  007740                        .WORD   7740
5669 005636  000403                        BR      6$               ;SKIP OVER THE INFORMATION REPORTING
5670 005640  042760  000004  002432  5$:   BIC     #BIT2,CSRINFO(R0) ;REPORT THAT THERE ARE NO ERROR ADDRESS BITS.
5671 005646  032760  000002  002432  6$:   BIT     #BIT1,CSRINFO(R0) ;IS THIS CSR AN ECC CSR?
5672 005654  001014                        BNE     7$               ;BRANCH IF NOT
5673 005656  032760  000001  002432        BIT     #BIT0,CSRINFO(R0) ;IS THE EUB BIT SET?
5674 005664  001410                        BEQ     7$               ;BRANCH IF IT IS
5675                                       ;WE MUST NOW TEST FOR MS11-M ON THE UNIBUS
5676 005666  012713  007760                MOV     #7760,(R3)       ;PUT PATTERN & SBE BIT INTO BITS 4->11
5677 005672  022713  007760                CMP     #7760,(R3)       ;ARE THEY STILL THERE?
5678 005676  001403                        BEQ     7$               ;YES - BRANCH FOR MF11S-K CSR
5679 005700  042760  000001  002432        BIC     #BIT0,CSRINFO(R0) ;NO - SET EUB BIT FOR MS11-M
5680 005706  005013                7$:     CLR     (R3)             ;CLEAR CSR
5681 005710  022760  000040  002432        CMP     #40,CSRINFO(R0)  ;IS THIS A LEGAL CONFIGURATION?
5682 005716  100004                        BPL     10$              ;BRANCH IF IT'S LEGAL
5683 005720  016037  002432  002050        MOV     CSRINFO(R0),BAD
5684 005726  104021                        ERROR   +21              ;ILLEGAL TYPE ERROR
5685 005730  032760  000004  002432  10$:  BIT     #BIT2,CSRINFO(R0) ;DOES THIS CSR HAVE ERROR BITS
5686 005736  001016                        BNE     2$               ;BRANCH IF TRUE
5687 005740  032760  000002  002432        BIT     #BIT1,CSRINFO(R0) ;ARE THE OTHER 2 BITS SET?
5688 005746  001404                        BEQ     11$              ;BRANCH IF NOT
5689 005750  032760  000001  002432        BIT     #BIT0,CSRINFO(R0)
5690 005756  001006                        BNE     2$
5691 005760  016037  002432  002050  11$:  MOV     CSRINFO(R0),BAD
5692 005766  104021                        ERROR   +21              ;ILLEGAL TYPE ERROR
5693 005770  005060  002432                CLR     CSRINFO(R0)      ;CLEAR THE CSR INFO-IT WILL NOT EXIST IN THE PROGRAM
5694 005774  062700  000002          2$:   ADD     #2,R0            ;INCREMENT TO NEXT CSR TABLE
5695 006000  062703  000002                ADD     #2,R3            ;INCREMENT TO NEXT CSR
5696 006004  006305                        ASL     R5
5697 006006  103001                        BCC     8$               ;IS THERE A CSR 0?
5698 006010  005204                        INC     R4               ;YES - SET CSR PRESENT FLAG
5699 006012  022700  000040          8$:   CMP     #40,R0           ;ARE WE DONE?
5700 006016  001221                        BNE     1$               ;BRANCH IF MORE TO DO
5701 006020  000241                        CLC
5702 006022  006005                        ROR     R5               ;RESYNC R5
5703 006024  005704                        TST     R4               ;WAS THERE A CSR 0?
5704 006026  001402                        BEQ     9$               ;BRANCH IF NOT
5705 006030  052705  100000                BIS     #BIT15,R5        ;YES - SET IN THE CSR TABLE
5706 006034  010537  002216          9$:   MOV     R5,TOTCSRS       ;STORE R5 IN TOTCSRS
5707 006040  000437                        BR      CTEST            ;JUMP OVER SUBROUTINE
5708                                       ;THIS SUBROUTINE TESTS THE CSR BITS
5709 006042  012501                TEST:   MOV     (R5)+,R1         ;GET THE BIT TO TEST
5710 006044  050113                        BIS     R1,(R3)          ;SET THAT IN THE CSR UNDER TEST
5711 006046  030113                        BIT     R1,(R3)          ;IS THE BIT STILL THERE?
5712 006050  001013                        BNE     1$               ;BRANCH IF STILL THERE
5713 006052  011337  002144                MOV     (R3),CSR         ;READ CSR
5714 006056  010137  002042                MOV     R1,GOOD
5715 006062  032713  100020                BIT     #BIT15!BIT4,(R3);IS IT BECAUSE OF A SBE OR DBE?
5716 006066  001004                        BNE     1$               ;BRANCH IF IT IS
5717 006070  104035                        ERROR   +35              ;BIT SET ERROR
```

```
5718 006072  042760  000010  002432          BIC     #BIT3,CSRINFO(R0) ;CLEAR CSR OK BIT
5719 006100  040113                  1$:     BIC     R1,(R3)           ;CLEAR THE SELECTED BIT
5720 006102  030113                          BIT     R1,(R3)           ;IS IT CLEARED?
5721 006104  001413                          BEQ     2$                ;BRANCH IF IT IS CLEARED
5722 006106  011337  002144                  MOV     (R3),CSR          ;READ CSR
5723 006112  010137  002042                  MOV     R1,GOOD
5724 006116  032713  100020                  BIT     #BIT15!BIT4,(R3);IS IT BECAUSE OF A SBE OR DBE?
5725 006122  001004                          BNE     2$                ;BRANCH IF TRUE
5726 006124  104010                          ERROR   +10               ;BIT CLEAR ERROR
5727 006126  042760  000010  002432          BIC     #BIT3,CSRINFO(R0) ;CLEAR CSR OK BIT
5728 006134  000205                  2$:     RTS     R5
5729 006136  000000                  TRACE:  .WORD   0
```

```
                                        ;THE FOLLOWING ROUTINE DETERMINES WHICH CSR CONTROLS PROGRAM SPACE
                                        ;
006140  104424                  CTEST:  CACHOFF
006142  012737  177777  002502          MOV     #177777,PGMCSR
006150  012737  002000  172350          MOV     #2000,KIPAR4            ;SET UP MAP REGISTER
006156  012701  002362                  MOV     #TESTADD,R1
006162  012737  100000  002362          MOV     #100000,TESTADD
006170  012737  100002  002364          MOV     #100002,TESTADD+2
006176  005000                          CLR     R0                      ;CLEAR CSP COUNTER
006200  005037  002146                  CLR     CSRNO
006204  013703  002216                  MOV     TOTCSRS,R3              ;OBTAIN CSR MAP
006210  000240                          NOP                             ;DEBUG AID
006212  006303                  4$:     ASL     R3                      ;PUT HIGH ORDER BIT INTO C BIT
006214  103407                          BCS     2$                      ;BRANCH IF CSR EXISTS
006216  062700  000002          1$:     ADD     #2,R0                   ;UPDATE CSR COUNTER
006222  010037  002146                  MOV     R0,CSRNO
006226  005703                          TST     R3                      ;IS MAP EMPTY?
006230  001464                          BEQ     3$                      ;BRANCH IF SO
006232  000767                          BR      4$
006234  000240                  2$:     NOP                             ;DEBUG AID
006236  000241                          CLC                             ;CLEAR CARRY
006240  032760  000002  002432          BIT     #BIT1,CSRINFO(R0)       ;IS THIS PARITY MEMORY?
006246  001414                          BEQ     5$                      ;BRACH IF NOT
006250  052760  000004  172100          BIS     #BIT2,CSRADD(R0)        ;SET WRITE WRONG PARITY
006256  012771  123456  000000          MOV     #123456,@(R1)           ;WRITE DATA
006264  012771  123456  000002          MOV     #123456,@2(R1)
006272  005060  172100                  CLR     CSRADD(R0)              ;RESTORE CSR
006276  000414                          BR      6$
006300  012760  000000  172100  5$:     MOV     #0,CSRADD(R0)           ;CLEAR THE CSR UNDER TEST
006306  012771  123456  000000          MOV     #123456,@(R1)           ;WRITE DATA
006314  012771  123456  000002          MOV     #123456,@2(R1)
006322  012760  020006  172100          MOV     #20006,CSRADD(R0)       ;SET DIAG CHECK MODE
006330  005771  000000          6$:     TST     @(R1)                   ;WRITE CHECKBITS TO CSR
006334  016004  172100                  MOV     CSRADD(R0),R4           ;WRITE CSR TO R4
006340  032760  000002  002432          BIT     #BIT1,CSRINFO(R0)       ;PARITY MEMORY?
006346  001403                          BEQ     7$                      ;BRACH IF NOT
006350  005704                          TST     R4                      ;PARITY ERROR?
006352  100411                          BMI     8$                      ;BRACH IF SO
006354  000720                          BR      1$                      ;TRY NEXT CSR
006356  000240                  7$:     NOP                             ;DEBUG AID
006360  072427  177773                  ASH     #-5,R4
006364  042704  177600                  BIC     #^C177,R4
006370  022704  000157                  CMP     #157,R4                 ;CORRECT CHECKBITS?
006374  001310                          BNE     1$                      ;BRANCH IF NOT
006376  010037  002502          8$:     MOV     R0,PGMCSR
006402  000240                  3$:     NOP                             ;DEBUG AID
006404  104502                          CLRCSR                          ;CLEAR ALL CSR'S
006406  012771  000000  000000          MOV     #0,@(R1)                ;RESTORE TEST LOCATIONS
006414  012771  000000  000002          MOV     #0,@2(R1)
006422  023727  002502  177777          CMP     PGMCSR,#177777
006430  001402                          BEQ     FINT                    ;IF PROGRAM CSR NOT FOUND GO TO FINT
006432  000137  007036                  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
                                        ;
006436                          FINT:   SET4    #2$                     ;NE MEMORY TRAPS GO TO 2$
006444  012771  123456  000000  1$:     MOV     #123456,@(R1)           ;WRITE DATA AT FIRST LOCATION OF BANK 2 IN BOARD
006452  012771  123456  000002          MOV     #123456,@2(R1)          ;WRITE DATA AT SECOND LOCATION OF BANK 2 IN BOARD
006460  062737  010000  172350          ADD     #10000,KIPAR4           ;UPDATE PAR4 TO POINT TO UPPER BOARDS
006466  000766                          BR      1$                      ;KEEP GOING TILL NO MORE MEMORY
006470  012700  177776          2$:     MOV     #-2,R0
006474  013703  002216                  MOV     TOTCSRS,R3              ;PUT CSR MAP IN R3
006500  062700  000002          3$:     ADD     #2,R0                   ;UPDATE CSR COUNTER
006504  010037  002146                  MOV     R0,CSRNO                ;UPDATE CSRNO
006510  006303                          ASL     R3
006512  103403                          BCS     4$                      ;BRANCH IF CSR EXISTS
006514  005703                          TST     R3                      ;ANY CSR'S LEFT?
006516  001405                          BEQ     5$                      ;BRANCH IF NOT
006520  000767                          BR      3$                      ;LOOK FOR NEXT CSR
006522  012760  020006  172100  4$:     MOV     #20006,CSRADD(R0)       ;SET DIAGNOSTIC CHECK MODE IN CSR
006530  000763                          BR      3$                      ;LOOK FOR NEXT CSR
006532                          5$:     SET4    #6$                     ;NE MEMORY TRAPS NOW GO TO 6$
006540  012700  177776                  MOV     #-2,R0                  ;RESET CSR POINTER
006544  012737  002000  172350          MOV     #2000,KIPAR4            ;REMAP PAR4 TO POINT TO BANK 2
006552  005771  000000                  TST     @(R1)                   ;TEST NONASSERTED LOCATIONS
006556  062700  000002          6$:     ADD     #2,R0                   ;UPDTAE CSR POINTER
006562  010037  002146                  MOV     R0,CSRNO
006566  022700  000040                  CMP     #40,R0                  ;NOT FOUND?
006572  001515                          BEQ     10$                     ;BRANCH IF NOT
006574  016004  172100                  MOV     CSRADD(R0),R4           ;GET CSR CONTENTS
006600  072427  177773                  ASH     #-5,R4
006604  042704  177600                  BIC     #^C177,R4               ;CLEAR ALL BUT CHECKBITS
006610  022704  000157                  CMP     #157,R4                 ;PROPER CHECKBITS?
006614  001401                          BEQ     7$                      ;BRANCH IF SO
006616  000757                          BR      6$                      ;TRY NEXT CSR IF NOT
006620  110037  002502          7$:     MOVB    R0,PGMCSR               ;WRITE NON-ASSERTED CSR # IN PGMCSR
006624                                  SET4    #8$                     ;NE TRAPS GO TO 8$
006632  012700  177776                  MOV     #-2,R0
006636  013703  002216                  MOV     TOTCSRS,R3              ;PUT CSR MAP IN R3
006642  062700  000002          23$:    ADD     #2,R0                   ;UPDATE CSR COUNTER
006646  010037  002146                  MOV     R0,CSRNO                ;UPDATE CSRNO
006652  006303                          ASL     R3
006654  103403                          BCS     24$                     ;BRANCH IF CSR EXISTS
006656  005703                          TST     R3                      ;ANY CSR'S LEFT?
006660  001405                          BEQ     25$                     ;BRANCH IF NOT
006662  000767                          BR      23$                     ;LOOK FOR NEXT CSR
006664  012760  020006  172100  24$:     MOV     #20006,CSRADD(R0)       ;SET DIAGNOSTIC CHECK MODE IN CSR
006672  000763                          BR      23$                     ;LOOK FOR NEXT CSR
006674  012700  177776          25$:    MOV     #-2,R0
006700  005771  000002                  TST     @2(R1)                  ;TEST ASSERTED LOCATIONS
006704  062700  000002          8$:     ADD     #2,R0
006710  010037  002146                  MOV     R0,CSRNO
006714  022700  000040                  CMP     #40,R0
006720  001442                          BEQ     10$
006722  016004  172100                  MOV     CSRADD(R0),R4
006726  072427  177773                  ASH     #-5,R4
006732  042704  177600                  BIC     #^C177,R4
006736  022704  000157                  CMP     #157,R4                 ;PROPER CHECKBITS?
```

```
5841 006742  001401                        BEQ     9$                  ;BRANCH IF SO
5842 006744  000757                        BR      8$                  ;TRY NEXT CSR IF NOT
5843 006746  110037  002503          9$:   MOVB    R0,PGMCSR+1         ;WRITE ASSERTED CSR # IN PGMCSR
5844 006752  052737  100000  002502        BIS     #BIT15,PGMCSR       ;SET INTERLEAVED INDICATOR IN PGMCSR
5845 006760  104502                        CLRCSR
5846 006762  012737  002000  172350        MOV     #2000,KIPAR4
5847 006770                                SET4    #12$                ;NE MEMORY TRAPS GO TO 12$
5848 006776  012771  000000  000000  11$:  MOV     #0,@(R1)            ;WRITE DATA AT FIRST LOCATION OF BANK 2 IN BOARD
5849 007004  012771  000000  000002        MOV     #0,@2(R1)           ;WRITE DATA AT SECOND LOCATION OF BANK 2 IN BOARD
5850 007012  062737  010000  172350        ADD     #10000,KIPAR4       ;UPDATE PAR4 TO POINT TO UPPER BOARDS
5851 007020  000766                        BR      11$
5852 007022  104423                  12$:  CACHON
5853 007024  000404                        BR      CLRMEM
5854 007026                          10$:  TYPE    MSG126              ;ERROR - PROGRAM CSR NOT FOUND!
5855 007032  005037  002502                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.
                                ;
007036                  CLRMEM: SET4    #CLREX          ;NONEM TRAPS GO TO CLREX
007044  005037  006136          CLR     TRACE
007050  012737  000001  002074  MOV     #1,NOPAR        ;IGNORE PARITY ERRORS
007056  012737  002000  172350  MOV     #2000,KIPAR4    ;SET UP MAP TO START AT BANK 2
007064  012701  100000          MOV     #100000,R1      ;R1 MAPS TO KIPAR4
007070  020127  117776  1$:     CMP     R1,#117776      ;WHOLE 16K BANK DONE?   ;R-C
007074  001003                  BNE     2$              ;KEEP GOING IF NOT
007076  012737  177777  006136  MOV     #-1,TRACE       ;USE TRACE FLAG TO FLAG END OF BANK
007104  005021          2$:     CLR     (R1)+           ;CLEAR CONTENTS & INCREMENT
007106  005737  006136          TST     TRACE           ;EOB FLAG SET?
007112  001001                  BNE     3$              ;GO TO NEXT BANK IF SO
007114  000765                  BR      1$
007116  062737  000200  172350  3$:     ADD     #200,KIPAR4     ;SET MAP FOR NEXT BANK ;R-C
007124  005037  006136          CLR     TRACE           ;RESET FLAG
007130  012701  100000          MOV     #100000,R1      ;RESET R1
007134  000755                  BR      1$              ;CLEAR NEXT BANK
007136  000240          CLREX:  NOP
007140  005037  006136          CLR     TRACE
007144                          RES4
```

```
5885 007166                             ANA2:   SUBTST  <<MATCH ALL CSR'S WITH MEMORY>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       MATCH ALL CSR'S WITH MEMORY
                                        ;*****************************************************************************
5886                                    ;* THE SECOND PART OF THE ANALYSIS MATCHES UP THE CSR'S WITH THE MEMORY, AND
5887                                    ;* INSTALLS ALL THE INFORMATION FOUND IN THE CONFIGURATION TABLE.  FOR ECC,
5888                                    ;* THIS IS DONE BY TAKING EACH CSR FOUND IN THE PREVIOUS SECTION SEQUENTIALLY
5889                                    ;* AND CHECKING THROUGH ALL OF MEMORY, ONE BANK AT AT TIME, TO SEE WHICH BANKS
5890                                    ;* RESPOND TO THE CSR IN QUESTION. THE FIRST DOUBLE WORD PAIR IN EACH BANK ARE
5891                                    ;* WRITTEN WITH DATA AND DIAGNOSTIC CHECK MODE SET IN THE CSR IN ORDER TO AC-
5892                                    ;* COMPLISH THIS. ALL POSSIBLE CONFIGURATIONS OF DOUBLE WORD PAIRS (NON-INTER-
5893                                    ;* LEAVED, 64K INTERLEAVED, OR 128K INTERLEAVED) ARE CHECKED FOR EACH BANK
5894                                    ;* THROUGH USE OF TESTADD AND KERNEL INSTRUCTION PAGE ADDRESS REGISTERS 4 AND
5895                                    ;* 5. IF WE GET THE PROPER CHECKBITS BACK, WE HAVE A MATCH. IF NOT, THE ROUT-
5896                                    ;* INE CHECKS FOR SINGLE OR DOUBLE BIT ERRORS.
5897                                    ;*                 IF ONE OR THE OTHER IS FOUND, THE ERROR ADDRESS IS CHECKED
5898                                    ;* TO SEE IF IT IS THAT BANK.  IF IT IS, WE HAVE A MATCH.  AT THE END OF EACH
5899                                    ;* BANK PASS, FOR EACH CSR PASS, THE PROGRAM COMES UP WITH A NUMBER, STORED IN
5900                                    ;* "I", WHICH DENOTES THE FOLLOWING:
5901                                    ;*
5902                                    ;*       I       MEMORY DESCRIPTION
5903                                    ;*       -       ------------------
5904                                    ;*       0       NON-EXISTANT MEMORY
5905                                    ;*       1       NON-INTERLEAVED MEMORY
5906                                    ;*       2       64K INTERLEAVED, A1 NOT ASSERTED MEMORY
5907                                    ;*       3       128K INTERLEAVED, A1 NOT ASSERTED MEMORY
5908                                    ;*       4       64K INTERLEAVED, A1 ASSERTED MEMORY
5909                                    ;*       5       128K INTERLEAVED, A1 ASSERTED MEMORY
5910                                    ;*
5911                                    ;*  NOTE - I=2 THROUGH I=5 CAN ONLY OCCUR WITH MS11-M MEMORY.
5912                                    ;*
5913                                    ;* NOTE THAT PARITY MEMORY WRITES WRONG PARITY TO THE DOUBLE WORDS, THEN LOOKS
5914                                    ;* FOR THE PARITY ERROR BIT TO BE SET.  IF THE BIT IS SET, WE HAVE A MATCH.
5915                                    ;*
5916 007166                                     SET4    #100$                   ;NE MEMORY TRAPS GO TO 100$
5917 007174  005037  002274                     CLR     CHECK                   ;CLEAR CHECK
5918 007200  012701  002362                     MOV     #TESTADD,R1             ;SET UP THE VIRTUAL ADDR. POINTER
5919 007204  013703  002216                     MOV     TOTCSRS,R3              ;MOVE CSR MAP INTO R3
5920 007210  005000                             CLR     R0                      ;CLEAR THE CSR POINTER
5921 007212  005005                             CLR     R5                      ;CLEAR THE PROGRAM CSR STATUS POINTER
5922 007214  005737  002424                     TST     NO22BIT                 ;IS THIS AN 11/44 OR 11/24?
5923 007220  001403                             BEQ     7$                      ;BRANCH IF IT IS
5924 007222  005037  002530                     CLR     LASTBLOCK               ;ADJUST LASTBLOCK INDICATOR FOR 124K MACHINE
5925 007226  000413                             BR      1$                      ;BRANCH OVER NEXT PIECE OF CODE
5926 007230  022737  000167  002526  7$:       CMP     #167,LASTBANK           ;IS THERE UNIBUS MEMORY ABOVE 17000000?
5927 007236  001407                             BEQ     1$                      ;BRANCH IF NOT
5928 007240  013702  002526                     MOV     LASTBANK,R2             ;SET UP A NEW LAST BLOCK INDICATOR
5929 007244  005202                             INC     R2
5930 007246  072227  000011                     ASH     #9.,R2
5931 007252  010237  002530                     MOV     R2,LASTBLOCK
5932 007256  012702  000004             1$:     MOV     #4,R2                   ;R2 IS INDEX FOR CONFIG TABLE
5933 007262  012737  001000  172350             MOV     #1000,KIPAR4            ;SET KIPAR4 FOR BANK 1
5934 007270  012737  001000  172352             MOV     #1000,KIPAR5            ;SET KIPAR5 FOR BANK 1
5935 007276  006303                     2$:     ASL     R3                      ;DOES THIS CSR EXIST?
5936 007300  103420                             BCS     3$                      ;BRANCH IF IT DOES EXIST
5937 007302  062700  000002                     ADD     #2,R0                   ;INCREMENT THE CSR POINTER
5938 007306  010037  002146                     MOV     R0,CSRNO                ;STORE IT IN CSRNO ALSO
```

```
5939 007312  005703                        TST     R3                      ;ARE THERE ANY MORE CSR'S TO DO?
5940 007314  001370                        BNE     2$                      ;BRANCH IF ALL CSRS NOT DONE
5941 007316  012737  001000  172350        MOV     #1000,KIPAR4            ;RESTORE KIPAR4
5942 007324  012737  001200  172352        MOV     #1200,KIPAR5            ;RESTORE KIPAR5
5943 007332  013706  002534                MOV     KSTACK,SP               ;RESTORE STACK
5944 007336  000137  010536                JMP     SUBAAS                  ;JUMP TO SUBAAS IF ALL CSR'S ARE DONE
5945 007342  010037  002146          3$:   MOV     R0,CSRNO                ;MAKE SURE CSRNO IS UPDATED
5946 007346  104424                 13$:   CACHOFF                         ;TURN THE CACHE OFF
5947 007350  000240                        NOP
5948 007352  012737  100000  002362  45$:   MOV     #100000,TESTADD         ;SET UP VIRTUAL ADDRESS TO KIPAR4
5949 007360  012737  120002  002364        MOV     #120002,TESTADD+2       ;SET UP VIRTUAL ADDRESS TO KIPAR5
5950 007366  032762  000040  002624        BIT     #BIT5,CONFIG(R2)        ;IS THIS A BANK TO SKIP?
5951 007374  001402                        BEQ     43$                     ;NO - BRANCH AROUND NEXT INSTRUCTION
5952 007376  000137  010452                JMP     6$                      ;YES - GO TO END OF BANK
5953 007402  005037  002422          43$:  CLR     I                       ;CLEAR THE MEMORY CONFIGURATION COUNTER
5954 007406  005771  000000          4$:   TST     @(R1)                   ;TEST TO SEE THAT THERE IS MEMORY PRESENT
5955 007412  005237  002422                INC     I
5956 007416                                PUSH    @(R1),@2(R1)            ;SAVE THE LOCATIONS UNDER TEST
5957 007426  032760  000002  002432        B'T     #BIT1,CSRINFO(R0)       ;IS THIS PARITY MEMORY?
5958 007434  001414                        BEQ     34$                     ;NO - BRANCH
5959 007436  052760  000004  172100        BIS     #BIT2,CSRADD(R0)        ;SET WRITE WRONG PARITY
5960 007444  012771  123456  000000        MOV     #123456,@(R1)           ;SET THE FIRST LOCATION UNDER TEST
5961 007452  012771  123456  000002        MOV     #123456,@2(R1)          ;SET THE SECOND LUT
5962 007460  005060  172100                CLR     CSRADD(R0)              ;CLEAR THE CSR
5963 007464  000411                        BR      41$                     ;TEST LOCATIONS
5964 007466  012771  123456  000000  34$:   MOV     #123456,@(R1)           ;SET THE FIRST LOCATION UNDER TEST
5965 007474  012771  123456  000002        MOV     #123456,@2(R1)          ;SET THE SECOND LUT
5966 007502  104503                        CLR1CSR                         ;RESET CSR
5967 007504  104475                        CB1CSR                          ;SET DIAG. CHECK MODE IN CSR UNDER TEST
5968 007506  000240                        NOP                             ;DEBUG AID
5969 007510  005771  000000          41$:  TST     @(R1)                   ;READ THE FIRST LUT TO WRITE CKBITS. INTO CSR
5970 007514  104426                        READCSR                         ;READ THE CSR UNDER TEST
5971 007516  000240                        NOP                             ;DEBUG AID
5972 007520  013704  002144                MOV     CSR,R4                  ;GET THE CHECKBITS FROM THE CSR
5973 007524  000240                        NOP                             ;DEBUG AID
5974 007526  010437  002404                MOV     R4,TEMP                 ;SAVE IN TEMP FOR LATER
5975 007532  104503                        CLR1CSR                         ;RESET CSR
5976 007534                                POP     @2(R1),@(R1)            ;RESTORE LOCATIONS UNDER TEST
5977 007544  032760  000002  002432        BIT     #BIT1,CSRINFO(R0)       ;IS THIS PARITY MEMORY?
5978 007552  001404                        BEQ     42$                     ;NO - BRANCH
5979 007554  005704                        TST     R4                      ;DID WE GET A PARITY ERROR?
5980 007556  100414                        BMI     25$                     ;YES - FILL IN CONFIG TABLE
5981 007560  000137  010452                JMP     6$                      ;NO - JUMP TO END OF BANK
5982 007564  072427  177773          42$:  ASH     #-5,R4                  ;MANIPULATE THE CSR BITS
5983 007570  042704  177600                BIC     #^C177,R4               ;INTO A USABLE FORM.
5984 007574  000240                        NOP                             ;DEBUG AID
5985 007576  022704  000157                CMP     #157,R4                 ;DO THE CHECKBITS COMPARE TO WHAT WAS WRITTEN?
5986 007602  001402                        BEQ     25$                     ;BRANCH IF THERE IS A MATCH
5987 007604  000137  010132                JMP     22$                     ;ELSE BRANCH IF NOT THE SAME
5988                                        ;*
5989                                        ;* WE COME HERE IF THERE IS A MATCH
5990                                        ;*
5991 007610  010004                 25$:   MOV     R0,R4                   ;GET THE CSR NUMBER
5992 007612  006204                        ASR     R4                      ;SET IT UP FOR USE IN THE
5993 007614  000304                        SWAB    R4                      ;CONFIGURATION TABLE.
5994 007616  042704  170377                BIC     #170377,R4              ;CLEAR OFF EXTRANEOUS BITS
5995 007622  032737  000004  002422        BIT     #BIT2,I                 ;INTERLEAVED A1 ASSERTED MEMORY FOUND?
```

```
5996 007630 001402                          BEQ     15$                     ;BRANCH IF NOT
5997 007632 072427 000004                   ASH     #4,R4                   ;PUT CSR NUMBER IN INTERLEAVED CSR SLOT
5998 007636 050462 002624           15$:    BIS     R4,CONFIG(R2)           ;PUT CSR NUMBER IN CONFIG. TABLE
5999 007642 016004 002432                   MOV     CSRINFO(R0),R4          ;GET MEMORY TYPE
6000 007646 042704 177770                   BIC     #^C7,R4                 ;CLEAR OFF THE EXTRANEOUS BITS
6001 007652 000304                          SWAB    R4                      ;MOVE INTO PROPER POSITION
6002 007654 050462 002626                   BIS     R4,CONFIG+2(R2)         ;SET IT INTO THE CONFIG TABLE
6003 007660 022737 000001 002422            CMP     #1,I                    ;WAS THIS NON-INTERLEAVED MEMORY?
6004 007666 001431                          BEQ     24$                     ;BRANCH IF IT WAS
6005 007670 052762 010000 002626            BIS     #BIT12,CONFIG+2(R2)     ;SET THE INTERLEAVED BIT
6006 007676 010204                          MOV     R2,R4                   ;SAVE THE CURRENT BANK INDEX
6007 007700 032737 000001 002422            BIT     #BIT0,I                 ;WAS THIS 128K INTERLEAVED?
6008 007706 001006                          BNE     5$                      ;BRANCH IF TRUE
6009 007710 052762 004000 002626            BIS     #BIT11,CONFIG+2(R2)     ;SET 64K INTERLEAVED FLAG IN CONFIG
6010 007716 062704 000020                   ADD     #20,R4                  ;SET NEW BANK POINTER TO 4 BANKS AHEAD
6011 007722 000402                          BR      16$                     ;JUMP OVER NEXT INSTRUCTION
6012 007724 062704 000040           5$:     ADD     #40,R4                  ;SET NEW BANK POINTER 8 BANKS AHEAD
6013 007730 052764 000040 002624    16$:    BIS     #BIT5,CONFIG(R4)        ;SET SKIP ECC LOGIC TESTS FLAG
6014 007736 056264 002624 002624            BIS     CONFIG(R2),CONFIG(R4)   ;SET OTHER INFO INTO THAT BANK
6015 007744 056264 002626 002626            BIS     CONFIG+2(R2),CONFIG+2(R4)
6016                                        ;*
6017                                        ;* THIS SECTION IS EXECUTED ONLY WHEN THE BANK=1
6018                                        ;*
6019 007752 022737 001000 172350    24$:    CMP     #1000,KIPAR4            ;IS THIS BANK 1 ?
6020 007760 001402                          BEQ     30$                     ;BRANCH IF TRUE
6021 007762 000137 010312                   JMP     33$                     ;ELSE JUMP TO END OF THIS BANK
6022 007766 032737 100020 002404    30$:    BIT     #BIT15!BIT4,TEMP        ;WAS THERE A SBE OR DBE?
6023 007774 001417                          BEQ     10$                     ;BRANCH IF NOT
6024 007776 013704 002404                   MOV     TEMP,R4                 ;GET CSR CONTENTS
6025 010002 072427 177767                   ASH     #-9.,R4                 ;MAKE ERROR ADDRESS INTO BANK #
6026 010006 022704 000001                   CMP     #1,R4                   ;ERROR IN BANKS 0 OR 1?
6027 010012 003010                          BGT     10$                     ;BRANCH IF NOT
6028 010014 052762 000001 002624            BIS     #BIT0,CONFIG(R2)        ;SET ERROR FLAG IN CONFIG TABLE
6029 010022 105262 002626                   INCB    CONFIG+2(R2)            ;ADD ONE TO BANK ERROR COUNT
6030 010026                                 SET     CONFGERROR              ;PRINT CONFIG TABLE
6031 010034 053737 002630 002624    10$:    BIS     CONFIG+4,CONFIG         ;SET UP INFORMATION IN BANK ZERO
6032 010042 053737 002632 002626            BIS     CONFIG+6,CONFIG+2
6033 010050 000240                          NOP                             ;DEBUG AID
6034 010052 022737 000001 002422            CMP     #1,I                    ;WAS THIS NON-INTERLEAVED MEMORY
6035 010060 001002                          BNE     46$                     ;NO - BRANCH OVER NEXT STMT.
6036 010062 000137 010452                   JMP     6$                      ;YES - JUMP TO END OF THIS BANK
6037 010066 012704 000020           46$:    MOV     #20,R4                  ;SET UP COUNTER FOR 64K INTERLEAVED
6038 010072 032737 000001 002422            BIT     #BIT0,I                 ;WAS IT 128K INTERLEAVED?
6039 010100 001402                          BEQ     26$                     ;BRANCH IF NOT
6040 010102 062704 000020                   ADD     #20,R4                  ;SET UP COUNTER FOR 128K INTERLEAVED
6041 010106 053764 002624 002624    26$:    BIS     CONFIG,CONFIG(R4)       ;SET OTHER BANK WITH SAME INFORMATION
6042 010114 053764 002626 002626            BIS     CONFIG+2,CONFIG+2(R4)   ;AS IN BANK 0
6043 010122 052764 000040 002624            BIS     #BIT5,CONFIG(R4)        ;SET SKIP ECC LOGIC TESTS FLAG
6044 010130 000470                          BR      33$                     ;BRANCH
6045                                        ;*
6046                                        ;* IF CHECKBITS DID NOT MATCH, WE COME HERE
6047                                        ;*
6048 010132 032737 100020 002144    22$:    BIT     #BIT15!BIT4,CSR         ;SBE OR DBE FLAGS SET?
6049 010140 001001                          BNE     8$                      ;BRANCH IF TRUE
6050 010142 000463                          BR      33$                     ;CHECK TO SEE IF IT IS MS11-M
6051 010144 013704 002146           8$:     MOV     CSRNO,R4                ;GET CSRNO
6052 010150 042764 000006 172100            BIC     #6,CSRADD(R4)           ;TURN OFF DIAG CHECK & ECC DISABLE
```

```
6053 010156                                        PUSH    R0,R1                   ;SAVE R0 & R1
6054 010162  016401  172100                        MOV     CSRADD(R4),R1           ;GET CSR INFORMATION
6055 010166  072127  177773                        ASH     #-5,R1                  ;SET UP ERROR ADDRESS
6056 010172  042701  177600                        BIC     #^C177,R1
6057 010176  005737  002424                        TST     NO22BIT                 ;IS THIS AN 11/44 OR 11/24?
6058 010202  001015                                BNE     27$                     ;BRANCH IF NOT
6059 010204  052764  040000  172100                BIS     #BIT14,CSRADD(R4)       ;GET EXTENDED ERROR ADDRESS BITS
6060 010212  016400  172100                        MOV     CSRADD(R4),R0           ;READ FROM CSR
6061 010216  042764  040000  172100                BIC     #BIT14,CSRADD(R4)       ;TURN OFF EUB BIT
6062 010224  042700  177037                        BIC     #^C740,R0               ;SET UP EXTENDED BITS
6063 010230  006300                                ASL     R0
6064 010232  006300                                ASL     R0
6065 010234  060001                                ADD     R0,R1                   ;SET UP TOTAL ERROR ADDRESS
6066 010236  010104                        27$:    MOV     R1,R4                   ;SAVE IN R4
6067 010240                                        POP     R1,R0                   ;RESTORE R0 & R1
6068 010244  072427  000005                        ASH     #5,R4                   ;SET ERROR ADDRESS UP IN PAR NOTATION
6069 010250  020437  172350                        CMP     R4,KIPAR4               ;DOES IT EQUAL KIPAR4?
6070 010254  001001                                BNE     28$                     ;BRANCH IF FALSE
6071 010256  000403                                BR      35$                     ;YES - MARK INFO IN CONFIG TABLE
6072 010260  020437  172352                28$:    CMP     R4,KIPAR5               ;DOES IT EQUAL KIPAR5?
6073 010264  001012                                BNE     33$                     ;BRANCH IF FALSE
6074 010266  052762  000001  002624        35$:    BIS     #BIT0,CONFIG(R2)        ;SET BANK ERROR FLAG
6075 010274  105262  002626                        INCB    CONFIG+2(R2)            ;INCREMENT BANK ERROR COUNTER
6076 010300                                        SET     CONFGERROR              ;PRINT CONFIG TABLE
6077 010306  000137  007610                        JMP     25$                     ;YES - MARK INFO IN CONFIG TABLE
6078                                               ;*
6079                                               ;* THIS SECTION SETS UP ALL THE POSSIBLE CONFIGURATIONS OF
6080                                               ;* MS11-M MEMORY.
6081                                               ;*
6082 010312  032760  000003  002432        33$:    BIT     #BIT0!BIT1,CSRINFO(R0)  ;IS THIS MS11-M MEMORY?
6083 010320  001054                                BNE     6$                      ;NO - GO TO END OF BANK
6084 010322  032760  000004  002432                BIT     #BIT2,CSRINFO(R0)
6085 010330  001450                                BEQ     6$
6086 010332  022737  000001  002422                CMP     #1,I                    ;IS THIS 1ST TIME THROUGH?
6087 010340  103410                                BLO     18$                     ;BRANCH IF NOT
6088 010342  162737  000002  002364                SUB     #2,TESTADD+2            ;TRY AS 64K INTERLEAVED
6089 010350  062737  004000  172352                ADD     #4000,KIPAR5            ;A1 NON-ASSERTED MEMORY
6090 010356  000137  007406                        JMP     4$                      ;TRY TO MATCH AGAIN
6091 010362  022737  000004  002422        18$:    CMP     #4,I                    ;4TH TIME THROUGH?
6092 010370  001404                                BEQ     20$                     ;YES - BRANCH
6093 010372  022737  000002  002422                CMP     #2,I                    ;2ND TIME THROUGH
6094 010400  103405                                BLO     12$                     ;NO - BRANCH
6095 010402  062737  004000  172352        20$:    ADD     #4000,KIPAR5            ;TRY AS 128K INTERLEAVED
6096 010410  000137  007406                        JMP     4$                      ;TRY TO MATCH AGAIN
6097 010414  022737  000003  002422        12$:    CMP     #3,I                    ;THIRD TIME THROUGH?
6098 010422  103413                                BLO     6$                      ;NO - BRANCH
6099 010424  062737  000002  002362                ADD     #2,TESTADD              ;TRY TESTING THE BANK
6100 010432  062737  000002  002364                ADD     #2,TESTADD+2            ;AS A1 ASSERTED
6101 010440  162737  004000  172352                SUB     #4000,KIPAR5            ;64K INTERLEAVED MEMORY
6102 010446  000137  007406                        JMP     4$                      ;TRY TO MATCH AGAIN
6103                                               ;*
6104                                               ;*END OF BANK ROUTINE
6105                                               ;*
6106 010452  104503                        6$:     CLR1CSR                         ;CLEAR THE CSR UNDER TEST
6107 010454  062702  000004                        ADD     #4,R2                   ;UPDATE CONFIGURATION POINTER
6108 010460  062737  001000  172350                ADD     #1000,KIPAR4            ;UPDATE KIPAR4 TO NEXT BANK
6109 010466  013737  172350  172352                MOV     KIPAR4,KIPAR5           ;AND UPDATE KIPAR5
```

```
 6110 010474  000240                        NOP                         ;DEBUG AID
 6111 010476  023737  002530  172350        CMP     LASTBLOCK,KIPAR4    ;HAVE WE DONE THE WHOLE MEMORY SPACE?
 6112 010504  101402                        BLOS    19$                 ;BRANCH IF DONE                    ;R-C
 6113 010506  000137  007352                JMP     45$                 ;JUMP IF NOT DONE
 6114 010512  062700  000002          19$:  ADD     #2,R0               ;INCREMENT CSR POINTER
 6115 010516  000240                        NOP                         ;DEBUG AID
 6116 010520  104423                        CACHON                      ;TURN ON THE CACHE
 6117 010522  000137  007256                JMP     1$                  ;JUMP TO TRY NEXT CSR
 6118                                    ;
 6119 010526  062706  000004          100$: ADD     #4,SP               ;RESTORE STACK                     ;R-C
 6120 010532  000137  010452                JMP     6$                  ;GO TO END OF BANK ROUTINE         ;R-C
```

```
                                SUBAAS: CACHON          ;MAKE SURE THE CACHE IS ON
                                        ECCINIT         ;TRAP ON DOUBLE BIT ERRORS (NORMAL)
                                        NEWTST  <<TEST BANK 0 ACCESSES>>
                                ;*****************************************************************************
                                ;*TEST 2         TEST BANK 0 ACCESSES
                                ;*****************************************************************************
     010542  000004             TST2:   SCOPE
                                        ;THIS DOES A "TST" INSTRUCTION ON EVERY LOCATION IN BANK #0 TO SEE
                                        ;IF IT GETS ANY PARITY TRAPS.
                                        ;SINCE EVERY LOCATION IS EITHER LOADED OR WRITTEN INTO BY THE PROGRAM
                                        ;PRIOR TO THIS POINT - THEN A PARITY ERROR IMPLIES THAT THERE IS A
                                        ;HARDWARE FAILURE IN THE MEMORY.
                                        ;THESE ERRORS ARE COUNTED AND A FATAL ACTION IS TAKEN
010544  005037  002070                  CLR     PARCNT                  ;CLEAR PARITY ERROR COUNTER
010550  012737  000001  002074          MOV     #1,NOPAR                ;SET THE NO PARITY ERROR FLAG
010556  005037  002066                  CLR     NEMCNT                  ;CLEAR NON-EXISTANT MEMORY ERROR COUNTER
010562  012737  000001  002076          MOV     #1,NONEM                ;SET THE NON-EXISTANT MEMORY ERROR MODE TO COUNT
010570                                  SET4    #NONEXIST               ;TRAPS TO 4 GOTO NONEXIST
010576  005000                          CLR     R0
010600  012701  040000                  MOV     #SIZE,R1
010604  104424                          CACHOFF                         ;TURN CACHE OFF
010606  005720                  1$:     TST     (R0)+                   ;SEE IF I CAN DO A READ ACCESS WITHOUT A PARITY TRAP
010610  077102                          SOB     R1,1$
010612  104423                          CACHON                          ;TURN CACHE ON
                                        ;SEE IF ANY FAILURES
010614  005737  002070                  TST     PARCNT                  ;ANY PARITY ERRORS?
010620  001403                          BEQ     2$                      ;NO - SKIP
010622                                  FATAL   3
010630  005737  002066          2$:     TST     NEMCNT                  ;ANY NON-EXISTANT MEMORY (HOLES)?
010634  001406                          BEQ     3$                      ;SKIP IF EQUAL
010636  162737  000002  002032          SUB     #2,ADDRESS              ;UPDATE 1ST ADDRESS FAILURE FROM AUTO INCREMENT #
010644                                  FATAL   4
010652  053737  002104  002624  3$:     BIS     CPUBIT,CONFIG           ;SET CORRECT ACCESSED BIT ON BANK 0
010660                                  RES4                            ;RESET TRAPS TO 4 TO DEFAULT

010702                                  SUBTST  <<ENABLE ECC FOR CORRECT TRAPS>>
                                ;*****************************************************************************
                                ;*SUBTEST        ENABLE ECC FOR CORRECT TRAPS
                                ;*****************************************************************************
010702                                  IF #SW0 SET.IN @SWR OR ACTFLAG IS TRUE
104506                                    ENASBE                        ;TRAP ON SINGLE BIT ERRORS
                                        ELSE
104472                                    ECCINIT                       ;TRAP ON DOUBLE BIT ERRORS (NORMAL)
010726                                  END ;OF IF #SW0
```

```
                                        NEWTST  <<TEST BANKS 1-200 (OCTAL) FOR ZEROS & ONES>>
                                ;*******************************************************************************
                                ;*TEST 3         TEST BANKS 1-200 (OCTAL) FOR ZEROS & ONES
                                ;*******************************************************************************
 010726  000004                 TST3:   SCOPE
                                        ;EACH BANK IS TESTED FOR EXISTANCE AND IF IT EXISTS
                                        ;THEN IS IS TESTED FOR ZEROS & ONES.
                                        ;EXCEPT -
                                        ;       PROTECTED BANKS (WHERE THE PROGRAM IS) ARE ONLY TESTED BY
                                        ;       "TST" INSTRUCTIONS LIKE BANK #0
                                        ;ANY BAD BANKS ARE LOGGED IN THE CONFIGURATION TABLE.
                                        ;THIS ROUTINE IS ONLY DOING A SMART SIZE - NOT ACTUAL TESTING!
 010730  005037  002100                 CLR     BANK
 010734  012737  000001  002074         MOV     #1,NOPAR                ;SET NO PARITY ERROR FLAG
 010742  012737  000002  002076         MOV     #2,NONEM                ;SET NON-EXISTANT MEMORY MODE TO EXIT TEST LOOP
 010750                                 SET4    #NONEXIST               ;TRAPS TO 4 GOTO NONEXIST
 010756  022737  000001  003716         CMP     #1,PROTYP               ;IS THIS AN 11/44?
 010764  001407                         BEQ     1$                      ;BRANCH IF TRUE
 010766  012737  011564  002472         MOV     #MTST3+4,LINK1          ;SET UP LINKS
 010774  012737  011566  002474         MOV     #MTST3+6,LINK2
 011002  000411                         BR      TAG9$
 011004                         1$:     BMOV    MTST3                   ;PUT IN FAST MEMORY
 011012  012737  177644  002472         MOV     #UIPAR2,LINK1           ;SET UP LINKS
 011020  012737  177646  002474         MOV     #UIPAR3,LINK2
 011026  005237  002100         TAG9$:  INC     BANK
 011032  023737  002526  002100         CMP     LASTBANK,BANK           ;DONE?
 011040  103457                         BLO     TAG2$                   ;YES - SKIP TO NEXT TEST
 011042  013701  002100                 MOV     BANK,R1
 011046  006301                         ASL     R1
 011050  006301                         ASL     R1                      ;BANK * 4
 011052  010137  002102                 MOV     R1,BANKINDEX
 011056  005037  002072                 CLR     PATERR                  ;CLEAR PATTERN ERROR COUNTER
 011062  005037  002070                 CLR     PARCNT                  ;CLEAR PARITY ERROR COUNTER
 011066  005037  002066                 CLR     NEMCNT                  ;CLEAR NON-EXISTANT MEMORY COUNTER (HOLES)
 011072                                 MAP     BANK                    ;MAP SUPERVISOR SPACE (TEST AREA) TO BANK
 011106  105761  002624                 TSTB    CONFIG(R1)              ;IS THIS BANK PROTECTED?
 011112  100555                         BMI     TSTBANK                 ;YES - GO TEST BANK SPECIAL
 011114  012777  000207  171350 WARN1:  MOV     #207,@LINK1             ;PUT "RETURN" INSTRUCTION AFTER WRITE ROUTINE
 011122  012700  060000                 MOV     #FIRST,R0
 011126  010004                         MOV     R0,R4
 011130  012701  040000                 MOV     #SIZE,R1
 011134  010103                         MOV     R1,R3
 011136  005002                         CLR     R2                      ;DATA IS ZEROS
 011140  104424                         CACHOFF                         ;TURN CACHE OFF
 011142                                 TESTAREA                        ;ENTER SUPERVISOR MODE
 011150  022737  000001  003716         CMP     #1,PROTYP               ;IS THIS AN 11/44?
 011156  001403                         BEQ     1$                      ;BRANCH IF TRUE
 011160  004737  011560                 CALL    MTST3
 011164  000402                         BR      2$
 011166  004737  177640         1$:     CALL    FASTCITY                ;CALL TO THE USER INSTRUCTION PAR'S
 011172  104417                 2$:     KERNEL                          ;ENTER KERNEL MODE
 011174  104423                         CACHON                          ;TURN CACHE ON
 011176  000416                         BR      TAG3$                   ;SKIP NEXT INSTRUCTION
 011200  005037  002100         TAG2$:  CLR     BANK
 011204                                 RES4                            ;RESET TRAPS TO 4 TO DEFAULT
 011226  005037  002074                 CLR     NOPAR                   ;INDICATE DEFAULT PARITY ACTION
 011232  000564                         BR      SUBAAI
```

```
6214 011234  005737  002066          TAG3$:  TST     NEMCNT          ;ANY TRAPS?
6215 011240  001401                          BEQ     1$              ;NO - SKIP
6216 011242  000671                          BR      TAG9$           ;NOW - TRY NEXT BANK
6217 011244  104424                  1$:     CACHOFF                 ;TURN CACHE OFF
6218 011246                                  TESTAREA                ;ENTER SUPERVISOR MODE
6219 011254  004777  171214                  CALL    @LINK2          ;FINISH PATTERN
6220 011260  104417                          KERNEL                  ;ENTER KERNEL MODE
6221 011262  104423                          CACHON                  ;TURN CACHE ON
6222 011264  005737  002072                  TST     PATERR          ;ANY PATTERN ERRORS
6223 011270  001040                          BNE     2$              ;YES - SKIP
6224 011272  005737  002070                  TST     PARCNT          ;ANY PARITY ERRORS
6225 011276  001035                          BNE     2$              ;YES - SKIP
6226 011300  005737  002066                  TST     NEMCNT          ;ANY NON EXISTANT MEMORY
6227 011304  001032                          BNE     2$              ;YES - SKIP
6228 011306  012700  060000                  MOV     #FIRST,R0
6229 011312  010004                          MOV     R0,R4
6230 011314  012701  040000                  MOV     #SIZE,R1
6231 011320  010103                          MOV     R1,R3
6232 011322  013702  002554                  MOV     ONES,R2         ;DATA IS ONES
6233 011326  012777  000240  171136          MOV     #000240,@LINK1  ;PUT "NOP" INSTRUCTION BACK IN SUBROUTINE
6234 011334  104424                          CACHOFF                 ;TURN CACHE OFF
6235 011336                                  TESTAREA                ;ENTER TEST MODE
6236 011344  022737  000001  003716          CMP     #1,PROTYP       ;IS THIS AN 11/44?
6237 011352  001403                          BEQ     5$              ;BRANCH IF IT IS
6238 011354  004737  011560                  CALL    MTST3           ;DO IN MEMORY IF NOT
6239 011360  000402                          BR      6$              ;JUMP OVER NEXT INSTRUCTION
6240 011362  004737  177640          5$:     CALL    FASTCITY        ;CALL TO THE USER INSTRUCTION PAR'S
6241 011366  104417                  6$:     KERNEL                  ;ENTER KERNEL MODE
6242 011370  104423                          CACHON                  ;TURN CACHE ON
6243 011372  013700  002102          2$:     MOV     BANKINDEX,R0
6244 011376  005737  002072                  TST     PATERR          ;ANY PATTERN ERRORS?
6245 011402  001006                          BNE     3$              ;YES - SKIP
6246 011404  005737  002070                  TST     PARCNT          ;ANY PARITY ERRORS?
6247 011410  001003                          BNE     3$              ;YES - SKIP
6248 011412  005737  002066                  TST     NEMCNT          ;ANY HOLES?
6249 011416  001406                          BEQ     4$              ;NONE - SKIP
6250 011420  052760  000001  002624  3$:     BIS     #BIT0,CONFIG(R0) ;SET ERROR BIT IN THIS BANK
6251 011426                                  SET     CONFGERROR      ;FORCE PRINTING OF CONFIGURATION TABLE
6252 011434  053760  002104  002624  4$:     BIS     CPUBIT,CONFIG(R0) ;SET ACCESSED BIT
6253 011442  000137  011026                  JMP     TAG9$
6254
6255                                         ;TEST A PROTECTED BANK
6256 011446                          TSTBANK:PUSH    R1
6257 011450  012737  000001  002076          MOV     #1,NONEM        ;SET NON-EXISTANT MEMORY TO COUNT
6258 011456  012700  060000                  MOV     #FIRST,R0
6259 011462  012701  020000                  MOV     #20000,R1
6260 011466  104424                          CACHOFF                 ;TURN CACHE OFF
6261 011470                                  TESTAREA                ;ENTER TEST MODE
6262 011476  005720                  4$:     TST     (R0)+
6263 011500  077102                          SOB     R1,4$
6264 011502  104417                          KERNEL                  ;ENTER KERNEL MODE
6265 011504  104423                          CACHON                  ;TURN CACHE ON
6266 011506  012737  000002  002076          MOV     #2,NONEM        ;RESET NON-EXISTANT MEMORY TO EXIT TEST LOOP
6267 011514                                  POP     R1
6268 011516                                  IF PARCNT NE #0
6269 011524  052761  000001  002624            BIS   #BIT0,CONFIG(R1) ;ERROR BANK
6270 011532                                    SET   CONFGERROR
```

```
 6271 011540                                    END ;OF IF PARCNT
 6272 011540                                    IF NEMCNT EQ #0
 6273 011546  053761  002104  002624              BIS   CPUBIT,CONFIG(R1)       ;ACCESSED BANK
 6274 011554                                    END ;OF IF NEMCNT
 6275 011554  000137  011026                    JMP   TAG9$
 6276 011560  010220                MTST3:      MOV   R2,(R0)+                ;V177640
 6277 011562  077102                            SOB   R1,MTST3                ;V177642
 6278 011564  000240                            NOP                           ;V177644
 6279 011566  012401                2$:         MOV   (R4)+,R1                ;V177646
 6280 011570  020102                            CMP   R1,R2                   ;V177650
 6281 011572  001402                            BEQ   3$                      ;V177652
 6282 011574  005237  002072                    INC   PATERR                  ;V177654
 6283 011600  077306                3$:         SOB   P3,2$                   ;V177660
 6284 011602  000207                            RETURN                        ;V177662
```

```
                                SUBAAI: SUBTST  <<FIND SHADOW INHIBIT MODE POINTERS>>
                                ;*****************************************************************************
                                ;*SUBTEST       FIND SHADOW INHIBIT MODE POINTERS
                                ;*****************************************************************************
                                ;* THIS SECTION LOOKS FOR INTERLEAVED MS11-M MEMORIES AND FIGURES OUT
                                ;* WHERE THE SHADOW INHIBIT MODE POINTERS ARE LOCATED.  THESE AREAS
                                ;* ARE THEN MARKED AS PROGRAM SPACE.
011604  005037  002100          SHADL1: CLR     BANK                    ;RESET BANK TO ZERO
011610  004737  044300                  CALL    EXBANK                  ;SET BANK PARAMETERS
011614  013700  002102                  MOV     BANKINDEX,R0
                                        IF ACFLAG IS TRUE AND INTFLAG IS TRUE
                                          IF INT64K IS TRUE
011642  062700  000020                      ADD #20,R0                  ;POINT TO BANKINDEX + 4
011646  062737  000010  002100              ADD #10,BANK                ;POINT TO BANK + 8
                                          ELSE
011656  062702  000040                      ADD #40,R2                  ;POINT TO BANKINDEX + 8
011662  062737  000020  002100              ADD #20,BANK                ;POINT TO BANK + 16
                                          END; OF IF INT64K
011670  052760  000200  002624            BIS   #BIT7,CONFIG(R0)        ;MAKE NEW BANK PROGRAM SPACE
                                        ELSE
011700  005237  002100                    INC   BANK                    ;GO TO NEXT BANK
                                        END; OF IF ACFLAG
011704  023737  002526  002100          CMP     LASTBANK,BANK           ;HAVE WE DONE ALL THE BANKS?
011712  002336                          BGE     SHADL1                  ;BRANCH IF NOT
```

```
 6309 011714                                    NEWTST  <<ECC INHIBIT MODE POINTER TEST>>
                                        ;*****************************************************************************
                                        ;*TEST 4        ECC INHIBIT MODE POINTER TEST
                                        ;*****************************************************************************
      011714  000004                    TST4:   SCOPE
 6310                                           ;THE MS11-M OR MF11S-K INHIBIT ECC DISABLE AND DIAGNOSTIC CHECK MODE
 6311                                           ;ON THE BOTTOM  FIRST OR SECOND 16K WORDS CONTROLLED BY A CSR.  THIS
 6312                                           ;IS CONSIDERED  TO BE A  PROTECTED BANK  BY THE  PROGRAM.  IT MAY BE
 6313                                           ;QUITE  COMPLEX TO DETERMINE ON A  GIVEN SYSTEM  CONFIGURATION WHICH
 6314                                           ;BANKS CAN BE PROTECTED;
 6315                                           ;SO
 6316                                           ;THIS ROUTINE ATTEMPS TO  CREATE A DOUBLE BIT ERROR IN ADDRESS 0 & 2
 6317                                           ;OF  EVERY ECC BANK.  ECC HARDWARE WILL PREVENT THIS  FROM HAPPENING
 6318                                           ;IN  PROTECTED BANKS  WHICH SHOULD ALWAYS  INCLUDE BANK ZERO - WHERE
 6319                                           ;THE PROGRAM IS.
 6320                                           ;
 6321                                           ;
 6322                                           ;WARNING:!!!!!!!!!!
 6323                                           ;  IN CASE OF HARDWARE FAILURE IT IS COMMON  THAT A DOUBLE BIT ERROR
 6324                                           ;  WILL BE  CREATED ON  THE KERNEL  STACK & "CRASH"  THE  DIAGNOSTIC
 6325                                           ;DURING  THIS ROUTINE.  YOUR ONLY CLUE IS THAT YOU CAN GET AS FAR AS
 6326                                           ;THIS ROUTINE BUT NOT PAST IT!
 6327 011716  104424                           CACHOFF                                 ;TURN CACHE OFF
 6328 011720  012737  177777  002150           MOV     #-1,OLDCSR
 6329 011726                                    FOR BANK := #0 TO LASTBANK
 6330 011732  012701  060000                      MOV   #FIRST,R1                       ;SET UP VIRT ADDR POINTER
 6331 011736  004737  044300                      CALL  EXBANK
 6332 011742  013700  002102                      MOV   BANKINDEX,R0
 6333 011746                                      IF ACFLAG IS TRUE
 6334 011754                                        IF MKFLAG IS TRUE
 6335 011762                                        IF SKIPMK IS FALSE
 6336 011770                                          IF INTFLAG IS TRUE
 6337 011776  012703  040000                            MOV     #40000,R3              ;SET INDEX COUNTER
 6338 012002  012737  000001  002232                    MOV     #1,SPLTCSR             ;MAP AS INTERLEAVED BANK
 6339 012010                                          ELSE
 6340 012012  012703  000002                            MOV     #2,R3                  ;SET INDEX COUNTER
 6341 012016                                          END; OF IF INTFLAG
 6342 012016  116002  002625                          MOVB    CONFIG+1(R0),R2
 6343 012022  006302                                  ASL     R2
 6344 012024  042702  177741                          BIC     #^C36,R2
 6345 012030  010237  002146                          MOV     R2,CSRNO
 6346 012034                                          IF CSRNO NE OLDCSR
 6347 012044  013737  002146  002150                    MOV     CSRNO,OLDCSR
 6348 012052                                            IF PFLAG IS FALSE
 6349 012060  052760  000100  002624                      BIS  #BIT6,CONFIG(R0)
 6350 012066                                            END; OF IF PFLAG
 6351 012066  004737  012222                            CALL    IMPTEST
```

```
6353 012072                                          IF INTFLAG IS TRUE
6354 012100  116002  002625                            MOVB  CONFIG+1(R0),R2
6355 012104  072227  177775                            ASH   #-3,R2
6356 012110  042702  177741                            BIC   #^C36,R2
6357 012114  010237  002146                            MOV   R2,CSRNO
6358 012120  062701  000002                            ADD   #2,R1            ;FIX POINTER FOR A1 ASSERTED HALF
6359 012124  004737  012222                            CALL  IMPTEST
6360 012130  005037  002232                            CLR   SPLTCSR
6361 012134                                          END; OF IF INTFLAG
6362 012134                                        END; OF IF CSRNO
6363 012134                                      END; OF IF SKIPMK
6364 012134                                      END; OF IF MKFLAG
6365 012134                                    END; OF IF ACFLAG
6366 012134                                  END; OF FOR BANK
6367 012150                                  MAP                       ;MAP TEST SPACE TO BANK 0
6368 012164  005037  002100                  CLR     BANK
6369 012170                                  IF #SW0 SET.IN @SWR OR ACTFLAG IS TRUE
6370 012206  104506                            ENASBE                  ;TRAP ON SINGLE BIT ERRORS
6371 012210                                  ELSE
6372 012212  104472                            ECCINIT                 ;TRAP ON DOUBLE BIT ERRORS (NORMAL)
6373 012214                                  END; OF IF #SW0
6374 012214  104423                          CACHON                    ;TURN THE CACHE BACK ON
6375 012216  000137  012464                  JMP     SUBAAR            ;JUMP OVER THE SUBROUTINE
```

```
6377 012222  005004                    IMPTEST:CLR     R4
6378 012224                                    MAP BANK                                ;MAP SUPERVISOR SPACE (TEST AREA) TO BANK
6379 012240  005005                            CLR     R5
6380 012242  012737  020000  002144            MOV     #BIT13,CSR
6381 012250                                    TESTAREA                                ;ENTER TEST MODE
6382 012256                                    PUSH    (R1)                            ;SAVE TEST LOCATION
6383 012260  060301                            ADD     R3,R1                   ;INDEX TO NEXT LOCATION
6384 012262                                    PUSH    (R1)                    ;SAVE TEST LOCATION
6385 012264  104505                            CHK1DIS                         ;DISABLE ECC & WRITE CHECKBITS FOR 1 CSR
6386 012266  010411                            MOV     R4,(R1)                 ;WRITE CHECKBITS (ALL ZEROS)
6387 012270  160301                            SUB     R3,R1
6388 012272  010411                            MOV     R4,(R1)
6389 012274  104503                            CLR1CSR                                 ;CLEAR CSR
6390 012276  005711                            TST     (R1)                            ;READ CHECKBITS INTO REAL CSR
6391 012300  104501                            WAS1DBE                                 ;WAS THERE A DOUBLE BIT ERROR
6392
6393                                           ;THIS MAKES SURE THAT SBE'S DON'T LOOK LIKE PROTECTED AREAS
6394
6395 012302                                    ON.NOERROR ;1
6396 012304  012737  020000  002144              MOV   #BIT13,CSR
6397 012312  104505                              CHK1DIS                               ;DISABLE ECC & WRITE CHECKBITS FOR 1 CSR
6398 012314  013711  002554                      MOV   ONES,(R1)
6399 012320  060301                              ADD   R3,R1
6400 012322  013711  002554                      MOV   ONES,(R1)
6401 012326  160301                              SUB   R3,R1
6402 012330  104503                              CLR1CSR                               ;CLEAR CSR
6403 012332  005711                              TST   (R1)
6404 012334  104501                              WAS1DBE                               ;WAS THERE A DOUBLE BIT ERROR
6405 012336                                      ON.NOERROR ;2
6406 012340  012737  027400  002144                MOV #27400,CSR
6407 012346  104505                                CHK1DIS                             ;DISABLE ECC & WRITE CHECKBITS FOR 1 CSR
6408 012350  010411                                MOV R4,(R1)
6409 012352  060301                                ADD R3,R1                   ;ADD INDEX TO GET TO SECOND WORD
6410 012354  010411                                MOV R4,(R1)
6411 012356  160301                                SUB R3,R1                   ;SUBTRACT INDEX TO FIRST WORD
6412 012360  104503                                CLR1CSR                             ;CLEAR CSR
6413 012362  005711                                TST (R1)
6414 012364  104501                                WAS1DBE                             ;WAS THERE A DOUBLE BIT ERROR
6415 012366                                        ON.NOERROR ;3
6416 012370  012737  074000  002144                  MOV       #74C00,CSR          '
6417 012376  104505                                  CHK1DIS                           ;DISABLE ECC & WRITE CHECKBITS FOR 1 CSR
6418 012400  010411                                  MOV       R4,(R1)
6419 012402  060301                                  ADD       R3,R1           ;INDEX TO SECOND WORD
6420 012404  010411                                  MOV       R4,(R1)
6421 012406  104503                                  CLR1CSR                           ;CLEAR CSR
6422 012410  160301                                  SUB       R3,R1           ;GO BACK TO FIRST WORD
6423 012412  005711                                  TST       (R1)
6424 012414  104501                                  WAS1DBE                           ;WAS THERE A DOUBLE BIT ERROR
6425 012416                                        END ;OF ON.NOERROR ;3
6426 012416                                    END ;OF ON.NOERROR ;2
6427 012416                                END ;OF ON.NOERROR ;1
6428 012416                                ON.ERROR
6429 012420  005205                          INC   R5                              ;IDENTIFY AS BAD BANK
6430 012422                                END ;OF ON.ERROR
6431 012422  104471                        ECC1DIS                                 ;DISABLE ERROR CORRECTION
6432 012424  010411                        MOV     R4,(R1)                 ;CLEAR OUT DOUBLE BIT ERROR!
6433 012426  060301                        ADD     R3,R1                   ;INDEX TO SECOND WORD
```

```
6434 012430  010411                        MOV     R4,(R1)             ;CLEAR OUT DOUBLE BIT ERROR!
6435 012432  104503                        CLR1CSR
6436 012434  005705                        TST     R5
6437 012436  001405                        BEQ     1$
6438 012440  050560  002624                BIS     R5,CONFIG(R0)
6439 012444  105260  002626                INCB    CONFIG+2(R0)
6440 012450  104036                        ERROR   +36
6441 012452                          1$:   POP     (R1)                ;RESTORE TEST LOCATION (2ND WORD)
6442 012454  160301                        SUB     R3,R1               ;GO BACK TO FIRST WORD
6443 012456                                POP     (R1)                ;RESTORE TEST LOCATION (1ST WORD)
6444 012460  104417                        KERNEL
6445 012462  000207                        RETURN
```

```
6789
6790 012464                    SUBAAR: SET     STOPOK          ;PROGRAM CAN NOW BE HALTED
```

```
6793 012472                                             SUBTST  <<LEGAL CONFIGURATION CHECK>>
                                                ;*****************************************************************************
                                                ;*SUBTEST       LEGAL CONFIGURATION CHECK
                                                ;*****************************************************************************
6794 012472  012700  000020                             MOV     #16.,R0
6795 012476  012701  002432                             MOV     #CSRINFO,R1
6796 012502  005021                             1$:     CLR     (R1)+
6797 012504  077002                                     SOB     R0,1$
6801 012506                                               FOR BANK := #0 TO LASTBANK
6802 012512  004737  044300                                 CALL        EXBANK
6803 012516  013700  002102                                 MOV BANKINDEX,R0
6825
6826 012522                                                 IF ACFLAG IS TRUE
6827 012530  116003  002625                                   MOVB      CONFIG+1(R0),R3
6828 012534  042703  177760                                   BIC       #^C17,R3
6829 012540  006303                                           ASL       R3
6830 012542  005263  002432                                   INC       CSRINFO(R3)
6831 012546                                                   IF MKFLAG IS TRUE
6832                                                          ;MAKE SURE THAT EACH BANK HAS NO MORE THAN 2 CSRS
6833 012554                                                     BEGIN   LEGALCSR
6834 012554
6835 012562  116003  002625                                       IF INTFLAG IS TRUE
6836 012566  010304                                                 MOVB        CONFIG+1(R0),R3
6837 012570  042703  177760                                         MOV R3,R4
6838 012574  072427  177774                                         BIC #^C17,R3
6839 012600  042704  177760                                         ASH #-4,R4
6840 012604                                                         BIC #^C17,R4
6841 012610  042760  014000  002626                                 IF R3 EQ R4
6842 012616  042760  170000  002624                                   BIC       #BIT11!BIT12,CONFIG+2(R0)
6843 012624                                                         BIC       #170000,CONFIG(R0)
6844 012626                                                         LEAVE LEGALCSR
6845 012626                                                       END; OF IF R3
6846 012634                                                       IF KFLAG IS FALSE
6847 012636                                                         LEAVE     LEGALCSR
6848 012636  006303                                               END; OF IF KFLAG
6849 012640  006304                                               ASL R3
6850 012642  005263  002432                                       ASL R4
6851 012646  005264  002432                                       INC CSRINFO(R3)
6852 012652                                                       INC CSRINFO(R4)
6853 012654                                                     ELSE
6854 012656                                                       LEAVE       LEGALCSR
6855 012656                                                     END; OF IF INTFLAG
6856 012662                                                   TYPE    MSG124                    ;# OF CSR'S IS WRONG
6857 012672                                                   TYPOCS  BANK,<TYPE BANK #>,3
6858 012700                                                   SET     CONFGERROR
6859 012700                                                   END     LEGALCSR
6860 012700                                                 END ;OF IF MKFLAG
6861 012700                                               ENF ;OF IF ACFLAG
6862 012714                                             END; OF FOR BANK
6863 012720  005000                             PUSH    R5,R0                                 ;SAVE CONTENTS OF R5, R0
6864 012722  005001                             CLR     R0                                    ;CLEAR REGISTERS
6865 012724  005005                             CLR     R1
6866 012726  005037  013134                     CLR     R5
6867 012732  022761  000010  002432  2$:       CLR     MBERR                                 ;CLEAR ERROR INDICATOR
6868 012740  002043                             CMP     #10,CSRINFO(R1)                       ;IS CURRENT CSR <= 10
6869 012742  022761  000020  002432             BGE     5$                                    ;BRANCH IF SO
6870 012750  002003                             CMP     #20,CSRINFO(R1)                       ;IS CURRENT CSR < 20
                                                BGE     3$                                    ;BRANCH IF SO
```

```
6871 012752  004737  013256                    CALL    ILLCSR                  ;CALL ERROR ROUTINE
6872 012756  000434                            BR      5$                      ;TRY NEXT CSR
6873 012760  016005  002624            3$:     MOV     CONFIG(R0),R5           ;MOVE LOW WORD TO R5
6874 012764  032705  000002                    BIT     #BIT1,R5                ;DOES MEMORY EXIST HERE?
6875 012770  001415                            BEQ     4$                      ;BRANCH IF NOT
6876 012772  042705  170377                    BIC     #^C7400,R5              ;ISOLATE CSR NUMBER IN
6877 012776  072527  177771                    ASH     #-7,R5                  ;REGISTER 5
6878 013002  020501                            CMP     R5,R1                   ;IS IT THE CURRENT CSR?
6879 013004  001007                            BNE     4$                      ;TRY NEXT WORD OF CONFIG IF NOT
6880 013006  032760  010000  002626            BIT     #BIT12,CONFIG+2(R0)     ;IS IT INTERLEAVED?
6881 013014  001003                            BNE     4$                      ;BRANCH IF SO
6882 013016  012737  000001  013134            MOV     #1,MBERR                ;SET ERROR INDICATOR
6883 013024  062700  000004            4$:     ADD     #4,R0                   ;UPDATE CONFIG COUNTER
6884 013030  022700  000340                    CMP     #340,R0                 ;CONFIG TABLE ALL DONE?
6885 013034  001351                            BNE     3$                      ;BRANCH IF NOT
6886 013036  005737  013134                    TST     MBERR                   ;ERRORS FOUND?
6887 013042  001402                            BEQ     5$                      ;TRY NEXT CSR IF NOT
6888 013044  004737  013256                    CALL    ILLCSR                  ;CALL ERROR ROUTINE
6889 013050  005000            5$:     CLR     R0                      ;REINITIALIZE CONFIG COUNTER
6890 013052  005037  013134                    CLR     MBERR                   ;CLEAR ERROR INDICATOR
6891 013056  062701  000002                    ADD     #2,R1                   ;UPDATE CSR COUNTER
6892 013062  022701  000040                    CMP     #40,R1                  ;ALL CSR'S DONE?
6893 013066  001321                            BNE     2$                      ;BRANCH IF NOT
6894 013070                                    POP     R0,R5                   ;RESTORE REGISTERS
6895 013074  005037  013134                    CLR     MBERR                   ;RESET ERROR INDICATOR
6899 013100  012700  000734                    MOV     #734,R0         ;INDEX TO TOP OF CONFIG TABLE   ;R-C
6900 013104  032760  000002  002624    6$:     BIT     #BIT1,CONFIG(R0)        ;MEMORY PRESENT?                ;R-C
6901 013112  001003                            BNE     7$              ;BRANCH IF SO                   ;R-C
6902 013114  162700  000004                    SUB     #4,R0           ;TRY NEXT LOWER ENTRY IN CONFIG TABLE   ;R-C
6903 013120  000771                            BR      6$                                              ;R-C
6904 013122  006200            7$:     ASR     R0                                                      ;R-C
6905 013124  006200                            ASR     R0              ;DIVIDE INDEX BY 4 TO GET BANK #;R-C
6906 013126  010037  002526                    MOV     R0,LASTBANK     ;STORE IN LASTBANK              ;R-C
6907 013132  000402                            BR      SKUJ
6908 013134  000000            MBERR:  .WORD 0                         ;SAVE SPACE FOR ERROR INDICATOR
6909 013136  000000            PHEBE:  .WORD 0                         ;SAVE SPACE FOR ODD BOUNDARY INTERLEAVED INDICATOR
6910 013140  005000            SKUJ:   CLR     R0                      ;CLEAR CONFIG COUNTER
6911 013142  005037  013136                    CLR     PHEBE                   ;CLEAR COUNTER
6912 013146  032760  000002  002624    1$:     BIT     #BIT1,CONFIG(R0)        ;IS THERE MEMORY PRESENT?
6913 013154  001431                            BEQ     3$                      ;BRANCH IF NOT
6914 013156  032760  010000  002626            BIT     #BIT12,CONFIG+2(R0)     ;IS IT INTERLEAVED?
6915 013164  001005                            BNE     2$                      ;BRANCH IF SO
6916 013166  005237  013136                    INC     PHEBE                   ;INCREMENT COUNTER
6917 013172  062700  000004                    ADD     #4,R0                   ;INCREMENT CONFIG COUNTER
6918 013176  000763                            BR      1$                      ;TRY NEXT BANK
6919 013200  023727  013136  000010    2$:     CMP     PHEBE,#10               ;IS THE COUNTER EQUAL TO...
6920 013206  001417                            BEQ     4$                      ;ONE OF THE SPECIAL VALUES.
6921 013210  023727  013136  000030            CMP     PHEBE,#30               ;IF IT IS...
6922 013216  001413                            BEQ     4$                      ;BRANCH TO 4$
6923 013220  023727  013136  000050            CMP     PHEBE,#50
6924 013226  001407                            BEQ     4$
6925 013230  023727  013136  000070            CMP     PHEBE,#70
6926 013236  001403                            BEQ     4$
6927 013240  005037  013136            3$:     CLR     PHEBE                   ;CLEAR INDICATOR
6928 013244  000403                            BR      5$
6929 013246  012737  000001  013136    4$:     MOV     #1,PHEBE                ;SET INDICATOR
6930 013254  000421            5$:     BR      SUBAAP          ;BRANCH TO NEXT SUBTEST
```

```
6931 013256  010102                ILLCSR: MOV     R1,R2           ;R2 HAS CSR NUMBER
6932 013260  006202                        ASR     R2              ;MAKE ACCEPTABLE FOR PRINTING
6933 013262  022702  000012                CMP     #10.,R2
6934 013266  100002                        BPL     1$
6935 013270  062702  000007                ADD     #7,R2
6936 013274  062702  000060        1$:     ADD     #60,R2
6937 013300  110237  075327                MOVB    R2,MSGA122      ;PUT NUMBER INTO ERROR MESSAGE
6938 013304                                TYPE    MSG122
6939 013310                                SET     CONFGERROR
6940 013316  000207                        RETURN
```

```
6943 013320                             SUBAAP: SUBTST  <<PRINT CONFIGURATION DETAILS>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       PRINT CONFIGURATION DETAILS
                                        ;*****************************************************************************
6944 013320                                     CLEAR   BSIZE,KSIZE,LSIZE,MSIZE,PSIZE
6945 013344  013702  002526                     MOV     LASTBANK,R2
6946 013350  006302                             ASL     R2
6947 013352  006302                             ASL     R2
6948 013354                                     FOR R1 := #0 TO R2 BY #4
6949 013356                                      IF CPUBIT SET.IN CONFIG(R1)
6950 013366                                        IF #BIT10 SET.IN CONFIG+2(R1)
6951 013376                                          IF #BIT8 SET.IN CONFIG+2(R1)
6952 013406                                            IF #BIT9 SET.IN CONFIG+2(R1)
6953 013416                                              LET PSIZE := PSIZE + #1
6954 013422                                            ELSE
6955 013424                                              LET KSIZE := KSIZE + #1
6956 013430                                            END;IF BIT9
6957 013430                                          ELSE
6958 013432                                            IF #BIT9 SET.IN CONFIG+2(R1)
6959 013442                                              LET LSIZE := LSIZE + #1
6960 013446                                            ELSE
6961 013450                                              LET MSIZE := MSIZE + #1
6962 013454                                            END; IF BIT9
6963 013454                                          END;IF BIT8
6964 013454                                        ELSE
6965 013456                                          IF #BIT9 SET.IN CONFIG+2(R1)
6966 013466                                            IF #BIT8 SET.IN CONFIG+2(R1)
6967 013476                                              LET BSIZE := BSIZE + #1
6968 013502                                            END; OF IF #BIT8
6969 013502                                          END; OF IF #BIT9
6970 013502                                        END;IF BIT10
6971 013502                                       END; OF IF CPUBIT
6972 013502                                     END ;OF FOR ALL BANKS IN TABLE
6973
6974 013512  005037  002422                     CLR     I
6975 013516                                     FOR R1 := #0 TO #10 BY #2
6976 013520  006361  002344                       ASL   BSIZE(R1)
6977 013524  006361  002344                       ASL   BSIZE(R1)
6978 013530  006361  002344                       ASL   BSIZE(R1)
6979 013534  006361  002344                       ASL   BSIZE(R1)                 ;BSIZE(R1) := BSIZE(R1) * 16.
6980 013540  066137  002344  002422               ADD   BSIZE(R1),I               ;I <- I + BSIZE(R1)
6981 013546                                     END; FOR R1
6982 013560                                     FOR R1 := #0 TO #200 BY #4
6983 013562                                        IF CPUBIT SET.IN CONFIG(R1)
6984 013572                                          LET UNITOP := UNITOP + #1
6985 013576                                        END; OF IF CPUBIT
6986 013576                                     END; OF FOR R1
6987 013610  006337  002366                     ASL     UNITOP
6988 013614  006337  002366                     ASL     UNITOP
6989 013620  006337  002366                     ASL     UNITOP
6990 013624  006337  002366                     ASL     UNITOP                  ;UNITOP := UNITOP * 16.
6991 013630                                     IF I LT UNITOP THEN LET I := UNITOP
6992 013646                                     TYPE    $CRLF
6993 013652  005737  002344                     TST     BSIZE
6994 013656  001405                             BEQ     1$
6995 013660                                     TYPDEC  BSIZE
6996 013666                                     TYPE    MSG071
```

```
 6997 013672  005737  002346          1$:     TST     KSIZE
 6998 013676  001405                          BEQ     2$
 6999 013700                                   TYPDEC  KSIZE
 7000 013706                                   TYPE    MSG072
 7001 013712  005737  002350          2$:     TST     LSIZE
 7002 013716  001405                          BEQ     3$
 7003 013720                                   TYPDEC  LSIZE
 7004 013726                                   TYPE    MSG112
 7005 013732  005737  002352          3$:     TST     MSIZE
 7006 013736  001405                          BEQ     4$
 7007 013740                                   TYPDEC  MSIZE
 7008 013746                                   TYPE    MSG113
 7009 013752  005737  002354          4$:     TST     PSIZE
 7010 013756  001405                          BEQ     5$
 7011 013760                                   TYPDEC  PSIZE
 7012 013766                                   TYPE    MSG114
 7013 013772                          5$:     TYPDEC  I
 7014 014000                                   TYPE    MSG070
 7015 014004                                   IF #SW6 OFF.IN @SWR
 7016 014014  004737  036630                     CALL  PCONFIG
 7017 014020                                   END; OF IF #SW6
```

```
7020 014020                                     SUBTST  <<CHECK APT SIZING>>
                                               ;*****************************************************************************
                                               ;*SUBTEST       CHECK APT SIZING
                                               ;*****************************************************************************
7021 014020                                            IF APTFLAG IS TRUE AND APTSIZE IS TRUE
7022 014034  005037  002404                              CLR    TEMP
7023 014040  012700  063056                              MOV    #$MAMS1,R0
7024 014044                                              FOR R2 := #0 TO #4
7025 014046                                                IFB 1(R0) NE #0
7026 014054  111001                                          MOVB       (R0),R1
7027 014056  042701  177400                                  BIC        #177400,R1
7028 014062                                                  IF 2(R0) LT #0
7029 014070  000261                                            SEC
7030 014072                                                  ELSE
7031 014074  000241                                            CLC
7032 014076                                                  END ;OF IF 2(R0)
7033 014076  006101                                          ROL        R1
7034 014100  005201                                          INC        R1                ;TO COMPENSATE FOR 4 BANKS BEING (0-3)
7035 014102  006301                                          ASL        R1
7036 014104  006301                                          ASL        R1
7037 014106  006301                                          ASL        R1
7038 014110  006301                                          ASL        R1
7039 014112  163701  002404                                  SUB        TEMP,R1
7040 014116  010137  002404                                  MOV        R1,TEMP
7041 014122                                                  IFB 1(R0) EQ #3
7042 014132  060137  002372                                    ADD      R1,APTPAR
7043 014136                                                  END ;OF IFB 1(R0)
7044 014136                                                  IFB 1(R0) EQ #4
7045 014146  060137  002374                                    ADD      R1,APTECC
7046 014152                                                  END ;OF IFB 1(R0)
7047 014152  062700  000004                                  ADD        #4,R0
7048 014156                                                IFB 1(R0)
7049 014156                                              END ;OF FOR R2
7050 014166                                              IF APTPAR NE LSIZE OR APTECC NE MSIZE
7051 014206  104046                                        ERROR      +46
7052 014210                                              END ;OF IF APTPAR
7064 014210                                            END ;OF IF APTFLAG
```

```
                                LOOP:   NEWTST  <<DIAGNOSTIC MODE DISPATCH ROUTINE>>
                                ;*****************************************************************************
                                ;*TEST 5         DIAGNOSTIC MODE DISPATCH ROUTINE
                                ;*****************************************************************************
     014210  000004             TST5:   SCOPE
7067 014212  005037  002214             CLR     CONTFLAG
7068 014216  017700  166354             MOV     @SWR,R0         ;GET SWITCHES
7069 014222  042700  177761             BIC     #^C16,R0        ;MASK TO ONLY MODE BITS
7070 014226  004770  014236             CALL    @DISPTBL(R0)    ;DISPATCH TO ROUTINE THROUGH NEXT TABLE
7071 014232  000137  014256             JMP     MEMDONE         ;GO TO NEXT TEST
7072 014236  014710             DISPTBL:BAFPAF  ;MODE 0;BANKS FORWARD, PATTERNS FORWARD
7073 014240  015016                     BAFPAR  ;MODE 1;BANKS FORWARD, PATTERNS REVERSE
7074 014242  015124                     BAWPAF  ;MODE 2;BANKS WORST FIRST, PATTERNS FORWARD
7075 014244  015254                     BAWPAR  ;MODE 3;BANKS WORST FIRST, PATTERNS REVERSE
7076 014246  015404                     PAFBAF  ;MODE 4;PATTERNS FORWARD, BANKS FORWARD
7077 014250  015534                     PAFBAW  ;MODE 5;PATTERNS FORWARD, BANKS WORST FIRST
7078 014252  015706                     PARBAF  ;MODE 6;PATTERNS REVERSE, BANKS FORWARD
7079 014254  016036                     PARBAW  ;MODE 7;PATTERNS REVERSE, BANKS WORST FIRST
7080
7081 014256  004737  014610     MEMDONE:CALL    DOBACK                          ;CHECK BACKGROUND PATTERN
7082
7083 014262                             NEWTST<<UNIQUE BANK TEST>>
                                ;*****************************************************************************
                                ;*TEST 6         UNIQUE BANK TEST
                                ;*****************************************************************************
     014262  000004             TST6:   SCOPE
7084                                    ;MAKE SURE THAT EACH BANK CAN HAVE UNIQUE DATA
7085                                    ;WRITE AND READ THE BANK NUMBER IN EACH BANK (EXCEPT WHERE THE PROGRAM IS)
7086 014264                             IF SELONLY IS FALSE
7087 014272                               SET   HEADER,MUT
7088 014306  004737  024232               CALL  MT0027
7089 014312                               SET   HEADER
7090 014320  005037  002106               CLR   MUT
7091 014324                             END ;OF IF SELONLY
7092 014324  004737  014610             CALL    DOBACK          ;RESTORE BACKROUND PATTERN
7096
7097 014330                     FLUSH:  SUBTST  <<FLUSH OUT DBE'S>>
                                ;*****************************************************************************
                                ;*SUBTEST        FLUSH OUT DBE'S
                                ;*****************************************************************************
7098 014330  004737  024716             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
014334  005037  002412          $EOP:   CLR     FSINFLAG
014340  012700  002626                  MOV     #CONFIG+2,R0    ;MOVE 2ND WORD OF CONFIG TO R0
014344  042710  020000          1$:     BIC     #BIT13,(R0)     ;CLEAR BACKGROUND VALID BIT
014350  062700  000004                  ADD     #4,R0           ;INCREMENT TO NEXT BANK
014354  020027  003620                  CMP     R0,#3620        ;DONE?
014360  003771                          BLE     1$              ;NO - BRANCH
014362  013737  002570  002014          MOV     $ERTTL,LASTERROR
014370  005237  063034                  INC     $PASS           ;;INCREMENT THE PASS NUMBER
014374  042737  100000  063034          BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
014402                                  TYPE    MSG077          ;;TYPE 'END PASS #"
014406                                  IF #SW11 SET.IN @SWR OR QVFLAG IS TRUE
014424                                    TYPE  MSG035          ;QV
014430  005037  002316                    CLR   QVFLAG
014434                                  END ;OF IF SW11
014434                                  TYPDEC  $PASS
014442  013700  000042                  MOV     42,R0           ;;GET MONITOR ADDRESS
014446  001456                          BEQ     $DOAGAIN        ;;BRANCH IF NO MONITOR
014450  022700  002000          $ZAP42: CMP     #STACK,R0       ;ARE WE UNDER RT11
014454  001453                          BEQ     $DOAGAIN        ;YES - BRANCH
                                        ;WE ARE UNDER (HEAVEN HELP US) XXDP!
014456                                  PUSH    R0
014460  004737  045166                  CALL    SHUTUP
014464                                  POP     R0
014466  000005                          RESET                   ;;CLEAR THE WORLD
014470  004710                  $ENDAD: CALL    (R0)            ;;GO TO MONITOR
014472  000240                          NOP                     ;;SAVE ROOM
014474  000240                          NOP                     ;;FOR
014476  000240                          NOP                     ;;ACT11
014500                          $DOAGN: ;UNDO SHUTUP STUFF
                                        ;       RESTORE STACK
                                        ;       ENERGIZE UNIBUS MAP & 22 BIT ADDRESSING
                                        ;       ENERGIZE MEMORY MANAGEMENT
                                        ;       PUT LOADERS BACK HOME
014500  013706  002534                  MOV     KSTACK,SP
014504  005737  002424                  TST     NO22BIT                 ;IS THIS AN 11/44 OR 11/24?
014510  001003                          BNE     1$
014512  052737  000060  172516          BIS     #BIT5!BIT4,MMR3
014520  104420                  1$:     ENERGIZE                        ;TURN ON MEMORY MANAGEMENT
014522  013700  002536                  MOV     LOADHOME,R0             ;DESTINATION BANK
014526  012701  000001                  MOV     #1,R1                   ;SOURCE BANK
014532  004737  043750                  CALL    BANKMOV
014536                                  IF APTFLAG IS TRUE
014544                                    IF $USWR EQ $PASS
014554  012701  000050          APTHANG:    MOV #50,R1
014560  077001                  2$:         SOB R0,2$
014562  062737  000001  063036              ADD #1,$DEVCT
014570  005537  063040                      ADC $UNIT
014574  077107                              SOB R1,2$
014576  005237  063034                      INC $PASS
014602  000764                              BR  APTHANG
```

END OF PASS ROUTINE

```
 7158 014604                                    END ;OF IF $USWR
 7159 014604                                  END ;OF IF APTFLAG
 7160 014604  000137  014210          $DOAGAIN: JMP   LOOP              ;RETURN
```

END OF PASS ROUTINE

```
 7163 014610                        DOBACK: SUBTST  <<WRITE BACKGROUND PATTERNS>>
                                    ;*****************************************************************************
                                    ;*SUBTEST       WRITE BACKGROUND PATTERNS
                                    ;*****************************************************************************
 7164 014610  005037  002110                CLR     PATTERN
 7165 014614                                FOR BANK := #0 TO LASTBANK
 7166 014620  004737  044300                  CAL'  EXBANK
 7167 014624                                  IF ACFLAG IS TRUE AND RRFLAG IS FALSE
 7168 014640                                    SET          HEADER,MUT
 7169 014654  004737  017516                    CALL         MKTEST          ;CALL   MJTEST WOULD ALSO WORK
 7170 014660  005037  002106                    CLR          MUT
 7171 014664                                    SET          HEADER
 7172 014672                                  END ;OF IF ACFLAG
 7173 014672                                END ;OF FOR BANK
 7174 014706  000207                        RETURN
```

```
                                        .SBTTL  MTEST MODES

                                BAFPAF: SUBTST  <<BANKS FORWARD,PATTERNS FORWARD      **RECURSIVE**>>
                                ;*********************************************************************
                                ;*SUBTEST        BANKS FORWARD,PATTERNS FORWARD      **RECURSIVE**
                                ;*********************************************************************
                                        CLR     BANK                ;SET BANK TO 0
                                        ;START OF BANK LOOP
                                1$:     CALL    EXBANK              ;EXAMINE BANK
                                        TST     ACFLAG              ;CAN WE ACCESS THIS BANK?
                                        BEQ     4$                  ;NO - GO TO BANK LOOP TERMINATION
                                        TST     RRFLAG              ;RELOCATION REQUIRED?
                                        BNE     4$                  ;YES - GO TO BANK LOOP TERMINATION
                                        CLR     PATTERN             ;SET PATTERN TO 0
                                        ;START OF PATTERN LOOP
                                2$:     CALL    MTEST               ;GO TEST CORRECT MEMORY
                                        ;TERMINATION OF PATTERN LOOP
                                        CALL    INCPAT              ;GO SEE IF THIS IS THE LAST PATTERN
                                        BNE     2$                  ;NO - LOOP ON THIS PATTERN
                                        ;TERMINATION OF BANK LOOP
                                4$:     CLR     CONTFLAG
                                        CALL    INCBNK              ;NEXT HIGHER BANK
                                        BGE     1$                  ;IF NOT DONE - LOOP ON THIS BANK
                                        ;END OF LOOPS
                                        TST     RLFLAG              ;HAVE WE BEEN RELOCATED?
                                        BEQ     5$                  ;NO - SKIP
                                        RETURN                      ;YES - RETURN
                                5$:     CALL    RELOCATE            ;MOVE & MAP PROGRAM
                                        ON.ERROR THEN $RETURN
                                        ;**NOTE** RECURSIVE CALL
                                        CALL    BAFPAF              ;CALL SELF
                                        CALL    UNRELOCATE          ;UNMOVE & UNMAP PROGRAM
                                        RETURN
```

```
7209 015016                          BAFPAR: SUBTST  <<BANKS FORWARD,PATTERNS REVERSE      **RECURSIVE**>>
                                     ;*****************************************************************************
                                     ;*SUBTEST       BANKS FORWARD,PATTERNS REVERSE      **RECURSIVE**
                                     ;*****************************************************************************
7210 015016  005037  002100                  CLR     BANK                ;SET BANK TO 0
7211                                         ;START OF BANK LOOP
7212 015022  004737  044300          1$:     CALL    EXBANK              ;EXAMINE BANK
7213 015026  005737  002114                  TST     ACFLAG              ;CAN WE ACCESS THIS BANK?
7214 015032  001412                          BEQ     4$                  ;NO - GO TO BANK LOOP TERMINATION
7215 015034  005737  002122                  TST     RRFLAG              ;RELOCATION REQUIRED?
7216 015040  001007                          BNE     4$                  ;YES - GO TO BANK LOOP TERMINATION
7217 015042  004737  045002                  CALL    SETPAT              ;SET HIGH PATTERN FOR CORRECT MEMORY
7218                                         ;START OF PATTERN LOOP
7219 015046  004737  016210          2$:     CALL    MTEST               ;GO TEST CORRECT MEMORY
7220                                         ;TERMINATION OF PATTERN LOOP
7221 015052  005337  002110                  DEC     PATTERN             ;IS THIS THE LAST PATTERN?
7222 015056  100373                          BPL     2$                  ;NO - LOOP ON THIS PATTERN
7223                                         ;TERMINATION OF BANK LOOP
7224 015060  005037  002214          4$:     CLR     CONTFLAG
7225 015064  004737  045012                  CALL    INCBNK              ;NEXT HIGHER BANK
7226 015070  002354                          BGE     1$                  ;IF NOT DONE - LOOP ON THIS BANK
7227                                         ;END OF LOOPS
7228 015072  005737  002124                  TST     RLFLAG              ;HAVE WE BEEN RELOCATED?
7229 015076  001401                          BEQ     5$                  ;NO - SKIP
7230 015100  000207                          RETURN                      ;YES - RETURN
7231 015102  004737  042526          5$:     CALL    RELOCATE            ;MOVE & MAP PROGRAM
7232 015106                                  ON.ERROR THEN $RETURN
7233                                         ;**NOTE** RECURSIVE CALL
7234 015112  004737  015016                  CALL    BAFPAR              ;CALL SELF
7235 015116  004737  043416                  CALL    UNRELOCATE          ;UNMOVE & UNMAP PROGRAM
7236 015122  000207                          RETURN
```

```
7239 015124                                BAWPAF: SUBTST  <<BANKS WORST FIRST,PATTERNS FORWARD  **RECURSIVE**>>
                                           ;*****************************************************************************
                                           ;*SUBTEST       BANKS WORST FIRST,PATTERNS FORWARD  **RECURSIVE**
                                           ;*****************************************************************************
7240 015124  005037  002100                        CLR     BANK            ;SET BANK TO 0
7241                                               ;START OF BANK LOOP
7242 015130  004737  044300                1$:     CALL    EXBANK          ;EXAMINE BANK
7243 015134  005737  002114                        TST     ACFLAG          ;CAN WE ACCESS THIS BANK?
7244 015140  001415                                BEQ     4$              ;NO - GO TO BANK LOOP TERMINATION
7245 015142  005737  002126                        TST     BMFLAG          ;IS THIS BAD MEMORY (WORST FIRST)?
7246 015146  001412                                BEQ     4$              ;NO - GO TO BANK LOOP TERMINATION
7247 015150  005737  002122                        TST     RRFLAG          ;RELOCATION REQUIRED?
7248 015154  001007                                BNE     4$              ;YES - GO TO BANK LOOP TERMINATION
7249 015156  005037  002110                        CLR     PATTERN         ;SET PATTERN TO 0
7250                                               ;START OF PATTERN LOOP
7251 015162  004737  016210                2$:     CALL    MTEST           ;GO TEST CORRECT MEMORY
7252                                               ;TERMINATION OF PATTERN LOOP
7253 015166  004737  044766                        CALL    INCPAT          ;GO SEE IF THIS IS THE LAST PATTERN
7254 015172  001373                                BNE     2$              ;NO - LOOP ON THIS PATTERN
7255                                               ;TERMINATION OF BANK LOOP
7256 015174  005037  002214                4$:     CLR     CONTFLAG
7257 015200  004737  045012                        CALL    INCBNK          ;NEXT HIGHER BANK
7258 015204  002351                                BGE     1$              ;IF NOT DONE - LOOP ON THIS BANK
7259                                               ;END OF LOOPS
7260 015206  005137  002540                        COM     WORST           ;IS THIS AN EVEN NUMBERED PASS?
7261 015212  001003                                BNE     5$              ;YES - SKIP
7262                                               ;**NOTE** RECURSIVE CALL
7263 015214  004737  015124                        CALL    BAWPAF          ;CALL SELF
7264 015220  000207                                RETURN
7265 015222  005737  002124                5$:     TST     RLFLAG          ;HAVE WE BEEN RELOCATED?
7266 015226  001401                                BEQ     6$              ;NO - SKIP
7267 015230  000207                                RETURN                  ;YES - RETURN
7268 015232  004737  042526                6$:     CALL    RELOCATE        ;MOVE & MAP PROGRAM
7269 015236                                        ON.ERROR THEN $RETURN
7270                                               ;**NOTE** RECURSIVE CALL
7271 015242  004737  015124                        CALL    BAWPAF          ;CALL SELF
7272 015246  004737  043416                        CALL    UNRELOCATE      ;UNMOVE & UNMAP PROGRAM
7273 015252  000207                                RETURN
```

```
7276 015254                                BAWPAR: SUBTST  <<BANKS WORST FIRST,PATTERNS REVERSE  **RECURSIVE**>>
                                           ;*****************************************************************************
                                           ;*SUBTEST       BANKS WORST FIRST,PATTERNS REVERSE  **RECURSIVE**
                                           ;*****************************************************************************
7277 015254  005037  002100                        CLR     BANK                ;SET BANK TO 0
7278                                               ;START OF BANK LOOP
7279 015260  004737  044300                1$:     CALL    EXBANK              ;EXAMINE BANK
7280 015264  005737  002114                        TST     ACFLAG              ;CAN WE ACCESS THIS BANK?
7281 015270  001415                                BEQ     4$                  ;NO - GO TO BANK LOOP TERMINATION
7282 015272  005737  002126                        TST     BMFLAG              ;IS THIS BAD MEMORY (WORST FIRST)
7283 015276  001412                                BEQ     4$                  ;NO - GO TO BANK LOOP TERMINATION
7284 015300  005737  002122                        TST     RRFLAG              ;RELOCATION REQUIRED?
7285 015304  001007                                BNE     4$                  ;YES - GO TO BANK LOOP TERMINATION
7286 015306  004737  045002                        CALL    SETPAT              ;SET HIGH PATTERN FOR CORRECT MEMORY
7287                                               ;START OF PATTERN LOOP
7288 015312  004737  016210                2$:     CALL    MTEST               ;GO TEST CORRECT MEMORY
7289                                               ;TERMINATION OF PATTERN LOOP
7290 015316  005337  002110                        DEC     PATTERN             ;IS THIS THE LAST PATTERN?
7291 015322  100373                                BPL     2$                  ;NO - LOOP ON THIS PATTERN
7292                                               ;TERMINATION OF BANK LOOP
7293 015324  005037  002214                4$:     CLR     CONTFLAG
7294 015330  004737  045012                        CALL    INCBNK              ;NEXT HIGHER BANK
7295 015334  002351                                BGE     1$                  ;IF NOT DONE - LOOP ON THIS BANK
7296                                               ;END OF LOOPS
7297 015336  005137  002540                        COM     WORST               ;IS THIS AN EVEN NUMBERED PASS?
7298 015342  001003                                BNE     5$                  ;YES - SKIP
7299                                               ;**NOTE** RECURSIVE CALL
7300 015344  004737  015254                        CALL    BAWPAR              ;CALL SELF
7301 015350  0002^7                                RETURN
7302 015352  005737  002124                5$:     TST     RLFLAG              ;HAVE WE BEEN RELOCATED?
7303 015356  001401                                BEQ     6$                  ;NO - SKIP
7304 015360  000207                                RETURN                      ;YES - RETURN
7305 015362  004737  042526                6$:     CALL    RELOCATE            ;MOVE & MAP PROGRAM
7306 015366                                        ON.ERROR THEN $RETURN
7307                                               ;**NOTE** RECURSIVE CALL
7308 015372  004737  015254                        CALL    BAWPAR              ;CALL SELF
7309 015376  004737  043416                        CALL    UNRELOCATE          ;UNMOVE & UNMAP PROGRAM
7310 015402  000207                                RETURN
```

```
7313 015404                          PAFBAF: SUBTST  <<PATTERNS FORWARD,BANKS FORWARD      **RECURSIVE**>>
                                     ;*****************************************************************************
                                     ;*SUBTEST       PATTERNS FORWARD,BANKS FORWARD      **RECURSIVE**
                                     ;*****************************************************************************
7314 015404  005037  002110                 CLR     PATTERN          ;SET PATTERN TO 0
7315                                        ;START OF PATTERN LOOP
7316 015410  005037  002100          1$:    CLR     BANK             ;SET BANK TO 0
7317                                        ;START OF BANK LOOP
7318 015414  004737  044300          2$:    CALL    EXBANK           ;EXAMINE BANK
7319 015420  004737  044750                 CALL    BANKOK           ;CORRECT MEMORY FOR THIS BANK?
7320 015424  001010                        BNE     4$               ;NO - GO TO BANK LOOP TERMINATOR
7321 015426  005737  002114                 TST     ACFLAG           ;CAN WE ACCESS THIS BANK?
7322 015432  001405                        BEQ     4$               ;NO - GO TO BANK LOOP TERMINATION
7323 015434  005737  002122                 TST     RRFLAG           ;RELOCATION REQUIRED?
7324 015440  001002                        BNE     4$               ;YES - GO TO BANK LOOP TERMINATION
7325 015442  004737  016210                 CALL    MTEST            ;GO TEST CORRECT MEMORY
7326                                        ;TERMINATION OF BANK LOOP
7327 015446  005037  002214          4$:    CLR     CONTFLAG
7328 015452  004737  045012                 CALL    INCBNK           ;NEXT HIGHER BANK
7329 015456  002356                        BGE     2$               ;IF NOT DONE - LOOP ON THIS BANK
7330                                        ;TERMINATION OF PATTERN LOOP
7331 015460  004737  044766                 CALL    INCRPT           ;NEXT HIGHER PATTERN
7332 015464  001351                        BNE     1$               ;OK - LOOP; ELSE CONTINUE
7333                                        ;END OF LOOPS
7334 015466  005137  002132                 COM     TMFLAG           ;COMPLEMENT TYPE OF MEMORY
7335                                                                 ;IS THIS AN EVEN NUMBER PASS?
7336 015472  001403                        BEQ     5$               ;YES - SKIP
7337                                        ;**NOTE** RECURSIVE CALL
7338 015474  004737  015404                 CALL    PAFBAF           ;CALL SELF
7339 015500  000207                        RETURN
7340 015502  005737  002124          5$:    TST     RLFLAG           ;HAVE WE BEEN RELOCATED?
7341 015506  001401                        BEQ     6$               ;NO - SKIP
7342 015510  000207                        RETURN                   ;YES - RETURN
7343 015512  004737  042526          6$:    CALL    RELOCATE         ;MOVE & MAP PROGRAM
7344 015516                                ON.ERROR THEN $RETURN
7345                                        ;**NOTE** RECURSIVE CALL
7346 015522  004737  015404                 CALL    PAFBAF           ;CALL SELF
7347 015526  004737  043416                 CALL    UNRELOCATE       ;UNMOVE & UNMAP PROGRAM
7348 015532  000207                        RETURN
```

```
7351 015534                          PAFBAW: SUBTST  <<PATTERNS FORWARD,BANKS WORST FIRST  **RECURSIVE**>>
                                     ;*****************************************************************************
                                     ;*SUBTEST       PATTERNS FORWARD,BANKS WORST FIRST  **RECURSIVE**
                                     ;*****************************************************************************
7352 015534  005037  002110                 CLR     PATTERN          ;SET PATTERN TO 0
7353                                        ;START OF PATTERN LOOP
7354 015540  005037  002100          1$:    CLR     BANK             ;SET BANK TO 0
7355                                        ;START OF BANK LOOP
7356 015544  004737  044300          2$:    CALL    EXBANK           ;EXAMINE BANK
7357 015550  004737  044750                 CALL    BANKOK           ;CORRECT MEMORY FOR THIS BANK?
7358 015554  001013                        BNE     4$               ;NO - GO TO BANK LOOP TERMINATOR
7359 015556  005737  002114                 TST     ACFLAG           ;CAN WE ACCESS THIS BANK?
7360 015562  001410                        BEQ     4$               ;NO - GO TO BANK LOOP TERMINATION
7361 015564  005737  002126                 TST     BMFLAG           ;IS THIS BAD MEMORY (WORST FIRST)
7362 015570  001405                        BEQ     4$               ;NO - GO TO BANK LOOP TERMINATION
7363 015572  005737  002122                 TST     RRFLAG           ;RELOCATION REQUIRED?
7364 015576  001002                        BNE     4$               ;YES - GO TO BANK LOOP TERMINATION
7365 015600  004737  016210                 CALL    MTEST            ;GO TEST CORRECT MEMORY
7366                                        ;TERMINATION OF BANK LOOP
7367 015604  005037  002214          4$:    CLR     CONTFLAG
7368 015610  004737  045012                 CALL    INCBNK           ;NEXT HIGHER BANK
7369 015614  002353                        BGE     2$               ;IF NOT DONE - LOOP ON THIS BANK
7370                                        ;TERMINATION OF PATTERN LOOP
7371 015616  004737  044766                 CALL    INCRPT           ;NEXT HIGHER PATTERN
7372 015622  001346                        BNE     1$               ;OK - LOOP; ELSE CONTINUE
7373                                        ;END OF LOOPS
7374 015624  005137  002132                 COM     TMFLAG           ;COMPLEMENT TYPE OF MEMORY
7375                                                                 ;IS THIS AN EVEN NUMBER PASS?
7376 015630  001403                        BEQ     5$               ;YES - SKIP
7377                                        ;**NOTE** RECURSIVE CALL
7378 015632  004737  015534                 CALL    PAFBAW           ;CALL SELF
7379 015636  000207                        RETURN
7380 015640  005137  002540          5$:    COM     WORST            ;4TH PASS?
7381 015644  001003                        BNE     6$               ;YES - SKIP
7382                                        ;**NOTE** RECURSIVE CALL
7383 015646  004737  015534                 CALL    PAFBAW           ;CALL SELF
7384 015652  000207                        RETURN
7385 015654  005737  002124          6$:    TST     RLFLAG           ;HAVE WE BEEN RELOCATED?
7386 015660  001401                        BEQ     7$               ;NO - SKIP
7387 015662  000207                        RETURN                   ;YES - RETURN
7388 015664  004737  042526          7$:    CALL    RELOCATE         ;MOVE & MAP PROGRAM
7389 015670                                 ON.ERROR THEN $RETURN
7390                                        ;**NOTE** RECURSIVE CALL
7391 015674  004737  015534                 CALL    PAFBAW           ;CALL SELF
7392 015700  004737  043416                 CALL    UNRELOCATE       ;UNMOVE & UNMAP PROGRAM
7393 015704  000207                        RETURN
```

```
7396 015706                          PARBAF: SUBTST  <<PATTERNS REVERSE,BANKS FORWARD      **RECURSIVE**>>
                                     ;*****************************************************************************
                                     ;*SUBTEST       PATTERNS REVERSE,BANKS FORWARD      **RECURSIVE**
                                     ;*****************************************************************************
7397 015706  004737  045002                  CALL    HIPAT           ;SET HIGHTEST PATTERNS
7398                                 ;START OF PATTERN LOOP
7399 015712  005037  002100          1$:     CLR     BANK            ;SET BANK TO 0
7400                                 ;START OF BANK LOOP
7401 015716  004737  044300          2$:     CALL    EXBANK          ;EXAMINE BANK
7402 015722  004737  044750                  CALL    BANKOK          ;CORRECT MEMORY FOR THIS BANK?
7403 015726  001010                          BNE     4$              ;NO - GO TO BANK LOOP TERMINATOR
7404 015730  005737  002114                  TST     ACFLAG          ;CAN WE ACCESS THIS BANK?
7405 015734  001405                          BEQ     4$              ;NO - GO TO BANK LOOP TERMINATION
7406 015736  005737  002122                  TST     RRFLAG          ;RELOCATION REQUIRED?
7407 015742  001002                          BNE     4$              ;YES - GO TO BANK LOOP TERMINATION
7408 015744  004737  016210                  CALL    MTEST           ;GO TEST CORRECT MEMORY
7409                                 ;TERMINATION OF BANK LOOP
7410 015750  005037  002214          4$:     CLR     CONTFLAG
7411 015754  004737  045012                  CALL    INCBNK          ;NEXT HIGHER BANK
7412 015760  002356                          BGE     2$              ;IF NOT DONE - LOOP ON THIS BANK
7413                                 ;TERMINATION OF PATTERN LOOP
7414 015762  005337  002110                  DEC     PATTERN         ;NEXT LOWER PATTERN
7415 015766  100351                          BPL     1$              ;OK - LOOP; ELSE CONTINUE
7416                                 ;END OF LOOPS
7417 015770  005137  002132                  COM     TMFLAG          ;COMPLEMENT TYPE OF MEMORY
7418                                                                 ;IS THIS AN EVEN NUMBER PASS?
7419 015774  001403                          BEQ     5$              ;YES - SKIP
7420                                 ;**NOTE** RECURSIVE CALL
7421 015776  004737  015706                  CALL    PARBAF          ;CALL SELF
7422 016002  000207                          RETURN
7423 016004  005737  002124          5$:     TST     RLFLAG          ;HAVE WE BEEN RELOCATED?
7424 016010  001401                          BEQ     6$              ;NO - SKIP
7425 016012  000207                          RETURN                  ;YES - RETURN
7426 016014  004737  042526          6$:     CALL    RELOCATE        ;MOVE & MAP PROGRAM
7427 016020                                  ON.ERROR THEN $RETURN
7428                                 ;**NOTE** RECURSIVE CALL
7429 016024  004737  015706                  CALL    PARBAF          ;CALL SELF
7430 016030  004737  043416                  CALL    UNRELOCATE      ;UNMOVE & UNMAP PROGRAM
7431 016034  000207                          RETURN
```

```
7434 016036                        PARBAW: SUBTST  <<PATTERNS REVERSE,BANKS WORST FIRST  **RECURSIVE**>>
                                   ;*****************************************************************************
                                   ;*SUBTEST       PATTERNS REVERSE,BANKS WORST FIRST  **RECURSIVE**
                                   ;*****************************************************************************
7435 016036  004737  045002              CALL    HIPAT           ;SET HIGHTEST PATTERN
7436                               ;START OF PATTERN LOOP
7437 016042  005037  002100        1$:   CLR     BANK            ;SET BANK TO 0
7438                               ;START OF BANK LOOP
7439 016046  004737  044300        2$:   CALL    EXBANK          ;EXAMINE BANK
7440 016052  004737  044750              CALL    BANKOK          ;CORRECT MEMORY FOR THIS BANK?
7441 016056  001013                      BNE     4$              ;NO - GO TO BANK LOOP TERMINATOR
7442 016060  005737  002114              TST     ACFLAG          ;CAN WE ACCESS THIS BANK?
7443 016064  001410                      BEQ     4$              ;NO - GO TO BANK LOOP TERMINATION
7444 016066  005737  002126              TST     BMFLAG          ;IS THIS BAD MEMORY (WORST FIRST)
7445 016072  001405                      BEQ     4$              ;NO - GO TO BANK LOOP TERMINATION
7446 016074  005737  002122              TST     RRFLAG          ;RELOCATION REQUIRED?
7447 016100  001002                      BNE     4$              ;YES - GO TO BANK LOOP TERMINATION
7448 016102  004737  016210              CALL    MTEST           ;GO TEST CORRECT MEMORY
7449                               ;TERMINATION OF BANK LOOP
7450 016106  005037  002214        4$:   CLR CONTFLAG
7451 016112  004737  045012              CALL    INCBNK          ;NEXT HIGHER BANK
7452 016116  002353                      BGE     2$              ;IF NOT DONE - LOOP ON THIS BANK
7453                               ;TERMINATION OF PATTERN LOOP
7454 016120  005337  002110              DEC     PATTERN         ;NEXT LOWER PATTERN
7455 016124  100346                      BPL     1$              ;OK - LOOP; ELSE CONTINUE
7456                               ;END OF LOOPS
7457 016126  005137  002132              COM     TMFLAG          ;COMPLEMENT TYPE OF MEMORY
7458                                                             ;IS THIS AN EVEN NUMBER PASS?
7459 016132  001403                      BEQ     5$              ;YES - SKIP
7460                               ;**NOTE** RECURSIVE CALL
7461 016134  004737  016036              CALL    PARBAW          ;CALL SELF
7462 016140  000207                      RETURN
7463 016142  005137  002540        5$:   COM     WORST           ;4TH PASS?
7464 016146  001003                      BNE     6$              ;YES - SKIP
7465                               ;**NOTE** RECURSIVE CALL
7466 016150  004737  016036              CALL    PARBAW          ;CALL SELF
7467 016154  000207                      RETURN
7468 016156  005737  002124        6$:   TST     RLFLAG          ;HAVE WE BEEN RELOCATED?
7469 016162  001401                      BEQ     7$              ;NO - SKIP
7470 016164  000207                      RETURN                  ;YES - RETURN
7471 016166  004737  042526        7$:   CALL    RELOCATE        ;MOVE & MAP PROGRAM
7472 016172                        ON.ERROR THEN $RETURN
7473                               ;**NOTE** RECURSIVE CALL
7474 016176  004737  016036              CALL    PARBAW          ;CALL SELF
7475 016202  004737  043416              CALL    UNRELOCATE      ;UNMOVE & UNMAP PROGRAM
7476 016206  000207                      RETURN
```

```
7479 016210                          MTEST:  SUBTST  <<SUBR  SETUP MEMORY TEST>>
                                     ;*****************************************************************************
                                     ;*SUBTEST        SUBR    SETUP MEMORY TEST
                                     ;*****************************************************************************
7480 016210                                  SET     HEADER                  ;INITIALIZE HEADER MESSAGE TYPEOUT
7481 016216                                  SET     MUT                     ;INDICATE THERE IS A MEMORY UNDER TEST
7482 016224  005037  002256                  CLR     PASFLG
7483 016230  005737  002116                  TST     MKFLAG                  ;ECC?
7484 016234  001413                          BEQ     MT1                     ;NO - SKIP
7485 016236                                  BEGIN   HOLDLOOP
7486 016236                                    IF CONTFLAG IS TRUE THEN LEAVE HOLDLOOP
7487 016244                                    IF SKIPMK IS FALSE
7488 016252  004737  016304                      CALL        MKCONTROL
7489 016256                                    END; OF IF SKIPMK
7490 016256                                  END     HOLDLOOP
7491 016256  004737  017516                  CALL    MKTEST                  ;YES - DO ECC TESTS
7492 016262  000402                          BR      MT2
7493 016264  004737  017736          MT1:    CALL    MJTEST                  ;DO PARITY TESTS
7494 016270  005037  002106          MT2:    CLR     MUT                     ;NOW - NO MEMORY UNDER TEST
7495 016274                                  SET     HEADER                  ;ALLOW HEADERS NORMAL
7496 016302  000207                          RETURN
```

```
7499 016304                                  MKCONTROL:SUBTST        <<SUBR  TEST ECC CSR LOGIC DISPATCH>>
                                             ;*****************************************************************************
                                             ;*SUBTEST       SUBR    TEST ECC CSR LOGIC DISPATCH
                                             ;*****************************************************************************
7500                                                 ;THE NEXT TWO MODULES SOLVE THE PROBLEM OF
7501                                                 ;HOW TO RUN THE CSR TESTS ON EACH ECC MEMORY
7502                                                 ;
7503 016304                                          IF SELONLY IS TRUE THEN $RETURN
7504 016314                                          IF INHECC IS TRUE THEN $RETURN
7505 016324                                          PUSH    BANK,R0,R1,R2,R3
7506 016340  012737  060000  002224                 MOV     #FIRST,CSRFBANK         ;SET FIRST TEST ADDRESS TO FIRST ADDR.
7507 016346  012737  157776  002226                 MOV     #LAST,CSRLBANK
7508 016354  005037  002230                         CLR     CSRINT
7509 016360  005037  002232                         CLR     SPLTCSR
7510 016364  005037  002302                         CLR     CSRLOOP                 ; AND ZERO THE LOOP COUNTER
7511 016370  013700  002102                         MOV     BANKINDEX,R0            ;GET THE BANK INDEX
7512 016374  016001  002624                         MOV     CONFIG(R0),R1           ;GET CSR NUMBER
7513 016400  000301                                 SWAB    R1
7514 016402  042701  177760                         BIC     #^C17,R1
7515 016406  006301                                 ASL     R1
7516 016410  010137  002476                         MOV     R1,CSRHOLD              ;STORE IN THE LOW BYTE
7517 016414  005737  002134                         TST     INTFLAG                 ;IS THIS BANK INTERLEAVED?
7518 016420  001421                                 BEQ     1$                      ;BRANCH IF NOT INTERLEAVED
7519 016422  005237  002232                         INC     SPLTCSR
7520 016426  012737  120000  002226                 MOV     #120000,CSRLBANK
7521 016434  005237  002302                         INC     CSRLOOP                 ;WE MUST LOOP TWICE FOR AN INTERLEAVED BANK
7522 016440  005237  002230                         INC     CSRINT
7523 016444  016001  002624                         MOV     CONFIG(R0),R1           ;GET THE INTERLEAVE CSR NUMBER
7524 016450  072127  177775                         ASH     #-3,R1
7525 016454  042701  160777                         BIC     #^C17000,R1
7526 016460  050137  002476                         BIS     R1,CSRHOLD              ;STORE IT IN CSRHOLD'S UPPER BYTE
7527 016464  005003                        1$:      CLR     R3
7528 016466  116337  002476  002146  MKLOOP: MOVB    CSRHOLD(R3),CSRNO
7529 016474  042737  177741  002146                 BIC     #^C36,CSRNO             ;CLEAR ANY UNNECESSARY BITS
7530 016502                                            FOR R2 := #0 TO CSRINT
7531 016504                                              FOR CSRFIRST := CSRFBANK TO CSRLBANK BY #4000
7532 016512                                              MAP BANK                    ;MAP TEST SPACE TO BANK
7533 016526  104511                                      INVALIDATE                  ;INVALIDATE BACKROUND PATTERN
7534 016530                                                BEGIN CSRSTUFF
7535 016530  005037  002304                               CLR       SUCCESS
7536 016534                                                 IF ACFLAG IS TRUE AND RRFLAG IS FALSE
7537 016550  013737  002220  002222                          MOV    CSRFIRST,CSRLAST
7538 016556  062737  004000  002222                          ADD    #4000,CSRLAST
7539 016564                                                    FOR TESTADD := CSRFIRST TO CSRLAST BY #4
7540 016572  013737  002362  002364                             MOV       TESTADD,TESTADD+2
7541 016600  005737  002232                                   TST       SPLTCSR
7542 016604  001404                                         BEQ       1$
7543 016606  062737  040000  002364                         ADD       #40000,TESTADD+2
7544 016614  000403                                         BR        2$
7545 016616  062737  000002  002364                   1$:   ADD       #2,TESTADD+2
7546 016624  004737  017112                           2$:   CALL      SBETEST
7547 016630                                                 ON.NOERROR
7548 016632  104424                                           CACHOFF                 ;TURN CACHE OFF
7549 016634  005037  002074                                   CLR     NOPAR           ;INDICATE PARITY ACTION
7550 016640                                                    FOR I := #0 TO #27
7551 016644                                                      SET   HEADER
7552 016652  005037  002256                                     CLR   PASFLG
```

```
7553 016656                                                    LET R0 := I
7554 016662                                                    PUSH  R3                ;SAVE LOOP COUNTER
7555 016664  010637  002142                                    MOV   SP,CTLKVEC        ;SAVE VECTOR IN CSR OF ^K
7556 016670  162737  000002  002142                            SUB   #2,CTLKVEC
7557 016676  004737  017416                                    CALL CSRCASE
7558 016702                                                    POP   R3                ;RESTORE LOOP COUNTER
7559 016704                                                  END ;OF FOR I
7560 016720  104423                                          CACHON                    ;TURN CACHE ON
7561 016722                                                  SET     SUCCESS
7562 016730                                                  LEAVE CSRSTUFF
7563 016732                                                END ;OF ON.NOERROR
7564 016732                                              END ;OF FOR TESTADD
7565 016750                                            END ;OF IF
7566 016750                                          END CSRSTUFF
7567 016750                                          IF SUCCESS IS FALSE
7568 016756                                            TYPE    MSGA34
7569 016762                                            TYPOCS  BANK,<TYPES BANK NUMBER>,3
7570 016772                                            TYPE    MSGB34
7571 016776                                          END ;OF IF SUCCESS
7572 016776                                        END; OF FOR CSRFIRST
7573 017014  005237  002232                        INC  SPLTCSR
7574 017020                                      END; OF FOR R2
7575 017030  062737  000002  002224          ADD     #2,CSRFBANK
7576 017036  012737  000001  002232          MOV     #1,SPLTCSR
7577 017044  005203                          INC     R3
7578 017046  020337  002302                  CMP     R3,CSRLOOP
7579 017052  003002                          BGT     1$
7580 017054  000137  016466                  JMP     MKLOOP
7581 017060  104472                  1$:     ECCINIT         ;TRAP ON DOUBLE BIT ERRORS (NORMAL)
7582 017062                          SET     CONTFLAG
7583 017070  005037  002232          CLR     SPLTCSR
7584 017074                          POP     R3,R2,R1,R0,BANK
7585 017110  000207                  RETURN
```

```
 7588 017112                    SBETEST:SUBTST  <<CHECK FOR SBE FREE LOCATIONS>>
                                ;*****************************************************************************
                                ;*SUBTEST       CHECK FOR SBE FREE LOCATIONS
                                ;*****************************************************************************
 7589                                   ;IN ORDER TO DETERMINE IF A LOCATION IS SBE FREE I DO THIS
 7590                                   ;
 7591                                   ;WRITE ZEROS WITH ECC DISABLE
 7592                                   ;READ ZEROS BACK
 7593                                   ;IF NOT ZEROS THEN RETURN ERROR
 7594                                   ;
 7595                                   ;WRITE ZEROS WITH ECC ENABLED BUT TRAPS DISABLED
 7596                                   ;READ ZEROS BACK
 7597                                   ;IF NOT ZEROS THEN RETURN ERROR
 7598                                   ;
 7599                                   ;TEST THE LOCATION FROM THE PAR'S (WITH NO PROGRAM FETCHES)
 7600                                   ;IF THERE WERE ANY SBE'S OR DBE'S THEN RETURN ERROR
 7601                                   ;
 7602                                   ;COMPLIMENT ZEROS TO ONES WITH ECC DISABLE
 7603                                   ;READ ONES BACK
 7604                                   ;IF NOT ONES THEN RETURN ERROR
 7605                                   ;
 7606                                   ;WRITE 100,,100000,00000 (CHECKBITS COMPLIMENT OF BEFORE)
 7607                                   ;  WITH ECC ENABLED BUT TRAPS DISABLED
 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                                   ;IF NONE OF THE ABOVE FORCES A RETURN ERROR THEN RETURN NO.ERROR
 7612                                   .ENABL  LSB
 7613 017112                    PUSH    R0,R1,R4                ;PUSH R0,R1,R4 ONTO STACK
 7614 017120  013701  002362    MOV     TESTADD,R1
 7615 017124  013704  002364    MOV     TESTADD+2,R4
 7616 017130                    TESTAREA                        ;ENTER TEST MODE
 7617 017136  104424            CACHOFF                         ;TURN CACHE OFF
 7618 017140  104471            ECC1DIS                         ;DISABLE ECC ON 1 SELECTED CSR
 7619 017142                    CLEAR   (R1),(R4)
 7620 017146  005711            TST     (R1)
 7621 017150  001107            BNE     SBENT
 7622 017152  005714            TST     (R4)
 7623 017154  001105            BNE     SBENT
 7624
 7625 017156  104503            CLR1CSR                         ;CLEAR 1 SELECTED CSR
 7626 017160                    CLEAR   (R1),(R4)
 7627 017164  005711            TST     (R1)
 7628 017166  001100            BNE     SBENT
 7629 017170  005714            TST     (R4)
 7630 017172  001076            BNE     SBENT
 7631
 7632 017174  104510            TSTREAD                 ;TEST LOC (R1) & TST FOR SBE (WITHOUT FETCHES)
 7633 017176                    IF #BIT15!BIT4 SET.IN CSR
 7634 017206                      SET SKPERR            ;DISABLE ERRGEN'S ERROR PRINTOUT
 7635 017214  104512              ERRGEN
 7636 017216  013700  002430      MOV   ERRADD,R0
 7637 017222  072027  177774      ASH   #-4,R0
 7638 017226  042700  177600      BIC   #^C177,R0
 7639 017232                      IF BANK EQ R0 THEN GOTO SBENT
 7640 017240                    END; OF IF #BIT15
 7641 017240  104471            ECC1DIS                         ;DISABLE ECC ON 1 SELECTED CSR
```

```
7642 017242  005111                    COM     (R1)
7643 017244  005114                    COM     (R4)
7644 017246  023711  002554            CMP     ONES,(R1)
7645 017252  001046                    BNE     SBENT
7646 017254  023714  002554            CMP     ONES,(R4)
7647 017260  001043                    BNE     SBENT
7648
7649 017262  104503                    CLR1CSR                         ;CLEAR 1 SELECTED CSR
7650 017264  005011                    CLR     (R1)
7651 017266  012714  100000            MOV     #BIT15,(R4)
7652 017272  005711                    TST     (R1)
7653 017274  001035                    BNE     SBENT
7654 017276  022714  100000            CMP     #BIT15,(R4)
7655 017302  001032                    BNE     SBENT
7656
7657 017304  104510                    TSTREAD                 ;TEST LOC (R1) & TST FOR SBE (WITHOUT FETCHES)
7658 017306                            IF #BIT15!BIT4 SET.IN CSR
7659 017316                              SET SKPERR                    ;DISABLE ERRGEN'S ERROR PRINTOUT
7660 017324  104512                      ERRGEN
7661 017326  013700  002430              MOV   ERRADD,R0
7662 017332  072027  177774              ASH   #-4,R0
7663 017336  042700  177600              BIC   #^C177,R0
7664 017342                              IF BANK EQ R0 THEN GOTO SBENT
7665 017350                            END; OF IF #BIT15
7666
7667 017350  104417                    KERNEL                          ;ENTER KERNEL MODE
7668 017352  104473                    ECC1INIT                        ;INITIALIZE 1 SELECTED CSR
7669 017354  104423                    CACHON                          ;TURN CACHE ON
7670 017356                            POP     R4,R1,R0                ;POP R0,R1 & R4 FROM STACK
7671 017364                            $RETURN NOERROR
7672
7673 017370  104503            SBENT:  CLR1CSR                         ;CLEAR 1 SELECTED CSR
7674 017372                            CLEAR   (R1),(R4)
7675 017376  104417                    KERNEL                          ;ENTER KERNEL MODE
7676 017400  104473                    ECC1INIT                        ;INITIALIZE 1 SELECTED CSR
7677 017402  104423                    CACHON                          ;TURN CACHE ON
7678 017404                            POP     R4,R1,R0                ;POP R0,R1 & R4 FROM STACK
7679 017412                            $RETURN ERROR
7680                                   .DSABL  LSB
```

```
7683 017416                          CSRCASE:SUBIST  <<CSR PATTERN CASE STATEMENT>>
                                     ;*****************************************************************************
                                     ;*SUBTEST       CSR PATTERN CASE STATEMENT
                                     ;*****************************************************************************
7684 017416                                  CASE R0
7685                                           ;WARNING IF YOU CHANGE THIS TABLE ALSO
7686                                           ;CHANGE "$DDW0" - "$DDW5" (THE PATTERN BIT MAP)
7687
7688                                           ;PAT  TIME              DISCRIPTION
7689 017426  021172                 MKCSRT:    MT0006          ;<1 SEC          INITIAL DATA TEST
7690 017430  021270                            MT0010          ;<1 SEC          BYTE ADDRESSING TEST
7691 017432  023662                            MT0025          ;<1 SEC          INTERRUPT ENABLE TEST
7692 017434  021324                            MT0011          ;<2 SEC          CREATE SINGLE BIT ERROR TEST
7693 017436  021372                            MT0012          ;<1 SEC          WRITE BYTE CLEARS SBE TEST
7694 017440  021466                            MT0013          ; 1 SEC          CREATE DOUBLE BIT ERROR TEST
7695 017442  021542                            MT0014          ; 1 SEC          WRITE INHIBIT DURING DATIP WITH DBE
7696 017444  021620                            MT0015          ; 1 SEC          WRITE INHIBIT OF BYTE WITH DBE
7697 017446  021666                            MT0016          ;<1 SEC          WRITE INHIBIT OF WORD WITH DBE
7698 017450  026414                            MT0999          ; 0 SEC          NULL TEST
7699 017452  026414                            MT0999          ; 0 SEC          NULL TEST
7700 017454  026414                            MT0999          ; 0 SEC          NULL TEST
7701 017456  026414                            MT0999          ; 0 SEC          NULL TEST
7702 017460  026414                            MT0999          ; 0 SEC          NULL TEST
7703 017462  026414                            MT0999          ; 0 SEC          NULL TEST
7704 017464  026414                            MT0999          ; 0 SEC          NULL TEST
7705 017466  026414                            MT0999          ; 0 SEC          NULL TEST
7706 017470  026414                            MT0999          ; 0 SEC          NULL TEST
7707 017472  026414                            MT0999          ; 0 SEC          NULL TEST
7708 017474  026414                            MT0999          ; 0 SEC          NULL TEST
7709 017476  026414                            MT0999          ; 0 SEC          NULL TEST
7710 017500  026414                            MT0999          ; 0 SEC          NULL TEST
7711 017502  026414                            MT0999          ; 0 SEC          NULL TEST
7712 017504  026414                            MT0999          ; 0 SEC          NULL TEST
7713 017506                          END ;OF CASE R0
7714 017514  000207                  RETURN
```

CSR PATTERN CASE STATEMENT

```
 7717 017516                                MKTEST: SUBTST  <<SUBR  ECC TEST DISPATCH>>
                                            ;*****************************************************************************
                                            ;*SUBTEST       SUBR    ECC TEST DISPATCH
                                            ;*****************************************************************************
 7721 017516                                        IF #SW0 SET.IN @SWR OR ACTFLAG IS TRUE
 7722 017534  104470                                  ECCDIS                        ;DISABLE ERROR CORRECTION
 7723 017536                                        ELSE
 7724 017540  104502                                  CLRCSR                        ;CLEAR ALL CSR'S
 7725 017542                                        END ;OF IF
 7726 017542  012737  000002  002074                MOV     #2,NOPAR              ;INDICATE PARITY ACTION
 7727 017550  012737  000002  002276                MOV     #2,PCBUMP             ;TRAPS ADD 2 TO PC
 7728 017556  013700  002110                        MOV     PATTERN,R0            ;GET PATTERN NUMBER
 7729 017562  006300                                ASL     R0                    ;MAKE IT A WORD ADDRESS
 7730 017564                                        IF MKPAT(R0) NE #MT0034 AND MKPAT(R0) NE #MT0999
 7731 017604  104511                                  INVALIDATE                  ;INVALIDATE BACKGROUND PATTERN ON 'BANK''
 7732 017606                                        END ;OF IF MKPAT(R0)
 7733 017606  010637  002142                        MOV     SP,CTLKVEC            ;SAVE VECTOR IN CASE OF ^K
 7734 017612  162737  000002  002142                SUB     #2,CTLKVEC
 7735 017620  004770  017656                        CALL    @MKPAT(R0)            ;INDEX OFF TABLE
 7736 017624                                        IF #SW0 SET.IN @SWR OR ACTFLAG IS TRUE
 7737 017642  104506                                  ENASBE                        ;TRAP ON SINGLE BIT ERRORS
 7738 017644                                        ELSE
 7739 017646  104472                                  ECCINIT                       ;TRAP ON DOUBLE BIT ERRORS (NORMAL)
 7740 017650                                        END ;OF IF #SW0
 7741 017650  005037  002074                        CLR     NOPAR                 ;INDICATE PARITY ACTION
 7742 017654  000207                                RETURN
 7743
 7744                                               ;WARNING IF YOU CHANGE THIS TABLE ALSO
 7745                                               ;CHANGE '$DDW0'' - ''$DDW5'' (THE PATTERN BIT MAP)
 7746                                               ;PAT    TIME              DISCRIPTION
 7747 017656                                MKPAT:  ;NOTE MT0034 MUST BE FIRST & LAST
 7748 017656  026130                                MT0034  ;<1 SEC           ;SOFT ERROR - BACKGROUND PATTERN TEST
 7749 017660  021734                                MT0017  ;<1 SEC           ;HOLDING 1'S & 0'S TEST
 7750 017662  021226                                MT0007  ;<1 SEC           ;ADDRESS BIT TEST
 7751 017664  020202                                MT0001  ;<1 SEC           ;ADDRESS TEST
 7752 017666  020322                                MT0002  ;<1 SEC           ;COMPLEMENT ADDRESS TEST
 7753 017670  020714                                MT0004  ; 1 SEC           ;ROTATING ZEROS TEST
 7754 017672  021036                                MT0005  ; 1 SEC           ;ROTATING ONES TEST
 7755 017674  023046                                MT0021  ; 1 SEC           ;MARCHING 0'S & 1'S TEST
 7756 017676  021756                                MT0020  ;<1 SEC           ;MARCHING 1'S & 0'S IN CHECK BITS
 7757 017700  023320                                MT0022  ;10 SEC           ;REFRESH & SHIFTING DIAGONAL  TEST
 7758 017702  023730                                MT0026  ;<1 SEC           ;RANDOM DATA TEST
 7759 017704  023416                                MT0024  ;20 SEC           ;FAST GALLOPING PATTERN TEST
 7760 017706  025220                                MT0031  ; 3 SEC           ;SOB-A-LONG TEST
 7761 017710  025410                                MT0032  ;<1 SEC           ;WRITE RECOVERY TEST
 7762 017712  025742                                MT0033  ;35 SEC           ;BRANCH GOBBLE TEST
 7763 017714  026130                                MT0034  ;<1 SEC           ;SOFT ERROR - BACKGROUND PATTERN TEST
 7764                                               ;NOTE MT0034 MUST BE FIRST & LAST
 7765 017716  026414                                MT0999  ; 0 SEC           ;NULL TEST
 7766 017720  026414                                MT0999  ; 0 SEC           ;NULL TEST
 7767 017722  026414                                MT0999  ; 0 SEC           ;NULL TEST
 7768 017724  026414                                MT0999  ; 0 SEC           ;NULL TEST
 7769 017726  026414                                MT0999  ; 0 SEC           ;NULL TEST
 7770 017730  026414                                MT0999  ; 0 SEC           ;NULL TEST
 7771 017732  026414                                MT0999  ; 0 SEC           ;NULL TEST
 7772 017734  026414                                MT0999  ; 0 SEC           ;NULL TEST
```

```
7775 017736                                MJTEST: SUBTST  <<SUBR  PARITY TEST DISPATCH>>
                                           ;*****************************************************************************
                                           ;*SUBTEST       SUBR    PARITY TEST DISPATCH
                                           ;*****************************************************************************
7779 017736  012737  000002  002074                MOV     #2,NOPAR                ;INDICATE PARITY ACTION
7780 017744  012737  000002  002276                MOV     #2,PCBUMP               ;TRAPS ADD 2 TO PC
7781 017752  012737  060000  002362                MOV     #FIRST,TESTADD
7782 017760  012737  060002  002364                MOV     #FIRST+2,TESTADD+2
7783 017766  013700  002110                        MOV     PATTERN,R0              ;GET PATTERN NUMBER
7784 017772  006300                                ASL     R0                      ;MAKE IT A WORD ADDRESS
7785 017774                                        IF MJPAT(R0) NE #MT0034 AND MJPAT(R0) NE #MT0999
7786 020014  104511                                  IA[illegible]TE               ;INVALIDATE BACKGROUND PATTERN ON 'BANK"
7787 020016                                        END [illegible] IF MJPAT(R0)
7788 020016  010637  002142                        MOV     SP,CTLKVEC              ;SAVE VECTOR IN CASE OF ^K
7789 020022  162737  000002  002142                SUB     #2,CTLKVEC
7790 020030  004770  020042                        CALL    @MJPAT(R0)              ;INDEX OFF TABLE
7791 020034  005037  002074                        CLR     NOPAR                   ;INDICATE PARITY ACTION
7792 020040  000207                                RETURN
7793
7794                                               ;WARNING IF YOU CHANGE THIS TABLE ALSO
7795                                               ;CHANGE "$DDW0" - "$DDW5" (THE PATTERN BIT MAP)
7796
7797                                               ;PAT    TIME            DISCRIPTION
7798 020042                                MJPAT:  ;NOTE MT0034 MUST BE FIRST & LAST
7799 020042  026130                                MT0034  ;<1 SEC         ;SOFT ERROR - BACKGROUND PATTERN TEST
7800 020044  021172                                MT0006  ;<1 SEC         ;INITIAL DATA TEST
7801 020046  021734                                MT0017  ;<1 SEC         ;HOLDING 1'S & 0'S TEST
7802 020050  021226                                MT0007  ;<1 SEC         ;ADDRESS BIT TEST
7803 020052  020202                                MT0001  ;<1 SEC         ;ADDRESS TEST
7804 020054  020322                                MT0002  ;<1 SEC         ;COMPLEMENT ADDRESS TEST
7805 020056  020462                                MT0003  ; 1 SEC         ;3 XOR 9 WORST CASE NOISE TEST
7806 020060  020714                                MT0004  ; 1 SEC         ;ROTATING ZEROS TEST
7807 020062  021036                                MT0005  ; 1 SEC         ;ROTATING ONES TEST
7808 020064  023046                                MT0021  ; 1 SEC         ;MARCHING 0'S & 1'S TEST
7809 020066  026302                                MT0035  ;<1 SEC         ;WORSE CASE NOISE PARITY TEST
7810 020070  023320                                MT0022  ;10 SEC         ;REFRESH TEST
7811 020072  023352                                MT0023  ;10 SEC         ;SHIFTING DIAGONAL  TEST
7812 020074  023730                                MT0026  ;<1 SEC         ;RANDOM DATA TEST
7813 020076  023416                                MT0024  ;20 SEC         ;FAST GALLOPING PATTERN TEST
7814 020100  025220                                MT0031  ; 3 SEC         ;SOB-A-LONG TEST
7815 020102  025410                                MT0032  ;<1 SEC         ;WRITE RECOVERY TEST
7816 020104  025742                                MT0033  ;35 SEC         ;BRANCH GOBBLE TEST
7817 020106  026130                                MT0034  ;<1 SEC         ;SOFT ERROR - BACKGROUND PATTERN TEST
7818                                               ;NOTE MT0034 MUST BE FIRST & LAST
7819 020110  026414                                MT0999  ; 0 SEC         ;NULL TEST
7820 020112  026414                                MT0999  ; 0 SEC         ;NULL TEST
7821 020114  026414                                MT0999  ; 0 SEC         ;NULL TEST
7822 020116  026414                                MT0999  ; 0 SEC         ;NULL TEST
7823 020120  026414                                MT0999  ; 0 SEC         ;NULL TEST
```

```
                                        .SBTTL  PATTERNS
                                        .SBTTL  MEMORY TEST SETUP ROUTINES
                                MT0000: SUBTST  <<MT0000        SETUP DATA PATTERN TEST>>
                                ;*****************************************************************************
                                ;*SUBTEST        MT0000  SETUP DATA PATTERN TEST
                                ;*****************************************************************************
                                        CLR     REALPAT                 ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
                                        MOV     #FIRST,R0
                                        MOV     #SIZE,R1
                                        CALL    REGCOPY
                                        CMP     #1,PROTYP               ;ARE WE ON AN 11/44?
                                        BEQ     1$                      ;BRANCH IF YES
                                        MOV     #MTP000,SUPDOADD        ;ELSE DO PATTERN IN MAIN MEMORY
                                        CALL    SUPD03
                                        RETURN
                                1$:     BMOV    MTP000
                                        CALL    SUPD01                  ;DO IT IN SUPERVISOR MODE
                                        RETURN
                                MT0001: SUBTST  <<MT0001        SETUP ADDRESS TEST>>
                                ;*****************************************************************************
                                ;*SUBTEST        MT0001  SETUP ADDRESS TEST
                                ;*****************************************************************************
                                        MOV     #1,REALPAT              ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
                                        MOV     #FIRST,R0
                                        MOV     #SIZE,R1
                                        TST     NOSUPER
                                        BNE     2$
                                        CMP     SIPAR5,SIPAR6
                                        BNE     4$
                                        BR      3$
                                2$:     CMP     UIPAR5,UIPAR6
                                        BNE     4$
                                3$:     MOV     #30000,R1
                                4$:     CLR     R2
                                        CALL    REGCOPY
                                        CMP     #1,PROTYP               ;IS THIS AN 11/44?
                                        BEQ     1$                      ;BRANCH IF IT IS
                                        MOV     #MTP001,SUPDOADD        ;SET UP CALLING ADDRESS
                                        CALL    SUPD03
                                        RETURN
                                1$:     BMOV    MTP001
                                        CALL    SUPD01                  ;DO IT IN SUPERVISOR MODE
                                        RETURN
                                MT0002: SUBTST  <<MT0002        SETUP COMPLEMENT ADDRESS TEST>>
                                ;*****************************************************************************
                                ;*SUBTEST        MT0002  SETUP COMPLEMENT ADDRESS TEST
                                ;*****************************************************************************
                                        MOV     #2,REALPAT              ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
                                        MOV     #LAST+2,R0
                                        MOV     #SIZE,R1
                                        MOV     #FIRST,R4
                                        MOV     #100001,R5
                                        TST     NOSUPER
                                        BNE     2$
                                        CMP     SIPAR5,SIPAR6
                                        BNE     4$
```

```
 7873 020366  000404                        BR      3$
 7874 020370  023737  177652  177654  2$:   CMP     UIPAR5,UIPAR6
 7875 020376  001006                        BNE     4$
 7876 020400  012701  030000          3$:   MOV     #30000,R1
 7877 020404  012700  140000                MOV     #140000,R0
 7878 020410  012705  120001                MOV     #120001,R5
 7879 020414  012702  000001          4$:   MOV     #1,R2
 7880 020420  010103                        MOV     R1,R3
 7881 020422  022737  000001  003716        CMP     #1,PROTYP               ;IS THIS AN 11/44?
 7882 020430  001406                        BEQ     1$                      ;BRANCH IF TRUE
 7883 020432  012737  027112  002254        MOV     #MTP002,SUPDOADD        ;SET UP CALLING ADDRESS
 7884 020440  004737  026642                CALL    SUPD03
 7885 020444  000207                        RETURN
 7886 020446                          1$:   BMOV    MTP002
 7887 020454  004737  026464                CALL    SUPD01
 7888 020460  000207                        RETURN
```

```
 7891 020462                                    MT0003: SUBTST  <<MT0003        SETUP 3 XOR 9 WORST CASE NOISE TEST>>
                                                ;*****************************************************************************
                                                ;*SUBTEST       MT0003  SETUP 3 XOR 9 WORST CASE NOISE TEST
                                                ;*****************************************************************************
 7892 020462                                            IF EUFLAG IS TRUE THEN $RETURN
 7893 020472  012737  000003  002260                    MOV     #3,REALPAT              ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
 7894 020500  005037  002276                    CLR     PCBUMP                  ;TRAPS DO NOT ADD TO PC
 7895 020504  004737  036400            1$:     CALL    FLIPWARN                ;SETUP WARNING CONSTANTS & R2
 7896 020510  012701  060000            2$:     MOV     #FIRST,R1               ;R1 <-- STARTING ADDRESS
 7897 020514  012703  020000                    MOV     #20000,R3
 7898 020520  072327  177770                    ASH     #-8.,R3                 ;R3 <-- R3 / 256.
 7899 020524  012702  000004                    MOV     #4,R2                   ;SMALL LOOP SIZE
 7900 020530  012705  000100                    MOV     #64.,R5                 ;MEDIUM LOOP SIZE
 7901 020534  022737  000001  003716            CMP     #1,PROTYP               ;IS THIS AN 11/44?
 7902 020542  001415                            BEQ     3$                      ;BRANCH IF IT IS
 7903 020544  104415                            SAVREG
 7904 020546  012737  027144  002254            MOV     #MTPA03,SUPDOADD
 7905 020554  004737  026642                    CALL    SUPD03                  ;DO IT IN MAIN MEMORY
 7906 020560  104416                            RESREG
 7907 020562  012737  027204  002254            MOV     #MTPB03,SUPDOADD
 7908 020570  004737  026656                    CALL    SUPD04
 7909 020574  000442                            BR      4$
 7910 020576                            3$:     BMOV    MTPA03
 7911 020604  104415                            SAVREG
 7912 020606  004737  026464                    CALL    SUPD01
 7913 020612                                    BMOV    MTPB03
 7914 020620                                    BMOV    MTPC03,KDPAR0,8.
 7915 020632                                    BMOV    MTPD03,SDPAR0,8.
 7916 020644  012737  172360  177642            MOV     #KDPAR0,UIPAR1          ;SET UP PAR LINKS
 7917 020652  012737  172260  172374            MOV     #SDPAR0,KDPAR6
 7918 020660  012737  177644  172276            MOV     #UIPAR2,SDPAR7
 7919 020666  012737  001032  172272            MOV     #1032,SDPAR5            ;CHANGE INST TO BR .+66 (BR TO KDPAR1)
 7920 020674  104416                            RESREG
 7921 020676  004737  026500                    CALL    SUPD02
 7922 020702  022737  000003  002556    4$:     CMP     #3,FLIPLOC              ;DONE WITH 4 PATTERNS
 7923                                                                           ;[(0,177777);(177777,0);(401,177777);(177777,401)]?
 7924 020710  001275                            BNE     1$                      ;NO - LOOP
 7925 020712  000207                            RETURN
 7926
 7927 020714                                    MT0004: SUBTST  <<MT0004        SETUP ROTATING ZEROS TEST>>
                                                ;*****************************************************************************
                                                ;*SUBTEST       MT0004  SETUP ROTATING ZEROS TEST
                                                ;*****************************************************************************
 7928 020714  012737  000004  002260            MOV     #4,REALPAT              ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
 7929 020722  012737  000004  002276            MOV     #4,PCBUMP               ;TRAPS ADD 4 TO PC
 7930 020730  013702  002554                    MOV     ONES,R2
 7931 020734  004737  036530                    CALL    BACKGND                 ;WRITE BACKGROUND OF ONES
 7932 020740  012700  060000                    MOV     #FIRST,R0
 7933 020744  012701  040000                    MOV     #SIZE,R1
 7934 020750  022737  000001  003716            CMP     #1,PROTYP               ;IS THIS AN 11/44?
 7935 020756  001406                            BEQ     1$                      ;BRANCH IF IT IS
 7936 020760  012737  027302  002254            MOV     #MTPA04,SUPDOADD        ;SET UP LINKS
 7937 020766  004737  026656                    CALL    SUPD04
 7938 020772  000207                            RETURN
 7939 020774                            1$:     BMOV    MTPA04
 7940 021002                                    BMOV    MTPB04,KDPAR0,8.
 7941 021014  012737  172360  177652            MOV     #KDPAR0,UIPAR5
```

```
 7942 021022  012737  177654  172376          MOV     #UIPAR6,KDPAR7
 7943 021030  004737  026500                  CALL    SUPD02
 7944 021034  000207                          RETURN
 7945 021036                          MT0005: SUBTST  <<MT0005        SETUP ROTATING ONES TEST>>
                                      ;*****************************************************************************
                                      ;*SUBTEST        MT0005  SETUP ROTATING ONES TEST
                                      ;*****************************************************************************
                                      ;
 7946 021036  012737  000005  002260          MOV     #5,REALPAT              ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
 7947 021044  012737  000004  002276          MOV     #4,PCBUMP               ;TRAPS ADD 4 TO PC
 7948 021052  005002                          CLR     R2
 7949 021054  004737  036530                  CALL    BACKGND                 ;WRITE BACKGROUND OF ZEROS
 7950 021060  012700  060000                  MOV     #FIRST,R0
 7951 021064  012701  040000                  MOV     #SIZE,R1
 7952 021070  022737  000001  003716          CMP     #1,PROTYP               ;IS THIS AN 11/44?
 7953 021076  001414                          BEQ     1$                      ;BRANCH IF IT IS
 7954 021100  012737  027356  002254          MOV     #MTP005,SUPD0ADD        ;SET UP LINKS
 7955 021106  012737  027372  027354          MOV     #MTP005+14,MTPB04+16
 7956 021114  004737  026656                  CALL    SUPD04
 7957 021120  012737  027316  027354          MOV     #MTPA04+14,MTPB04+16    ;RESET TEST'S ORIGINAL VALUE
 7958 021126  000207                          RETURN
 7959 021130                          1$:     BMOV    MTP005
 7960 021136                                  BMOV    MTPB04,KDPAR0,8.
 7961 021150  012737  172360  177652          MOV     #KDPAR0,UIPAR5
 7962 021156  012737  177654  172376          MOV     #UIPAR6,KDPAR7
 7963 021164  004737  026500                  CALL    SUPD02
 7964 021170  000207                          RETURN
```

```
 7967 021172                                MT0006: SUBTST  <<MT0006        SETUP INITIAL DATA TEST>>
                                            ;*****************************************************************************
                                            ;*SUBTEST       MT0006  SETUP INITIAL DATA TEST
                                            ;*****************************************************************************
 7968 021172  012737  000006  002260               MOV     #6,REALPAT              ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
 7969 021200  012737  000004  002276               MOV     #4,PCBUMP               ;TRAPS ADD 4 TO PC
 7970 021206  012701  002362                       MOV     #TESTADD,R1
 7971 021212  012737  027412  002254               MOV     #MTP006,SUPDOADD
 7972 021220  004737  026642                       CALL    SUPD03                  ;DO IT IN SUPERVISOR MODE
 7973 021224  000207                               RETURN
 7974 021226                                MT0007: SUBTST  <<MT0007        SETUP ADDRESS BIT TEST>>
                                            ;*****************************************************************************
                                            ;*SUBTEST       MT0007  SETUP ADDRESS BIT TEST
                                            ;*****************************************************************************
 7975 021226  012737  000007  002260               MOV     #7,REALPAT              ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
 7976 021234  005002                               CLR     R2
 7977 021236  004737  036530                       CALL    BACKGND                 ;OF ZEROS
 7978 021242  012701  060000                       MOV     #FIRST,R1
 7979 021246  012702  000001                       MOV     #1,R2
 7980 021252  050201                               BIS     R2,R1
 7981 021254  012737  027612  002254               MOV     #MTP007,SUPDOADD
 7982 021262  004737  026642                       CALL    SUPD03                  ;DO IT IN SUPERVISOR MODE
 7983 021266  000207                               RETURN
 7984 021270                                MT0010: SUBTST  <<MT0010        SETUP BYTE ADDRESSING TEST>>
                                            ;*****************************************************************************
                                            ;*SUBTEST       MT0010  SETUP BYTE ADDRESSING TEST
                                            ;*****************************************************************************
 7985 021270  012737  000010  002260               MOV     #10,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
 7986 021276  012737  000004  002276               MOV     #4,PCBUMP               ;TRAPS ADD 4 TO PC
 7987 021304  013704  002362                       MOV     TESTADD,R4
 7988 021310  012737  027712  002254               MOV     #MTP010,SUPDOADD
 7989 021316  004737  026642                       CALL    SUPD03                  ;DO IT IN SUPERVISOR MODE
 7990 021322  000207                               RETURN
```

```
 7993 021324                            MT0011: SUBTST  <<MT0011        SETUP CREATE SINGLE BIT ERROR TEST>>
                                        ;*****************************************************************************
                                        ;*SUBTEST        MT0011  SETUP CREATE SINGLE BIT ERROR TEST
                                        ;*****************************************************************************
 7994 021324                                    IF ACTFLAG IS TRUE OR APTFLAG IS TRUE
 7995 021340                                      IF $PASS NE #0 THEN $RETURN
 7996 021350                                    END; OF IF ACTFLAG
 7997 021350  012737  000011  002260            MOV     #11,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
 7998 021356  012737  030020  002254            MOV     #MTP011,SUPDOADD
 7999 021364  004737  026642                    CALL    SUPD03                  ;DO IT IN SUPERVISOR MODE
 8000 021370  000207                            RETURN
 8001 021372                            MT0012: SUBTST  <<MT0012        SETUP WRITE BYTE CLEARS SBE TEST>>
                                        ;*****************************************************************************
                                        ;*SUBTEST        MT0012  SETUP WRITE BYTE CLEARS SBE TEST
                                        ;*****************************************************************************
 8002 021372                                    IF ACTFLAG IS TRUE OR APTFLAG IS TRUE
 8003 021406                                      IF $PASS NE #0 THEN $RETURN
 8004 021416                                    END; OF IF ACTFLAG
 8005 021416  012737  000012  002260            MOV     #12,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
 8006 021424  013700  002102                    MOV     BANKINDEX,R0
 8007 021430                                    IF #BIT12 SET.IN CONFIG+2(R0)
 8008 021440  012705  040000                      MOV   #40000,R5
 8009 021444                                    ELSE
 8010 021446  012705  000002                      MOV   #2,R5
 8011 021452                                    END; OF IF #BIT12
 8012 021452  012737  030616  002254            MOV     #MTP012,SUPDOADD
 8013 021460  004737  026642                    CALL    SUPD03                  ;DO IT IN SUPERVISOR MODE
 8014 021464  000207                            RETURN
 8015 021466                            MT0013: SUBTST  <<MT0013        SETUP CREATE DOUBLE BIT ERROR TEST>>
                                        ;*****************************************************************************
                                        ;*SUBTEST        MT0013  SETUP CREATE DOUBLE BIT ERROR TEST
                                        ;*****************************************************************************
 8016 021466                                    IF ACTFLAG IS TRUE OR APTFLAG IS TRUE
 8017 021502                                      IF $PASS NE #0 THEN $RETURN
 8018 021512                                    END; OF IF ACTFLAG
 8019 021512  012737  000013  002260            MOV     #13,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
 8020 021520  012737  031204  002254            MOV     #MTP013,SUPDOADD
 8021 021526  012737  000003  002074            MOV     #3,NOPAR                ;INDICATE PARITY ACRION
 8022 021534  004737  026642                    CALL    SUPD03                  ;DO IT IN SUPERVISOR MODE
 8023 021540  000207                            RETURN
 8024 021542                            MT0014: SUBTST  <<MT0014        SETUP WRITE INHIBIT DURING DATIP WITH DBE>>
                                        ;*****************************************************************************
                                        ;*SUBTEST        MT0014  SETUP WRITE INHIBIT DURING DATIP WITH DBE
                                        ;*****************************************************************************
 8025 021542                                    IF ACTFLAG IS TRUE OR APTFLAG IS TRUE
 8026 021556                                      IF $PASS NE #0 THEN $RETURN
 8027 021566                                    END; OF IF ACTFLAG
 8028 021566                                    IF KFLAG IS FALSE THEN $RETURN
 8029 021576  012737  000014  002260            MOV     #14,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
 8030 021604  012737  031720  002254            MOV     #MTP014,SUPDOADD
 8031 021612  004737  026642                    CALL    SUPD03                  ;DO IT IN SUPERVISOR MODE
 8032 021616  000207                            RETURN
```

```
                                MT0015: SUBTST  <<MT0015        SETUP WRITE INHIBIT OF BYTE WITH DBE>>
                                ;*****************************************************************************
                                ;*SUBTEST        MT0015  SETUP WRITE INHIBIT OF BYTE WITH DBE
                                ;*****************************************************************************
                                        IF ACTFLAG IS TRUE OR APTFLAG IS TRUE
                                          IF $PASS NE #0 THEN $RETURN
                                        END ;OF IF ACTFLAG
8039 021644  012737  000015  002260    MOV     #15,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
8040 021652  012737  032502  002254    MOV     #MTP015,SUPDOADD
8041 021660  004737  026642            CALL    SUPD03                  ;DO IT IN SUPERVISOR MODE
8042 021664  000207                    RETURN
8043 021666                     MT0016: SUBTST  <<MT0016        SETUP WRITE INHIBIT OF WORD WITH DBE>>
                                ;*****************************************************************************
                                ;*SUBTEST        MT0016  SETUP WRITE INHIBIT OF WORD WITH DBE
                                ;*****************************************************************************
                                        IF ACTFLAG IS TRUE OR APTFLAG IS TRUE
                                          IF $PASS NE #0 THEN $RETURN
                                        END ;OF IF ACTFLAG
8047 021712  012737  000016  002260    MOV     #16,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
8048 021720  012737  033246  002254    MOV     #MTP016,SUPDOADD
8049 021726  004737  026642            CALL    SUPD03                  ;DO IT IN SUPERVISOR MODE
8050 021732  000207                    RETURN
8051 021734                     MT0017: SUBTST  <<MT0017        SETUP HOLDING 1'S & 0'S>>
                                ;*****************************************************************************
                                ;*SUBTEST        MT0017  SETUP HOLDING 1'S & 0'S
                                ;*****************************************************************************
8052 021734  012737  000017  002260    MOV     #17,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
8053 021742  012737  034030  002254    MOV     #MTP017,SUPDOADD
8054 021750  004737  026642            CALL    SUPD03                  ;DO IT IN SUPERVISOR MODE
8055 021754  000207                    RETURN
```

```
8058 021756                                MT0020: SUBTST  <<MT0020        SETUP MARCHING 0'S & 1'S IN CHECKBITS TEST>>
                                           ;*****************************************************************************
                                           ;*SUBTEST       MT0020  SETUP MARCHING 0'S & 1'S IN CHECKBITS TEST
                                           ;*****************************************************************************
8059 021756                                        IF ACTFLAG IS TRUE OR ACTFLAG IS TRUE
8060 021772                                          IF $PASS NE #0 THEN $RETURN
8061 022002                                        END ;OF IF ACTFLAG
8062 022002  012737  000020  002260                 MOV     #20,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
8063 022010  012737  000003  002074                 MOV     #3,NOPAR                ;INDICATE PARITY ACTION
8064 022016  005001                                 CLR     R1                      ;CLEAR LOOP COUNTER
8065 022020  005004                                 CLR     R4                      ;CLEAR INTERLEAVE ODD/EVEN FLAG
8066 022022  012700  060000                       MOV     #FIRST,R0
8067 022026  013702  002102                       MOV     BANKINDEX,R2            ;SET BANK INDEX
8068 022032                                MTL020: IF INTFLAG IS FALSE
8069 022040                                          BEGIN MTB020
8070 022040                                          IF NO22BIT IS TRUE
8071 022046                                            IF BANK EQ #7
8072 022056  012737  140000  002362                     MOV       #140000,TESTADD ;SET UP 12K NON-INTERLEAVED VIRT ADDR
8073 022064  012705  140002                           MOV       #140002,R5
8074 022070                                           LEAVE     MTB020
8075 022072                                          END; OF IF BANK
8076 022072                                        END; OF IF NO22BIT
8077 022072                                        IF NO22BIT IS FALSE
8078 022100                                          IF BANK EQ #177
8079 022110  012737  140000  002362                     MOV       #140000,TESTADD
8080 022116  012705  140000                           MOV       #140000,R5
8081 022122                                           LEAVE     MTB020
8082 022124                                          END; OF IF BANK
8083 022124                                        END; OF IF NO22BIT
8084 022124  012737  160000  002362                 MOV   #LAST+2,TESTADD
8085 022132  012705  160002                       MOV   #LAST+4,R5
8086 022136                                       END   MTB020
8087 022136  010537  002364                       MOV   R5,TESTADD+2              ;SET UP NON-INTERLEAVED VIRT. ADDR.
8088 022142                                      ELSE
8089 022144  005737  002312                       TST   SKIPMK                    ;IS THIS BANK IN SKIP RANGE?
8090 022150  001401                               BEQ   1$                        ;BANK IS OUT OF RANGE - DO TEST
8091 022152  000207                               RETURN                          ;LEAVE TEST-BANK'S ALREADY TESTED
8092 022154  012737  120000  002362  1$:          MOV   #120000,TESTADD           ;SET UP 1ST INTERLEAVED VIRT. ADDR.
8093 022162  012705  160000                       MOV   #LAST+2,R5                ;SET UP END OF BANK FLAG
8094 022166  010537  002364                       MOV   R5,TESTADD+2              ;SET UP 2ND INT'L. VIRT. ADDR.
8095 022172  005237  002232                       INC   SPLTCSR                   ;FLAG THE MAPPING ROUTINE FOR INTERLEAVING
8096 022176  005201                               INC   R1                        ;SET LOOP COUNTER FOR INTERLEAVING
8097 022200  005204                               INC   R4                        ;SET ODD/EVEN FLAG
8098 022202                                      END; OF IF INTFLAG
8099 022202  016203  002624                     MOV     CONFIG(R2),R3           ;SET UP CSR NUMBER
8100 022206                                      IF R4 EQ #2                     ;IF THE SECOND TIME AROUND
8101 022214  060437  002362                       ADD   R4,TESTADD
8102 022220  060437  002364                       ADD   R4,TESTADD+2              ;TEST THE A1 ASSERTED ADDRESSES
8103 022224  060400                               ADD   R4,R0
8104 022226  060405                               ADD   R4,R5
8105 022230  072327  177775                       ASH   #-3,R3                    ;MOVE INTERLEAVED CSR NUMBER
8106 022234                                      ELSE
8107 022236  006303                               ASL   R3                        ;MOVE CSR NUMBER
8108 022240                                      END; IF R4
8109 022240  000303                              SWAB    R3
8110 022242  042703  177741                      BIC     #^C36,R3
8111 022246  010337  002146                      MOV     R3,CSRNO                ;MOVE R3 INTO CSR NUMBER
```

```
 8112 022252                                    IF #SW0 SET.IN @SWR
 8113 022262  104506                              ENASBE                         ;TRAP ON SINGLE BIT ERRORS
 8114 022264                                    ELSE
 8115 022266  104472                              ECCINIT                        ;TRAP ON UNCORRECTABLE ERRORS
 8116 022270                                    END; OF IF #SW0
 8117 022270                                    PUSH    R2,R4
 8118 022274                                    FOR MTV020 := #0 TO R1
 8119 022300                                      PUSH  R1
 8120 022302  005002                              CLR   R2                       ;PATTERN TO WRITE INTO BANK
 8121 022304  004737  036530                      CALL  BACKGND                  ;SET UP ZEROS IN BANK
 8122 022310                                      IF NO22BIT IS TRUE AND MTV020 EQ #1 AND BANK EQ #3
 8123 022336  162737  020000  002362                SUB #20000,TESTADD           ;SET UP 12K INTERLEAVED BANK
 8124 022344  162705  020000                       SUB #20000,R5
 8125 022350  010537  002364                       MOV R5,TESTADD+2
 8126 022354                                      END; OF IF NO22BIT
 8127 022354  004737  022430                      CALL  MT020Z                   ;START TEST
 8128 022360  005237  002232                      INC   SPLTCSR                  ;UPDATE INTERLEAVED MAPPING FLAG
 8129 022364                                      POP   R1
 8130 022366                                    END; OF FOR MTV020
 8131 022400                                    POP     R4,R2
 8132 022404  005001                            CLR     R1                       ;RESET LOOP FLAG
 8133 022406  005037  002232                    CLR     SPLTCSR                  ;RESET INTERLEAVED MAP FLAG
 8134 022412  022704  000001                    CMP     #1,R4                    ;ODD/EVEN FLAG SET?
 8135 022416  001605                            BEQ     MTL020                   ;BRANCH IF TRUE
 8136 022420  005037  002074                    CLR     NOPAR                    ;INDICATE PARITY ACTION
 8137 022424  000207                            RETURN
 8138 022426  000000                MTV020:     0                                ;VARIABLE FOR PAT 20
```

```
 8140 022430  012702  000004          MT020Z: MOV     #4,R2                   ;SET UP WORD INCR/DECR AMOUNT
 8141 022434  013701  002362                  MOV     TESTADD,R1
 8142 022440  013704  002364                  MOV     TESTADD+2,R4
 8143 022444  012703  100000                  MOV     #BIT15,R3
 8144 022450                                  IF #SW11 SET.IN @SWR OR QVFLAG IS TRUE
 8145 022466                                    GOTO  MT020Y
 8146 022470                                  END ;OF IF #SW11
 8147 022470  022737  000001  003716          CMP     #1,PROTYP               ;IS THIS AN 11/44?
 8148 022476  001411                          BEQ     1$                      ;BRANCH IF IT IS
 8149 022500  012737  034106  002254          MOV     #MTPA20,SUPDOADD
 8150 022506  012737  034122  002264          MOV     #MTPA20+14,PARTHERE     ;VECTOR FOR TRAPS
 8151 022514  004737  026642                  CALL    SUPD03
 8152 022520  000410                          BR      2$
 8153 022522                          1$:     BMOV    MTPA20
 8154 022530  012737  177654  002264          MOV     #UIPAR6,PARTHERE        ;VECTOR FOR TRAPS
 8155 022536  004737  026464                  CALL    SUPD01
 8156 022542  022737  000001  003716  2$:     CMP     #1,PROTYP               ;IS THIS AN 11/44?
 8157 022550  001411                          BEQ     4$                      ;BRANCH IF IT IS
 8158 022552  012737  034136  002254          MOV     #MTPB20,SUPDOADD
 8159 022560  012737  034146  002264          MOV     #MTPB20+10,PARTHERE     ;VECTOR FOR TRAPS
 8160 022566  004737  026656                  CALL    SUPD04
 8161 022572  000410                          BR      MT020Y
 8162 022574                          4$:     BMOV    MTPB20
 8163 022602  012737  177650  002264          MOV     #UIPAR4,PARTHERE        ;VECTOR FOR TRAPS
 8164 022610  004737  026500                  CALL    SUPD02
 8165 022614  005737  002134          MT020Y: TST     INTFLAG                 ;ARE WE INTERLEAVED?
 8166 022620  001405                          BEQ     7$                      ;BRANCH IF NOT INTERLEAVED
 8167 022622  162701  040000                  SUB     #40000,R1               ;RESET FIRST WORD TO BEGINNING OF BANK
 8168 022626  162704  040000                  SUB     #40000,R4               ;RESET SECOND WORD TO BEGINNING OF BANK
 8169 022632  000404                          BR      8$
 8170 022634  012701  060000          7$:     MOV     #FIRST,R1               ;RESET FIRST WORD TO BEGINNING OF BANK
 8171 022640  012704  060002                  MOV     #FIRST+2,R4             ;RESET SECOND WORD TO BEGINNING OF BANK
 8172 022644  022737  000001  003716  8$:     CMP     #1,PROTYP               ;IS THIS AN 11/44?
 8173 022652  001411                          BEQ     1$                      ;BRANCH IF IT IS
 8174 022654  012737  034166  002254          MOV     #MTPC20,SUPDOADD
 8175 022662  012737  034176  002264          MOV     #MTPC20+10,PARTHERE     ;VECTOR FOR TRAPS
 8176 022670  004737  026642                  CALL    SUPD03
 8177 022674  000410                          BR      2$
 8178 022676                          1$:     BMOV    MTPC20
 8179 022704  012737  177650  002264          MOV     #UIPAR4,PARTHERE        ;VECTOR FOR TRAPS
 8180 022712  004737  026464                  CALL    SUPD01
 8181 022716  022737  000001  003716  2$:     CMP     #1,PROTYP               ;IS THIS AN 11/44?
 8182 022724  001411                          BEQ     3$                      ;BRANCH IF IT IS
 8183 022726  012737  034216  002254          MOV     #MTPD20,SUPDOADD
 8184 022734  012737  034232  002264          MOV     #MTPD20+14,PARTHERE     ;VECTOR FOR TRAPS
 8185 022742  004737  026656                  CALL    SUPD04
 8186 022746  000410                          BR      4$
 8187 022750                          3$:     BMOV    MTPD20
 8188 022756  012737  177654  002264          MOV     #UIPAR6,PARTHERE        ;VECTOR FOR TRAPS
 8189 022764  004737  026500                  CALL    SUPD02
 8190 022770  022737  000001  003716  4$:     CMP     #1,PROTYP               ;IS THIS AN 11/44?
 8191 022776  001411                          BEQ     5$                      ;BRANCH IF IT IS
 8192 023000  012737  034246  002254          MOV     #MTPE20,SUPDOADD
 8193 023006  012737  034256  002264          MOV     #MTPE20+10,PARTHERE     ;VECTOR FOR TRAPS
 8194 023014  004737  026656                  CALL    SUPD04
 8195 023020  000410                          BR      6$
 8196 023022                          5$:     BMOV    MTPE20
```

```
 8197 023030  012737  177650  002264          MOV     #UIPAR4,PARTHERE        ;VECTOR FOR TRAPS
 8198 023036  004737  026500                  CALL    SUPD02
 8199 023042  104503                  6$:     CLR1CSR                         ;CLEAR 1 SELECTED CSR
 8200 023044  000207                          RETURN
```

```
8203 023046                                MT0021: SUBTST  <<MT0021        SETUP MARCHING 0'S & 1'S TEST>>
                                           ;*****************************************************************************
                                           ;*SUBTEST       MT0021  SETUP MARCHING 0'S & 1'S TEST
                                           ;*****************************************************************************
8204 023046                                        SET NOSCOPE
8205 023054  012737  000021  002260                MOV     #21,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
8206 023062  013702  002572                        MOV     BAKPAT,R2
8207 023066  004737  036530                        CALL    BACKGND
8208 023072  010203                                MOV     R2,R3
8209 023074  000303                                SWAB    R3
8210 023076  012701  160000                        MOV     #LAST+2,R1
8211 023102  010105                                MOV     R1,R5
8212 023104  012704  060000                        MOV     #FIRST,R4
8213 023110  022737  000001  003716                CMP     #1,PROTYP       ;IS THIS AN 11/44?
8214 023116  001441                                BEQ     1$              ;BRANCH IF IT IS
8215 023120  022737  000003  003716                CMP     #3,PROTYP       ;IS THIS AN 11/24?
8216 023126  001407                                BEQ     3$              ;BRANCH IF SO
8217 023130  022737  000007  002100                CMP     #7,BANK
8218 023136  001003                                BNE     3$
8219 023140  012701  140000                        MOV     #140000,R1
8220 023144  010105                                MOV     R1,R5
8221 023146  012737  034272  002254  3$:           MOV     #MTPA21,SUPDOADD
8222 023154  004737  026642                        CALL    SUPD03
8223 023160  012737  034322  002254                MOV     #MTPB21,SUPDOADD
8224 023166  004737  026656                        CALL    SUPD04
8225 023172  010401                                MOV     R4,R1
8226 023174  012737  034356  002254                MOV     #MTPC21,SUPDOADD
8227 023202  004737  026656                        CALL    SUPD04
8228 023206  012737  034412  002254                MOV     #MTPD21,SUPDOADD
8229 023214  004737  026656                        CALL    SUPD04
8230 023220  000434                                BR      2$
8231 023222  022737  000177  002100  1$:           CMP     #177,BANK
8232 023230  001003                                BNE     4$
8233 023232  012701  140000                        MOV     #140000,R1
8234 023236  010105                                MOV     R1,R5
8235 023240                             4$:        BMOV    MTPA21
8236 023246  004737  026464                        CALL    SUPD01
8237
8238 023252                                        BMOV    MTPB21
8239 023260  004737  026500                        CALL    SUPD02
8240
8241 023264  010401                                MOV     R4,R1
8242 023266                                        BMOV    MTPC21
8243 023274  004737  026500                        CALL    SUPD02
8244
8245 023300                                        BMOV    MTPD21
8246 023306  004737  026500                        CALL    SUPD02
8247 023312  005037  002410             2$:        CLR     NOSCOPE
8248 023316  000207                                RETURN
```

```
 8250 023320                            MT0022: SUBTST  <<MT0022        SETUP REFRESH & SHIFTING DIAGONAL TEST>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       MT0022  SETUP REFRESH & SHIFTING DIAGONAL TEST
                                        ;*****************************************************************************
 8251 023320  004737  026430                    CALL    KAMITEST                ;CHECK FOR KAMIKAZE MODE
 8252 023324                                    ON.ERROR THEN $RETURN           ;IF NOT IN KAMIKAZE MODE RETURN
 8253 023330  012737  000022  002260            MOV     #22,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
 8254 023336  012737  034442  002254            MOV     #MTP022,SUPDOADD
 8255 023344  004737  026642                    CALL    SUPDO3                  ;DO IT IN SUPERVISOR MODE
 8256 023350  000207                            RETURN
 8257
 8258 023352                            MT0023: SUBTST  <<MT0023        SHIFTING DIAGONAL TEST>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       MT0023  SHIFTING DIAGONAL TEST
                                        ;*****************************************************************************
 8259 023352  004737  026430                    CALL    KAMITEST                ;CHECK FOR KAMIKAZE MODE
 8260 023356                                    ON.ERROR THEN $RETURN           ;IF NOT IN KAMIKAZE MODE RETURN
 8261 023362  012737  000023  002260            MOV     #23,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
 8262 023370  012737  034442  002254            MOV     #MTP022,SUPDOADD
 8263 023376                                    SET     DIAGFLAG                ;IDENTIFY DIAGONAL TEST TO MTP022
 8264 023404  004737  026642                    CALL    SUPDO3                  ;DO IT IN SUPERVISOR MODE
 8265 023410  005037  002002                    CLR     DIAGFLAG
 8266 023414  000207                            RETURN
```

```
 8268 023416                                MT0024: SUBTST  <<MT0024        SETUP FAST GALLOPING PATTERN TEST>>
                                            ;*****************************************************************************
                                            ;*SUBTEST       MT0024  SETUP FAST GALLOPING PATTERN TEST
                                            ;*****************************************************************************
 8269 023416  004737  026430                        CALL    KAMITEST                ;CHECK FOR KAMIKAZE MODE
 8270 023422                                        ON.ERROR THEN $RETURN           ;IF NOT IN KAMIKAZE MODE RETURN
 8271 023426                                        SET     NOSCOPE
 8272 023434  012737  000024  002260              MOV     #24,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
 8273 023442  013702  002572                        MOV     BAKPAT,R2
 8274 023446  004737  036530                        CALL    BACKGND
 8275 023452  010203                                MOV     R2,R3
 8276 023454  010304                                MOV     R3,R4
 8277 023456  000304                                SWAB    R4
 8278 023460  012701  060000                        MOV     #FIRST,R1
 8279 023464  012705  157776                        MOV     #LAST,R5
 8280 023470  022737  000001  003716                CMP     #1,PROTYP
 8281 023476  001417                                BEQ     1$
 8282 023500  022737  000003  003716                CMP     #3,PROTYP
 8283 023506  001406                                BEQ     3$
 8284 023510  022737  000007  002100                CMP     #7,BANK
 8285 023516  001002                                BNE     3$
 8286 023520  012705  137776                        MOV     #137776,R5
 8287 023524  104415                      3$:       SAVREG
 8288 023526  012737  035156  002254                MOV     #MTPB24,SUPDOADD
 8289 023534  000440                                BR      2$
 8290 023536  022737  000177  002526      1$:       CMP     #177,LASTBANK
 8291 023544  001002                                BNE     4$
 8292 023546  012705  137776                        MOV     #137776,R5
 8293 023552  104415                      4$:       SAVREG
 8294 023554                                        BMOV    MTPA24
 8295 023562                                        BMOV    MTPB24,SDPAR0,8.
 8296 023574                                        BMOV    MTPC24,KDPAR0,8.
 8297 023606  012737  172260  002254                MOV     #SDPAR0,SUPDOADD
 8298 023614  012737  172260  177676                MOV     #SDPAR0,UDPAR7          ;SET UP PAR LINKS
 8299 023622  012737  172360  172272                MOV     #KDPAR0,SDPAR5
 8300 023630  012737  177660  172374                MOV     #UDPAR0,KDPAR6
 8301 023636  004737  026656              2$:       CALL    SUPD04
 8302
 8303                                               ;DO IT AGAIN FOR COMPLEMENT DATA
 8304 023642  104416                                RESREG
 8305 023644  000302                                SWAB    R2
 8306 023646  000303                                SWAB    R3
 8307 023650  004737  026656                        CALL    SUPD04
 8308 023654  005037  002410                        CLR     NOSCOPE
 8309 023660  000207                                RETURN
 8310 023662                                MT0025: SUBTST  <<MT0025        SETUP INTERRUPT ENABLE TEST>>
                                            ;*****************************************************************************
                                            ;*SUBTEST       MT0025  SETUP INTERRUPT ENABLE TEST
                                            ;*****************************************************************************
 8311 023662                                        IF ACTFLAG IS TRUE OR APTFLAG IS TRUE
 8312 023676                                          IF $PASS NE #0 THEN $RETURN
 8313 023706                                        END ;OF IF ACTFLAG
 8314 023706  012737  000025  002260                MOV     #25,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
 8315 023714  012737  035210  002254                MOV     #MTP025,SUPDOADD
 8316 023722  004737  026642                        CALL    SUPD03                  ;DO IT IN SUPERVISOR MODE
 8317 023726  000207                                RETURN
```

```
 8320 023730                                MT0026: SUBTST  <<MT0026        SETUP RANDOM DATA TEST>>
                                            ;*****************************************************************************
                                            ;*SUBTEST       MT0026  SETUP RANDOM DATA TEST
                                            ;*****************************************************************************
 8321 023730  012737  000026  002260              MOV     #26,REALPAT
 8322 023736  005037  002276                      CLR     PCBUMP                  ;TRAPS DO NOT ADD TO THE PC
 8323 023742  013703  002544                      MOV     SEEDLO,R3               ;INITIALIZE RANDOM NUMBERS
 8324 023746  013702  002542                      MOV     SEEDHI,R2
 8325 023752  010305                              MOV     R3,R5
 8326 023754  010204                              MOV     R2,R4
 8327 023756  012701  060000                      MOV     #FIRST,R1
 8328 023762  012700  020000                      MOV     #SIZE/2,R0
 8329 023766  022737  000001  003716              CMP     #1,PROTYP               ;DO WE HAVE AN 11/44?
 8330 023774  001437                            BEQ     1$                      ;BRANCH IF WE DO
 8331 023776  022737  000003  003716              CMP     #3,PROTYP               ;11/24?
 8332 024004  001406                            BEQ     3$                      ;BRANCH IF SO
 8333 024006  022737  000007  002100              CMP     #7,BANK
 8334 024014  001002                            BNE     3$
 8335 024016  012700  014000                      MOV     #14000,R0
 8336 024022  104415                    3$:     SAVREG
 8337 024024  012737  035662  035762              MOV     #MTPA26+4,MTPD26+14
 8338 024032  012737  035656  002254              MOV     #MTPA26,SUPDOADD
 8339 024040  004737  026642                      CALL    SUPD03
 8340 024044  005037  035706                      CLR     RANODD                  ;FOR ERROR REPORTING
 8341 024050  012737  035676  035762              MOV     #MTPB26+4,MTPD26+14     ;SET UP NEXT LINK
 8342 024056  012737  035672  002254              MOV     #MTPB26,SUPDOADD
 8343 024064  104416                            RESREG
 8344 024066  004737  026642                      CALL    SUPD03
 8345 024072  000452                            BR      2$
 8346 024074  022737  000177  002100    1$:     CMP     #177,BANK
 8347 024102  001002                            BNE     4$
 8348 024104  012700  014000                      MOV     #14000,R0
 8349 024110  104415                    4$:     SAVREG
 8350 024112                                      BMOV    MTPA26                  ;WRITE ROUTINE TO FAST MEMORY
 8351 024120                                      BMOV    MTPC26,KDPAR0,8.        ;RANDOM SUBPROGRAM TO FAST MEMORY
 8352 024132  012737  000730  172376              MOV     #730,KDPAR7             ;WRITES 'BR .-116' IN (BR SDPAR0)
 8353 024140                                      BMOV    MTPD26,SDPAR0,8.        ;RANDOM SUBSUBPROGRAM TO FAST MEMORY
 8354 024152  012737  172360  177642              MOV     #KDPAR0,UIPAR1
 8355 024160  012737  177644  172274              MOV     #UIPAR2,SDPAR6
 8356 024166  004737  026464                      CALL    SUPD01                  ;WRITE RANDOM DATA
 8357 024172  005037  035706                      CLR     RANODD                  ;FOR ERROR REPORTING
 8358 024176                                      BMOV    MTPB26                  ;READ ROUTINE TO FAST MEMORY
 8359 024204  012737  172360  177642              MOV     #KDPAR0,UIPAR1          ;SET UP PAR LINK
 8360 024212  104416                            RESREG
 8361 024214  004737  026464                      CALL    SUPD01                  ;READ RANDOM DATA
 8362 024220  010337  002544            2$:     MOV     R3,SEEDLO               ;UPDATE FOR NEW RANDOM NUMBERS
 8363 024224  010237  002542                      MOV     R2,SEEDHI
 8364 024230  000207                            RETURN
```

```
 8367 024232                                MT0027: SUBTST  <<MT0027        UNIQUE BANK TEST>>
                                            ;*****************************************************************************
                                            ;*SUBTEST       MT0027  UNIQUE BANK TEST
                                            ;*****************************************************************************
 8368                                               ;MAKE SURE THAT EACH BANK CAN HAVE UNIQUE DATA
 8369                                               ;WRITE AND READ THE BANK NUMBER IN EACH BANK (EXCEPT WHERE THE PROGRAM IS)
 8370 024232  012737  000027  002260                 MOV     #27,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
 8371 024240  104502                                CLRCSR                  ;CLEAR CSRS
 8372 024242  022737  000001  003716                 CMP     #1,PROTYP               ;IS THIS AN 11/44?
 8373 024250  001404                                BEQ     1$                      ;BRANCH IF TRUE
 8374 024252  012737  026642  002472                 MOV     #SUPD03,LINK1           ;SET UP LINK
 8375 024260  000414                                BR      STAR27                  ;BRANCH TO RUN
 8376 024262                                1$:     BMOV    MTP034
 8377 024270  012737  177646  002254     WARN7:      MOV     #UIPAR3,SUPDOADD
 8378 024276  012737  026464  002472                 MOV     #SUPD01,LINK1           ;SET UP LINK
 8379 024304                                        SET NOFSMODE
 8380 024312                                STAR27: FOR I := #1 TO #2
 8381 024320                                          FOR BANK := #0 TO LASTBANK
 8382 024324  004737  044300                            CALL EXBANK
 8383 024330                                            IF ACFLAG IS TRUE AND RRFLAG IS FALSE
 8384 024344  104511                                      INVALIDATE                ;INVALIDATE BACKGROUND PATTERN ON 'BANK''
 8385 024346                                              LET R2 := BANK
 8386 024352  012700  060000                              MOV     #FIRST,R0
 8387 024356  010004                                      MOV     R0,R4
 8388 024360  012701  040000                              MOV     #SIZE,R1
 8389 024364  010103                                      MOV     R1,R3
 8390 024366                                              IF ' EQ #1
 8391 024376  C22737  000001  003716                        CMP     #1,PROTYP
 8392 024404  001403                                      BEQ     2$
 8393 024406  012737  036202  002254                      MOV     #MTP034,SUPDOADD
 8394 024414  004777  156052                          2$: CALL    @LINK1
 8395 024420                                              END ;OF IF
 8396 024420                                              IF I EQ #2
 8397 024430  022737  000001  003716                        CMP     #1,PROTYP
 8398 024436  001403                                      BEQ     3$
 8399 024440  012737  036210  002254                      MOV     #MTP034+6,SUPDOADD
 8400 024446  004737  026642                          3$: CALL    SUPD03
 8401 024452                                              END ;OF IF
 8402 024452                                            END ;OF IF
 8403 024452                                          END ;OF FOR BANK
 8404 024466                                        END ;OF FOR I
 8405 024502                                        IF FS7FLAG IS TRUE
 8406 024510  005037  002376                          CLR   NOFSMODE
 8407 024514  000207                                  RETURN
 8408 024516                                        END ;OF IF FS7FLAG
 8409 024516                                        FOR I := #1 TO #2
 8410 024524                                          FOR BANK := LASTBANK DOWNTO #0
 8411 024532  004737  044300                            CALL EXBANK
 8412 024536                                            IF ACFLAG IS TRUE AND RRFLAG IS FALSE
 8413 024552                                              LET R2 := BANK
 8414 024556  005102                                      COM     R2
 8415 024560  012700  060000                              MOV     #FIRST,R0
 8416 024564  010004                                      MOV     R0,R4
 8417 024566  012701  040000                              MOV     #SIZE,R1
 8418 024572  010103                                      MOV     R1,R3
 8419 024574                                              IF I EQ #1
 8420 024604  022737  000001  003716                        CMP     #1,PROTYP
```

```
 8421 024612  001403                              BEQ     4$
 8422 024614  012737  036202  002254              MOV     #MTP034,SUPDOADD
 8423 024622  004777  155644                    4$: CALL    @LINK1
 8424 024626                                      END ;OF IF
 8425 024626                                      IF I EQ #2
 8426 024636  022737  000001  003716                CMP     #1,PROTYP
 8427 024644  001403                              BEQ     5$
 8428 024646  012737  036210  002254              MOV     #MTP034+6,SUPDOADD
 8429 024654  004737  026642                    5$: CALL    SUPD03
 8430 024660                                      END ;OF IF
 8431 024660                                    END ;OF IF
 8432 024660                                  END ;OF FOR BANK
 8433 024674                                END ;OF FOR I
 8434 024710  005037  002376                  CLR     NOFSMODE
 8435 024714  000207                          RETURN
```

```
8438 024716                                MT0030: SUBTST  <<MT0030        SETUP FLUSH OUT DBE'S TEST>>
                                           ;*****************************************************************************
                                           ;*SUBTEST       MT0030  SETUP FLUSH OUT DBE'S TEST
                                           ;*****************************************************************************
8439 024716  005037  002256                        CLR     PASFLG
8440 024722                                        SET     FULLREL
8441 024730  012737  000030  002260  MTA030: MOV     #30,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
8442 024736  012737  000001  002074        MOV     #1,NOPAR                ;INDICATE COUNT PARITY ERRORS
8443 024744  022737  000001  003716        CMP     #1,PROTYP
8444 024752  001007                        BNE     4$
8445 024754                                BMOV    MTP030
8446 024762  012737  026464  002472        MOV     #SUPD01,LINK1
8447 024770  000406                        BR      1$
8448 024772  012737  026642  002472  4$:   MOV     #SUPD03,LINK1
8449 025000  012737  035764  002254        MOV     #MTP030,SUPDOADD
8450 025006  104470                  1$:   ECCDIS                          ;DISABLE ERROR CORRECTION
8451 025010                                SET     NOFSMODE,NOSCOPE
8452 025024                                FOR BANK := #0 TO LASTBANK
8453 025030  004737  044300                  CALL  EXBANK
8454 025034                                  IF MKFLAG IS TRUE
8455 025042                                    IF ACFLAG IS TRUE AND RRFLAG IS FALSE
8456 025056  012701  040000                      MOV       #SIZE,R1
8457 025062  012700  060000                      MOV       #FIRST,R0
8458 025066  004777  155400                      CALL      @LINK1
8459 025072                                    END ;OF IF ACFLAG
8460 025072                                  END ;OF IF MKFLAG
8461 025072                                END ;OF FOR
8462 025106                                IF PASFLG IS FALSE
8463 025114                                  SET PASFLG
8464 025122  104502                          CLRCSR                          ;CLEAR CSRS
8465 025124  004737  042526                  CALL  RELOCATE
8466 025130                                  ON.ERROR
8467 025132  104472                            ECCINIT                 ;TRAP ON DOUBLE BIT ERRORS (NORMAL)
8468 025134                                    CLEAR       NOFSMODE,NOSCOPE,FULLREL
8469 025150  000207                            RETURN
8470 025152                                  END ;OF ON.ERROR
8471 025152  013737  002270  002100          MOV   NEWBANK,BANK
8472 025160  004737  044300                  CALL  EXBANK
8473 025164  004737  024730                  CALL  MTA030
8474 025170  104472                          ECCINIT                 ;TRAP ON DOUBLE BIT ERRORS (NORMAL)
8475 025172  004737  043416                  CALL  UNRELOCATE
8476 025176  000207                          RETURN
8477 025200                                END ;OF IF PASFLG
8478 025200  104472                      ECCINIT             ;TRAP ON DOUBLE BIT ERRORS (NORMAL)
8479 025202                              CLEAR   NOFSMODE,NOSCOPE,FULLREL
8480 025216  000207                      RETURN
```

```
 8483 025220                                MT0031: SUBTST  <<MT0031        SETUP SOB-A-LONG TEST>>
                                            ;*****************************************************************************
                                            ;*SUBTEST       MT0031  SETUP SOB-A-LONG TEST
                                            ;*****************************************************************************
 8484 025220  004737  026430                        CALL    KAMITEST                ;CHECK FOR KAMIKAZE MODE
 8485 025224                                        ON.ERROR THEN $RETURN           ;IF NOT IN KAMIKAZE MODE RETURN
 8486 025230                                        SET     NOSCOPE
 8487 025236  012737  000031  002260               MOV     #31,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
 8488 025244  005037  002074                        CLR     NOPAR                   ;SETUP PARITY ACTION
 8489 025250                                        MAP     BANK                    ;MAP FIRST SO BLOCK MOVE WORKS
 8490 025264                                        TESTAREA                        ;ENTER TEST MODE
 8491 025272                                        BMOV    MTP031,FIRST,SOBLENGTH/2
 8492 025304  104417                                KERNEL                          ;ENTER KERNEL MODE
 8493 025306  013702  002532                        MOV     SOBK,R2
 8494 025312  010200                                MOV     R2,R0
 8495 025314  012701  100776                        MOV     #100776,R1              ;COMPLEMENT OF INSTRUCTION "SOB R0,DOT"
 8496 025320  012705  060056                        MOV     #FIRST+SOBLENGTH,R5
 8497 025324  012737  060002  002254               MOV     #FIRST+2,SUPDOADD
 8498 025332  012737  160000  002472               MOV     #LAST+2,LINK1
 8499 025340  005737  002426                        TST     NOSUPER
 8500 025344  001005                               BNE     1$
 8501 025346  023737  172252  172254               CMP     SIPAR5,SIPAR6
 8502 025354  001405                               BEQ     2$
 8503 025356  000407                               BR      3$
 8504 025360  023737  177652  177654  1$:          CMP     UIPAR5,UIPAR6
 8505 025366  001003                               BNE     3$
 8506 025370  012737  140000  002472  2$:          MOV     #140000,LINK1
 8507 025376  004737  026656          3$:          CALL    SUPD04
 8508 025402  005037  002410                        CLR     NOSCOPE
 8509 025406  000207                               RETURN
```

```
8512 025410                          MT0032: SUBTST  <<MT0032        SETUP WRITE RECOVERY TEST>>
                                     ;*****************************************************************************
                                     ;*SUBTEST       MT0032  SETUP WRITE RECOVERY TEST
                                     ;*****************************************************************************
8513 025410  004737  026430                     CALL    KAMITEST                ;CHECK FOR KAMIKAZE MODE
8514 025414                                     ON.ERROR THEN $RETURN           ;IF NOT IN KAMIKAZE MODE RETURN
8515 025420                                     SET     NOSCOPE
8516 025426  012737  000032  002260             MOV     #32,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
8517 025434  005037  002074                     CLR     NOPAR                   ;SETUP PARITY ACTION
8518 025440                                     MAP     BANK                    ;MAP FIRST SO THAT THE BLOCK MOVE WORKS
8519 025454  012700  010247                     MOV     #10247,R0               ;OP CODE OF INSTRUCTION 'MOV    R2,-(PC)''
8520 025460  012701  177667                     MOV     #177667,R1              ;OP CODE OF COMPLEMENT OF INSTRUCTION ''JMP (R0)''
8521 025464  012702  020000                     MOV     #SIZE/2,R2              ;USED FOR 1/2 BANK LOOP
8522 025470  010237  002472                     MOV     R2,LINK1
8523 025474  012703  060000                     MOV     #FIRST,R3
8524 025500  012704  160000                     MOV     #LAST+2,R4
8525 025504  005037  002474                     CLR     LINK2
8526 025510  005737  002426                     TST     NOSUPER
8527 025514  001005                             BNE     1$
8528 025516  023737  172252  172254             CMP     SIPAR5,SIPAR6
8529 025524  001405                             BEQ     2$
8530 025526  000415                             BR      3$
8531 025530  023737  177652  177654  1$:        CMP     UIPAR5,UIPAR6
8532 025536  001011                             BNE     3$
8533 025540  012704  140000          2$:        MOV     #140000,R4
8534 025544  012702  014000                     MOV     #14000,R2
8535 025550  010237  002472                     MOV     R2,LINK1
8536 025554  012737  000001  002474             MOV     #1,LINK2
8537
8538 025562                          3$:        TESTAREA                        ;ENTER TEST MODE
8539                                             ;MOVE TEST TO MEMORY UNDER TEST
8540 025570  010023                  4$:        MOV     R0,(R3)+
8541 025572  010144                             MOV     R1,-(R4)
8542 025574  077203                             SOB     R2,4$
8543
8544 025576  022737  000001  003716             CMP     #1,PROTYP
8545 025604  001003                             BNE     5$
8546                                             ;MOVE LAST PART OF TEST TO FASTCITY
8547 025606                                     BMOV    MTP032
8548 025614  104417                  5$:        KERNEL                          ;ENTER KERNEL MODE
8549
8550 025616  012702  005141                     MOV     #5141,R2                ;OP CODE OF INSTRUCTION ''COM    -(R1)''
8551 025622  012700  025740                     MOV     #10$,R0                 ;ADDRESS TO RETURN TO IN R0
8552 025626  012701  160000                     MOV     #LAST+2,R1              ;TOP OF BANK
8553 025632  012737  060000  002254             MOV     #FIRST,SUPDOADD
8554 025640  005737  002474                     TST     LINK2
8555 025644  001402                             BEQ     6$
8556 025646  012701  140000                     MOV     #140000,R1
8557 025652  004737  026656          6$:        CALL    SUPD04
8558 025656  012703  020000                     MOV     #SIZE/2,R3
8559 025662  012705  000110                     MOV     #110,R5
8560 025666  012704  060000                     MOV     #FIRST,R4
8561 025672  005737  002474                     TST     LINK2
8562 025676  001402                             BEQ     7$
8563 025700  012703  014000                     MOV     #14000,R3
8564 025704  022737  000001  003716  7$:        CMP     #1,PROTYP
8565 025712  001406                             BEQ     8$
```

```
 8566 025714  012737  036052  002254          MOV     #MTP032,SUPDOADD
 8567 025722  004737  026656                  CALL    SUPD04
 8568 025726  000402                          BR      9$
 8569 025730  004737  026500          8$:     CALL    SUPD02
 8570 025734  005037  002410          9$:     CLR     NOSCOPE
 8571 025740  000207                  10$:    RETURN                  ;THIS RETURN ACTS AS A NORMAL RETURN FROM MT0032
 8572                                                                 ;ALSO A RETURN FROM THE ''CALL   SUPD04'' ABOVE
 8573
```

```
8576 025742                          MT0033: SUBTST  <<MT0033        SETUP BRANCH GOBBLE TEST>>
                                     ;*****************************************************************************
                                     ;*SUBTEST        MT0033  SETUP BRANCH GOBBLE TEST
                                     ;*****************************************************************************
8577 025742  004737  026430                 CALL    KAMITEST                ;CHECK FOR KAMIKAZE MODE
8578 025746                                 ON.ERROR THEN $RETURN           ;IF NOT IN KAMIKAZE MODE RETURN
8579 025752                                 SET     NOSCOPE
8580 025760  012737  000033  002260         MOV     #33,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
8581 025766  005037  002074                 CLR     NOPAR                   ;SETUP PARITY ACTION
8582 025772                                 MAP     BANK                    ;MAP FIRST SO THAT BLOCK MOVE WORKS
8583
8584 026006                                 TESTAREA                        ;ENTER TEST MODE
8585 026014                                 BMOV    MTP033,FIRST,GBLENGTH/2
8586 026026  104417                         KERNEL                          ;ENTER KERNEL MODE
8587
8588 026030  012705  060076                 MOV     #FIRST+GBLENGTH,R5
8589 026034  012737  060004  002254         MOV     #FIRST+4,SUPDOADD
8590 026042  012701  060002                 MOV     #FIRST+2,R1
8591 026046  012702  060003                 MOV     #FIRST+3,R2
8592 026052  012737  160000  002472         MOV     #LAST+2,LINK1
8593 026060  005737  002426                 TST     NOSUPER
8594 026064  001005                         BNE     1$
8595 026066  023737  172252  172254         CMP     SIPAR5,SIPAR6
8596 026074  001405                         BEQ     2$
8597 026076  000407                         BR      3$
8598 026100  023737  177652  177654  1$:    CMP     UIPAR5,UIPAR6
8599 026106  001003                         BNE     3$
8600 026110  012737  140000  002472  2$:    MOV     #140000,LINK1
8601
8602 026116  004737  026656          3$:    CALL    SUPD04
8603 026122  005037  002410                 CLR     NOSCOPE
8604 026126  000207                         RETURN
8605
8606 026130                          MT0034: SUBTST  <<MT0034        SOFT ERROR - BACKGROUND PATTERN TEST>>
                                     ;*****************************************************************************
                                     ;*SUBTEST        MT0034  SOFT ERROR - BACKGROUND PATTERN TEST
                                     ;*****************************************************************************
8607 026130  012737  000034  002260         MOV     #34,REALPAT
8608 026136  012700  060000                 MOV     #FIRST,R0
8609 026142  012701  040000                 MOV     #SIZE,R1
8610 026146  013702  002560                 MOV     SOFTPAT,R2
8611 026152  010103                         MOV     R1,R3
8612 026154  013705  002102                 MOV     BANKINDEX,R5
8613 026160  010004                         MOV     R0,R4
8614 026162  022737  000001  003716         CMP     #1,PROTYP               ;IS THIS AN 11/44?
8615 026170  001006                         BNE     1$                      ;BRANCH IF NOT
8616 026172                                 BMOV    MTP034
8617 026200  012737  177646  002254         MOV     #UIPAR3,SUPDOADD
8618 026206                          1$:    IF #BIT13 SET.IN CONFIG+2(R5)
8619                                           ;BACKGROUND PATTERN IS VALID
8620 026216  022737  000001  003716           CMP   #1,PROTYP
8621 026224  001403                           BEQ   2$
8622 026226  012737  036210  002254           MOV   #MTP034+6,SUPDOADD
8623 026234  004737  026642          2$:      CALL  SUPD03                  ;READ IT
8624 026240                                 ELSE
8625                                           ;BACKGROUND PATTERN HAS BEEN INVALIDATED
8626 026242  022737  000001  003716           CMP   #1,PROTYP
```

```
8627 026250  001406                          BEQ   3$
8628 026252  012737  036202  002254          MOV   #MTP034,SUPDOADD
8629 026260  004737  026642                  CALL  SUPD03
8630 026264  000402                          BR    4$
8631 026266  004737  026464          3$:     CALL  SUPD01                    ;WRITE IT
8632 026272  052765  020000  002626  4$:     BIS   #BIT13,CONFIG+2(R5)       ;VALIDATE IT
8633 026300                                 END ;OF IF #BIT13
8634 026300  000207                         RETURN
8635
8636 026302                                 MT0035: SUBTST  <<MT0035        SETUP WORST CASE NOISE PARITY TEST>>
                                            ;*****************************************************************************
                                            ;*SUBTEST       MT0035  SETUP WORST CASE NOISE PARITY TEST
                                            ;*****************************************************************************
8637 026302  012737  000035  002260          MOV     #35,REALPAT             ;SET UP TEST NUMBER FOR DISPLAY
8638 026310  013703  002102                  MOV     BANKINDEX,R3
8639 026314  016301  002624                  MOV     CONFIG(R3),R1
8640 026320  000301                          SWAB    R1
8641 026322  042701  177760                  BIC     #^C17,R1
8642 026326  006301                          ASL     R1
8643 026330  010137  002146                  MOV     R1,CSRNO
8644 026334  023737  002146  002502          CMP     CSRNO,PGMCSR
8645 026342  001001                          BNE     1$
8646 026344  000207                          RETURN
8647 026346  012702  052524          1$:     MOV     #52524,R2
8648 026352  004737  036530                  CALL    BACKGND                 ;WRITE BACKROUND OF ALMOST ALT. 1'S AND 0'S
8649 026356  012737  036226  002254          MOV     #MTP035,SUPDOADD
8650 026364  004737  026642                  CALL    SUPD03
8651 026370                                 IF QVFLAG IS TRUE THEN $RETURN
8652 026400  005102                          COM     R2
8653 026402  004737  036530                  CALL    BACKGND                 ;WRITE COMPLEMENT PATTERN INTO MUT
8654 026406  004737  026656                  CALL    SUPD04
8655 026412  000207                         RETURN
```

```
 8658 026414                        MT0999: SUBTST  <<MT0999        SETUP NULL TEST>>
                                    ;*****************************************************************************
                                    ;*SUBTEST       MT0999  SETUP NULL TEST
                                    ;*****************************************************************************
 8659 026414  005037  002260                CLR     REALPAT
 8660 026420                                SET     NULLFLAG
 8661 026426  000207                        RETURN
 8662
 8663 026430                        KAMITEST:SUBTST <<CHECK FOR KAMIKAZE MODE>>
                                    ;*****************************************************************************
                                    ;*SUBTEST       CHECK FOR KAMIKAZE MODE
                                    ;*****************************************************************************
 8664 026430                                IF KAMIKAZE IS TRUE OR ACTFLAG IS TRUE OR APTFLAG IS TRUE
 8665 026452                                  $RETURN NOERROR                ;RUN THE TEST
 8666 026456                                ELSE
 8667 026460                                  $RETURN ERROR                  ;DON'T RUN THE TEST
 8668 026464                                END ;OF IF KAMIKAZE
```

```
                                                                    B 1
8671 026464                         SUPD01: SUBTST  <<SUBR  EXECUTE PATTERN IN SUPERVISOR>>
                                    ;*****************************************************************************
                                    ;*SUBTEST       SUBR    EXECUTE PATTERN IN SUPERVISOR
                                    ;*****************************************************************************
8672 026464                                 MAP     BANK                        ;MAP SUPERVISOR SPACE (TEST AREA) TO BANK
8673 026500  004737  055656            SUPD02: CALL    GETDIS
8674 026504                                 PUSH    $LPERR,$LPADR
8675 026514  010037  002152                MOV     R0,SUPDR0
8676 026520  012700  002154                MOV     #SUPDR1,R0
8677 026524  010120                        MOV     R1,(R0)+
8678 026526  010220                        MOV     R2,(R0)+
8679 026530  010320                        MOV     R3,(R0)+
8680 026532  010420                        MOV     R4,(R0)+
8681 026534  010520                        MOV     R5,(R0)+
8682 026536  010620                        MOV     SP,(R0)+
8683 026540  013700  002152                MOV     SUPDR0,R0
8684 026544  012737  026560  002562        MOV     #TAG4$,$LPADR
8685 026552  013737  002562  002564        MOV     $LPADR,$LPERR
8686 026560  012700  002170            TAG4$:  MOV     #SUPDR6+2,R0
8687 026564  014006                        MOV     -(R0),SP
8688 026566  014005                        MOV     -(R0),R5
8689 026570  014004                        MOV     -(R0),R4
8690 026572  014003                        MOV     -(R0),R3
8691 026574  014002                        MOV     -(R0),R2
8692 026576  014001                        MOV     -(R0),R1
8693 026600  014000                        MOV     -(R0),R0
8694 026602                                 SUPERVISOR              ;ENTER SUPERVISOR MODE
8695 026610  012706  000740                MOV     #SUPSTK,SSP
8696 026614  104424                        CACHOFF                 ;TURN CACHE OFF
8697 026616  004737  177640                CALL    FASTCITY                ;CALL TO THE USER INSTRUCTION PAR'S
8698 026622  104423                        CACHON                  ;TURN CACHE ON
8699 026624  104417                        KERNEL                  ;ENTER KERNEL MODE
8700 026626  000004                        SCOPE
8701 026630                                 POP     $LPADR,$LPERR
8702 026640  000207                        RETURN
```

```
 8705 026642                                SUPD03: MAP     BANK                    ;MAP SUPERVISOR SPACE (TEST AREA) TO BANK
 8706 026656  004737  055656                SUPD04: CALL    GETDIS
 8707 026662                                        PUSH    $LPERR,$LPADR
 8708 026672  010037  002152                        MOV     R0,SUPDR0
 8709 026676  012700  002154                        MOV     #SUPDR1,R0
 8710 026702  010120                        1$:     MOV     R1,(R0)+
 8711 026704  010220                                MOV     R2,(R0)+
 8712 026706  010320                                MOV     R3,(R0)+
 8713 026710  010420                                MOV     R4,(R0)+
 8714 026712  010520                                MOV     R5,(R0)+
 8715 026714  010620                                MOV     SP,(R0)+
 8716 026716  013700  002152                        MOV     SUPDR0,R0
 8717 026722  012737  026736  002562                MOV     #TBG4$,$LPADR
 8718 026730  013737  002562  002564                MOV     $LPADR,$LPERR
 8719 026736  012700  002170                TBG4$:  MOV     #SUPDR6+2,R0
 8720 026742  014006                        2$:     MOV     -(R0),SP
 8721 026744  014005                                MOV     -(R0),R5
 8722 026746  014004                                MOV     -(R0),R4
 8723 026750  014003                                MOV     -(R0),R3
 8724 026752  014002                                MOV     -(R0),R2
 8725 026754  014001                                MOV     -(R0),R1
 8726 026756  014000                                MOV     -(R0),R0
 8727 026760                                        TESTAREA                        ;ENTER SUPERVISOR MODE
 8728 026766  005737  002426                        TST     NOSUPER
 8729 026772  001403                                BEQ     1$
 8730 026774  012706  000700                        MOV     #USESTK,USP
 8731 027000  000402                                BR      2$
 8732 027002  012706  000740                1$:     MOV     #SUPSTK,SSP
 8733 027006  104424                        2$:     CACHOFF                         ;TURN CACHE OFF
 8734 027010  004777  153240                        CALL    @SUPDOADD
 8735 027014  104423                                CACHON                          ;TURN CACHE ON
 8736 027016  104417                                KERNEL                          ;ENTER KERNEL MODE
 8737 027020  000004                                SCOPE
 8738 027022                                        POP     $LPADR,$LPERR
 8739 027032  000207                                RETURN
```

```
                                        .SBTTL  MEMORY TEST PATTERN ROUTINES
                                ;*******************************************************************************
                                ; PATTERN REGISTER CONVENTIONS
                                ;       R0      FIRST ADDRESS OF PATTERN (FIRST,LAST+2,ETC)
                                ;       R1      NUMBER OF ADDRESSES IN PATTERN (SIZE)
                                ;       R2      DATA FOR PATTERN (ONES,52525,ETC)
                                ;       R3      COPY OF R1 (IF NECESSARY)
                                ;       R4      COPY OF R0 (IF NECESSARY)
                                ;       R5      COPY OF R2 (IF NECESSARY)
                                ;*******************************************************************************
8752 027034                     MTP000: SUBTST  <<MTP000        BASIC DATA TEST>>
                                ;*******************************************************************************
                                ;*SUBTEST       MTP000  BASIC DATA TEST
                                ;*******************************************************************************
8753 027034  010220             1$:     MOV     R2,(R0)+        ;V177640
8754 027036  077102                     SOB     R1,MTP000       ;V177642
8755 027040  000240                     NOP                     ;V177644
8756 027042  012401             2$:     MOV     (R4)+,R1        ;V177646
8757 027044  020102                     CMP     R1,R2           ;V177650
8758 027046  001402                     BEQ     3$              ;V177652
8759 027050  104430                     PERR02                  ;V177654
8760 027052  000240                     NOP                     ;V177656
8761 027054  077306             3$:     SOB     R3,2$           ;V177660
8762 027056  000207                     RETURN                  ;V177662
8763 027060                     MTP001: SUBTST  <<MTP001        ADDRESS TEST>>
                                ;*******************************************************************************
                                ;*SUBTEST       MTP001  ADDRESS TEST
                                ;*******************************************************************************
8764 027060  010220             3$:     MOV     R2,(R0)+        ;V177640
8765 027062  062702  000002             ADD     #2,R2           ;V177642
8766 027066  077104                     SOB     R1,3$           ;V177646
8767 027070  000240                     NOP                     ;V177650
8768 027072  012400             1$:     MOV     (R4)+,R0        ;V177652
8769 027074  020005                     CMP     R0,R5           ;V177654
8770 027076  001401                     BEQ     2$              ;V177656
8771 027100  104427                     PERR01                  ;V177660
8772 027102  062705  000002     2$:     ADD     #2,R5           ;V177662
8773 027106  077307                     SOB     R3,1$           ;V177666
8774 027110  000207                     RETURN                  ;V177672
8775 027112                     MTP002: SUBTST  <<MTP002        COMPLEMENT ADDRESS TEST (WRITE DOWN, READ UP)>>
                                ;*******************************************************************************
                                ;*SUBTEST       MTP002  COMPLEMENT ADDRESS TEST (WRITE DOWN, READ UP)
                                ;*******************************************************************************
8776 027112  010540             3$:     MOV     R5,-(R0)        ;V177640
8777 027114  062705  000002             ADD     #2,R5           ;V177642
8778 027120  077104                     SOB     R1,3$           ;V177646
8779 027122  000240                     NOP                     ;V177650
8780 027124  162702  000002     1$:     SUB     #2,R2           ;V177652
8781 027130  012401                     MOV     (R4)+,R1        ;V177656
8782 027132  020102                     CMP     R1,R2           ;V177660
8783 027134  001401                     BEQ     2$              ;V177662
8784 027136  104430                     PERR02                  ;V177664
8785 027140  077307             2$:     SOB     R3,1$           ;V177666
8786 027142  000207                     RETURN                  ;V177670
```

```
                                MTPA03: SUBTST  <<MTPA03        3 XOR 9 WORST CASE NOISE TEST (WRITE)>>
                                ;*****************************************************************************
                                ;*SUBTEST        MTPA03  3 XOR 9 WORST CASE NOISE TEST (WRITE)
                                ;*****************************************************************************
                                        ;R1 = ADDRESS
                                        ;R2 = SMALL LOOP CONSTANT
                                        ;R3 = NUM OF ADD TO TEST (LARGE LOOP)
                                        ;R4 = GOOD DATA
                                        ;R5 = MEDIUM LOOP CONSTANT
                                        .ENABL  LSB
027144  010421          1$:    MOV     R4,(R1)+        ;V177640
027146  010421                 MOV     R4,(R1)+        ;V177642
027150  077203                 SOB     R2,1$           ;V177644
027152  005104                 COM     R4              ;V177646
027154  052704                 BIS     (PC)+,R4        ;V177650
027156  000401          WARN2: 401                     ;V177652 WARNING LOCATION IS MODIFIED BEFORE LOADING
027160  012702  000004         MOV     #4,R2           ;V177654
027164  077511                 SOB     R5,1$           ;V177660
027166  005104                 COM     R4              ;V177662
027170  052704                 BIS     (PC)+,R4        ;V177664
027172  000401          WARN3: 401                     ;V177666 WARNING LOCATION IS MODIFIED BEFORE LOADING
027174  012705  000100         MOV     #64.,R5         ;V177670
027200  077317                 SOB     R3,1$           ;V177674
027202  000207                 RETURN                  ;V177676
                                        .DSABL  LSB

027204                          MTPB03: SUBTST  <<MTPB03        3 XOR 9 WORST CASE NOISE TEST (READ)>>
                                ;*****************************************************************************
                                ;*SUBTEST        MTPB03  3 XOR 9 WORST CASE NOISE TEST (READ)
                                ;*****************************************************************************
                                        .ENABL  LSB
027204  000137  027244  1$:    JMP     @#MTPC03        ;V177640        GO TO V172360
027210  077203                 SOB     R2,1$           ;V177644
027212  005104                 COM     R4              ;V177646
027214  052704                 BIS     (PC)+,R4        ;V177650
027216  000401          WARN4: 401                     ;V177652 WARNING LOCATION IS MODIFIED BEFORE LOADING
027220  012702  000004         MOV     #4,R2           ;V177654
027224  077511                 SOB     R5,1$           ;V177660
027226  005104                 COM     R4              ;V177662
027230  052704                 BIS     (PC)+,R4        ;V177664
027232  000401          WARN5: 401                     ;V177666 WARNING LOCATION IS MODIFIED BEFORE LOADING
027234  012705  000100         MOV     #64.,R5         ;V177670
027240  077317                 SOB     R3,1$           ;V177674
027242  000207                 RETURN                  ;V177676
                                        .DSABL  LSB
```

```
 8830 027244                          MTPC03: SUBTST  <<MTPC03        TEST DATA SUBPROGRAM>>
                                      ;*****************************************************************************
                                      ;*SUBTEST        MTPC03  TEST DATA SUBPROGRAM
                                      ;*****************************************************************************
 8831 027244  020421                          CMP     R4,(R1)+        ;V172360
 8832 027246  001401                          BEQ     1$              ;V172362
 8833 027250  104431                          PERR03                  ;V172364
 8834 027252  005141                  1$:     COM     -(R1)           ;V172366
 8835 027254  005111                          COM     (R1)            ;V172370
 8836 027256  000137  027262                  JMP     @#MTPD03        ;V172372        GO TO V172260
 8837
 8838 027262                          MTPD03: SUBTST  <<MTPD03        TEST DATA SUBSUBPROGRAM>>
                                      ;*****************************************************************************
                                      ;*SUBTEST        MTPD03  TEST DATA SUBSUBPROGRAM
                                      ;*****************************************************************************
 8839 027262  020421                          CMP     R4,(R1)+        ;V172260
 8840 027264  001401                          BEQ     1$              ;V172262
 8841 027266  104431                          PERR03                  ;V172264
 8842 027270  005127                  1$:     COM     (PC)+           ;V172266
 8843 027272  000000                          0                       ;V172270
 8844 027274  001363                          BNE     MTPC03          ;V172272        GO TO V172360
 8845 027276  000137  027210                  JMP     @#MTPB03+4      ;V172274        GO TO V177644
```

```
                                                    MTPA04: SUBTST  <<MTPA04        ROTATING ZEROS TEST>>
                                                    ;*****************************************************************************
                                                    ;*SUBTEST        MTPA04  ROTATING ZEROS TEST
                                                    ;*****************************************************************************
 8849 027302  012705  000010                       1$:     MOV     #8.,R5          ;V177640
 8850 027306  010504                                       MOV     R5,R4           ;V177644
 8851 027310  000241                                       CLC                     ;V177646
 8852 027312  000137  027336                               JMP     @#MTPB04        ;V177650
 8853 027316  016004  177776                               MOV     -2(R0),R4       ;V177654
 8854 027322  103402                                       BCS     2$              ;V177660
 8855 027324  020204                                       CMP     R2,R4           ;V177662
 8856 027326  001401                                       BEQ     3$              ;V177664
 8857 027330  104432                                2$:    PERR04                  ;V177666
 8858 027332  077115                                3$:    SOB     R1,1$           ;V177670
 8859 027334  000207                                       RETURN                  ;V177672
 8860
 8861 027336                                        MTPB04: SUBTST  <<MTPB04        SUBR    ROTATING BIT>>
                                                    ;*****************************************************************************
                                                    ;*SUBTEST        MTPB04  SUBR    ROTATING BIT
                                                    ;*****************************************************************************
 8862 027336  106110                                1$:     ROLB    (R0)            ;V172360
 8863 027340  077502                                       SOB     R5,1$           ;V172362
 8864 027342  106120                                       ROLB    (R0)+           ;V172364
 8865 027344  106110                                2$:    ROLB    (R0)            ;V172366
 8866 027346  077402                                       SOB     R4,2$           ;V172370
 8867 027350  106120                                       ROLB    (R0)+           ;V172372
 8868 027352  000137  027316                               JMP     @#MTPA04+14     ;V172374
 8869
 8870 027356                                        MTP005: SUBTST  <<MTP005        ROTATION ONES TEST>>
                                                    ;*****************************************************************************
                                                    ;*SUBTEST        MTP005  ROTATION ONES TEST
                                                    ;*****************************************************************************
 8871 027356  012705  000010                       1$:     MOV     #8.,R5          ;V177640
 8872 027362  010504                                       MOV     R5,R4           ;V177644
 8873 027364  000261                                       SEC                     ;V177646
 8874 027366  000137  027336                               JMP     @#MTPB04        ;V177650
 8875 027372  016004  177776                               MOV     -2(R0),R4       ;V177654
 8876 027376  103002                                       BCC     2$              ;V177660 IF THIS HAPPENS THE GOOD & BAD MATCH
 8877 027400  020204                                       CMP     R2,R4           ;V177662
 8878 027402  001401                                       BEQ     3$              ;V177664
 8879 027404  104432                                2$:    PERR04                  ;V177666
 8880 027406  077115                                3$:    SOB     R1,1$           ;V177670
 8881 027410  000207                                       RETURN                  ;V177672
```

```
                                        MTP006: SUBTST  <<MTP006        INITIAL DATA TEST>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       MTP006  INITIAL DATA TEST
                                        ;*****************************************************************************
                                                        ;       THIS TEST CHECKS THE DI/DO LINES BY
                                                        ;       SHIFTING A 1 THROUGH THE WORD.
 8887 027412  012737  000001  002234          MOV     #1,DATBUF       ;SET THE FIRST TEST BIT
 8888 027420  005037  002236                  CLR     DATBUF+2        ;CLEAR 2ND WORD
 8889 027424  013771  002234  000000  1$:     MOV     DATBUF,@(R1)    ;WRITE TEST WORD 1
 8890 027432  013771  002236  000002          MOV     DATBUF+2,@2(R1) ;AND TEST WORD 2
 8891 027440  017102  000000                  MOV     @(R1),R2
 8892 027444  023702  002234                  CMP     DATBUF,R2       ;NOW READ THEM
 8893 027450  001401                          BEQ     2$              ;BR IF FIRST 16 OK
 8894 027452  104433                          PERR07                  ;ERROR TRAP
 8895
 8896 027454  017102  000002          2$:     MOV     @2(R1),R2
 8897 027460  023702  002236                  CMP     DATBUF+2,R2     ;NOW READ SECOND WORD
 8898 027464  001401                          BEQ     3$              ;BR IF OK
 8899 027466  104434                          PERR10                  ;ERROR TRAP
 8900
 8901 027470  005737  002236          3$:     TST     DATBUF+2        ;HAS LAST BIT BEEN TESTED ?
 8902 027474  100405                          BMI     4$              ;MINUS MEANS BIT 31
 8903 027476                                  DLEFT   DATBUF          ;NO, SHIFT TEST BIT LEFT
 8904 027506  000746                          BR      1$              ;GO WRITE NEW TEST DATA
 8905                                                                 ;NOW GOING TO SHIFT A 0 IN DATA DIRECTION
 8906 027510  012737  177776  002234  4$:     MOV     #177776,DATBUF  ;PUT A 0 IN BIT 0
 8907 027516  012737  177777  002236          MOV     #-1, DATBUF+2   ;AND 1'S IN ALL OTHERS
 8908 027524  013771  002234  000000  5$:     MOV     DATBUF,@(R1)    ;WRITE THE DATA
 8909 027532  013771  002236  000002          MOV     DATBUF+2,@2(R1) ;2 WORDS WORTH
 8910 027540  017102  000000                  MOV     @(R1),R2
 8911 027544  023702  002234                  CMP     DATBUF,R2       ;NOW READ FIRST WORD
 8912 027550  001401                          BEQ     6$              ;BR IF OK
 8913 027552  104433                          PERR07
 8914
 8915 027554  017102  000002          6$:     MOV     @2(R1),R2
 8916 027560  023702  002236                  CMP     DATBUF+2,R2     ;NOW, READ SECOND WORD
 8917 027564  001401                          BEQ     7$              ;BR IF OK
 8918 027566  104434                          PERR10
 8919
 8920 027570  005737  002236          7$:     TST     DATBUF+2        ;TESTED BIT 31 YET?
 8921 027574  100005                          BPL     8$              ;BR IF YES, WE'RE DONE
 8922 027576                                  DLEFT   DATBUF
 8923 027606  000746                          BR      5$              ;KEEP GOING
 8924 027610  000207          8$:     RETURN
```

```
 8927 027612                    MTP007: SUBTST  <<MTP007        ADDRESS BIT TEST>>
                                ;*****************************************************************************
                                ;*SUBTEST       MTP007  ADDRESS BIT TEST
                                ;*****************************************************************************
 8928                                           ;       THIS TEST CHECKS TO SEE THAT EACH ADDRESS
 8929                                           ;       BIT IN EACH 16K BANK CAN BE ASSERTED UNIQUELY.
 8930                                           ;       IT CHECKS FOR ADDRESS BITS THAT MAY BE STUCK
 8931                                           ;       HIGH, STUCK LOW OR STUCK TOGETHER.
 8932 027612  111100                    MOVB    (R1),R0
 8933 027614  105700                    TSTB    R0              ;READ AND COMPARE FOR ZEROS
 8934 027616  001401                    BEQ     1$              ;BR IF OK
 8935 027620  104435                    PERR11
 8936
 8937 027622  105111            1$:     COMB    (R1)            ;COMPLEMENT THE BYTE
 8938 027624  111100                    MOVB    (R1),R0
 8939 027626  105700                    TSTB    R0              ;READ FOR NON ZERCS
 8940 027630  001001                    BNE     2$              ;BR IF OK
 8941 027632  104436                    PERR12
 8942
 8943 027634  040201            2$:     BIC     R2,R1           ;MASK OFF THE ASSERTED BIT
 8944 027636  006302                    ASL     R2              ;SHIFT R2 FOR NEXT BIT
 8945 027640  050201                    BIS     R2,R1           ;SET THE NEW BIT INTO R1
 8946 027642  011100                    MOV     (R1),R0
 8947 027644  005700                    TST     R0              ;READ THE NEW ADDRESS
 8948 027646  001401                    BEQ     3$              ;READ FOR ZEROS
 8949 027650  104437                    PERR13
 8950
 8951 027652  005111            3$:     COM     (R1)            ;COMPL THE WORD
 8952 027654  011100                    MOV     (R1),R0
 8953 027656  005700                    TST     R0              ;READ IT AGAIN
 8954 027660  001001                    BNE     4$
 8955 027662  104440                    PERR14
 8956
 8957 027664  022702  100000    4$:     CMP     #100000,R2
 8958 027670  001407                    BEQ     5$
 8959 027672  022702  010000            CMP     #10000,R2       ;CHECK FOR MSB IN 4K BANK
 8960 027676  001356                    BNE     2$              ;NOT LAST BIT, BRANCH
 8961 027700  006302                    ASL     R2
 8962 027702  012701  160000            MOV     #160000,R1
 8963 027706  000752                    BR      2$
 8964 027710  000207            5$:     RETURN
```

```
                                        MTP010: SUBTST  <<MTP010        BYTE ADDRESSING TEST>>
                                        ;*************************************************************************
                                        ;*SUBTEST       MTP010  BYTE ADDRESSING TEST
                                        ;*************************************************************************
                                                        ;TEST 3 THIS TEST CHECKS FOR PROPER
                                                        ;       BYTE ADDRESSING WITH ECC DISABLED
 8970 027712  010402                            MOV     R4,R2           ;R4 HAS LOWEST ADDRESS
 8971 027714  010403                            MOV     R4,R3           ;PUT IT IN R3 ALSO
 8972 027716  062702  000004                    ADD     #4,R2           ;POINT R2 TO LAST BYTE +1
 8973 027722  012713  177777                    MOV     #-1,(R3)        ;WRITE ALL ONES IN
 8974 027726  012763  177777  000002            MOV     #-1,2(R3)       ;THE 4 TEST BYTES
 8975 027734  105013                    1$:     CLRB    (R3)            ;CLEAR A BYTE
 8976 027736  010401                            MOV     R4,R1           ;INITIALIZE R1 FOR EACH PASS
 8977 027740  020201                    2$:     CMP     R2,R1           ;IF EQUAL, JUST READ LAST BYTE
 8978 027742  001420                            BEQ     6$              ;BR IF EQUAL
 8979 027744  020301                            CMP     R3,R1           ;IS THIS THE BYTE OF ZEROS
 8980 027746  001007                            BNE     4$              ;BR IF NOT
 8981 027750  111100                            MOVB    (R1),R0
 8982                                           ;WARNING IF YOU OPTOMIZE CHANGE THE PCBUMP FOR THIS ERROR INCASE OF TRAPS
 8983 027752  022700  000000                    CMP     #0,R0           ;IT IS, COMPARE FOR ZEROS
 8984 027756  001401                            BEQ     3$
 8985 027760  104435                            PERR11
 8986
 8987 027762  005201                    3$:     INC     R1              ;NEXT BYTE
 8988 027764  000765                            BR      2$              ;RETURN
 8989 027766  111100                    4$:     MOVB    (R1),R0
 8990 027770  122700  177777                    CMPB    #-1,R0          ;ITS NOT THE BYTE OF 0'S, READ 1'S
 8991 027774  001401                            BEQ     5$
 8992 027776  104436                            PERR12
 8993
 8994 030000  005201                    5$:     INC     R1              ;MOVE TO NEXT BYTE
 8995 030002  000756                            BR      2$
 8996 030004  112713  177777            6$:     MOVB    #-1,(R3)        ;RESTORE 1'S TO BYTE JUST TESTED
 8997 030010  005203                            INC     R3              ;INC TO NEXT BYTE
 8998 030012  020302                            CMP     R3,R2           ;WAS THAT JUST THE LAST ONE?
 8999 030014  001347                            BNE     1$              ;BR IF NO
 9000 030016  000207                            RETURN
```

```
 9003 030020                            MTP011: SUBTST  <<MTP011        SINGLE BIT ERROR TEST>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       MTP011  SINGLE BIT ERROR TEST
                                        ;*****************************************************************************
 9004                                           ;(1)    CREATE A SINGLE BIT ERROR
 9005                                           ;
 9006                                           ;(2)    READ BACK SBE UNCORRECTED (WITH ECC DISABLE)
 9007                                           ;
 9008                                           ;(3)    ENABLE ECC & READ CORRECTED DATA
 9009                                           ;
 9010                                           ;(4)    CHECK THAT THE SBE FLAG WAS SET FROM THE LAST READ
 9011                                           ;
 9012                                           ;(5)    DO (1-4) FOR DATA CONSISTING OF 1 BIT SET IN EACH OF 32
 9013                                           ;       POSITIONS OF A DOUBLE WORD
 9014                                           ;       THEN DO IT AGAIN FOR 1 BIT CLEARED IN EACH OF 32 POSITIONS OF
 9015                                           ;       A DOUBLE WORD
 9016                                           ;       IE (64 TIMES)
 9017                                           ;
 9018                                           ;(6)    DO (1-5) FOR A SBE IN EACH OF 32 BIT POSITIONS
 9019                                           ;       IE (RUN TEST 64 * 32 = 2048 TIMES)
 9020 030020  104503                            CLR1CSR                 ;CLEAR 1 SELECTED CSR
 9021 030022  005737  013136                    TST     PHEBE           ;TEST SPECIAL CASE INDICATOR
 9022 030026  001407                            BEQ     MTLA11          ;BRANCH IF NOT SET
 9023 030030  013702  172246                    MOV     SIPAR3,R2       ;SAVE CONTENTS OF SIPAR #3
 9024 030034  013737  172252  172246            MOV     SIPAR5,@#SIPAR3 ;COPY CONTENTS OF #5 INTO #3
 9025 030042  010237  172252                    MOV     R2,@#SIPAR5     ;COPY CONTENTS OF #3 INTO #5
 9026                                           ;BIG LOOP
 9027 030046  012737  000001  002234  MTLA11:  MOV     #1,DATBUF       ;INITIAL DATA
 9028 030054  005037  002236                    CLR     DATBUF+2        ;32 BITS WORTH
 9029                                           ;MEDIUM LOOP
 9030 030060  012737  000001  002244  MTLB11:  MOV     #1,SBEMSK       ;INITIAL ERROR MASK
 9031 030066  005037  002246                    CLR     SBEMSK+2        ;32 BITS WORTH
 9032                                           ;LITTLE LOOP
 9033 030072  013737  002234  002240  MTLC11:  MOV     DATBUF,TSTDAT   ;
 9034 030100  013737  002236  002242            MOV     DATBUF+2,TSTDAT+2;TO SAVE ORIG DATA
 9035 030106  105737  002256                    TSTB    PASFLG  ;COMP DATA ON SECOND PASSONLY
 9036 030112  001404                            BEQ     4$              ;BR IF FIRST PASS
 9037 030114  005137  002240                    COM     TSTDAT          ;SECOND PASS, COMP BOTH WORDS
 9038 030120  005137  002242                    COM     TSTDAT+2
 9039 030124  013702  002240            4$:     MOV     TSTDAT,R2
 9040 030130  013703  002242                    MOV     TSTDAT+2,R3
 9041 030134  012737  002240  002272            MOV     #TSTDAT,SOURCE  ;SET UP ADDRESS FOR CHKGEN
 9042 030142  004737  041722                    CALL    CHKGEN          ;GEN CHECKBITS ON TSTDAT
 9043                                           ;*********************************
 9044                                           ;** CREATE A SINGLE BIT ERROR **
 9045                                           ;*********************************
 9046 030146  013701  002244                    MOV     SBEMSK,R1
 9047 030152  074137  002240                    XOR     R1,TSTDAT
 9048 030156  013701  002246                    MOV     SBEMSK+2,R1
 9049 030162  074137  002242                    XOR     R1,TSTDAT+2
 9050 030166  013701  002362            MTLD11: MOV     TESTADD,R1      ;FIRST TEST ADDRESS
 9051 030172  013705  002364                    MOV     TESTADD+2,R5    ;SECOND TEST ADDRESS
 9052 030176  104471                            ECC1DIS                 ;DISABLE ECC ON 1 SELECTED CSR
 9053 030200  013711  002240                    MOV     TSTDAT,(R1)     ;WRITE FIRST 16 BITS
 9054 030204  104475                            CB1CSR                  ;WRITE GENERATED CHECKBITS IN 1 SELECTED CSR
 9055 030206  013715  002242                    MOV     TSTDAT+2,(R5)   ;WRITE SECOND 16 BITS AND
 9056                                                                   ;CHECK BITS. WE NOW HAVE CHECKBITS
```

```
                                                                ;GENERATED ON DATBUF AND DATA WITH
                                                                ;ONE BIT IN ERROR (AS PER SBEMSK).
030212  104471                          ECC1DIS                 ;DISABLE ECC ON 1 SELECTED CSR
030214  011100                          MOV     (R1),R0
030216  020037  002240                  CMP     R0,TSTDAT       ;READ THE LOW WORD (UNCORRECTED)
030222  001403                          BEQ     6$              ;BR IF OK
030224  010137  002032                  MOV     R1,ADDRESS
030230  104455                          PERR31

030232  011500                  6$:     MOV     (R5),R0
030234  020037  002242                  CMP     R0,TSTDAT+2     ;READ THE HIGH WORD (UNCORRECTED)
030240  001403                          BEQ     7$              ;BR IF OK
030242  010537  002032                  MOV     R5,ADDRESS
030246  104455                          PERR31

030250                          7$:     IF KFLAG IS FALSE
030256  104426                            READCSR
030260                                  IF #BIT4 OFF.IN CSR OR #BIT15 OFF.IN CSR
030300  104045                            ERROR       +45
030302                                  END; OF IF #BIT4
030302                                END; OF IF KFLAG
030302  005737  013136                  TST     PHEBE
030306  001001                          BNE     17$
030310  104512                          ERRGEN
030312  104503                  17$:    CLR1CSR                 ;CLEAR 1 SELECTED CSR
030314  011100                          MOV     (R1),R0
030316  020002                          CMP     R0,R2           ;SEE IF ITS BEEN CORRECTED
030320  001401                          BEQ     8$              ;IT SHOULD HAVE BEEN
030322  104456                          PERR32

030324  104510                  8$:     TSTREAD                 ;TEST LOC (R1) & TST FOR SBE (WITHOUT FETCHES)
030326  103411                          BCS     9$              ;BR IF IT IS SET
030330                                  SET     HEADER          ;ENABLE PRINTING OF ERROR HEADER INFO
030336  010137  002032                  MOV     R1,ADDRESS
030342  104460                          PERR34
030344                                  SET     HEADER          ;ENABLE PRINTING OF ERROR HEADER INFO

030352  104503                  9$:     CLR1CSR                 ;CLEAR 1 SELECTED CSR
030354  011500                          MOV     (R5),R0
030356  020003                          CMP     R0,R3           ;SEE IF ITS BEEN CORRECTED
030360  001401                          BEQ     10$             ;BR IF OK
030362  104456                          PERR32

030364  104510                  10$:    TSTREAD                 ;TEST LOC (R1) & TST FOR SBE (WITHOUT FETCHES)
030366  103411                          BCS     11$             ;BR IF YES
030370                                  SET     HEADER          ;ENABLE PRINTING OF ERROR HEADER INFO
030376  010137  002032                  MOV     R1,ADDRESS
030402  104460                          PERR34
030404                                  SET     HEADER          ;ENABLE PRINTING OF ERROR HEADER INFO
030412  104512                  11$:    ERRGEN                  ;TEST ERROR ADDRESS
030414  105737  002256                  TSTB    PASFLG
030420  100452                          BMI     15$
030422  005737  002246                  TST     SBEMSK+2        ;TEST FOR LAST MASK BIT
030426  100405                          BMI     12$             ;MINUS MEANS BIT 31
030430                                  DLEFT   SBEMSK
030440  000614                          BR      MTLC11
030442                          12$:    IF #SW11 SET.IN @SWR THEN GOTO 13$
```

```
 9114 030452                                        IF QVFLAG IS TRUE THEN GOTO 13$
 9115 030460  005737  002236                        TST     DATBUF+2        ;LAST DATA BIT ?
 9116 030464  100406                                BMI     13$             ;WHICH IS BIT 31
 9117 030466                                        DLEFT   DATBUF
 9118 030476  000137  030060                        JMP     MTLB11
 9119 030502  105737  002256          13$:          TSTB    PASFLG  ;FIRST OR SECOND PASS ?
 9120 030506  001004                                BNE     14$             ;NON ZERO MEANS WE'RE DONE
 9121 030510  105237  002256                        INCB    PASFLG  ;NOT DONE, GO DO SECOND PASS
 9122 030514  000137  030046                        JMP     MTLA11
 9123 030520  052737  000200  002256  14$:          BIS     #BIT7,PASFLG
 9124 030526  005002                                CLR     R2
 9125 030530  005003                                CLR     R3
 9126 030532  005037  002240                        CLR     TSTDAT
 9127 030536  005037  002242                        CLR     TSTDAT+2
 9128 030542  012704  000040                        MOV     #40,R4
 9129 030546  012737  003740  002274  15$:          MOV     #3740,CHECK
 9130 030554  C74437  002274                        XOR     R4,CHECK
 9131 030560  006304                                ASL     R4
 9132 030562  032704  020000                        BIT     #BIT13,R4
 9133 030566  001002                                BNE     16$
 9134 030570  000137  030166                        JMP     MTLD11
 9135                                               ;CLEAR OUT ANY DBE'S OR SBE'S
 9136 030574  104471                    16$:        ECC1DIS                 ;DISABLE ECC ON 1 SELECTED CSR
 9137 030576  013701  002362                        MOV     TESTADD,R1
 9138 030602  013705  002364                        MOV     TESTADD+2,R5
 9139 030606                                        CLEAR   (R1),(R5)
 9140 030612  104503                                CLR1CSR                 ;CLEAR 1 SELECTED CSR
 9141 030614  000207                                RETURN
```

```
                                                    MTP012: SUBTST  <<MTP012        WRITE BYTE CLEARS SBE TEST>>
                                                    ;*****************************************************************************
                                                    ;*SUBTEST       MTP012  WRITE BYTE CLEARS SBE TEST
                                                    ;*****************************************************************************
                                                                    ;SINGLE BIT ERROR TEST TO INSURE THAT A WRITE
                                                                    ;BYTE CLEARS SINGLE BIT ERRORS.
030616  104503                                              CLR1CSR                 ;CLEAR 1 SELECTED CSR
030620  012737  000001  002234                              MOV     #1,DATBUF       ;INITIAL DATA
030626  005037  002236                                      CLR     DATBUF+2        ;32 BITS WORTH
030632  012737  000001  002244                      1$:     MOV     #1,SBEMSK       ;INITIAL ERROR MASK
030640  005037  002246                                      CLR     SBEMSK+2        ;32 BITS WORTH
030644  013737  002234  002240                      2$:     MOV     DATBUF,TSTDAT   ;SAVE ORIGINAL DATA
030652  013737  002236  002242                              MOV     DATBUF+2,TSTDAT+2;BOTH WORDS
030660  012737  002240  002272                              MOV     #TSTDAT,SOURCE  ;NEED ADDRESS FOR CHKGEN
030666  004737  041722                                      CALL    CHKGEN          ;GENERATE CHECK BITS
030672  013701  002244                                      MOV     SBEMSK,R1
030676  074137  002240                                      XOR     R1,TSTDAT
030702  013701  002246                                      MOV     SBEMSK+2,R1
030706  074137  002242                                      XOR     R1,TSTDAT+2
030712  013704  002362                                      MOV     TESTADD,R4      ;FIRST TEST ADDRESS
030716  010401                                              MOV     R4,R1           ;PUT IT IN R1 ALSO
030720  10447                                               ECC1DIS                 ;DISABLE ECC ON 1 SELECTED CSR
030722  013711  002240                                      MOV     TSTDAT,(R1)     ;WRITE 16 BITS
030726  104475                                              CB1CSR                  ;WRITE GENERATED CHECKBITS IN 1 SELECTED CSR
030730  060501                                              ADD     R5,R1           ;INDEX UP TO SECOND WORD
030732  013711  002242                                      MOV     TSTDAT+2,(R1)   ;WRITE HIGH WORD+CHECKBITS
030736  104503                                              CLR1CSR                 ;CLEAR 1 SELECTED CSR
                                                                                    ;IT'S DANGEROUS IF WE DON'T
030740  012702  002244                                      MOV     #SBEMSK,R2      ;ADDRESS OF ERROR MASK
030744  160501                                              SUB     R5,R1           ;RETURN TO FIRST WORD
030746  112711  177777                      3$:     MOVB    #-1,(R1)        ;WRITE A BYTE OF 1'S
030752  005737  002500                              TST     KFLAG           ;IS THIS MF11S-K
030756  001403                                      BEQ     4$              ;BRANCH IF NOT - IT'S MS11-M
030760  132712  177777                              BITB    #-1,(R2)        ;DID THIS BYTE HAVE THE BAD BIT IN IT?
030764  001420                                      BEQ     6$              ;NO - BRANCH
030766  104510                      4$:     TSTREAD                 ;TEST LOC (R1) & TST FOR SBE (WITHOUT FETCHES)
030770  103011                              BCC     5$              ;NO - SKIP
030772                                      SET     HEADER          ;ENABLE PRINTING OF ERROR HEADER INFO
031000  010137  002032                              MOV     R1,ADDRESS
031004  104017                                      ERROR   +17
031006                                      SET     HEADER          ;ENABLE PRINTING OF ERROR HEADER INFO

031014  111100                      5$:     MOVB    (R1),R0
031016  122700  177777                      CMPB    #-1,R0          ;CHECK DATA
031022  001414                              BEQ     7$              ;BR IF OK
031024  104457                              PERR33

031026  104510                      6$:     TSTREAD                 ;TEST LOC (R1) & TST FOR SBE (WITHOUT FETCHES)
                                                                    ;READ THE BYTE
                                                                    ;SBE ERROR BIT ONLY SET ?
031030  103771                              BCS     5$              ;SHOULD BE SET, BR IF OK
031032                                      SET     HEADER          ;ENABLE PRINTING OF ERROR HEADER INFO
031040  010137  002032                              MOV     R1,ADDRESS
031044  104460                              PERR34
031046                                      SET     HEADER          ;ENABLE PRINTING OF ERROR HEADER INFO

031054  132712  177777                      7$:     BITB    #-1,(R2)        ;CHECK FOR LAST BYTE
```

```
9198 031060  001012                        BNE     8$              ;
9199 031062  005202                        INC     R2
9200 031064  005201                        INC     R1              ;MOVE TO NEXT BYTE
9201 031066  013704  002362                MOV     TESTADD,R4      ;FIRST TEST ADDRESS
9202 031072  032701  000002                BIT     #2,R1           ;TEST FOR LOWER WORD
9203 031076  001723                        BEQ     3$              ;BR IF IT'S LOW 16 BITS
9204 031100  062704  000002                ADD     #2,R4           ;ADJUST POINTER FOR ERROR REPT.
9205 031104  000720                        BR      3$
9206 031106  005737  002246          8$:   TST     SBEMSK+2        ;LAST ERROR BIT ?
9207 031112  100405                        BMI     9$              ;MINUS MEANS BIT 31
9208 031114                                DLEFT   SBEMSK
9209 031124  000647                        BR      2$
9210 031126                          9$:   IF #SW11 SET.IN @SWR THEN GOTO 10$
9211 031136                                IF QVFLAG IS TRUE THEN GOTO 10$
9212 031144  005737  002236                TST     DATBUF+2        ;LAST DATA BIT?
9213 031150  100405                        BMI     10$             ;MINUS = BIT 31
9214 031152                                DLEFT   DATBUF
9215 031162  000623                        BR      1$
9216                                       ;CLEAR OUT ANY DBE'S OR SBE'S
9217 031164  104471                  10$:  ECC1DIS                 ;DISABLE ECC ON 1 SELECTED CSR
9218 031166  013701  002362                MOV     TESTADD,R1
9219 031172  005011                        CLR     (R1)
9220 031174  060501                        ADD     R5,R1
9221 031176  005011                        CLR     (R1)
9222 031200  104503                        CLR1CSR                 ;CLEAR 1 SELECTED CSR
9223 031202  000207                        RETURN
```

```
 9226 031204                                MTP013: SUBTST  <<MTP013        CREATE DOUBLE BIT ERROR TEST>>
                                            ;*****************************************************************************
                                            ;*SUBTEST       MTP013  CREATE DOUBLE BIT ERROR TEST
                                            ;*****************************************************************************
 9227                                                   ;DOUBLE BIT ERROR FORCE TO CHECK DOUBLE ERROR LOGIC
 9228 031204  104503                              CLR1CSR                 ;CLEAR 1 SELECTED CSR
 9229 031206  012701  002362                      MOV     #TESTADD,R1
 9230 031212  005037  002234              1$:     CLR     DATBUF          ;MAKE INITIAL DATA
 9231 031216  005037  002236                      CLR     DATBUF+2        ;ALL ZEROS
 9232 031222  012737  000001  002244      2$:     MOV     #1,SBEMSK       ;INITIAL SINGLE ERROR MASK
 9233 031230  005037  002246                      CLR     SBEMSK+2        ;SECOND WORD
 9234 031234  012737  000001  002250      3$:     MOV     #1,DBEMSK       ;INITIAL DOUBLE ERROR MASK
 9235 031242  005037  002252                      CLR     DBEMSK+2        ;32 BITS HERE ALSO
 9236 031246  013737  002234  002240      4$:     MOV     DATBUF,TSTDAT   ;
 9237 031254  013737  002236  002242              MOV     DATBUF+2,TSTDAT+2
 9238 031262  105737  002256                      TSTB    PASFLG  ;NO COMPLEMENTING FIRST PASS
 9239 031266  001404                              BEQ     5$
 9240 031270  005137  002240                      COM     TSTDAT          ;COMP FIRST WORD
 9241 031274  005137  002242                      COM     TSTDAT+2        ;SECOND WORD
 9242 031300  104505                      5$:     CLR1CSR                 ;CLEAR 1 SELECTED CSR
 9243 031302  023737  002244  002250              CMP     SBEMSK,DBEMSK   ;CAN'T HAVE THE SAME ERROR BIT SET
 9244 031310  001004                              BNE     6$              ;IN BOTH MASKS
 9245 031312  023737  002246  002252              CMP     SBEMSK+2,DBEMSK+2;COULD BE EQUAL IN SECOND WORD
 9246 031320  001460                              BEQ     13$             ;GO MAKE THEM NOT EQUAL
 9247 031322  012737  002240  002272  6$:     MOV     #TSTDAT,SOURCE  ;SOURCE ADDRESS FOR CHKGEN
 9248 031330  004737  041722                      CALL    CHKGEN          ;GO GENERATE CHECK BITS
 9249 031334  013702  002244                      MOV     SBEMSK,R2
 9250 031340  074237  002240                      XOR     R2,TSTDAT
 9251 031344  013702  002246                      MOV     SBEMSK+2,R2
 9252 031350  074237  002242                      XOR     R2,TSTDAT+2
 9253 031354  013702  002250                      MOV     DBEMSK,R2
 9254 031360  074237  002240                      XOR     R2,TSTDAT
 9255 031364  013702  002252                      MOV     DBEMSK+2,R2
 9256 031370  074237  002242                      XOR     R2,TSTDAT+2
 9257 031374  104471                      16$:    ECC1DIS                 ;DISABLE ECC ON 1 SELECTED CSR
 9258 031376  013731  002240                      MOV     TSTDAT,@(R1)+   ;WRITE 16 BITS
 9259 031402  104475                              CB1CSR                  ;WRITE GENERATED CHECKBITS IN 1 SELECTED CSR
 9260 031404  013771  002242  000000              MOV     TSTDAT+2,@(R1)  ;WRITE HIGH WORD
 9261 031412  104503                              CLR1CSR                 ;CLEAR 1 SELECTED CSR
 9262 031414  162701  000002                      SUB     #2,R1           ;ADJUST TEST ADDRESS
 9263 031420  005771  000000                      TST     @(R1)           ;READ THE LOCATION
 9264 031424  104501                              WAS1DBE                 ;WAS THERE ANY DOUBLE BIT ERRORS ON 1 SELECTED CSR
 9265 031426  103411                              BCS     9$              ;IT SHOULD BE SET
 9266 031430                                      SET     HEADER
 9267 031436  011137  002032                      MOV     (R1),ADDRESS
 9268 031442  104030                              ERROR   +30
 9269 031444                                      SET     HEADER
```

```
9272 031452  104512                      9$:     ERRGEN
9273 031454  105737  002256                      TSTB    PASFLG
9274 031460  100452                              BMI     14$
9275 031462  005737  002252              13$:    TST     DBEMSK+2        ;CHECK MASK FOR LAST BIT
9276 031466  100405                              BMI     10$             ;MINUS = BIT31
9277 031470                                      DLEFT   DBEMSK
9278 031500  000662                              BR      4$
9279 031502                              10$:    IF #SW11 SET.IN @SWR THEN GOTO 11$
9280 031512                                      IF QVFLAG IS TRUE THEN GOTO 11$
9281 031520  005737  002246                      TST     SBEMSK+2        ;CHECK SINGLE ERROR MASK TOO
9282 031524  100405                              BMI     11$             ;BR IF DONE
9283 031526                                      DLEFT   SBEMSK
9284 031536  000636                              BR      3$
9285 031540  105737  002256              11$:    TSTB    PASFLG  ;FIRST PASS
9286 031544  001003                              BNE     12$             ;NON ZERO MEANS WE'RE DONE
9287 031546  105237  002256                      INCB    PASFLG  ;FIRST PASS, NOT DONE
9288                                             ;CLEAR OUT ANY DBE'S OR SBE'S
9289 031552  000617                              BR      1$              ;KEEP GOING
9290 031554  052737  000200  002256  12$:     BIS     #BIT7,PASFLG    ;SET UP FOR CHECK BIT TEST
9291 031562  005037  002240                      CLR     TSTDAT
9292 031566  005037  002242                      CLR     TSTDAT+2
9293 031572  012737  000040  002244              MOV     #40,SBEMSK
9294 031600  012737  000100  002250              MOV     #100,DBEMSK
9295 031606  012737  003740  002274  14$:     MOV     #3740,CHECK
9296 031614  013702  002244                      MOV     SBEMSK,R2
9297 031620  074237  002274                      XOR     R2,CHECK
9298 031624  013702  002250                      MOV     DBEMSK,R2
9299 031630  074237  002274                      XOR     R2,CHECK
9300 031634  006337  002250                      ASL     DBEMSK
9301 031640  032737  020000  002250              BIT     #BIT13,DBEMSK
9302 031646  001652                              BEQ     16$
9303 031650  006337  002244                      ASL     SBEMSK
9304 031654  032737  004000  002244              BIT     #BIT11,SBEMSK
9305 031662  001006                              BNE     15$
9306 031664  013737  002244  002250              MOV     SBEMSK,DBEMSK
9307 031672  006337  002250                      ASL     DBEMSK
9308 031676  000743                              BR      14$
9309 031700  104471                      15$:    ECC1DIS                 ;DISABLE ECC ON 1 SELECTED CSR
9310 031702  012701  002362                      MOV     #TESTADD,R1
9311 031706                                      CLEAR   @(R1)+,@(R1)
9312 031714  104503                              CLR1CSR                 ;CLEAR 1 SELECTED CSR
9313 031716  000207                              RETURN
```

```
 9316 031720                                MTP014: SUBTST  <<MTP014        WRITE INHIBIT DURING DATIP WITH DBE TEST>>
                                            ;*****************************************************************************
                                            ;*SUBTEST       MTP014  WRITE INHIBIT DURING DATIP WITH DBE TEST
                                            ;*****************************************************************************
 9317                                                       ;THIS TEST CHECKS THE WRITE INHIBIT ON DOUBLE
 9318                                                       ;BIT ERRORS DURING A DATIP OPERATION BY USE
 9319                                                       ;OF AN 'ASRB' INSTRUCTION.
 9320 031720                                        IF KFLAG IS TRUE THEN $RETURN
 9321
 9327                                                       ;NOTE- THIS TEST WILL ONLY BE RUN FOR MF11S-K MEMORY.
 9328 031730  005037  002234            1$:     CLR     DATBUF          ;INITIAL DATA
 9329 031734  005037  002236                    CLR     DATBUF+2        ;2 WORDS WORTH
 9330 031740  012737  000001  002244    2$:     MOV     #1,SBEMSK       ;INITIAL ERROR MASK
 9331 031746  005037  002246                    CLR     SBEMSK+2        ;
 9332 031752  012737  000001  002250    3$:     MOV     #1,DBEMSK       ;DOUBLE ERROR MASK
 9333 031760  005037  002252                    CLR     DBEMSK+2        ;2 WORDS
 9334 031764  013737  002234  002240    4$:     MOV     DATBUF,TSTDAT   ;PRESERVE ORIG DATA
 9335 031772  013737  002236  002242            MOV     DATBUF+2,TSTDAT+2
 9336 032000  105737  002256                    TSTB    PASFLG  ;SECOND PASS YET ?
 9337 032004  001404                            BEQ     5$              ;BR IF NO
 9338 032006  005137  002240                    COM     TSTDAT          ;COMPL DATA  ON SECOND PASS
 9339 032012  005137  002242                    COM     TSTDAT+2
 9340 032016  104503                    5$:     CLR1CSR                 ;CLEAR 1 SELECTED CSR
 9341 032020  023737  002250  002244            CMP     DBEMSK,SBEMSK   ;CHECK FOR SAME MASKS
 9342 032026  001004                            BNE     6$              ;BR IF OK
 9343 032030  023737  002252  002246            CMP     DBEMSK+2,SBEMSK+2
 9344 032036  001476                            BEQ     11$             ;BR IF THEY'RE EQUAL
 9345 032040  012737  002240  002272    6$:     MOV     #TSTDAT,SOURCE  ;SET UP ADDRESS FOR CHKGEN
 9346 032046  004737  041722                    CALL    CHKGEN          ;GENERATE CHECK BITS
 9347 032052  013701  002244                    MOV     SBEMSK,R1
 9348 032056  074137  002240                    XOR     R1,TSTDAT
 9349 032062  013701  002246                    MOV     SBEMSK+2,R1
 9350 032066  074137  002242                    XOR     R1,TSTDAT+2
 9351 032072  013701  002250                    MOV     DBEMSK,R1
 9352 032076  074137  002240                    XOR     R1,TSTDAT
 9353 032102  013701  002252                    MOV     DBEMSK+2,R1
 9354 032106  074137  002242                    XOR     R1,TSTDAT+2
 9355 032112  012701  002362            7$:     MOV     #TESTADD,R1     ;TEST ADDRESS
 9356 032116  104471                            ECC1DIS                 ;DISABLE ECC ON 1 SELECTED CSR
 9357 032120  013731  002240                    MOV     TSTDAT,@(R1)+   ;WRITE FIRST 16 BITS
 9358 032124  104475                            CB1CSR                  ;WRITE GENERATED CHECKBITS IN 1 SELECTED CSR
 9359 032126  013771  002242  000000            MOV     TSTDAT+2,@(R1)  ;SECOND 16 BITS+CHECKBITS
 9360 032134  105037  002257                    CLRB    UPPFLG          ;INDICATE LOWER WORD
 9361 032140  013703  002362                    MOV     TESTADD,R3      ;TEST ADDRESS
 9362 032144  104503                    8$:     CLR1CSR                 ;CLEAR 1 SELECTED CSR
 9363 032146  106223                            ASRB    (R3)+           ;SPECIAL DATIP INSTRUCTION
 9364 032150  015100                            MOV     @-(R1),R0
 9365 032152  023700  002240                    CMP     TSTDAT,R0       ;CHECK FOR UNCHANGED DATA
 9366 032156  001404                            BEQ     9$              ;SHOULD BE UNCHANGED
 9367 032160  017137  000000  002032            MOV     @(R1),ADDRESS
 9368 032166  104455                            PERR31
 9369
 9370 032170  062701  000002            9$:     ADD     #2,R1           ;POINT TO UPPER WORD
 9371 032174  017100  000000                    MOV     @(R1),R0
 9372 032200  023700  002242                    CMP     TSTDAT+2,R0     ;READ IT
 9373 032204  001404                            BEQ     10$             ;BR IF UNCHANGED
 9374 032206  017137  000000  002032            MOV     @(R1),ADDRESS
```

```
                                                PERR31
                                
002257  10$:    CMPB    #3,UPPFLG               ;LOWER WORD
                BEQ     11$             ;BR IF NO
002257          INCB    UPPFLG
                BR      8$
002256  11$:    TSTB    PASFLG
                BMI     15$             ;BRANCH IF WE'RE TESTING CHECK BITS
002252          TST     DBEMSK+2        ;LAST BIT IN MASK ?
                BMI     12$             ;BR IF BIT 31
                DLEFT   DBEMSK
                BR      4$
        12$:    IF #SW11 SET.IN @SWR THEN GOTO 13$
                IF QVFLAG IS TRUE THEN GOTO 13$
002246          TST     SBEMSK+2        ;LAST BIT IN SINGLE ERROR MASK ?
                BMI     13$             ;BR IF YES
                DLEFT   SBEMSK
                BR      3$
002256  13$:    TSTB    PASFLG  ;WHICH PASS
                BNE     14$             ;BR IF WE'RE DONE
002256          INCB    PASFLG  ;INDICATE SECOND PASS COMING
                ;CLEAR OUT ANY DBE'S OR SBE'S
031730          JMP     1$              ;GO DO IT!
002256  14$:    BIS     #BIT7,PASFLG
002240          CLR     TSTDAT
002242          CLR     TSTDAT+2
000040  002244  MOV     #40,SBEMSK
000100  002250  MOV     #100,DBEMSK
003740  002274  15$:    MOV     #3740,CHECK
002244          MOV     SBEMSK,R2
002274          XOR     R2,CHECK
002250          MOV     DBEMSK,R2
002274          XOR     R2,CHECK
002250          ASL     DBEMSK
020000  002250  BIT     #BIT13,DBEMSK
                BEQ     7$
002244          ASL     SBEMSK
004000  002244  BIT     #BIT11,SBEMSK
                BNE     16$
002244  002250  MOV     SBEMSK,DBEMSK
002250          ASL     DBEMSK
                BR      15$
                ECC1DIS                 ;DISABLE ECC ON 1 SELECTED CSR
002362  16$:    MOV     #TESTADD,R1
                CLEAR   @(R1)+,@(R1)
                CLR1CSR                 ;CLEAR 1 SELECTED CSR
                RETURN
```

```
 9424 032502                                MTP015: SUBTST  <<MTP015        WRITE INHIBIT OF BYTE WITH DBE>>
                                            ;*****************************************************************************
                                            ;*SUBTEST       MTP015  WRITE INHIBIT OF BYTE WITH DBE
                                            ;*****************************************************************************
 9425                                                   ;CHECK FOR WRITE INHIBIT DURING A WRITE BYTE.
 9426                                                   ;CHECKS FOR UNCORRECTED DATA.
 9427 032502  005037  002234          1$:      CLR      DATBUF                ;INITIAL DATA
 9428 032506  005037  002236                   CLR      DATBUF+2              ;32 BITS WORTH
 9429 032512  012737  000001  002244  2$:      MOV      #1,SBEMSK             ;SINGLE ERROR MASK
 9430 032520  005037  002246                   CLR      SBEMSK+2              ;
 9431 032524  012737  000001  002250  3$:      MOV      #1,DBEMSK             ;DOUBLE ERROR MASK
 9432 032532  005037  002252                   CLR      DBEMSK+2              ;
 9433 032536  013737  002234  002240  4$:      MOV      DATBUF,TSTDAT         ;PRESERVE ORIG DATA
 9434 032544  013737  002236  002242           MOV      DATBUF+2,TSTDAT+2
 9435 032552  105737  002256                   TSTB     PASFLG  ;WHICH PASS ?
 9436 032556  001404                           BEQ      5$                    ;FIRST PASS, NO COMPLEMENTING
 9437 032560  005137  002240                   COM      TSTDAT
 9438 032564  005137  002242                   COM      TSTDAT+2              ;SECOND PASS, COMPLEMENT TSTDAT
 9439 032570  104503                5$:      CLR1CSR                        ;CLEAR 1 SELECTED CSR
 9440 032572  023737  002244  002250           CMP      SBEMSK,DBEMSK         ;CHECK FOR SAME MASKS
 9441 032600  001004                           BNE      6$                    ;BR IF NOT EQUAL
 9442 032602  023737  002246  002252           CMP      SBEMSK+2,DBEMSK+2           ;SECOND WORD ALSO
 9443 032610  001474                           BEQ      11$                   ;BR TO MAKE THEM NOT EQUAL
 9444 032612  012737  002240  002272  6$:      MOV      #TSTDAT,SOURCE        ;ADDRESS FOR CHKGEN
 9445 032620  004737  041722                   CALL     CHKGEN                ;GO GENERATE CHECK BITS
 9446 032624  013701  002244                   MOV      SBEMSK,R1
 9447 032630  074137  002240                   XOR      R1,TSTDAT
 9448 032634  013701  002246                   MOV      SBEMSK+2,R1
 9449 032640  074137  002242                   XOR      R1,TSTDAT+2
 9450 032644  013701  002250                   MOV      DBEMSK,R1
 9451 032650  074137  002240                   XOR      R1,TSTDAT
 9452 032654  013701  002252                   MOV      DBEMSK+2,R1
 9453 032660  074137  002242                   XOR      R1,TSTDAT+2
 9454 032664  012701  002362          7$:      MOV      #TESTADD,R1           ;TEST LOCATION
 9455 032670  104471                           ECC1DIS                        ;DISABLE ECC ON 1 SELECTED CSR
 9456 032672  013731  002240                   MOV      TSTDAT,@(R1)+         ;WRITE FIRST 16 BITS
 9457                                           ;LOAD CSR WITH IMAGE FROM R2
 9458 032676  104475                           CB1CSR                         ;WRITE GENERATED CHECKBITS IN 1 SELECTED CSR
 9459 032700  013771  002242  000000           MOV      TSTDAT+2,@(R1)        ;WRITE SECOND 16 BITS + CHECKBITS
 9460 032706  104503                           CLR1CSR                        ;CLEAR 1 SELECTED CSR
 9461 032710  013702  002362                   MOV      TESTADD,R2            ;GET ADDRESS OF TEST LOC
 9462 032714  010203                           MOV      R2,R3                 ;R2 DESIGNATES FIRST BYTE
 9463 032716  062703  000003                   ADD      #3,R3                 ;R3 DESIGNATES LAST BYTE
 9464 032722  112722  000360          8$:      MOVB     #360,(R2)+            ;TRY WRITING A BYTE
 9465 032726  012701  002362                   MOV      #TESTADD,R1
 9466 032732  017100  000000                   MOV      @(R1),R0
 9467 032736  023700  002240                   CMP      TSTDAT,R0             ;CHECK FOR UNCHANGED DATA
 9468 032742  001404                           BEQ      9$                    ;BR IF OK
 9469 032744  017137  000000  002032           MOV      @(R1),ADDRESS
 9470 032752  104455                           PERR31
 9471
 9472 032754  017100  000002          9$:      MOV      @2(R1),R0
 9473 032760  023700  002242                   CMP      TSTDAT+2,R0           ;READ SECOND WORD
 9474 032764  001404                           BEQ      10$                   ;BR IF UNCHANGED
 9475 032766  017137  000002  002032           MOV      @2(R1),ADDRESS
 9476 032774  104455                           PERR31
 9477
```

```
9478 032776  020203                    10$:    CMP     R2,R3           ;TESTED LAST BYTE ?
9479 033000  001350                            BNE     8$              ;BR IF NO
9480 033002  105737  002256             11$:    TSTB    PASFLG
9481 033006  100452                            BMI     15$             ;BRANCH IF TESTING CHECK BITS
9482 033010  005737  002252                    TST     DBEMSK+2        ;CHECKING FOR LAST ERROR BIT
9483 033014  100405                            BMI     12$             ;BR IF DONE HERE
9484 033016                                    DLEFT   DBEMSK
9485 033026  000643                            BR      4$
9486 033030                             12$:    IF #SW11 SET.IN @SWR THEN GOTO 13$
9487 033040                                     IF QVFLAG IS TRUE THEN GOTO 13$
9488 033046  005737  002246                    TST     SBEMSK+2        ;LAST SBE MASK
9489 033052  100405                            BMI     13$             ;BR IF DONE WITH THIS PASS
9490 033054                                    DLEFT   SBEMSK
9491 033064  000617                            BR      3$
9492 033066  105737  002256             13$:    TSTB    PASFLG  ;TEST PASS FLAG
9493 033072  001003                            BNE     14$             ;NON ZERO MEANS WE'RE DONE
9494 033074  105237  002256                    INCB    PASFLG  ;NOT DONR
9495 033100  000600                            BR      1$
9496 033102  052737  000200  002256  14$:     BIS     #BIT7,PASFLG
9497 033110  005037  002240                    CLR     TSTDAT
9498 033114  005037  002242                    CLR     TSTDAT+2
9499 033120  012737  000040  002244            MOV     #40,SBEMSK
9500 033126  012737  000100  002250            MOV     #100,DBEMSK
9501 033134  012737  003740  002274  15$:     MOV     #3740,CHECK
9502 033142  013702  002244                    MOV     SBEMSK,R2
9503 033146  074237  002274                    XOR     R2,CHECK
9504 033152  013702  002250                    MOV     DBEMSK,R2
9505 033156  074237  002274                    XOR     R2,CHECK
9506 033162  006337  002250                    ASL     DBEMSK
9507 033166  032737  020000  002250            BIT     #BIT13,DBEMSK
9508 033174  001633                            BEQ     7$
9509 033176  006337  002244                    ASL     SBEMSK
9510 033202  032737  004000  002244            BIT     #BIT11,SBEMSK
9511 033210  001006                            BNE     16$
9512 033212  013737  002244  002250            MOV     SBEMSK,DBEMSK
9513 033220  006337  002250                    ASL     DBEMSK
9514 033224  000743                            BR      15$
9515 033226  104471                    16$:    ECC1DIS                 ;DISABLE ECC ON 1 SELECTED CSR
9516 033230  012701  002362                    MOV     #TESTADD,R1     ;TEST LOCATION
9517 033234                                    CLEAR   @(R1)+,@(R1)    ;TO ERASE ANY DBE'S FROM TESTING
9518                                           ;RESTORE CSR
9519 033242  104503                            CLR1CSR                 ;CLEAR 1 SELECTED CSR
9520 033244  000207                            RETURN
```

```
                                        MTP016: SUBTST  <<MTP016        WRITE INHIBIT OF WORD WITH DBE>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       MTP016  WRITE INHIBIT OF WORD WITH DBE
                                        ;*****************************************************************************
                                                ;DOUBLE BIT ERROR WRITE CANCEL WITH
                                                ;WORD WRITE.
                                                ;CHECKS WRITE INHIBIT WITH WORD WRITES TO
                                                ;WORD WITH DOUBLE ERROR.
033246  005037  002234          T12A:   CLR     DATBUF          ;BACKGROUND FOR DOUBLE ERRORS
033252  005037  002236                  CLR     DATBUF+2        ;2 WORDS WORTH
033256  012737  000001  002244          MOV     #1,SBEMSK       ;SINGLE ERROR MASK
033264  005037  002246                  CLR     SBEMSK+2        ;
033270  012737  000001  002250  T12B:   MOV     #1,DBEMSK       ;DOUBLE ERROR MASK
033276  005037  002252                  CLR     DBEMSK+2
033302  013737  002234  002240  1$:     MOV     DATBUF,TSTDAT   ;DATA FOR TEST
033310  013737  002236  002242          MOV     DATBUF+2,TSTDAT+2;BOTH WORDS
033316  105737  002256                  TSTB    PASFLG  ;COMP DATA ON SECOND PASS ONLY
033322  001404                          BEQ     2$              ;BR IF FIRST PASS
033324  005137  002240                  COM     TSTDAT          ;COMP FIRST WORD
033330  005137  002242                  COM     TSTDAT+2        ;NOW SECOND WORD
033334  023737  002244  002250  2$:     CMP     SBEMSK,DBEMSK   ;CHECK FOR IDENTICAL MASKS
033342  001004                          BNE     3$              ;BR IF DIFFERENT
033344  023737  002246  002252          CMP     SBEMSK+2,DBEMSK+2;UPPER WORD TOO
033352  001502                          BEQ     8$              ;BR TO MAKE THEM NOT EQUAL
033354  012737  002240  002272  3$:     MOV     #TSTDAT,SOURCE  ;NEED ADDR OF DATA FOR CHKGEN
033362  004737  041722                  CALL    CHKGEN          ;GO GENERATE CHECK BITS
033366  013701  002244                  MOV     SBEMSK,R1
033372  074137  002240                  XOR     R1,TSTDAT
033376  013701  002246                  MOV     SBEMSK+2,R1
033402  074137  002242                  XOR     R1,TSTDAT+2
033406  013701  002250                  MOV     DBEMSK,R1
033412  074137  002240                  XOR     R1,TSTDAT
033416  013701  002252                  MOV     DBEMSK+2,R1
033422  074137  002242                  XOR     R1,TSTDAT+2
033426  012701  002362          4$:     MOV     #TESTADD,R1     ;FIRST TEST ADRRESS
033432  104471                          ECC1DIS                 ;DISABLE ECC ON 1 SELECTED CSR
033434  013731  002240                  MOV     TSTDAT,@(R1)+   ;WRITE FIRST 16 BITS
033440  104475                          CB1CSR                  ;WRITE GENERATED CHECKBITS IN 1 SELECTED CSR
033442  013771  002242  000000          MOV     TSTDAT+2,@(R1)  ;WRITE SECOND 16 BITS + CHECKBITS
033450  105037  002257                  CLRB    UPPFLG          ;SET FOR 2 LOOPS
033454  162701  000002                  SUB     #2,R1           ;POINT TO LOW WORD
033460  104503                  5$:     CLR1CSR                 ;CLEAR 1 SELECTED CSR
033462  012771  177400  000000          MOV     #177400,@(R1)   ;TRY WRITING LOCATION
033470  012701  002362                  MOV     #TESTADD,R1
033474  017100  000000                  MOV     @(R1),R0
033500  023700  002240                  CMP     TSTDAT,R0       ;CHECK FOR ORIGINAL DATA
033504  001404                          BEQ     6$              ;SHOULD BE UNCHANGED
033506  017137  000000  002032          MOV     @(R1),ADDRESS
033514  104455                          PERR31
                                        
033516  062701  000002          6$:     ADD     #2,R1
033522  017100  000000                  MOV     @(R1),R0
033526  023700  002242                  CMP     TSTDAT+2,R0     ;THIS SHOULD BE UNCHANGED ALSO
033532  001404                          BEQ     7$
033534  017137  000000  002032          MOV     @(R1),ADDRESS
033542  104455                          PERR31
```

```
9578 033544  105737  002257          7$:   TSTB    UPPFLG          ;WHICH LOOP ?
9579 033550  001003                        BNE     8$              ;SECOND, BR OUT
9580 033552  105237  002257                INCB    UPPFLG          ;FIRST, KEEP GOING
9581 033556  000740                        BR      5$
9582 033560  105737  002256          8$:   TSTB    PASFLG
9583 033564  100454                        BMI     12$
9584 033566  005737  002252                TST     DBEMSK+2        ;LAST BIT ?
9585 033572  100405                        BMI     9$              ;MINUS = BIT 31
9586 033574                                DLEFT   DBEMSK
9587 033604  000636                        BR      1$
9588 033606                        9$:     IF #SW11 SET.IN @SWR THEN GOTO 10$
9589 033616                                IF QVFLAG IS TRUE THEN GOTO 10$
9590 033624  005737  002246                TST     SBEMSK+2        ;LAST BIT IN THIS MASK ?
9591 033630  100406                        BMI     10$             ;BR IF LAST BIT
9592 033632                                DLEFT   SBEMSK
9593 033642  000137  033270                JMP     T12B
9594 033646  105737  002256          10$:  TSTB    PASFLG  ;FIRST PASS ?
9595 033652  001004                        BNE     11$             ;BR IF SECOND
9596 033654  105237  002256                INCB    PASFLG  ;INDICATE SECOND PASS COMING
9597 033660  000137  033246                JMP     T12A
9598 033664  052737  000200  002256  11$:  BIS     #BIT7,PASFLG
9599 033672  005037  002240                CLR     TSTDAT
9600 033676  005037  002242                CLR     TSTDAT+2
9601 033702  012737  000040  002244        MOV     #40,SBEMSK
9602 033710  012737  000100  002250        MOV     #100,DBEMSK
9603 033716  012737  003740  002274  12$:  MOV     #3740,CHECK
9604 033724  013702  002244                MOV     SBEMSK,R2
9605 033730  074237  002274                XOR     R2,CHECK
9606 033734  013702  002250                MOV     DBEMSK,R2
9607 033740  074237  002274                XOR     R2,CHECK
9608 033744  006337  002250                ASL     DBEMSK
9609 033750  032737  020000  002250        BIT     #BIT13,DBEMSK
9610 033756  001623                        BEQ     4$
9611 033760  006337  002244                ASL     SBEMSK
9612 033764  032737  004000  002244        BIT     #BIT11,SBEMSK
9613 033772  001006                        BNE     13$
9614 033774  013737  002244  002250        MOV     SBEMSK,DBEMSK
9615 034002  006337  002250                ASL     DBEMSK
9616 034006  000743                        BR      12$
9617 034010  104471                13$:    ECC1DIS                 ;DISABLE ECC ON 1 SELECTED CSR
9618 034012  012701  002362                MOV     #TESTADD,R1     ;RESTORE TEST ADDRESS
9619 034016  005031                        CLR     @(R1)+          ;CLEAR ANY DBE'S FROM TEST
9620 034020  005071  000000                CLR     @(R1)
9621 034024  104503                        CLR1CSR                 ;CLEAR 1 SELECTED MK11 CSR
9622 034026  000207                        RETURN
```

```
9625 034030                          MTP017: SUBTST  <<MTP017        HOLDING 1'S & 0'S TEST>>
                                     ;*********************************************************************
                                     ;*SUBTEST       MTP017  HOLDING 1'S & 0'S TEST
                                     ;*********************************************************************
9626                                         ;*(1)   THIS TEST CHECKS THE MEMORY FOR THE CAPABILITY
9627                                         ;*      OF HOLDING 1'S AND 0'S BY WRITING A BACKGROUND
9628                                         ;*      OF 000377 AND READING IT
9629                                         ;*(2)   MEMORY IS WRITTEN USING A BYTE AT A TIME
9630                                         ;*(3)   STEPS 1 & 2 ARE REPEATED WITH A SWAPPED BACKGROUND PATTERN
9631                                         ;NOTE:  THIS TEST WRITES BYTES & READS WORDS
9632 034030  012701  060000                 MOV     #FIRST,R1
9633 034034  010104                         MOV     R1,R4
9634 034036  012705  160000                 MOV     #LAST+2,R5
9635 034042  012700  000377                 MOV     #377,R0         ;GET THE PATTERN INTO R0
9636 034046  010003                         MOV     R0,R3
9637 034050  000303                         SWAB    R3
9638 034052  110021                 1$:     MOVB    R0,(R1)+        ;WRITE A BYTE
9639 034054  110321                         MOVB    R3,(R1)+        ;WRITE THE MEMORY WITH THE BYTE STORED IN BAKPAT+1
9640 034056  020105                         CMP     R1,R5           ;COMPARE TEST LOC TO TOP + 2
9641 034060  103774                         BLO     1$              ;BRANCH IF LOWER
9642
9643 034062  014102                 2$:     MOV     -(R1),R2
9644 034064  020002                         CMP     R0,R2           ;TEST THE MEMORY TO SEE IF IT CONTAINS
9645                                                                 ;THE WORD STORED IN BAKPAT
9646 034066  001401                         BEQ     3$
9647 034070  104446                         PERR22
9648
9649 034072  020104                 3$:     CMP     R1,R4           ;KEEP ON TESTING THE MEMORY UNTIL
9650 034074  101372                         BHI     2$              ;R1 EQUALS THE LOWEST ADDRESS
9651 034076  000303                         SWAB    R3              ;CHANGE THE DATA PATTERN
9652 034100  000300                         SWAB    R0
9653 034102  001763                         BEQ     1$              ;IF THE DATA PATTERN DOES NOT HAVE LOW
9654                                                                 ; BYTE =0 THEN FALL THRU
9655 034104  000207                         RETURN
```

```
 9658 034106                    MTP020: SUBTST  <<MTP020        MARCHING 1'S & 0'S IN CHECK BITS TEST>>
                                ;*****************************************************************************
                                ;*SUBTEST        MTP020  MARCHING 1'S & 0'S IN CHECK BITS TEST
                                ;*****************************************************************************
 9659                                   ;*THIS TEST IS CONCERNED ONLY WITH THE INTEGRITY
 9660                                   ;*OF THE MOS RAMS THAT STORE THE CHECKBITS.
 9661
 9662                                   ;077 --> 100 DOWN
 9663 034106  160201            MTPA20: SUB     R2,R1           ;V177640
 9664 034110  160204                    SUB     R2,R4           ;V177642
 9665 034112  005711                    TST     (R1)            ;V177644        ;1ST WORD OK?
 9666 034114  001002                    BNE     1$              ;V177646        ;NO - SKIP
 9667 034116  005714                    TST     (R4)            ;V177650        ;2ND WORD OK?
 9668 034120  001401                    BEQ     2$              ;V177652        ;YES - SKIP
 9669 034122  104453            1$:     PERR27                  ;V177654        ;GOOD=000000,,000000,,077
 9670 034124  010314            2$:     MOV     R3,(R4)         ;V177656        ;2ND WORD <= 100000
 9671 034126  005011                    CLR     (R1)            ;V177660        ;CLEAR 1ST WORD
 9672 034130  020100                    CMP     R1,R0           ;V177662        ;ARE WE DONE?
 9673 034132  101365                    BHI     MTPA20          ;V177664        ;BRANCH IF NOT
 9674 034134  000207                    RETURN                  ;V177666
 9675
 9676                                   ;100 --> 077 UP
 9677 034136  005711            MTPB20: TST     (R1)            ;V177640        ;1ST WORD OK?
 9678 034140  001002                    BNE     3$              ;V177642        ;NO - SKIP
 9679 034142  020314                    CMP     R3,(R4)         ;V177644        ;2ND WORD OK?
 9680 034144  001401                    BEQ     4$              ;V177646        ;YES - SKIP
 9681 034146  104452            3$:     PERR26                  ;V177650        ;GOOD=000000,,100000,,100
 9682 034150  005014            4$:     CLR     (R4)            ;V177652        ;CLEAR 2ND WORD
 9683 034152  005011                    CLR     (R1)            ;V177654        ;CLEAR 1ST WORD
 9684 034154  060201                    ADD     R2,R1           ;V177656
 9685 034156  060204                    ADD     R2,R4           ;V177660
 9686 034160  020405                    CMP     R4,R5           ;V177662        ;TOP + 2 YET?
 9687 034162  001365                    BNE     MTPB20          ;V177664        ;NO - LOOP
 9688 034164  000207                    RETURN                  ;V177666
 9689
 9690                                   ;077 --> 100 UP
 9691 034166  005711            MTPC20: TST     (R1)            ;V177640        ;1ST WORD OK?
 9692 034170  001002                    BNE     5$              ;V177642        ;NO - SKIP
 9693 034172  005714                    TST     (R4)            ;V177644        ;2ND WORD OK?
 9694 034174  001401                    BEQ     6$              ;V177646        ;YES - SKIP
 9695 034176  104453            5$:     PERR27                  ;V177650        ;GOOD=000000,,000000,,077
 9696 034200  010314            6$:     MOV     R3,(R4)         ;V177652        ;WRITE 1ST WORD
 9697 034202  005011                    CLR     (R1)            ;V177654        ;WRITE 2ND WORD
 9698 034204  060204                    ADD     R2,R4           ;V177656
 9699 034206  060201                    ADD     R2,R1           ;V177660
 9700 034210  020405                    CMP     R4,R5           ;V177662        ;TOP + 2 YET?
 9701 034212  001365                    BNE     MTPC20          ;V177664        ;NO - LOOP
 9702 034214  000207                    RETURN                  ;V177666
```

```
 9705                                          ;100 --> 077 DOWN
 9706 034216  160201                MTPD20: SUB     R2,R1           ;V177640
 9707 034220  160204                        SUB     R2,R4           ;V177642
 9708 034222  020314                        CMP     R3,(R4)         ;V177644        ;2ND WORD OK?
 9709 034224  001002                        BNE     7$              ;V177646        ;NO - SKIP
 9710 034226  005711                        TST     (R1)            ;V177650        ;1ST WORD OK?
 9711 034230  001401                        BEQ     8$              ;V177652        ;YES - SKIP
 9712 034232  104452                7$:     PERR26                  ;V177654        ;GOOD=000000,,100000,,100
 9713 034234  005014                8$:     CLR     (R4)            ;V177656        ;WRITE 1ST WORD
 9714 034236  005011                        CLR     (R1)            ;V177660        ;WRITE 2ND WORD
 9715 034240  020100                        CMP     R1,R0           ;V177662
 9716 034242  101365                        BHI     MTPD20          ;V177664
 9717 034244  000207                        RETURN                  ;V177666
 9718
 9719                                          ;077 UP
 9720 034246  005711                MTPE20: TST     (R1)            ;V177640        ;1ST WORD OK?
 9721 034250  001002                        BNE     9$              ;V177642        ;NO - SKIP
 9722 034252  005714                        TST     (R4)            ;V177644        ;2ND WORD OK?
 9723 034254  001401                        BEQ     10$             ;V177646        ;YES - SKIP
 9724 034256  104453                9$:     PERR27                  ;V177650        ;GOOD=000000,,000000,,077
 9725 034260  060201                10$:    ADD     R2,R1           ;V177652
 9726 034262  060204                        ADD     R2,R4           ;V177654
 9727 034264  020405                        CMP     R4,R5           ;V177656        ;TOP + 2 YET?
 9728 034266  001367                        BNE     MTPE20          ;V177660        ;NO - LOOP
 9729 034270  000207                        RETURN                  ;V177662
```

```
 9732 034272                    MTPA21: SUBTST  <<MTPA21        MARCHING 1'S & 0'S PATTERN TEST>>
                                ;*****************************************************************************
                                ;*SUBTEST       MTPA21  MARCHING 1'S & 0'S PATTERN TEST
                                ;*****************************************************************************
 9733                                   ;READ,BYTESWAP-MODIFY,READ,DOWN
 9734 034272  014100            1$:     MOV     -(R1),R0;V177640
 9735 034274  020200                    CMP     R2,R0   ;V177642
 9736 034276  001401                    BEQ     2$      ;V177644
 9737 034300  104443                    PERR17          ;V177646
 9738
 9739 034302  000311            2$:     SWAB    (R1)    ;V177650
 9740 034304  011100                    MOV     (R1),R0 ;V177652
 9741 034306  020300                    CMP     R3,R0   ;V177654
 9742 034310  001401                    BEQ     3$      ;V177656
 9743 034312  104444                    PERR20          ;V177660
 9744
 9745 034314  020401            3$:     CMP     R4,R1   ;V177662        ;DONE?
 9746 034316  001365                    BNE     1$      ;V177664        ;NO - LOOP
 9747 034320  000207                    RETURN          ;V177666        ;YES - RETURN
 9748
 9749 034322                    MTPB21: ;READ,BYTESWAP-MODIFY,READ,UP
 9750 034322  011100            1$:     MOV     (R1),R0 ;V177640
 9751 034324  020300                    CMP     R3,R0   ;V177642
 9752 034326  001401                    BEQ     2$      ;V177644
 9753 034330  104444                    PERR20          ;V177646
 9754
 9755 034332  000311            2$:     SWAB    (R1)    ;V177650
 9756 034334  011100                    MOV     (R1),R0 ;V177652
 9757 034336  020200                    CMP     R2,R0   ;V177654
 9758 034340  001401                    BEQ     3$      ;V177656
 9759 034342  104443                    PERR17          ;V177660
 9760
 9761 034344  062701  000002    3$:     ADD     #2,R1   ;V177662
 9762 034350  020501                    CMP     R5,R1   ;V177666        ;DONE?
 9763 034352  001363                    BNE     1$      ;V177670        ;NO - LOOP
 9764 034354  000207                    RETURN          ;V177672        ;YES - RETURN
 9765
 9766 034356                    MTPC21: ;READ,BYTESWAP-MODIFY,READ,UP
 9767 034356  011100            1$:     MOV     (R1),R0 ;V177640
 9768 034360  020200                    CMP     R2,R0   ;V177642
 9769 034362  001401                    BEQ     2$      ;V177644
 9770 034364  104443                    PERR17          ;V177646
 9771
 9772 034366  000311            2$:     SWAB    (R1)    ;V177650
 9773 034370  011100                    MOV     (R1),R0 ;V177652
 9774 034372  020300                    CMP     R3,R0   ;V177654
 9775 034374  001401                    BEQ     3$      ;V177656
 9776 034376  104444                    PERR20          ;V177660
 9777
 9778 034400  062701  000002    3$:     ADD     #2,R1   ;V177662
 9779 034404  020501                    CMP     R5,R1   ;V177666        ;DONE?
 9780 034406  001363                    BNE     1$      ;V177670        ;NO - LOOP
 9781 034410  000207                    RETURN          ;V177672        ;YES - RETURN
```

```
 9784 034412                    MTPD21: ;READ,BYTESWAP-MODIFY,READ,DOWN
 9785 034412  014100            1$:     MOV     -(R1),R0;V177640
 9786 034414  020300                    CMP     R3,R0   ;V177642
 9787 034416  001401                    BEQ     2$      ;V177644
 9788 034420  104444                    PERR20          ;V177646
 9789
 9790 034422  000311            2$:     SWAB    (R1)    ;V177650
 9791 034424  011100                    MOV     (R1),R0 ;V177652
 9792 034426  020200                    CMP     R2,R0   ;V177654
 9793 034430  001401                    BEQ     3$      ;V177656
 9794 034432  104443                    PERR17          ;V177660
 9795
 9796 034434  020401            3$:     CMP     R4,R1   ;V177662        ;DONE?
 9797 034436  001365                    BNE     1$      ;V177664        ;NO - LOOP
 9798 034440  000207                    RETURN          ;V177666        ;YES - RETURN
 9799
```

```
 9802 034442                            MTP022: SUBTST  <<MTP022        REFRESH & SHIFTING DIAGONAL TEST>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       MTP022  REFRESH & SHIFTING DIAGONAL TEST
                                        ;*****************************************************************************
 9803                                           ;(1)    WE WRITE A DIAGONAL PATTERN IN MEMORY (WITH CACHE ON).
 9804                                           ;(2)    IF A REFRESH TEST WE DISTURB ALL ROWS FOR > 2 MS (WITH CACHE ON).
 9805                                           ;(3)    WE READ & CHECK FOR CORRECTNESS THE DIAGONAL PATTERN
 9806                                           ;       (WITH CACHE OFF).
 9807          000010                   KDIAG=8.                ;HOW OFTEN A DIAGONAL STRIPE OCCURS (MUST BE A POWER OF 2)
 9808 034442                                    FOR EVEN := #1 TO #2                    ;FOR DATA & COMPLEMENT DATA
 9809 034450                                      IF EVEN EQ #1
 9810 034460                                        LET R2 := ZEROS
 9811 034464                                        LET R3 := ONES
 9812 034470                                      ELSE
 9813 034472                                        LET R2 := ONES
 9814 034476                                        LET R3 := ZEROS
 9815 034502                                      END ;OF IF EVEN
 9816 034502                                      FOR STRIPES := #0 TO #KDIAG-1         ;FOR THE NUMBER OF STRIPES
 9817
 9818                                               ;WRITE LOOP
 9819 034506  104423                               CACHON                        ;TURN CACHE ON
 9820 034510                                       LET COUNT := STRIPES
 9821 034516                                       LET R1 := #FIRST
 9822 034522                                       WHILE R1 LOS #LAST
 9823 034530                                         IF COUNT LT #0 THEN LET COUNT := #KDIAG-1
 9824 034544                                         IF #374 OFF.IN R1 THEN LET COUNT := COUNT - #1
 9825 034556                                         IF COUNT NE #0
 9826 034564                                           LET (R1) := R2
 9827 034566                                           LET 2(R1) := R2
 9828 034572                                         ELSE
 9829 034574                                           LET (R1) := R3
 9830 034576                                           LET 2(R1) := R3
 9831 034602                                         END ;OF IF COUNT
 9832 034602                                         LET COUNT := COUNT - #1
 9833 034606                                         LET R1 := R1 + #4
 9834 034612                                       END ;OF WHILE
 9835                                              ;END OF WRITE LOOP
 9836
 9837 034614                                       IF DIAGFLAG IS FALSE THEN $CALL REFRESH
 9838                                              ;READ LOOP
 9839 034626                                       LET COUNT := STRIPES
 9840 034634                                       LET R1 := #FIRST
 9841 034640  104424                               CACHOFF                       ;TURN CACHE OFF
```

```
9843 034642                                     WHILE R1 LOS #LAST
9844 034650                                       IF COUNT LT #0 THEN LET COUNT := #KDIAG-1
9845 034664                                       IF #374 OFF.IN R1 THEN LET COUNT := COUNT - #1
9846 034676                                       IF COUNT NE #0
9847 034704                                         LET R0 := (R1)
9848 034706                                         IF R2 NE R0
9849 034712  104443                                   PERR17
9850 034714                                         END ;OF IF R2
9851 034714                                         LET R0 := 2(R1)
9852 034720                                         IF R2 NE R0
9853 034724  104443                                   PERR17
9854 034726                                         END ;OF IF R2
9855 034726                                       ELSE
9856 034730                                         LET R0 := (R1)
9857 034732                                         IF R3 NE R0
9858 034736  104444                                   PERR20
9859 034740                                         END ;OF IF R3
9860 034740                                         LET R0 := 2(R1)
9861 034744                                         IF R3 NE R0
9862 034750  104444                                   PERR20
9863 034752                                         END ;OF IF R3
9864 034752                                       END ;OF IF COUNT
9865 034752                                       LET COUNT := COUNT - #1
9866 034756                                       LET R1 := R1 + #4
9867 034762                                     END ;OF WHILE
9868                                             ;END OF READ LOOP
9869
9870 034764                                   END ;OF FOR STRIPES
9871 035000                                 END ;OF FOR EVEN
9872 035014  000207                         RETURN
9873
9874 035016                           REFRESH:SUBTST  <<SUBR  REFRESH DELAY>>
                                      ;*****************************************************************************
                                      ;*SUBTEST       SUBR    REFRESH DELAY
                                      ;*****************************************************************************
9875                                          ;DISTURB EACH ROW FOR > 3.2 MS
9876 035016                                   FOR R0 := #FIRST TO #FIRST+374 BY #4
9877 035022  004737  035066                     CALL REFSUB
9878 035026                                   END ;OF FOR R0
9879 035040                                   LET R0 := #FIRST+BIT14
9880 035044                                   WHILE R0 LOS #LAST+BIT14+374
9881 035052  004737  035066                     CALL REFSUB
9882 035056                                     LET R0 := R0 + #4
9883 035062                                   END ;OF WHILE
9884 035064  000207                           RETURN
9885 035066  012704  000640           REFSUB: MOV     #640,R4                 ;TIME FOR A > 3.2 MS LOOP
9886 035072  062700  000002                   ADD     #2,R0
9887 035076  005140                   1$:     COM     -(R0)
9888 035100  005120                           COM     (R0)+
9889 035102  005110                           COM     (R0)
9890 035104  005110                           COM     (R0)
9891 035106  077405                           SOB     R4,1$
9892 035110  162700  000002                   SUB     #2,R0
9893 035114  000207                           RETURN
```

```
 9897 035116                    MTPA24: SUBTST  <<MTPA24        FAST GALLOPING PATTERN TEST>>
                                ;*****************************************************************************
                                ;*SUBTEST       MTPA24  FAST GALLOPING PATTERN TEST
                                ;*****************************************************************************
 9898                                   ;THE TOTAL TEST (INCLUDING SETUP) IS AS FOLLOWS
 9899                                   ;*(1)   THIS TEST WRITES THE MEMORY WITH A BACK GROUND PATTERN
 9900                                   ;*      STORED AT LOCATION BAKPAT
 9901                                   ;*(2)   TEST BEGINS AT LOWEST LOCATION BEING TESTED
 9902                                   ;*      (LETS NAME IT 'A')
 9903                                   ;*(3)   LETS NAME THE 1ST LOCATION IN THE ROW/COLUMN UNDER TEST AS 'B'.
 9904                                   ;*(4)   SWAPS BYTES FOR LOCATION 'A'.
 9905                                   ;*(5)   READS 'A', READS 'B'
 9906                                   ;*(6)   'B' = 'B'+400   (ADDS 64 DOUBLE WORDS TO 'B')
 9907                                   ;*(7)   REPEATS STEPS 5 AND 6 UNTIL 'B' IS GREATER THAN THE
 9908                                   ;*(8)   END OF THE BANK.  A+2
 9909                                   ;*(9)   REPEATS  STEPS 3-8 UNTILL 'A' REACHES THE END OF THE BANK
 9910                                   ;*(10)  AFTER EXECUTING THE TEST  DATA IS COMPLEMENTED
 9911                                   ;*      AND STEPS 1-9 ARE REPEATED
 9912                                   ;REGISTERS ARE USED AS FOLLOWS
 9913                                   ;R0     TEST DATA
 9914                                   ;R1     'A'
 9915                                   ;R2     'B'
 9916                                   ;R3     BAKPAT
 9917                                   ;R4     SWAPAT
 9918                                   ;R5     LAST
 9919
 9920                                   ;NOTE THE PATTERN STARTS AT MTPB24!!!!!!!!!!!!!!!
 9921
 9922                                   ;UIPAR'S
 9923 035116  011100            1$:     MOV     (R1),R0         ;V177640        ;READ 'A'
 9924 035120  020004                    CMP     R0,R4           ;V177642        ;CHECK 'A'
 9925 035122  001401                    BEQ     2$              ;V177644        ;BR IF OK
 9926 035124  104447                    PERR23                  ;V177646        ;REPORT ERROR
 9927
 9928 035126  011200            2$:     MOV     (R2),R0         ;V177650        ;READ 'B'
 9929 035130  020003                    CMP     R0,R3           ;V177652        ;CHECK 'B'
 9930 035132  001401                    BEQ     3$              ;V177654        ;BR IF OK
 9931 035134  104450                    PERR24                  ;V177656        ;REPORT ERROR
 9932
 9933 035136  062702  000400    3$:     ADD     #400,R2         ;V177660        ;BUMP 'B'
 9934 035142  020205                    CMP     R2,R5           ;V177664        ;AT END YET?
 9935 035144  101764                    BLOS    1$              ;V177666        ;BR IF NO
 9936
 9937 035146  062701  000002            ADD     #2,R1           ;V177670        ;BUMP 'A'
 9938 035152  000137  035156            JMP     @#MTPB24        ;V177674        ;GOTO V177260
```

```
 9941 035156                          MTPB24: SUBTST  <<MTPB24        FAST GALLOP PART B>>
                                      ;*****************************************************************************
                                      ;*SUBTEST       MTPB24  FAST GALLOP PART B
                                      ;*****************************************************************************
 9942                                         ;SDPAR'S
 9943 035156  010411                          MOV     R4,(R1)         ;V172260        ;WRITE 'A'
 9944 035160  020105                          CMP     R1,R5           ;V172262        ;DONE?
 9945 035162  001001                          BNE     1$              ;V172264        ;BR IF NO
 9946 035164  000207                          RETURN                  ;V172266        ;YES - RETURN
 9947 035166  000137  035172     1$:          JMP     @#MTPC24        ;V172270        ;GOTO V172360
 9948
 9949 035172                          MTPC24: SUBTST  <<MTPC24        FAST GALLOP PART C>>
                                      ;*****************************************************************************
                                      ;*SUBTEST       MTPC24  FAST GALLOP PART C
                                      ;*****************************************************************************
 9950                                         ;KDPAR'S
 9951 035172  010102                          MOV     R1,R2           ;V172360        ;RESET 'B' <--- 'A'
 9952 035174  011100                          MOV     (R1),R0         ;V172362        ;READ 'A'
 9953 035176  020004                          CMP     R0,R4           ;V172364        ;CHECK 'A'
 9954 035200  001401                          BEQ     1$              ;V172366        ;BR IF OK
 9955 035202  104447                          PERR23                  ;V172370        ;REPORT ERROR
 9956 035204  000137  035136     1$:          JMP     @#MTPA24+20     ;V172372        ;GOTO V177660
```

```
 9959 035210                                MTP025: SUBTST  <<MTP025        INTERRUPT ENABLE TEST>>
                                            ;*****************************************************************************
                                            ;*SUBTEST       MTP025  INTERRUPT ENABLE TEST
                                            ;*****************************************************************************
 9960 035210  005037  002240                        CLR     TSTDAT          ;GENERATE CHECKBITS ON 0,,0
 9961 035214  005037  002242                        CLR     TSTDAT+2
 9962 035220  012737  002240  002272                MOV     #TSTDAT,SOURCE
 9963 035226  004737  041722                        CALL    CHKGEN
 9964 035232  012737  000003  002074                MOV     #3,NOPAR                ;SETUP PARITY ACTION
 9965 035240  012701  002362                        MOV     #TESTADD,R1     ;FIRST TEST ADDRESS
 9966 035244  012737  035304  002264                MOV     #1$,PARTHERE    ;SETUP TRAP DESTINATION
 9967 035252  004737  035526                        CALL    MTPA25          ;WRITE DATA & CHECKBITS
 9968 035256  104473                                ECC1INIT                ;INITIALIZE 1 SELECTED MK11 CSR
 9969 035260  005771  000000                        TST     @(R1)           ;ACCESS LOCATIONS FOR DBE TRAPS
 9970 035264  005771  000002                        TST     @2(R1)
 9971                                               ;NONE - GOOD - ACCESS FOR SBE TRAPS
 9972 035270  104507                                ENA1SBE                 ;DISABLE TRAPS ON SBE'S FROM 1 SELECTED CSR
 9973 035272  005771  000000                        TST     @(R1)
 9974 035276  005771  000002                        TST     @2(R1)
 9975 035302  000404                                BR      2$              ;NONE - GOOD - SKIP
 9976 035304  104426                        1$:     READCSR
 9977 035306                                        FATAL   27
 9978 035314  005237  002240                2$:     INC     TSTDAT          ;CHECK FOR CORRECT ACTION ON SBE'S
 9979 035320  004737  035454                        CALL    MTPD25          ;IN ALL 4 BYTES
 9980 035324  012737  000400  002240                MOV     #400,TSTDAT
 9981 035332  004737  035454                        CALL    MTPD25
 9982 035336  005037  002240                        CLR     TSTDAT
 9983 035342  005237  002242                        INC     TSTDAT+2
 9984 035346  004737  035454                        CALL    MTPD25
 9985 035352  012737  000400  002242                MOV     #400,TSTDAT+2
 9986 035360  004737  035454                        CALL    MTPD25
 9987
 9988 035364  005037  002242                        CLR     TSTDAT+2        ;CHECK FOR CORRECT ACTION ON DBE'S
 9989 035370  012737  000003  002240                MOV     #3,TSTDAT       ;IN ALL 4 BYTES
 9990 035376  004737  035476                        CALL    MTPE25
 9991 035402  012737  001400  002240                MOV     #1400,TSTDAT
 9992 035410  004737  035476                        CALL    MTPE25
 9993 035414  005037  002240                        CLR     TSTDAT
 9994 035420  012737  000003  002242                MOV     #3,TSTDAT+2
 9995 035426  004737  035476                        CALL    MTPE25
 9996 035432  012737  001400  002242                MOV     #1400,TSTDAT+2
 9997 035440  004737  035476                        CALL    MTPE25
 9998 035444  104503                                CLR1CSR                 ;CLEAR 1 SELECTED MK11 CSR
 9999 035446  005037  002074                        CLR     NOPAR           ;INDICATE PARITY ACTION
10000 035452  000207                                RETURN
10001
10002 035454  004737  035526          MTPD25:       CALL    MTPA25          ;WRITE DATA & CHECKBITS
10003 035460  104471                                ECC1DIS                 ;DISABLE ECC ON 1 SELECTED CSR
10004 035462  004737  035550                        CALL    MTPB25          ;CHECK FOR NO TRAPS
10005 035466  104507                                ENA1SBE                 ;DISABLE TRAPS ON SBE'S FROM 1 SELECTED CSR
10006 035470  004737  035610                        CALL    MTPC25          ;CHECK FOR EXPECTED TRAP
10007 035474  000207                                RETURN
```

```
10010 035476  004737  035526          MTPE25: CALL    MTPA25          ;WRITE DATA & CHECKBITS
10011 035502  104471                          ECC1DIS                 ;DISABLE ECC ON 1 SELECTED CSR
10012 035504  004737  035550                  CALL    MTPB25          ;CHECK FOR NO TRAPS
10013                                         ;ENABLE DBE TRAPS
10014 035510  104473                          ECC1INIT                ;INITIALIZE 1 SELECTED MK11 CSR
10015 035512  004737  035610                  CALL    MTPC25          ;CHECK FOR EXPECTED TRAP
10016 035516  104507                          ENA1SBE                 ;DISABLE TRAPS ON SBE'S FROM 1 SELECTED CSR
10017 035520  004737  035610                  CALL    MTPC25          ;CHECK FOR EXPECTED TRAP
10018 035524  000207                          RETURN
10019
10020                                         ;WRITE TSTDAT & TSTDAT+2 & CHECKBITS
10021 035526  104471                  MTPA25: ECC1DIS                 ;DISABLE ECC ON 1 SELECTED CSR
10022 035530  013771  002240  000000          MOV     TSTDAT,@(R1)    ;WRITE FIRST 16 BITS
10023 035536  104475                          CB1CSR                  ;WRITE GENERATED CHECKBITS IN 1 SELECTED CSR
10024 035540  013771  002242  000002          MOV     TSTDAT+2,@2(R1) ;WRITE 2ND 16 BITS & CHECKBITS
10025 035546  000207                          RETURN
10026
10027                                         ;CHECK FOR NO TRAP OCCURING CONDITION
10028 035550  012737  035570  002264  MTPB25: MOV     #1$,PARTHERE    ;SETUP TRAP DESTINATION
10029 035556  005771  000000                  TST     @(R1)           ;ACCESS LOCATIONS
10030 035562  005771  000002                  TST     @2(R1)
10031 035566  000207                          RETURN                  ;NO TRAP - GOOD - RETURN
10032
10033 035570  104426                  1$:     READCSR
10034 035572  011137  002032                  MOV     (R1),ADDRESS    ;SAVE VIRTUAL ADDRESS
10035 035576  104024                          ERROR   +24
10036 035600                                  SET     HEADER
10037 035606  000207                          RETURN
10038
10039                                         ;TRAP SHOULD OCCURE TEST
10040 035610  012737  035624  002264  MTPC25: MOV     #1$,PARTHERE    ;SETUP TRAP DESTINATION
10041 035616  005771  000000                  TST     @(R1)           ;ACCESS 1ST LOCATION
10042 035622  000405                          BR      2$              ;NO TRAP - BAD NEWS - SKIP
10043 035624  012737  035654  002264  1$:     MOV     #3$,PARTHERE    ;SETUP TRAP DESTINATION
10044 035632  005771  000002                  TST     @2(R1)          ;ACCESS 2ND LOCATION
10045 035636  104426                  2$:     READCSR                 ;NO TRAP - BAD NEWS
10046 035640  011137  002032                  MOV     (R1),ADDRESS    ;SAVE VIRTUAL ADDRESS
10047 035644  104025                          ERROR   +25
10048 035646                                  SET     HEADER
10049 035654  000207                  3$:     RETURN
10050
```

```
10053 035656                          MTPA26: SUBTST  <<MTPA26        RANDOM DATA (WRITE)>>
                                      ;*****************************************************************************
                                      ;*SUBTEST       MTPA26  RANDOM DATA (WRITE)
                                      ;*****************************************************************************
10054 035656  000137  035726          1$:     JMP     @#MTPC26        ;V177640        GOTO V172360
10055 035662  010221                          MOV     R2,(R1)+        ;V177644
10056 035664  010321                          MOV     R3,(R1)+        ;V177646
10057 035666  077005                          SOB     R0,1$           ;V177650
10058 035670  000207                          RETURN                  ;V177652
10059
10060 035672                          MTPB26: SUBTST  <<MTPB26        RANDOM DATA (READ)>>
                                      ;*****************************************************************************
                                      ;*SUBTEST       MTPB26  RANDOM DATA (READ)
                                      ;*****************************************************************************
10061                                         .DSABL  AMA
10062                                         .ENABL  LSB
10063 035672  000137  035726          1$:     JMP     @#MTPC26        ;V177640        GOTO V172360
10064 035676  020221                          CMP     R2,(R1)+        ;V177644
10065 035700  001401                          BEQ     2$              ;V177646
10066 035702  104451                          PERR25                  ;V177650
10067 035704  005127                  2$:     COM     (PC)+           ;V177652
10068 035706  000000                  RANODD: 0                       ;V177654        FOR ERROR REPORTING
10069 035710  020321                          CMP     R3,(R1)+        ;V177656
10070 035712  001401                          BEQ     3$              ;V177660
10071 035714  104451                          PERR25                  ;V177662
10072 035716  005167  177764          3$:     COM     RANODD          ;V177664
10073 035722  077015                          SOB     R0,1$           ;V177670
10074 035724  000207                          RETURN                  ;V177672
10075                                         .DSABL  LSB
10076                                         .ENABL  AMA
10077
10078 035726                          MTPC26: SUBTST  <<RANDOM NUMBER SUBPROGRAM>>
                                      ;*****************************************************************************
                                      ;*SUBTEST       RANDOM NUMBER SUBPROGRAM
                                      ;*****************************************************************************
10079                                         ;CALLER MUST SETUP
10080                                         ;       MOV     SEEDLO,R3
10081                                         ;       MOV     SEEDHI,R2
10082                                         ;       MOV     R3,R5
10083                                         ;       MOV     R2,R4
10084 035726  073427  000007                  ASHC    #7,R4           ;V172360
10085 035732  060305                          ADD     R3,R5           ;V172364
10086 035734  005504                          ADC     R4              ;V172366
10087 035736  060204                          ADD     R2,R4           ;V172370
10088 035740  062705  001057                  ADD     #1057,R5        ;V172372
10089 035744  000240                          NOP                     ;V172376        GOTO V172260
10090
10091 035746                          MTPD26: SUBTST  <<RANDOM NUMBER SUBSUBPROGRAM>>
                                      ;*****************************************************************************
                                      ;*SUBTEST       RANDOM NUMBER SUBSUBPROGRAM
                                      ;*****************************************************************************
10092 035746  005504                          ADC     R4              ;V172260
10093 035750  062704  047401                  ADD     #47401,R4       ;V172262
10094 035754  010503                          MOV     R5,R3           ;V172266
10095 035756  010402                          MOV     R4,R2           ;V172270
10096 035760  000137  035662                  JMP     @#MTPA26+4      ;V172272        GOTO V177644
```

```
10099 035764                        MTP030: SUBTST  <<MT0030        FLUSH OUT DBE'S>>
                                    ;*****************************************************************************
                                    ;*SUBTEST       MT0030  FLUSH OUT DBE'S
                                    ;*****************************************************************************
10100 035764  011002                1$:     MOV     (R0),R2         ;V177640
10101 035766  010220                        MOV     R2,(R0)+        ;V177642
10102 035770  077103                        SOB     R1,1$           ;V177644
10103 035772  000207                        RETURN                  ;V177646
10104
10105 035774                        MTP031: SUBTST  <<MTP031        SOB-A-LONG TEST>>
                                    ;*****************************************************************************
                                    ;*SUBTEST       MTP031  SOB-A-LONG TEST
                                    ;*****************************************************************************
10106                                       .DSABL  AMA
10107 035774  000000                        0                       ;MOVE TERMINATOR
10108 035776  077001                1$:     SOB     R0,1$           ;SOB TILL R0 UNDERFLOWS
10109 036000  005167  177772                COM     1$              ;WRITE COMPLEMENT OF SOB
10110 036004  020167  177766                CMP     R1,1$           ;READ & CHECK FOR NOT ''SOB R0,DOT''
10111 036010  001403                        BEQ     2$              ;OK - SKIP
10112 036012  104454                        PERR30
10113 036014  010167  177756                MOV     R1,1$
10114 036020  005167  177752        2$:     COM     1$              ;CORRECT SOB INSTRUCTION
10115 036024  010200                        MOV     R2,R0           ;REINITIALIZE SOB CONSTANT
10116                                       ;UPDATE MOVE REGISTERS
10117 036026  010503                        MOV     R5,R3
10118 036030  005725                        TST     (R5)+           ;BUMP (SAFELY) BY 2
10119 036032  010504                        MOV     R5,R4
10120 036034  020537  002472                CMP     R5,@#LINK1      ;DONE?
10121 036040  001001                        BNE     3$              ;NO - SKIP
10122 036042  000207                        RETURN                  ;YES
10123
10124 036044  014344                3$:     MOV     -(R3),-(R4)
10125 036046  001376                        BNE     3$
10126 036050  000752                        BR      1$
10127         000056                SOBLENGTH=.-MTP031
10128                                       .ENABL  AMA
```

```
10156 036052                    MTP032: SUBTST  <<MTP032        WRITE RECOVERY TEST>>
                                ;*****************************************************************************
                                ;*SUBTEST       MTP032  WRITE RECOVERY TEST
                                ;*****************************************************************************
10157                                   ;THE TEST ACTUALLY EXECUTED ALREADY IN THE MEMORY UNDER TEST.
10158                                   ;THIS CODE INSURES THAT IT CHANGED MEMORY TO HAVE
10159                                   ;1/2 BANK OF #5141      WHICH IS A "COM -(R1)" INSTRUCTION AND
10160                                   ;1/2 BANK OF #110       WHICH IS A "JMP (R0)"   INSTRUCTION.
10161
10162 036052  012401              1$:      MOV     (R4)+,R1        ;V177640        ;GET DATA FROM LOWER 1/2 BANK
10163 036054  020102                       CMP     R1,R2           ;V177642        ;IS IT #5141?
10164 036056  001401                       BEQ     2$              ;V177644        ;YES - SKIP
10165 036060  104430                       PERR02                  ;V177646        ;NO - TAKE ERROR TRAP
10166 036062  077305              2$:      SOB     R3,1$           ;V177650        ;LOOP FOR 1/2 BANK
10167 036064  013703  002472               MOV     @#LINK1,R3      ;V177652        ;RESTORE LOOP SIZE
10168 036070  012400              3$:      MOV     (R4)+,R0        ;V177656        ;GET DATA FROM UPPER 1/2 BANK
10169 036072  020005                       CMP     R0,R5           ;V177660        ;IS IT #110?
10170 036074  001401                       BEQ     4$              ;V177662        ;YES - SKIP
10171 036076  104427                       PERR01                  ;V177664        ;NO- TAKE ERROR TRAP
10172 036100  077305              4$:      SOB     R3,3$           ;V177666        ;LOOP FOR 1/2 BANK
10173 036102  000207                       RETURN
```

```
10176 036104                          MTP033: SUBTST  <<MTP033        BRANCH GOBBLE TEST>>
                                      ;*****************************************************************************
                                      ;*SUBTEST       MTP033  BRANCH GOBBLE TEST
                                      ;*****************************************************************************
10177                                         .DSABL  AMA
10178 036104  000000                  0                               ;MOVE TERMINATOR
10179 036106  000000                  BGTEST: 0                       ;TEST WORD (TWO BYTES)
10180 036110  000261                  BRGOBB: SEC                     ;SET CARRY (TO BE ADDED TO 'BGTEST')
10181 036112  105511                          ADCB    (R1)            ;INCREMENT LOW BYTE OF 'BGTEST'
10182 036114  100402                          BMI     1$              ;BRANCH WHEN BIT7 IS SET
10183 036116  105212                          INCB    (R2)            ;INCREMENT HIGH BYTE OF 'BGTEST'
10184 036120  000773                          BR      BRGOBB          :LOOP 128 TIMES
10185
10186                                         ;NOW CHECK FOR CORRECT CONDITION CODES
10187 036122  102401                  1$:     BVS     2$              ;BR IF V-BIT SET (SHOULD BE)
10188 036124  104461                          PERR35                  ;NO - REPORT ERROR AND ABORT TEST
10189                                                                 ;COND CODES NOT EQUAL TO 1010
10190 036126  000242                  2$:     CLV                     ;CLEAR V-BIT
10191 036130  105212                          INCB    (R2)            ;INCREMENT HIGH BYTE OF 'BGTEST' ONCE MORE
10192 036132  103402                          BCS     3$              ;BR IF C-BIT SET (SHOULD NOT BE)
10193 036134  102001                          BVC     3$              ;BR IF V-BIT CLEAR (SHOULD NOT BE)
10194 036136  100401                          BMI     4$              ;BR IF N-BIT SET (SHOULD BE)
10195 036140  104461                  3$:     PERR35                  ;NO - REPORT ERROR AND ABORT TEST
10196                                                                 ;COND CODES NOT EQUAL TO 1010
10197
10198                                         ;UPDATE TEST POINTERS
10199 036142  010701                  4$:     MOV     PC,R1
10200 036144  162701  000036          5$:     SUB     #5$-BGTEST,R1
10201 036150  010102                          MOV     R1,R2
10202 036152  005202                          INC     R2
10203
10204                                         ;UPDATE MOVE REGISTERS
10205 036154  010503                          MOV     R5,R3
10206 036156  005725                          TST     (R5)+           ;BUMP (SAFELY) BY 2
10207 036160  010504                          MOV     R5,R4
10208
10209                                         ;DONE?
10210 036162  020537  002472                  CMP     R5,@#LINK1      ;DONE?
10211 036166  001001                          BNE     6$              ;NO - SKIP
10212 036170  000207                          RETURN                  ;YES - RETURN
10213
10214                                         ;MOVE CODE 1 LOCATION
10215 036172  014344                  6$:     MOV     -(R3),-(R4)
10216 036174  001376                          BNE     6$
10217 036176  005011                          CLR     (R1)            ;CLEAR TEST WORD 'BGTEST'
10218 036200  000743                          BR      BRGOBB          ;RUN MOVED CODE AGAIN
10219         000076                  GBLENGTH=.-MTP033
10220                                         .ENABL  AMA
```

```
036202                  MTP034: SUBTST  <<MTP034        SOFT ERROR - BACKROUND PATTERN TEST>>
                        ;*************************************************************************
                        ;*SUBTEST       MTP034  SOFT ERROR - BACKROUND PATTERN TEST
                        ;*************************************************************************
036202  010220          1$:     MOV     R2,(R0)+        ;V177640
036204  077102                  SOB     R1,MTP034       ;V177642
036206  000207                  RETURN                  ;V177644
036210  012401          2$:     MOV     (R4)+,R1        ;V177646
036212  020102                  CMP     R1,R2           ;V177650
036214  001402                  BEQ     3$              ;V177652
036216  104430                  PERR02                  ;V177654
036220  000240                  NOP                     ;V177656
036222  077306          3$:     SOB     R3,2$           ;V177660
036224  000207                  RETURN                  ;V177662
```

```
10234 036226                          MTP035:SUBTST   <<MTP035        WORST CASE NOISE PARITY TEST>>
                                      ;*****************************************************************************
                                      ;*SUBTEST       MTP035  WORST CASE NOISE PARITY TEST
                                      ;*****************************************************************************
10235 036226  012737  000003  002074          MOV     #3,NOPAR        ;SET PARITY TRAPS TO RETURN TO 'PARTHERE'
10236
10237 036234                              FOR R0 := #FIRST TO #LAST BY #4000
10238 036240  012737  000005  002144          MOV   #BIT2!BIT0,CSR  ;SET WRITE WRONG PARITY & PAR. TRAPS INTO CSR
10239 036246  104425                          LOADCSR
10240 036250  012737  036304  002264          MOV   #1$,PARTHERE
10241 036256  011010                          MOV   (R0),(R0)       ;WWP TEST LOCATION
10242 036260  005710                          TST   (R0)
10243 036262  010037  002032                  MOV   R0,ADDRESS
10244 036266  104050                          ERROR +50
10245 036270  004737  054704                  CALL  PERBNK
10246 036274  032763  002000  002626          BIT   #BIT10,CONFIG+2(R3)
10247 036302  001002                          BNE   2$
10248 036304  104426                    1$:   READCSR
10249 036306  104512                          ERRGEN
10250
10251 036310  104503                    2$:   CLR1CSR
10252 036312  011010                          MOV   (R0),(R0)       ;CLEAR WRONG PARITY IN MEMORY
10253 036314  012737  000001  002144          MOV   #BIT0,CSR
10254 036322  104425                          LOADCSR
10255 036324  012737  036336  002264          MOV   #3$,PARTHERE
10256 036332  005710                          TST   (R0)
10257 036334  000405                          BR    4$
10258 036336  010037  002032           3$:    MOV   R0,ADDRESS
10259 036342  104050                          ERROR +50
10260 036344  004737  054704                  CALL  PERBNK
10261 036350                           4$:    END; OF FOR
10262
10263 036362  005037  002074              CLR     NOPAR           ;RESET PARITY TRAP ACTION
10264 036366  000207                      RETURN
```

```
                                        .SBTTL  MISC SUBROUTINES

                                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
                                ;*****************************************************************************
                                        MOV     R0,R4
                                        MOV     R1,R3
                                        MOV     R2,R5
                                        RETURN

                                FLIPWARN:SUBTST <<FLIP WARNING CONSTANTS IN WORST CASE NOISE TESTS>>
                                ;*****************************************************************************
                                ;*SUBTEST        FLIP WARNING CONSTANTS IN WORST CASE NOISE TESTS
                                ;*****************************************************************************
                                        PUSH    R0
                                        INC     FLIPLOC
                                        BIC     #^C3,FLIPLOC
                                        CMP     #1,FLIPLOC
                                        BEQ     1$
                                        CMP     #2,FLIPLOC
                                        BEQ     2$
                                        CMP     #3,FLIPLOC
                                        BEQ     3$
                                        CLR     R0
                                        MOV     ONES,R4
                                        BR      4$
                                1$:     CLEAR   R0,R4
                                        BR      4$
                                2$:     MOV     #401,R0
                                        MOV     ONES,R4
                                        BR      4$
                                3$:     MOV     #401,R0
                                        MOV     #401,R4
                                4$:     MOV     R0,WARN2
                                        MOV     R0,WARN3
                                        MOV     R0,WARN4
                                        MOV     R0,WARN5
                                        POP     R0
                                        RETURN
```

```
10301 036530                                BACKGND:SUBTST  <<SUBR  WRITE BACKGROUND>>
                                            ;*****************************************************************************
                                            ;*SUBTEST        SUBR    WRITE BACKGROUND
                                            ;*****************************************************************************
10302                                               ;WRITES DATA FROM R2
10303 036530  104415                                SAVREG
10304 036532  012700  060000                        MOV     #FIRST,R0
10305 036536  012701  040000                        MOV     #SIZE,R1
10306 036542  022737  000001  003716                CMP     #1,PROTYP
10307 036550  001415                                BEQ     WARN6B
10308 036552  012737  000207  027040  WARN6A:       MOV     #207,MTP000+4           ;WARNING PUTTING 'RETURN'' AFTER WRITE
10309 036560  012737  027034  002254                MOV     #MTP000,SUPDOADD
10310 036566  004737  026642                        CALL    SUPD03
10311 036572  012737  000240  027040                MOV     #240,MTP000+4           ;RESTORE 'NOP' AFTER WRITE
10312 036600  104416                                RESREG
10313 036602  000207                                RETURN
10314 036604                                WARN6B: BMOV    MTP000
10315 036612  012737  000207  177644  WARN6:       MOV     #207,UIPAR2             ;WARNING PUTTING 'RETURN'' INSTRUCTION AFTER WRITE
10316 036620  004737  026464                        CALL    SUPD01
10317 036624  104416                                RESREG
10318 036626  000207                                RETURN
```

```
10321 036630                                  PCONFIG:SUBTST  <<SUBR  PRINT CONFIGURATION MAP>>
                                              ;*****************************************************************************
                                              ;*SUBTEST       SUBR    PRINT CONFIGURATION MAP
                                              ;*****************************************************************************
10322 036630                                          PUSH    TKVEC,TKVEC+2,R0
10323 036642  010637  037130                          MOV     SP,PCONFS                       ;SAVE LAST GOOD SP
10324 036646  012737  037076  000060                  MOV     #PCONF2,TKVEC
10325 036654  012737  000340  000062                  MOV     #340,TKVEC+2
10326 036662  017700  143716                          MOV     @$TKB,R0                        ;KILL ANY OLD INTERRUPT
10327 036666  042737  000200  177776                  BIC     #BIT7,PSW                       ;LOWER CPU PRIORITY TO 140
10328 036674  052777  000100  143700                  BIS     #BIT6,@$TKS                     ;ENABLE KEYBOARD INTERRUPTS
10329
10330 036702                                          TYPE    MSG001
10331 036706                                          TYPE    MSG002
10332 036712                                          TYPE    MSG003
10333 036716  022737  000060  002526                  CMP     #60,LASTBANK
10334 036724  002006                                  BGE     NOOJ
10335                                                 ;IF FAT PAPER ON TERMINAL GOTO 1$
10336 036726                                          IF #SW4 SET.IN @SWR THEN JUMPTO PCONF1
10337 036742  012700  000074                  NOOJ:   MOV     #60.,R0
10338 036746  010004                                  MOV     R0,R4
10339 036750                                          CLEAR   R1,R3
10340 036754                                          TYPE    MSG004
10341 036760  004737  037132                          CALL    TCONFIG                         ;GO TYPE CONFIGURATION (1ST HALF)
10342 036764  022737  000060  002526                  CMP     #60,LASTBANK
10343 036772  002041                                  BGE     PCONF2
10344 036774                                          TYPE    $CRLF
10345 037000                                          TYPE    MSG017                          ;PRINT SPACE(S)
10346 037004                                          TYPE    MSG011
10347 037010                                          TYPE    $CRLF
10348 037014                                          TYPE    MSG017                          ;PRINT SPACE(S)
10349 037020                                          TYPE    MSG012
10350 037024  012701  000360                          MOV     #60.*2*2,R1
10351 037030  010103                                  MOV     R1,R3
10352 037032  004737  037132                          CALL    TCONFIG
10353 037036  000417                                  BR      PCONF2
10354
10355 037040  012700  000170                  PCONF1: MOV     #120.,R0
10356 037044  010004                                  MOV     R0,R4
10357 037046                                          CLEAR   R1,R3
10358 037052                                          TYPE    MSG014                          ;SPACE
10359 037056                                          TYPE    MSG011
10360 037062                                          TYPE    MSG004
10361 037066                                          TYPE    MSG012
10362 037072  004737  037132                          CALL    TCONFIG
10363
10364 037076  013706  037130                  PCONF2: MOV     PCONFS,SP                       ;RESTORE STACK
10365 037102  042777  000100  143472                  BIC     #BIT6,@$TKS
10366 037110  117700  143470                          MOVB    @$TKB,R0                        ;READ CHAR TO KILL FLAG
10367 037114                                          POP     R0,TKVEC+2,TKVEC
10368 037126  000207                                  RETURN
10369
10370 037130  000000                  PCONFS: 0                                               ;STACK SAVED HERE!
```

```
10373 037132                                    SUBTST  <<SUBR  TYPE CONFIGURATION>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       SUBR    TYPE CONFIGURATION
                                        ;*****************************************************************************
10374                                   ;*****************************************************************************
10375                                   ;CALL:  MOV     #N,R0           ;N=NUMBER OF CHARACTERS
10376                                   ;       MOV     R0,R4           ;BACKUP
10377                                   ;       MOV     #K,R1           ;INDEX CONSTANT
10378                                   ;       MOV     R1,R3           ;BACKUP
10379                                   ;       CALL    TCONFIG         ;ACTUAL CALL
10380                                   ;       RETURN                  ;ONLY RETURN
10381                                   ;*****************************************************************************
10382
10383                                           ;***********
10384                                           ;** ERROR **
10385                                           ;***********
10386 037132                           TCONFIG:TYPE    MSG005
10387 037136  032761  000001  002624   1$:     BIT     #BIT0,CONFIG(R1)        ;ERROR ON THIS BANK?
10388 037144  001403                           BEQ     2$                      ;NO - SKIP
10389 037146                                   TYPE    MSG013                  ;PRINT "X"
10390 037152  000402                           BR      3$
10391 037154                           2$:     TYPE    MSG014                  ;PRINT SPACE
10392 037160  062701  000004           3$:     ADD     #4,R1                   ;BUMP POINTER
10393 037164  077014                           SOB     R0,1$                   ;LOOP TILL DONE
10394 037166  010400                           MOV     R4,R0
10395 037170  010301                           MOV     R3,R1
10396
10397                                          ;***********
10398                                          ;** CPU'S **
10399                                          ;***********
10400 037172                                    TYPE    MSG008
10401 037176  016105  002624           4$:     MOV     CONFIG(R1),R5
10402 037202  006205                           ASR     R5                      ;GET CPU BITS
10403 037204  042705  177760                   BIC     #^C17,R5                ;CLEAR NON INTERESTING BITS
10404 037210  005705                           TST     R5                      ;IS THERE ANYTHING THERE?
10405 037212  001003                           BNE     8$                      ;YES - BRANCH.
10406 037214  112705  000040                   MOVB    #' ,R5                  ;NO - MOVE A BLANK INTO R5
10407 037220  000402                           BR      9$                      :BRANCH OVER NEXT INSTRUCTION
10413 037222  062705  000060           8$:     ADD     #60,R5                  ;MAKE ASCII
10414 037226  110537  071574           9$:     MOVB    R5,MSG015               ;PLUG INTO MEMORY
10415 037232                                   TYPE    MSG015
10416 037236  062701  000004                   ADD     #4,R1                   ;BUMP POINTER
10417 037242  077023                           SOB     R0,4$                   ;LOOP TILL DONE
10418 037244  010400                           MOV     R4,R0
10419 037246  010301                           MOV     R3,R1
```

```
10422                                                   ;*****************
10423                                                   ;** INTERLEAVE **
10424                                                   ;*****************
10425 037250                                            TYPE    MSG007
10426                                                   ;THIS IS AN ENTRY POINT FROM ERROR REPORTS
10427 037254  032761  010000  002626  TCFIG1: BIT     #BIT12,CONFIG+2(R1)
10428 037262  001014                        BNE     1$
10429 037264  032761  000002  002624        BIT     #BIT1,CONFIG(R1)        ;IS THERE ANY MEMORY HERE?
10430 037272  001004                        BNE     18$                     ;BRANCH IF MEMORY PRESENT.
10431 037274  112737  000040  071574        MOVB    #' ,MSG015              ;MOVE A BLANK IN TO BE PRINTED
10432 037302  000424                        BR      16$                     ;BRANCH TO TYPE ROUTINE
10433 037304  112737  000055  071574  18$:  MOVB    #'-,MSG015
10434 037312  000420                        BR      16$
10435 037314  016105  002624          1$:   MOV     CONFIG(R1),R5
10436 037320  042705  007777                BIC     #^C170000,R5            ;GET CSR INTERLEAVE
10437 037324  000305                        SWAB    R5
10438 037326  072527  177774                ASH     #-4,R5
10439 037332  022705  000012                CMP     #10.,R5
10440 037336  100002                        BPL     2$
10441 037340  062705  000007                ADD     #7,R5
10442 037344  062705  000060          2$:   ADD     #60,R5                  ;MAKE ASCII
10443 037350  110537  071574                MOVB    R5,MSG015               ;PLUG INTO MEMORY
10444 037354                          16$:  TYPE    MSG015
10445 037360                                IF NOTAB NE #0 THEN $RETURN
10446 037370  062701  000004                ADD     #4,R1                   ;BUMP POINTER
10447 037374  077051                        SOB     R0,TCFIG1               ;LOOP TILL DONE
10448 037376  010400                        MOV     R4,R0
10449 037400  010301                        MOV     R3,R1
10450
10451                                                   ;******************
10452                                                   ;** MEMORY TYPE **
10453                                                   ;******************
10454                                                   .ENABL  LSB
10455 037402                                            TYPE    MSG009
10456 037406  033761  002104  002624  TCFIG2: BIT     CPUBIT,CONFIG(R1)
10457 037414  001447                        BEQ     17$
10458 037416  016105  002626                MOV     CONFIG+2(R1),R5
10459 037422  000305                        SWAB    R5                      ;GET MEMORY TYPE
10460 037424  042705  177770                BIC     #^C7,R5                 ;CLEAR NON INTERESTING BITS
10461 037430  005705                        TST     R5
10462 037432  001440                        BEQ     17$
10463 037434  032705  000004                BIT     #BIT2,R5
10464 037440  001004                        BNE     4$
10465 037442  112737  000102  071574        MOVB    #'B,MSG015
10466 037450  000434                        BR      8$
10467 037452  032705  000002          4$:   BIT     #BIT1,R5
10468 037456  001013                        BNE     6$
10469 037460  032705  000001                BIT     #BIT0,R5
10470 037464  001004                        BNE     5$
10471 037466  112737  000115  071574        MOVB    #'M,MSG015
10472 037474  000422                        BR      8$
10473 037476  112737  000113  071574  5$:   MOVB    #'K,MSG015
10474 037504  000416                        BR      8$
10475 037506  032705  000001          6$:   BIT     #BIT0,R5
10476 037512  001004                        BNE     7$
10477 037514  112737  000114  071574        MOVB    #'L,MSG015
10478 037522  000407                        BR      8$
```

```
10479 037524  112737  000120  071574  7$:     MOVB    #'P,MSG015
10480 037532  000403                          BR      8$
10481 037534  112737  000040  071574  17$:    MOVB    #' ,MSG015
10482 037542                          8$:     TYPE    MSG015
10483 037546                                  IF NOTAB NE #0 THEN $RETURN
10484 037556  062701  000004                  ADD     #4,R1                   ;BUMP POINTER
10485 037562  077067                          SOB     R0,TCFIG2               ;LOOP TILL DONE
10486 037564  010400                          MOV     R4,R0
10487 037566  010301                          MOV     R3,R1
10488                                         .DSABL  LSB
10489
10490                                         ;*********
10491                                         ;** CSR **
10492                                         ;*********
10493 037570                                   TYPE    MSG016
10494 037574  112737  000040  071574  TCFIG3: MOVB    #' ,MSG015
10495 037602  016105  002624                  MOV     CONFIG(R1),R5
10496 037606  032705  000002                  BIT     #BIT1,R5
10497 037612  001414                          BEQ     16$
10498 037614  042705  170377                  BIC     #^C7400,R5
10499 037620  000305                          SWAB    R5
10500 037622  022705  000012                  CMP     #10.,R5
10501 037626  100002                          BPL     10$
10502 037630  062705  000007                  ADD     #7,R5
10503 037634  062705  000060          10$:    ADD     #60,R5                  ;MAKE ASCII
10504 037640  110537  071574                  MOVB    R5,MSG015               ;PLUG INTO MEMORY
10505 037644                          16$:    TYPE    MSG015
10506 037650                                  IF NOTAB NE #0 THEN $RETURN
10507 037660  062701  000004                  ADD     #4,R1                   ;BUMP POINTER
10508 037664  077035                          SOB     R0,TCFIG3               ;LOOP TILL DONE
10509 037666  010400                          MOV     R4,R0
10510 037670  010301                          MOV     R3,R1
10511
10512                                         ;***************
10513                                         ;** PROTECTED **
10514                                         ;***************
10515 037672                                   TYPE    MSG010
10516 037676  105761  002624          11$:    TSTB    CONFIG(R1)              ;BANK PROTECTED?
10517 037702  100004                          BPL     12$                     ;NO - SKIP
10518 037704  112737  000120  071574          MOVB    #'P,MSG015
10519 037712  000407                          BR      13$
10520 037714  032761  000100  002624  12$:    BIT     #BIT6,CONFIG(R1)        ;PROTECTED REGION OF ECC?
10521 037722  001406                          BEQ     14$                     ;NO - SKIP
10522 037724  112737  000111  071574          MOVB    #'I,MSG015
10523 037732                          13$:    TYPE    MSG015
10524 037736  000402                          BR      15$
10525 037740                          14$:    TYPE    MSG014                  ;PRINT SPACE
10526 037744  062701  000004          15$:    ADD     #4,R1                   ;BUMP POINTER
10527 037750  077026                          SOB     R0,11$                  ;LOOP TILL DONE
10528 037752  010400                          MOV     R4,R0
10529 037754  010301                          MOV     R3,R1
10530 037756  000207                          RETURN
```

```
10533                                   .SBTTL  TRAP    PARITY ERROR HANDLER
10534                           ;*******************************************************************************
10535                           ;VECTOR TO HERE FROM TRAPS TO 114
10536                           ;IGNORE ERRORS BUT COUNT IF NOPAR FLAG = 1.
10537                           ;*******************************************************************************
10538                           ;
10539                           ;       CODE    ACTION
10540                           ;
10541                           ;       --0--   PRINT "UNEXPECTED PARITY TRAP
10542                           ;         1     COUNT ERROR
10543                           ;         2     SET "ABORT" / SETUP "BADPC" / RETURN VIA PCBUMP
10544                           ;         3     RETURN VIA "PARTHERE"
10545                           ;
10546 037760  022737  000001  002074  PARITY: CMP     #1,NOPAR                ;COUNTING PARITY ERRORS?
10547 037766  001003                          BNE     1$                      ;NO - SKIP
10548 037770  005237  002070                  INC     PARCNT                  ;PARITY ERROR COUNTER + 1
10549 037774  000002                          RTI
10550 037776  022737  000002  002074  1$:     CMP     #2,NOPAR                ;ACTION CODE = 2 ?
10551 040004  001013                          BNE     2$                      ;NO - SKIP
10552 040006                                  SET     ABORTFLAG               ;YES
10553 040014  004737  040166                  CALL    BADSTACK                ;FIND BAD SP,PC,PSW OFF STACK
10554 040020  063716  002276                  ADD     PCBUMP,(SP)             ;UPDATE RETURN PC
10555 040024  042766  000004  000002          BIC     #BIT2,2(SP)             ;SHOW FAILURE BY .NE.
10556 040032  000002                          RTI
10557 040034  022737  000003  002074  2$:     CMP     #3,NOPAR                ;ACTION CODE = 3 ?
10558 040042  001003                          BNE     3$                      ;NO - SKIP
10559 040044  013716  002264                  MOV     PARTHERE,(SP)
10560 040050  000002                          RTI
10561 040052  004737  040166          3$:     CALL    BADSTACK                ;FIND BAD SP,PC,PSW OFF STACK
10562 040056                                  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
                                ;*******************************************************************************

040064  022737  000001  002076  NONEXIST:CMP     #1,NONEM                ;COUNTING NON-EXISTANT MEMORY ERRORS?
040072  001011                          BNE     2$                      ;NO - SKIP
040074  005237  002066                  INC     NEMCNT                  ;BUMP NON-EXISTANT MEMORY COUNTER
040100  022737  000001  002066          CMP     #1,NEMCNT               ;FIRST ERROR?
040106  001002                          BNE     1$                      ;NO - SKIP
040110  010037  002032                  MOV     R0,ADDRESS              ;ASSUME R0 CONTAINS THE ADDRESS ACCESSED
040114  000002                  1$:     RTI
040116  005237  002066          2$:     INC     NEMCNT                  ;BUMP NON-EXISTANT MEMORY COUNTER
040122  012701  000001                  MOV     #1,R1                   ;DUMMY UP R1 FOR A FORCED SOB EXIT
040126  000002                          RTI

                                ;*******************************************************************************
                                        .SBTTL  TRAP    TIMEOUT (TRAP TO 4) HANDLER
040130  004737  040166          TIMEOUT:CALL    BADSTACK                ;FIND BAD SP,PC,PSW OFF STACK
040134                                  FATAL   6
                                ;*******************************************************************************
                                        .SBTTL  TRAP    MEMORY MANAGEMENT (TRAP TO 250) HANDLER
040142  004737  040166          MMTRAP: CALL    BADSTACK                ;FIND BAD SP,PC,PSW OFF STACK
040146                                  FATAL   7
                                        .SBTTL  TRAP    RESERVED INSTRUCTION HANDLER
040154  004737  040166          PDP1105:CALL    BADSTACK                ;FIND BAD SP,PC,PSW OFF STACK
040160                                  FATAL   5


040166                          BADSTACK:SUBTST <<FIND BAD SP, PC, & PSW FROM STACK>>
                                ;*******************************************************************************
                                ;*SUBTEST       FIND BAD SP, PC, & PSW FROM STACK
                                ;*******************************************************************************
040166  010637  002024                  MOV     SP,BADSP
040172  062737  000002  002024          ADD     #2,BADSP
040200  016637  000002  002020          MOV     2(SP),BADPC
040206  016637  000004  002030          MOV     4(SP),BADPSW
040214  000207                          RETURN
```

```
10610                                           .SBTTL  TRAP    KERNEL TRAP HANDLER
10611                                   ;*****************************************************************************
10612                                   ;KERNEL IS A TRAP THAT COMES HERE
10613                                   ;*****************************************************************************
10614
10615 040216  042766  140000  000002  $KERNEL:       BIC     #140000,2(SP)
10616 040224  000002                          RTI
10617                                   ;*****************************************************************************
10618                                           .SBTTL  TRAP    ENERGIZE TRAP HANDLER
10619 040226  052737  000001  177572  $ENERGIZE:BIS   #BIT0,MMR0
10620 040234  000002                          RTI
10621                                   ;*****************************************************************************
10622                                           .SBTTL  TRAP    DEENERGIZE TRAP HANDLER
10623 040236  042737  000001  177572  $DEENERGIZE:BIC #BIT0,MMR0
10624 040244  000002                          RTI
10625                                   ;*****************************************************************************
10626                                           .SBTTL  TRAP    CACHON TRAP HANDLER
10627 040246  005737  002514          $CACHN: TST     CACHKN          ;IS THERE A CACHE
10628 040252  001406                          BEQ     1$              ;NO - RETURN
10629 040254  013737  002514  177746          MOV     CACHKN,CONTRL   ;SETUP CACHE AS PER CONSTANT (USUALLY 1 = FULLY ON)
10630 040262  052737  000001  177746          BIS     #BIT0,CONTRL    ;DISABLE TRAPS (BUT NOT ABORTS)
10631 040270  000002                  1$:     RTI
10632                                   ;*****************************************************************************
10633                                           .SBTTL  TRAP    CACHOFF TRAP HANDLER
10634 040272  005737  002514          $CACHF: TST     CACHKN          ;IS THERE A CACHE?
10635 040276  001403                          BEQ     1$              ;NO - RETURN
10636                                           ;DISABLE TRAPS (NOT ABORTS), FORCE MISSES, FLUSH, BYPASS
10637 040300  053737  002520  177746          BIS     CACHKF,CONTRL
10638 040306  000002                  1$:     RTI
```

```
10641                                                   .SBTTL  TRAP    LOAD CSR TRAP HANDLER
10642                                                   ;LOAD CORRECT CSR WITH DATA IN CSR
10643                                                   ;PROGRAM CSR'S ASSERT INHIBIT MODE POINTER WHEN LOADED
10644 040310                                    $LOADC: PUSH    R0,R1
10645 040314  013700  002146                          MOV     CSRNO,R0                ;CREATE CSR ADDRESS
10646 040320                                          IF INHECC IS TRUE THEN GOTO 3$  ;DON'T WANT INH. MODE POINTER ON
10647 040326  005737  002502                          TST     PGMCSR                  ;PROGRAM IN INTERLEAVED SPACE?
10648 040332  100007                                  BPL     1$                      ;BRANCH IF NOT
10649 040334  113701  002503                          MOVB    PGMCSR+1,R1             ;CHECK SECOND CSR
10650 040340  042701  177740                          BIC     #^C37,R1                ;CLEAR UNNECESSARY BITS
10651 040344  020137  002146                          CMP     R1,CSRNO                ;IS THIS THE CURRENT CSR?
10652 040350  001404                                  BEQ     2$                      ;BRANCH IF IT IS
10653 040352  123737  002502  002146  1$:             CMPB    PGMCSR,CSRNO            ;IS THIS THE CURRENT CSR?
10654 040360  001003                                  BNE     3$                      ;BRANCH IF NOT
10655 040362  052737  020000  002144  2$:             BIS     #BIT13,CSR              ;SET THE INHIBIT MODE POINTER TO  1ST 16K
10656 040370  013760  002144  172100  3$:             MOV     CSR,CSRADD(R0)          ;LOAD THE CSR
10657 040376                                          POP     R1,R0                   ;RESTORE REGISTERS
10658 040402  000002                                  RTI
10659
10660                                                   .SBTTL  TRAP    READ CSR TRAP HANDLER
10661                                                   ;READ THE CORRECT CSR INTO LOCATIONS CSR
10662 040404                                    $READC: PUSH    R0
10663 040406  013700  002146                          MOV     CSRNO,R0
10664 040412  016037  172100  002144                  MOV     CSRADD(R0),CSR          ;READ IT
10665 040420                                          POP     R0
10666 040422  000002                                  RTI
```

```
10668                                                   .SBTTL  TRAP    TEST (R1) & READ CSR CAREFULLY
10669 040424                                    $TSTRD: PUSH    R0,R2,R3
10670 040432  012700  172100                            MOV     #CSRADD,R0              ;CREATE CSR ADDRESS
10671 040436  063700  002146                            ADD     CSRNO,R0
10672 040442  005002                                    CLR     R2
10673 040444  005737  002502                            TST     PGMCSR
10674 040450  100007                                    BPL     1$
10675 040452  113703  002503                            MOVB    PGMCSR+1,R3
10676 040456  042703  000200                            BIC     #BIT7,R3
10677 040462  020337  002146                            CMP     R3,CSRNO
10678 040466  001404                                    BEQ     2$
10679 040470  123737  002502  002146  1$:       CMPB    PGMCSR,CSRNO
10680 040476  001002                                    BNE     3$
10681 040500  012702  020000          2$:       MOV     #BIT13,R2
10682 040504  022737  000001  003716  3$:       CMP     #1,PROTYP       ;IS THIS AN 11/44?
10683 040512  001403                                    BEQ     4$              ;BRANCH IF IT IS
10684 040514  004737  040602                            CALL    TSTRD1
10685 040520  000405                                    BR      5$
10686 040522                                    4$:     BMOV    TSTRD1
10687 040530  004737  177640                            CALL    FASTCITY                ;CALL TO THE USER INSTRUCTION PAR'S
10688                                                   ;IF SINGLE BIT ERROR ONLY - SET CARRY BIT
10689 040534                                    5$:     POP     R3,R2,R0
10690 040542                                            IF #BIT4 SET.IN CSR AND #BIT15 OFF.IN CSR
10691 040562  052766  000001  000002                      BIS   #BIT0,2(SP)
10692 040570                                            ELSE
10693 040572  042766  000001  000002                      BIC   #BIT0,2(SP)
10694 040600                                            END ;OF IF #BIT4
10695 040600  000002                                    RTI
10696
10697 040602  010210                            TSTRD1: MOV     R2,(R0)                 ;V177640
10698 040604                                            TESTAREA                        ;V177642 ;ENTER SUPERVISOR MODE
10699 040612  105711                                    TSTB    (R1)                    ;V177646
10700 040614  042737  140000  177776                    BIC     #BIT15!BIT14,PSW        ;V177650
10701 040622  011037  002144                            MOV     (R0),CSR                ;V177656
10702 040626  000207                                    RETURN                          ;V177662
```

```
10705                                                   .SBTTL  TRAP    ECC DISABLE ALL CSR'S TRAP HANDLER
10706 040630  012737  000002  002144  $ECCDIS:MOV       #BIT1,CSR
10707 040636  004737  041354                  CALL      CSROUT
10708 040642  000002                          RTI
10709                                                   .SBTTL  TRAP    ECC DISABLE OF 1 SELECTED CSR TRAP HANDLER
10710 040644  012737  000002  002144  $ECC1DIS:MOV      #BIT1,CSR
10711 040652  104425                        LOADCSR
10712 040654  000002                          RTI
10713                                                   .SBTTL  TRAP    INITIALIZE ALL CSR'S TRAP HANDLER
10714 040656  012737  000001  002144  $ECCINIT:MOV      #BIT0,CSR
10715 040664  004737  041354                  CALL      CSROUT
10716 040670  000002                          RTI
10717                                                   .SBTTL  TRAP    INITIALIZE 1 SELECTED CSR TRAP HANDLER
10718 040672  012737  000001  002144  $ECC1INIT:MOV     #BIT0,CSR
10719 040700  104425                        LOADCSR
10720 040702  000002                          RTI
10721                                                   .SBTTL  TRAP    ENABLE SBE PARITY TRAPS ON ALL CSR'S
10722 040704  012737  000003  002144  $ENASBE:MOV       #BIT0!BIT1,CSR
10723 040712  004737  041354                  CALL      CSROUT
10724 040716  000002                          RTI
10725                                                   .SBTTL  TRAP    ENABLE SBE PARITY TRAPS ON 1 SELECTED CSR
10726 040720  012737  000003  002144  $ENA1SBE:MOV      #BIT0!BIT1,CSR
10727 040726  104425                        LOADCSR
10728 040730  000002                          RTI
10729                                                   .SBTTL  TRAP    WRITE CHECKBITS THRU ALL CSR'S TRAP HANDLER
10730 040732  013737  002274  002144  $CBCSR: MOV       CHECK,CSR               ;BITS 11-5
10731 040740  052737  000006  002144          BIS       #BIT1!BIT2,CSR          ;CHECK MODE
10732 040746  004737  041354                  CALL CSROUT
10733 040752  000002                          RTI
10734                                                   .SBTTL  TRAP    WRITE CHECKBITS THRU 1 SELECTED CSR TRAP HANDLER
10735 040754  013737  002274  002144  $CB1CSR:MOV       CHECK,CSR               ;BITS 11-5
10736 040762  052737  000006  002144          BIS       #BIT1!BIT2,CSR          ;CHECK MODE
10737 040770  104425                        LOADCSR
10738 040772  000002                          RTI
```

```
10741                                           .SBTTL  TRAP    WAS THERE A SBE ON ANY CSR TRAP HANDLER
10742 040774                              $WASSBE:PUSH    R1,R4
10743 041000  013701  002216                      MOV     TOTCSRS,R1      ;GET CSR'S BYTE
10744 041004  005004                              CLR     R4
10745 041006                                      BEGIN LWSBE
10746 041006                                        FOR CSRNO := #0 TO #36 BY #2
10747 041012  006301                                  ASL         R1
10748 041014                                          ON.ERROR
10749 041016  104426                                    READCSR
10750 041020                                            IF #BIT4 SET.IN CSR
10751 041030                                              SET     R4
10752 041034                                              LEAVE LWSBE
10753 041036                                            END ;OF IF #BIT4
10754 041036                                          END ;OF ON.ERROR
10755 041036                                          IF R1 EQ #0 THEN LEAVE LWSBE
10756 041042                                        END ;OF FOR CSRNO
10757 041060                                      END LWSBE
10758 041060  006004                              ROR     R4              ;SET C BIT FOR ERROR
10759 041062                                      POP     R4,R1
10760 041066                                      ON.ERROR
10761 041070  052766  000001  000002                BIS   #BIT0,2(SP)
10762 041076                                      ELSE
10763 041100  042766  000001  000002                BIC   #BIT0,2(SP)
10764 041106                                      END ;OF ON.ERROR
10765 041106  000002                              RTI
10766                                           .SBTTL  TRAP    WAS THERE A SBE IN 1 SELECTED CSR TRAP HANDLER
10767                                           ;ON RETURN IF CARRY IS SET THERE WAS A SBE
10768 041110  104426                      $WAS1SBE:READCSR
10769 041112  042766  000001  000002              BIC     #BIT0,2(SP)     ;CLR C BIT ON STACK
10770 041120  032737  000020  002144              BIT     #BIT4,CSR
10771 041126  001403                              BEQ     1$
10772 041130  052766  000001  000002              BIS     #BIT0,2(SP)     ;SET C BIT ON STACK
10773 041136  000002                        1$:   RTI
```

```
10776                                                   .SBTTL  TRAP    WAS THERE A DBE ON ANY CSR TRAP HANDLER
10777 041140                                    $WASDBE:PUSH    R1,R4
10778 041144  013701  002216                            MOV     TOTCSRS,R1      ;GET CSR'S BYTE
10779 041150  005004                                    CLR     R4
10780 041152                                            BEGIN LWDBE
10781 041152                                              FOR CSRNO := #0 TO #36 BY #2
10782 041156  006301                                        ASL         R1
10783 041160                                                ON.ERROR
10784 041162  104426                                          READCSR
10785 041164                                                  IF #BIT15 SET.IN CSR
10786 041174                                                    SET     R4
10787 041200                                                    LEAVE LWDBE
10788 041202                                                  END ;OF IF #BIT4
10789 041202                                                END ;OF ON.ERROR
10790 041202                                              IF R1 EQ #0 THEN LEAVE LWDBE
10791 041206                                              END ;OF FOR CSRNO
10792 041224                                            END LWDBE
10793 041224  006004                                    ROR     R4              ;SET C BIT FOR ERROR
10794 041226                                            POP     R4,R1
10795 041232                                            ON.ERROR
10796 041234  052766  000001  000002                      BIS   #BIT0,2(SP)
10797 041242                                            ELSE
10798 041244  042766  000001  000002                      BIC   #BIT0,2(SP)
10799 041252                                            END ;OF ON.ERROR
10800 041252  000002                                    RTI
10801                                                   .SBTTL  TRAP    WAS THERE A DBE ON 1 SELECTED CSR TRAP HANDLER
10802                                                   ;ON RETURN IF CARRY IS SET THERE WAS A DBE
10803 041254  104426                            $WAS1DBE:READCSR
10804 041256  005737  002144                            TST     CSR             ;DBE?
10805 041262  100004                                    BPL     3$              ;NO - SKIP
10806 041264  052766  000001  000002                    BIS     #BIT0,2(SP)     ;SET C BIT ON STACK
10807 041272  000002                                    RTI
10808 041274  042766  000001  000002  3$:               BIC     #BIT0,2(SP)     ;CLR C BIT ON STACK
10809 041302  000002                                    RTI
```

```
10812                                                   .SBTTL  TRAP    CLEAR ALL ECC CSR'S TRAP HANDLER
10813 041304                                    $CLRCSR:CLEAR   CSR
10814 041310  004737  041354                            CALL    CSROUT
10815 041314  000002                                    RTI
10816                                                   .SBTTL  TRAP    CLEAR 1 SELECTED CSR TRAP HANDLER
10817 041316                                    $CLR1CSR:CLEAR  CSR
10818 041322  104425                                    LOADCSR
10819 041324  000002                                    RTI
10820                                                   .SBTTL  TRAP    ECC DISABLE, CHECK MODE, & WRITE CHECKBITS IN ALL CSR'S TRAP HANDLER
10821                                                   ;CHECKBITS ALREADY IN LOC "CSR"
10822 041326  052737  000006  002144  $CHKDIS:BIS     #BIT1!BIT2,CSR          ;ECC DISABLE & DIAG CHECK MODE
10823 041334  004737  041354                            CALL    CSROUT
10824 041340  000002                                    RTI
10825                                                   .SBTTL  TRAP    ECC DISABLE, CHECK MODE, & WRITE CHECKBITS IN 1 SELECTED CSR
10826                                                   ;CHECKBITS ALREADY IN LOC "CSR"
10827 041342  052737  000006  002144  $CHK1DIS:BIS    #BIT1!BIT2,CSR          ;ECC DISABLE & DIAG CHECK MODE
10828 041350  104425                                    LOADCSR
10829 041352  000002                                    RTI
```

```
10832 041354                          CSROUT: SUBTST  <<SUBR  WRITE IN ALL CSR'S>>
                                      ;*****************************************************************************
                                      ;*SUBTEST       SUBR    WRITE IN ALL CSR'S
                                      ;*****************************************************************************
10833 041354                                  PUSH    R1
10834 041356  013701  002216                  MOV     TOTCSRS,R1      ;GET CSR'S BYTE
10835 041362                                  BEGIN LCSROUT
10836 041362                                    FOR CSRNO := #0 TO #36 BY #2
10837 041366  006301                                ASL         R1
10838 041370                                      ON.ERROR
10839 041372  104425                                    LOADCSR
10840 041374                                      END ;OF ON.ERROR
10841 041374                                    IF R1 EQ #0 THEN LEAVE LCSROUT
10842 041400                                    END ;OF FOR CSRNO
10843 041416                                  END LCSROUT
10844 041416                                  POP     R1
10845 041420  000207                          RETURN
10846
10847 041422                          $INVALID:       SUBTST  <<TRAP  INVALIDATE BACKGROUND PATTERN>>
                                      ;*****************************************************************************
                                      ;*SUBTEST       TRAP    INVALIDATE BACKGROUND PATTERN
                                      ;*****************************************************************************
10848 041422                                  PUSH    R0,R1
10849 041426  013701  002100                  MOV     BANK,R1
10850 041432  006301                          ASL     R1
10851 041434  006301                          ASL     R1
10852 041436  042761  020000  002626          BIC     #BIT13,CONFIG+2(R1)
10853 041444                                  POP     R1,R0
10854 041450  000002                          RTI
```

```
10856 041452                                $ERRGEN:    SUBTST<<TRAP    GENERATE AND TEST ERROR ADDRESS>>
                                            ;*************************************************************************
                                            ;*SUBTEST    TRAP    GENERATE AND TEST ERROR ADDRESS
                                            ;*************************************************************************
10857 041452                                        PUSH    R0,R1,R2,R3
10858 041462  013703  002102                  MOV     BANKINDEX,R3
10859 041466  005737  002426                  TST     NOSUPER
10860 041472  001003                          BNE     6$
10861 041474  013700  172246                  MOV     SIPAR3,R0               ;GENERATE WHAT ERROR ADDR SHOULD BE
10862 041500  000402                          BR      7$
10863 041502  013700  177646          6$:     MOV     UIPAR3,R0
10864 041506  072027  177773          7$:     ASH     #-5,R0
10865 041512  005737  002130                  TST     EUFLAG
10866 041516  001002                          BNE     1$
10867 041520  042700  177600                  BIC     #^C177,R0
10868 041524  000301                1$:       SWAB    R1                      ;GET CURRENT ADDRESS BITS 11 AND 12
10869 041526  006201                          ASR     R1
10870 041530  006201                          ASR     R1
10871 041532  006201                          ASR     R1
10872 041534  042701  177775                  BIC     #^C2,R1
10873 041540  060100                          ADD     R1,R0                   ;ADD THEM TO THE ADJUSTED PAR VALUE
10674                                         ;GET ERROR ADDRESS FROM CSR UNDER TEST
10875 041542  013701  002144                  MOV     CSR,R1
10876 041546  072127  177773                  ASH     #-5,R1
10877 041552  042701  177600                  BIC     #^C177,R1
10878 041556  005737  002424                  TST     NO22BIT                 ;IS THIS AN 11/44 OR 11/24?
10879 041562  001024                          BNE     2$                      ;BRANCH IF NOT NECESSARY
10880 041564  005737  002130                  TST     EUFLAG                  ;IS IT EUB?
10881 041570  001421                          BEQ     2$                      ;BRANCH IF NOT
10882 041572                                  PUSH    R0                      ;SAVE GENERATED ERROR ADDRESS
10883 041574  013702  002146                  MOV     CSRNO,R2                ;GET CSR NUMBER
10884 041600  052762  040000  172100          BIS     #BIT14,CSRADD(R2)       ;TURN ON EUB BIT CAREFULLY
10885 041606  016200  172100                  MOV     CSRADD(R2),R0           ;GET CSR CONTENTS
10886 041612  042762  040000  172100          BIC     #BIT14,CSRADD(R2)       ;TURN OFF EUB BIT CAREFULLY
10887 041620  042700  177037                  BIC     #^C740,R0               ;CLEAR EVERYTHING BUT ERROR ADDR
10888 041624  006300                          ASL     R0
10889 041626  006300                          ASL     R0                      ;SHIFT ADDR BITS 18-21 INTO POSITION
10890 041630  060001                          ADD     R0,R1                   ;ADD TO CURRENT ERROR ADDRESS
10891 041632                                  POP     R0
10892 041634  020001                2$:       CMP     R0,R1                   ;COMPARE REAL AND GENERATED ERR. ADDR.
10893 041636  001420                          BEQ     5$                      ;BRANCH IF THEY ARE THE SAME
10894 041640  005737  002134                  TST     INTFLAG                 ;INTERLEAVED?
10895 041644  001411                          BEQ     3$                      ;NO - WE HAVE AN ERROR
10896 041646  062700  000100                  ADD     #100,R0
10897 041652  005737  002136                  TST     INT64K                  ;64K INTERLEAVED MEMORY?
10898 041656  001002                          BNE     4$
10899 041660  062700  000100                  ADD     #100,R0
10900 041664  020001                4$:       CMP     R0,R1
10901 041666  001404                          BEQ     5$
10902 041670  005737  002064          3$:     TST     SKPERR                  ;ARE WE SUPPOSED TO SKIP ERROR P.O.?
10903 041674  001001                          BNE     5$                      ;YES - SKIP ERROR PRINTOUT
10904 041676  104462                          PERR36                          ;ELSE PRINT ERROR ADDRESS ERROR
10905 041700  010137  002430          5$:     MOV     R1,ERRADD               ;SAVE CSR'S ERROR ADDRESS
10906 041704  005037  002064                  CLR     SKPERR                  ;ENABLE THE ERROR PRINTOUT AGAIN
10907 041710                                  POP     R3,R2,R1,R0             ;RESTORE REGISTERS
10908 041720  000002                          RTI
```

```
10911 041722                          CHKGEN: SUBTST<<SUBR    GENERATE CHECK BITS>>
                                      ;*****************************************************************************
                                      ;*SUBTEST       SUBR    GENERATE CHECK BITS
                                      ;*****************************************************************************
10912                                         ;CHECK BIT GENERATOR ROUTINE
10913                                         ;CALLING SEQUENCE IS:
10914                                         ;       MOV     #WORD1,SOURCE   ;SOURCE = ADDRESS OF DATA
10915                                         ;       CALL    CHKGEN
10916                                         ;
10917                                         ;CHECK BITS RETURNED IN BITS 11-5 OF LOCATION CHECK
10918                                         ;
10919 041722                                  PUSH    R0,R1,R2,R3,R4,R5
10920 041736  012702  000077                  MOV     #77,R2          ;DEFAULT CHECKBITS FOR DOUBLE WORD OF ZEROS
10921 041742  012703  042030                  MOV     #CHKTAB,R3      ;ADDRESS OF CHECKBIT TABLE
10922 041746  013705  002272                  MOV     SOURCE,R5       ;GET SOURCE ADDRESS
10923 041752  012501                          MOV     (R5)+,R1        ;GET LSB'S
10924 041754  011500                          MOV     (R5),R0         ;GET MSB'S
10925
10926 041756  006704                  1$:     SXT     R4              ;EXTEND SIGN OF DOUBLE WORD TO R4
10927 041760  142304                          BICB    (R3)+,R4        ;ELIMINATE BITS THAT DON'T COUNT
10928 041762  074402                          XOR     R4,R2           ;COMPLEMENT MASKED BITS IN CHECKBITS
10929 041764  073027  000001                  ASHC    #1,R0           ;DOUBLE PRECISION LEFT SHIFT R0,,R1
10930 041770  001372                          BNE     1$              ;LOOP TILL ALL BITS ARE CHECKED
10931
10932 041772  042702  177600                  BIC     #^C177,R2               ;KILL ALL JUNK BITS
10933 041776  000302                          SWAB    R2              ;POSITION CHECKBITS IN BITS 11-5
10934 042000  006202                          ASR     R2
10935 042002  006202                          ASR     R2
10936 042004  006202                          ASR     R2
10937 042006  010237  002274                  MOV     R2,CHECK
10938 042012                                  POP     R5,R4,R3,R2,R1,R0
10939 042026  000207                          RETURN
```

```
10942 042030                    CHKTAB: ;BYTE #3
10943 042030      200                   .BYTE   ^C177           ;BIT 31
10944 042031      301                   .BYTE   ^C076           ;BIT 30
10945 042032      302                   .BYTE   ^C075           ;BIT 29
10946 042033      203                   .BYTE   ^C174           ;BIT 28
10947 042034      304                   .BYTE   ^C073           ;BIT 27
10948 042035      205                   .BYTE   ^C172           ;BIT 26
10949 042036      206                   .BYTE   ^C171           ;BIT 25
10950 042037      307                   .BYTE   ^C070           ;BIT 24
10951                                   ;BYTE #2
10952 042040      310                   .BYTE   ^C067           ;BIT 23
10953 042041      211                   .BYTE   ^C166           ;BIT 22
10954 042042      212                   .BYTE   ^C165           ;BIT 21
10955 042043      313                   .BYTE   ^C064           ;BIT 20
10956 042044      214                   .BYTE   ^C163           ;BIT 19
10957 042045      315                   .BYTE   ^C062           ;BIT 18
10958 042046      316                   .BYTE   ^C061           ;BIT 17
10959 042047      217                   .BYTE   ^C160           ;BIT 16
10960                                   ;BYTE #1
10961 042050      320                   .BYTE   ^C057           ;BIT 15
10962 042051      221                   .BYTE   ^C156           ;BIT 14
10963 042052      222                   .BYTE   ^C155           ;BIT 13
10964 042053      323                   .BYTE   ^C054           ;BIT 12
10965 042054      224                   .BYTE   ^C153           ;BIT 11
10966 042055      325                   .BYTE   ^C052           ;BIT 10
10967 042056      326                   .BYTE   ^C051           ;BIT  9
10968 042057      227                   .BYTE   ^C150           ;BIT  8
10969                                   ;BYTE #0
10970 042060      340                   .BYTE   ^C037           ;BIT  7
10971 042061      241                   .BYTE   ^C136           ;BIT  6
10972 042062      242                   .BYTE   ^C135           ;BIT  5
10973 042063      343                   .BYTE   ^C034           ;BIT  4
10974 042064      244                   .BYTE   ^C133           ;BIT  3
10975 042065      345                   .BYTE   ^C032           ;BIT  2
10976 042066      346                   .BYTE   ^C031           ;BIT  1
10977 042067      247                   .BYTE   ^C130           ;BIT  0
```

```
10980 042070                            SUBTST<<SUBR    MAPPER>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       SUBR    MAPPER
                                        ;*****************************************************************************
10981                                   ;THIS SUBROUTINE MAPS THE MEMORY BANK (16K WORDS = 1 BANK)
10982                                   ;IN R3 TO THE TEST PATTERN AREA (SUPERVISOR VIRTUAL (60000 - 157777) FOR
10983                                   ;THE 11/44 AND 11/45-55; USER VIRTUAL (60000 - 157777) FOR ALL OTHER
10984                                   ;PDP-11'S).
10985                                   ;
10986                                   ;CALL   MOV     BANKNO,R3               ;SET UP BANK ARGUEMENT
10987                                   ;       CALL    MAPPER                  ;ACTUAL CALL
10988                                   ;       RETURN                          ;ONLY RETURN
10989
10990                                   ;SET SUPERVISOR/USER UP FOR 1 TO 1 MAP
10991 042070                            MAPPER: PUSH    R0,R1,R2,R4,R5
10992 042102  012700  172340                    MOV     #KIPAR0,R0              ;FIRST AREA TO MAP TO
10993 042106  012701  172240                    MOV     #SIPAR0,R1              ;FIRST ADDRESS REGISTER
10994 042112  012704  172200                    MOV     #SIPDR0,R4              ;FIRST DESCRIPTOR REGISTER
10995 042116  005737  002426                    TST     NOSUPER                 ;CAN WE USE SUPERVISOR MODE?
10996 042122  001404                            BEQ     4$                      ;YES, BRANCH
10997 042124  012701  177640                    MOV     #UIPAR0,R1              ;FIRST ADDRESS REGISTER
10998 042130  012704  177600                    MOV     #UIPDR0,R4              ;FIRST DESCRIPTOR REGISTER
10999 042134  012702  077406            4$:     MOV     #77406,R2               ;CONSTANT FOR 4K PAGE, UP, R/W
11000 042140  012705  000010                    MOV     #8.,R5                  ;COUNTER
11001 042144  012021                    1$:     MOV     (R0)+,(R1)+             ;PUT IN SUPERVISOR ADDRESS
11002 042146  010224                            MOV     R2,(R4)+                ;PUT IN SUPERVISOR DESCRIPTOR
11003 042150  077503                            SOB     R5,1$                   ;LOOP TILL DONE
11004 042152  012741  177600                    MOV     #177600,-(R1)           ;CORRECT LAST FIELD FOR PERIPHERALS PAGE
11005
11006                                   ;SET UP SUPERVISOR/USER FOR TEST AREA
11007 042156  022703  000170                    CMP     #120.,R3                ;MAP NOTHING (1 TO 1)?
11008 042162  001516                            BEQ     3$                      ;YES - SKIP
11009 042164  072327  000011                    ASH     #9.,R3                  ;BANK 1 STARTS AT 100,000 LESS 6 LSB'S
11010                                                                           ;FOR MEMORY MANAGEMENT = 1000
11011 042170  012701  172246                    MOV     #SIPAR3,R1              ;SETUP FOR AUTO INCREMENTING
11012 042174  005737  002426                    TST     NOSUPER                 ;DO WE HAVE SUPERVISOR MODE?
11013 042200  001402                            BEQ     5$                      ;YES - BRANCH
11014 042202  012701  177646                    MOV     #UIPAR3,R1              ;SETUP FOR AUTO INCREMENTING
11015 042206  012702  000004            5$:     MOV     #4,R2                   ;COUNTER
11016 042212  010321                    2$:     MOV     R3,(R1)+                ;PLUG IN PAR INFO
11017 042214  062703  000200                    ADD     #200,R3                 ;BUMP ADDRESS 4K
11018 042220  077204                            SOB     R2,2$                   ;LOOP TILL DONE
11019 042222  005737  002232                    TST     SPLTCSR
11020 042226  001442                            BEQ     9$
11021 042230  162701  000010                    SUB     #10,R1
11022 042234  010102                            MOV     R1,R2
11023 042236  062702  000004                    ADD     #4,R2
11024 042242  022737  000001  002232            CMP     #1,SPLTCSR
11025 042250  001403                            BEQ     10$
11026 042252  010200                            MOV     R2,R0
11027 042254  010102                            MOV     R1,R2
11028 042256  010001                            MOV     R0,R1
11029 042260  012122                    10$:    MOV     (R1)+,(R2)+
11030 042262  011112                            MOV     (R1),(R2)
11031 042264  013700  002102                    MOV     BANKINDEX,R0
11032 042270  005737  002136                    TST     INT64K
11033 042274  001403                            BEQ     11$
```

```
11034 042276 012700 004000                 MOV     #4000,R0
11035 042302 000402                        BR      12$
11036 042304 012700 010000          11$:   MOV     #10000,R0
11037 042310 005737 002426          12$:   TST     NOSUPER
11038 042314 001403                        BEQ     13$
11039 042316 012701 177652                 MOV     #UIPAR5,R1
11040 042322 000402                        BR      14$
11041 042324 012701 172252          13$:   MOV     #SIPAR5,R1
11042 042330 060021                 14$:   ADD     R0,(R1)+
11043 042332 060011                        ADD     R0,(R1)
11044                                      ;IF WE ONLY HAVE AN 124K SYSTEM, WE DON'T WANT TO TEST THE
11045                                      ;LAST 4K, WHERE THE UNIBUS DEVICE PAGE IS. INSTEAD, THE
11046                                      ;PROGRAM WILL REMAP THE LAST 4K TO 8-12K. ALSO, IF THERE
11047                                      ;IS A BANK 177 ON AN 11/44, THE PROGRAM WILL REMAP THE LAST
11048                                      ;4K TO 8-12K FOR THE SAME REASON.
11049 042334 022737 000007 002526   9$:    CMP     #7,LASTBANK
11050 042342 001010                        BNE     7$
11051 042344 005737 002424                 TST     NO22BIT                 ;11/44 OR 24?
11052 042350 001423                        BEQ     3$                      ;BRANCH IF SO
11053 042352 022737 000007 002100          CMP     #7,BANK                 ;BANK 7?
11054 042360 001017                        BNE     3$                      ;NO - BRANCH
11055 042362 000404                        BR      8$
11056 042364 022737 000177 002526   7$:    CMP     #177,LASTBANK
11057 042372 001012                        BNE     3$
11058 042374 005737 002426          8$:    TST     NOSUPER
11059 042400 001404                        BEQ     6$
11060 042402 013737 177652 177654          MOV     UIPAR5,UIPAR6
11061 042410 000403                        BR      3$
11062 042412 013737 172252 172254   6$:    MOV     SIPAR5,SIPAR6
11063 042420                        3$:    POP     R5,R4,R2,R1,R0
11064 042432 000207                        RETURN
11065                                      .SBTTL  TRAP    MAP KERNEL (ALMOST 1 TO 1) TRAP HANDLER
11066 042434                        $KMAP: PUSH    R0,R1,R2,R3,R4
11067 042446 005000                        CLR     R0                      ;1ST AREA TO MAP TO
11068 042450 012701 172340                 MOV     #KIPAR0,R1              ;FIRST ADDRESS
11069 042454 012702 077406                 MOV     #77406,R2               ;CONSTANT FOR 4K PAGE,UP,R/W
11070 042460 012703 172300                 MOV     #KIPDR0,R3              ;1ST PAGE DESCRIPTOR REGISTER
11071 042464 012704 000010                 MOV     #8.,R4                  ;COUNTER
11072 042470 010021                 1$:    MOV     R0,(R1)+                ;PUT IN KERNEL ADDRESS
11073 042472 010223                        MOV     R2,(R3)+                ;PUT IN KERNEL DISCRIPTOR
11074 042474 062700 000200                 ADD     #200,R0                 ;ADD ADDRESS CONSTANT FOR 4K CHANGE
11075 042500 077405                        SOB     R4,1$                   ;LOOP TILL DONE
11076 042502 012741 177600                 MOV     #177600,-(R1)           ;THE PERIPHERALS PAGE TO KIPAR7
11077 042506 012741 177400                 MOV     #177400,-(R1)           ;AND NEXT LOWER PAGE TO KIPAR6
11084 042512                               POP     R4,R3,R2,R1,R0
11085 042524 000002                        RTI
```

```
 11088 042526                          RELOCATE:SUBTST <<RELOCATE PROGRAM>>
                                       ;*****************************************************************************
                                       ;*SUBTEST       RELOCATE PROGRAM
                                       ;*****************************************************************************
 11089 042526                                  IF #SW12 SET.IN @SWR THEN $RETURN ERROR
 11090 042542                                  IF APTFLAG IS TRUE OR ACTFLAG IS TRUE
 11091 042556                                    IF $PASS NE #0 THEN $RETURN ERROR
 11092 042570                                  END; OF IF APTFLAG
 11093 042570                                  BEGIN LOADERBANK
 11094 042570                                    FOR BANK := #1 TO LASTBANK
 11095 042576  004737  044300                      CALL EXBANK
 11096 042602                                      IF ACFLAG IS TRUE AND PFLAG IS FALSE AND BMFLAG IS FALSE
 11097 042624  013700  002100                        MOV       BANK,R0
 11098 042630  010037  002402                        MOV       R0,LOADBANK
 11099 042634  013701  002536                        MOV       LOADHOME,R1
 11100 042640  004737  043750                        CALL      BANKMOV
 11101 042644  004737  044246                        CALL      NEWLOAD          ;MAP NEW LOADER BANK IN KERNEL
 11102 042650  013701  002102                        MOV       BANKINDEX,R1
 11103 042654  052761  100000  002626                BIS       #BIT15,CONFIG+2(R1)     ;MARK LOADER
 11104 042662  042761  020000  002626                BIC       #BIT13,CONFIG+2(R1)     ;INVALIDATE BACKGROUND PATTERN
 11105 042670                                        LEAVE LOADERBANK
 11106 042672                                      END ;OF IF ACFLAG
 11107 042672                                    END ;OF FOR BANK
 11108 042706                                    IF #SW13 OFF.IN @SWR
 11109 042716                                      TYPE        MSG075           ;RELOCATION NOT POSSIBLE
 11110 042722                                    END ;OF IF #SW13
 11111 042722                                    $RETURN ERROR
 11112 042726                                  END LOADERBANK
 11113 042726                                  BEGIN FINDBANK
 11114 042726  013702  002526                  MOV     LASTBANK,R2
 11115 042732  006302                          ASL     R2
 11116 042734  006302                          ASL     R2                ;R2 <- R2 * 4
 11117 042736                                    FOR R1 := #2*2 TO R2 BY #4
 11118 042742                                      IF #BIT7!BIT0 OFF.IN CONFIG(R1) ;IF NO ERRORS & NOT PROGRAM SPACE
 11119 042752                                        IF #BIT15 OFF.IN CONFIG+2(R1) ;IF NOT LOADER BANK
 11120 042762                                          IF CPUBIT SET.IN CONFIG(R1) ;IF ACCESSABLE
 11121 042772                                            IF #BIT9 SET.IN CONFIG+2(R1) THEN LEAVE FINDBANK ;IF PARITY
 11122 043002                                              IF #BIT6 SET.IN CONFIG(R1) AND #BIT7 OFF.IN CONFIG(R1)
 11123                                                       ;IF 1ST PROTECTABLE ECC BANK
 11124 043022                                                LEAVE FINDBANK
 11125 043024                                              END ;OF IF #BIT6
 11126 043024                                              IF INHECC IS FALSE
 11127 043032                                                SET   INHECC
 11128 043040  010137  002510                              MOV   R1,INHBANK
 11129 043044                                              END; OF IF INHECC
 11130 043044                                          END ;OF IF CPUBIT
 11131 043044                                        END ;OF IF #BIT15
 11132 043044                                      END ;OF IF #BIT7
 11133 043044                                    END ;OF FOR
 11134 043054                                    IF FULLREL IS FALSE
 11135 043062                                      IF INHECC IS TRUE
 11136 043070  013701  002510                        MOV     INHBANK,R1
 11137 043074  023727  002260  000030                CMP     REALPAT,#30       ;IS THIS PATTERN 30?
 11138 043102  001423                                BEQ     RELENT1           ;YES - SKIP MESSAGE
 11139 043104                                        TYPE    MSG123
 11140 043110  000420                                BR      RELENT1
 11141 043112                                      END; OF IF INHECC
```

```
11142 043112                                        END; OF IF FULLREL
11143 043112  005037  002506                        CLR   INHECC                ;MAKE SURE FLAG IS TURNED OFF!
11144 043116                                        IF #SW13 OFF.IN @SWR
11145 043126  023727  002260  000030                  CMP         REALPAT,#30   ;IS THIS PATTERN 30?
11146 043134  001402                                  BEQ         SKUB          ;YES - SKIP MESSAGE
11147 043136                                          TYPE        MSG075        ;RELOCATION NOT POSSIBLE
11148 043142                                        END ;OF IF #SW13
11149 043142                              SKUB:     $RETURN ERROR
11150 043146                                    END FINDBANK
11151 043146                                    CLEAR   INHECC                  ;IF WE RELOCATED PROPERLY, THIS SHOULD BE OFF!
11152 043152  042761  020000  002626  RELENT1: BIC     #BIT13,CONFIG+2(R1)      ;INVALIDATE BACKGROUND PATTERN
11153 043160  005000                            CLR     R0
11154 043162  071027  000004                    DIV     #4,R0
11155 043166                              RELOC1: LET NEWBANK := R0
11156 043172  013737  002502  002504            MOV     PGMCSR,PGMCSR+2         ;SAVE CURRENT PGM. CSR
11157 043200  004737  044116                    CALL    USERMAP                 ;MAP NEWBANK TO USER PAR
11158 043204                                    USER                            ;ENTER USER MODE
11159 043212                                    BMOV    0,100000,SIZE           ;MOVE PROGRAM
11160 043224  104417                            KERNEL                          ;ENTER KERNEL MODE
11161 043226  022737  000001  003716            CMP     #1,PROTYP               ;IS THIS AN 11/44 ?
11162 043234  001021                            BNE     JMPRL1                  ;JUMP IF NOT
11163 043236  042737  000040  172516            BIC     #BIT5,MMR3              ;TURN OFF UNIBUS MAP
11164 043244  013700  002270                    MOV     NEWBANK,R0
11165 043250  006200                            ASR     R0
11166 043252                                    ON.ERROR
11167 043254  012737  100000  170200              MOV   #BIT15,MAPL0
11168 043262                                    END ;OF ON.ERROR
11169 043262  010037  170202                    MOV     R0,MAPH0
11170 043266  004737  043704                    CALL    LOWMAP                  ;SETUP LOWER 16K IN UNIBUS MAP
11171 043272  052737  000040  172516            BIS     #BIT5,MMR3              ;ENERGIZE UNIBUS MAP
11172 043300  042737  000001  177572  JMPRL1:   BIC     #BIT0,MMR0              ;DEENERGIZE MEMORY MANAGEMENT
11173 043306  004737  044200                    CALL    NEWKERNEL
11174 043312  013700  002270                    MOV     NEWBANK,R0
11175 043316  006300                            ASL     R0
11176 043320  006300                            ASL     R0                      ;R0 <- R0 * 4
11177 043322  016002  002624                    MOV     CONFIG(R0),R2
11178 043326  000302                            SWAB    R2
11179 043330  042702  177760                    BIC     #^C17,R2
11180 043334  006302                            ASL     R2
11181 043336  052737  000001  177572            BIS     #BIT0,MMR0              ;ENERGIZE MEMORY MANAGEMENT
11182 043344  010237  002502                    MOV     R2,PGMCSR               ;PUT NEW PGM. CSR INTO PGMCSR
11183 043350  032760  010000  002626            BIT     #BIT12,CONFIG+2(R0)     ;IS THE NEW BANK INTERLEAVED?
11184 043356  001412                            BEQ     1$                      ;BRANCH IF NOT INTERLEAVED
11185 043360  016002  002624                    MOV     CONFIG(R0),R2
11186 043364  042702  007777                    BIC     #^C170000,R2
11187 043370  072227  177775                    ASH     #-3,R2
11188 043374  052702  100000                    BIS     #BIT15,R2
11189 043400  050237  002502                    BIS     R2,PGMCSR
11190 043404                              1$:   SET     RLFLAG
11191 043412                                    $RETURN NOERROR
```

```
11194 043416                                UNRELOCATE:SUBTST       <<UNRELOCATE PROGRAM>>
                                            ;*************************************************************************
                                            ;*SUBTEST       UNRELOCATE PROGRAM
                                            ;*************************************************************************
11195                                               ;RESTORE LOADERS
11196 043416                                        PUSH    R0
11197 043420  013701  002402                        MOV     LOADBANK,R1
11198 043424  013700  002536                        MOV     LOADHOME,R0
11199 043430  004737  043750                        CALL    BANKMOV
11200 043434  004737  044246                        CALL    NEWLOAD                 ;MAP NEW LOADER BANK IN KERNEL SPACE
11201 043440                                        PUSH    BANK
11202 043444  013737  002402  002100                MOV     LOADBANK,BANK
11203 043452  004737  044300                        CALL    EXBANK
11204 043456  013701  002102                        MOV     BANKINDEX,R1
11205 043462  042761  100000  002626                BIC     #BIT15,CONFIG+2(R1)     ;CLEAR LOADER FLAG
11206 043470  013737  002536  002100                MOV     LOADHOME,BANK
11207 043476  004737  044300                        CALL    EXBANK
11208 043502  013701  002102                        MOV     BANKINDEX,R1
11209 043506  042761  020000  002626                BIC     #BIT13,CONFIG+2(R1)     ;INVALIDATE BACKGROUND PATTERN
11210 043514                                        POP     BANK
11211 043520                                        CLEAR   INHECC                  ;MAKE SURE ECC TESTS ARE NOT INHIBITED!
11212
11213                                               ;RESTORE BANK 0
11214 043524  042737  020000  002626                BIC     #BIT13,CONFIG+2         ;INVALIDATE BACKGROUND PATTERN
11215 043532                                        LET NEWBANK := #0
11216 043536  004737  044116                        CALL    USERMAP                 ;MAP NEWBANK TO USER PAR
11217 043542                                        USER                            ;ENTER USER MODE
11218 043550                                        BMOV    0,100000,SIZE           ;MOVE PROGRAM
11219 043562  104417                                KERNEL                          ;ENTER KERNEL MODE
11220 043564  042737  000001  177572                BIC     #BIT0,MMR0              ;DEENERGIZE MEMORY MANAGEMENT
11221 043572  004737  044200                        CALL    NEWKERNEL
11222 043576  013737  002504  002502                MOV     PGMCSR+2,PGMCSR         ;RESTORE PREVIOUS PGM. CSR
11223 043604  052737  000001  177572                BIS     #BIT0,MMR0              ;ENERGIZE MEMORY MANAGEMENT
11224 043612  005037  002124                        CLR     RLFLAG
11225 043616  022737  000001  003716                CMP     #1,PROTYP               ;IS THIS AN 11/44 ?
11226 043624  001014                                BNE     1$
11227 043626  042737  000040  172516                BIC     #BIT5,MMR3              ;TURN OFF UNIBUS MAP
11228 043634                                        CLEAR   MAPL0,MAPH0
11229 043644  004737  043704                        CALL    LOWMAP                  ;SETUP LOWER 16K OF UNIBUS MAP
11230 043650  052737  000040  172516                BIS     #BIT5,MMR3              ;ENERGIZE UNIBUS MAP
11231 043656  012700  002626          1$:           MOV     #CONFIG+2,R0    ;MOVE 2ND WORD OF CONFIG TO R0
11232 043662  042710  020000          2$:           BIC     #BIT13,(R0)     ;CLEAR BACKGROUND VALID BIT
11233 043666  062700  000004                        ADD     #4,R0           ;INCREMENT TO NEXT BANK
11234 043672  020027  003620                        CMP     R0,#3620        ;DONE?
11235 043676  003771                                BLE     2$              ;NO - BRANCH
11236 043700                                        POP     R0
11237 043702  000207                                RETURN
11238
11239 043704                                LOWMAP: SUBTST  <<SETUP LOWER 16K OF UNIBUS MAP>>
                                            ;*************************************************************************
                                            ;*SUBTEST       SETUP LOWER 16K OF UNIBUS MAP
                                            ;*************************************************************************
11240 043704                                        PUSH    R0,R1,R2
11241 043712  012700  170200                        MOV     #MAPL0,R0
11242 043716  012701  170204                        MOV     #MAPL1,R1
11243 043722  012702  000003                        MOV     #3,R2
11244 043726  012011                          1$:   MOV     (R0)+,(R1)
```

SETUP LOWER 16K OF UNIBUS MAP

```
11245 043730  062721  020000          ADD     #BIT13,(R1)+
11246 043734  012021                  MOV     (R0)+,(R1)+
11247 043736  077205                  SOB     R2,1$
11248 043740                          POP     R2,R1,R0
11249 043746  000207                  RETURN
```

```
 11252 043750                           BANKMOV:SUBTST  <<MOVE BANKS>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       MOVE BANKS
                                        ;*****************************************************************************
 11253                                          ;MOVE 3/4 OF A BANK
 11254                                          ;CALLING SEQUENCE
 11255                                          ;R0 = DESTINATION BANK
 11256                                          ;R1 = SOURCE BANK
 11257 043750  104415                           SAVREG
 11258 043752  004737  044116                   CALL    USERMAP
 11259 043756  104416                           RESREG
 11260 043760  104415                           SAVREG
 11261 043762  072027  000011                   ASH     #9.,R0
 11262 043766  072127  000011                   ASH     #9.,R1
 11263 043772  012702  177650                   MOV     #UIPAR4,R2
 11264 043776  012703  000200                   MOV     #200,R3
 11265
 11266 044002  010122                           MOV     R1,(R2)+                ;MAP 1ST HALF BANK
 11267 044004  060301                           ADD     R3,R1                   ;BUMP BY 4K
 11268 044006  010122                           MOV     R1,(R2)+
 11269 044010  060301                           ADD     R3,R1
 11270
 11271 044012  010022                           MOV     R0,(R2)+
 11272 044014  060300                           ADD     R3,R0
 11273 044016  010022                           MOV     R0,(R2)+
 11274 044020  060300                           ADD     R3,R0
 11275
 11276 044022                                   USER
 11277 044030                                   BMOV    100000,140000,SIZE/2    ;MOV 1ST HALF BANK
 11278 044042  104417                           KERNEL                          ;ENTER KERNEL MODE
 11279
 11280 044044  012702  177650                   MOV     #UIPAR4,R2
 11281
 11282 044050  010122                           MOV     R1,(R2)+                ;MAP 2ND HALF BANK
 11283 044052  060301                           ADD     R3,R1                   ;BUMP BY 4K
 11284 044054  010122                           MOV     R1,(R2)+
 11285 044056  060301                           ADD     R3,R1
 11286
 11287 044060  010022                           MOV     R0,(R2)+
 11288 044062  060300                           ADD     R3,R0
 11289 044064  010022                           MOV     R0,(R2)+
 11290 044066  060300                           ADD     R3,R0
 11291
 11292 044070                                   USER
 11293 044076                                   BMOV    100000,140000,SIZE/4    ;MOV 3ND FOURTH OF BANK
 11294 044110  104417                           KERNEL                          ;ENTER KERNEL MODE
 11295
 11296 044112  104416                           RESREG
 11297 044114  000207                           RETURN
```

```
 11300 044116                           USERMAP:SUBTST  <<SUBR  MAP USER TO NEW BANK>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       SUBR    MAP USER TO NEW BANK
                                        ;*****************************************************************************
 11301 044116  012701  177640                   MOV     #UIPAR0,R1              ;COPY KERNEL PAR'S & PDR'S (0-3)
 11302 044122  012702  172340                   MOV     #KIPAR0,R2
 11303 044126  012703  177600                   MOV     #UIPDR0,R3
 11304 044132  012704  172300                   MOV     #KIPDR0,R4
 11305 044136  012705  000004                   MOV     #4,R5
 11306 044142  012221                   1$:     MOV     (R2)+,(R1)+
 11307 044144  011423                           MOV     (R4),(R3)+
 11308 044146  077503                           SOB     R5,1$
 11309
 11310 044150  013700  002270                   MOV     NEWBANK,R0
 11311 044154  072027  000011                   ASH     #9.,R0                  ;BANK 1 STARTS AT 100,000 LESS 6 LSB'S
 11312                                                                          ;FOR MEMORY MANAGEMENT = 1000
 11313 044160  012705  000004                   MOV     #4,R5
 11314 044164  010021                   2$:     MOV     R0,(R1)+                ;SETUP UIPAR(4-7)
 11315 044166  062700  000200                   ADD     #200,R0                 ;BUMP ADDRESS 4K
 11316 044172  011423                           MOV     (R4),(R3)+              ;SETUP UIPDR(4-7)
 11317 044174  077505                           SOB     R5,2$
 11318 044176  000207                           RETURN
 11319
 11320 044200                           NEWKERNEL:SUBTST        <<SUBR  SETUP KERNEL PAR'S FOR NEW BANK>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       SUBR    SETUP KERNEL PAR'S FOR NEW BANK
                                        ;*****************************************************************************
 11321 044200                                   PUSH    R0,R1,R5
 11322 044206  012700  172340                   MOV     #KIPAR0,R0
 11323 044212  013701  002270                   MOV     NEWBANK,R1
 11324 044216  072127  000011                   ASH     #9.,R1                  ;BANK 1 STARTS AT 100,000 LESS 6 LSB'S
 11325                                                                          ;FOR MEMORY MANAGEMENT = 1000
 11326 044222  012705  000004                   MOV     #4,R5
 11327 044226  010120                   1$:     MOV     R1,(R0)+                ;SETUP KIPAR(0-3)
 11328 044230  062701  000200                   ADD     #200,R1
 11329 044234  077504                           SOB     R5,1$
 11330 044236                                   POP     R5,R1,R0
 11331 044244  000207                           RETURN
 11332
 11333 044246                           NEWLOAD:SUBTST  <<SUBR  SETUP KERNEL PAR'S FOR NEW LOADER BANK>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       SUBR    SETUP KERNEL PAR'S FOR NEW LOADER BANK
                                        ;*****************************************************************************
 11334                                          ;R0 CONTAINS THE DESTINATION BANK
 11335 044246                                   PUSH    R0,R1
 11336 044252  012701  172350                   MOV     #KIPAR4,R1
 11337 044256  072027  000011                   ASH     #9.,R0                  ;BANK 1 STARTS AT 100000 LESS 6 LSB'S (1000)
 11338 044262  010021                           MOV     R0,(R1)+                ;SETUP KIPAR4
 11339 044264  062700  000200                   ADD     #200,R0
 11340 044270  010021                           MOV     R0,(R1)+                ;SETUP KIPAR5
 11341 044272                                   POP     R1,R0
 11342 044276  000207                           RETURN
```

```
 11345 044300                    EXBANK: SUBTST  <<SUBR  EXAMINE BANK>>
                                 ;*****************************************************************************
                                 ;*SUBTEST        SUBR    EXAMINE BANK
                                 ;*****************************************************************************
 11346                           ;DOES THE FOLLOWING:
 11347                           ;(1)  SETS UP "BANKINDEX" AND R1 BASED ON VALUE OF "BANK".
 11348                           ;(2)  SETS THE "MKFLAG" IF THE BANK IS ECC.
 11349                           ;(3)  SETS THE "KFLAG" IF THE BANK IS MF11S-K.
 11350                           ;(4)  SETS THE "KPFLAG" IF THE BANK IS THE PROTECTED REGION OF ECC MEMORY.
 11351                           ;(5)  SETS THE "ACFLAG" IF THE BANK CAN BE ACCESSED BY THIS CPU.
 11352                           ;(6)  SETS THE "PFLAG" IF THE BANK IS IN PROGRAM SPACE.
 11353                           ;(7)  SETS THE  "RRFLAG" IF RELOCATION IS REQUIRED TO TEST THIS BANK; HOWEVER,
 11354                           ;     IT COMPLEMENTS THIS FLAG IF THE RELOCATION FLAG "RLFLAG" IS SET (THIS IS
 11355                           ;     NECESSARY FOR THE USE OF THE RECURSIVE "MODE" SUBROUTINES). THE "RRFLAG"
 11356                           ;     IS ALWAYS SET TO DISABLE TESTING IF FIELD SERVICE MODE  "SELECTED BANKS"
 11357                           ;     ARE BEING TESTED AND THIS BANK IS NOT SELECTED.
 11358                           ;(8)  SETS THE "BMFLAG" IF THE BANK IS A BAD MEMORY;  HOWEVER, IT  COMPLEMENTS
 11359                           ;     THIS FLAG IF THE "WORST" FLAG IS NOT SET  (THIS IS NECESSARY FOR THE USE
 11360                           ;     OF THE RECURSIVE "MODE" SUBROUTINES).
 11361                           ;(9)  SETS THE "EUFLAG" IF THE BANK HAS EXTENDED UNIBUS MEMORY.
 11362                           ;(10) SETS THE "INTFLAG" IF THE BANK IS INTERLEAVED.
 11363                           ;(11) SETS THE "INT64K" FLAG IF THE BANK IS INTERLEAVED ON 64K WORD BOUNDS.
 11364                           ;(12) SETS THE "SKIPMK" FLAG IF THIS BANK IS INTERLEAVED, AND HAS ALREADY
 11365                           ;     BEEN TESTED.
 11366                           ;
 11367 044300                      PUSH    R0,R1,R2
 11368 044306                      CLEAR   MKFLAG,KPFLAG,KFLAG,EUFLAG
 11369 044326                      SET     ACFLAG
 11370 044334                      CLEAR   PFLAG,RRFLAG,BMFLAG
 11371 044350                      CLEAR   INTFLAG,INT64K,SKIPMK
 11372 044364  013701  002100      MOV     BANK,R1
 11373 044370  006301              ASL     R1
 11374 044372  006301              ASL     R1                  ;R1 <- R1 * 4
 11375 044374  010137  002102      MOV     R1,BANKINDEX
 11376 044400  032761  000100  002624  BIT     #BIT6,CONFIG(R1)        ;PROTECTED REGION OF ECC MEMORY?
 11377 044406  001403              BEQ     1$                      ;NO - SKIP
 11378 044410                      SET     KPFLAG
 11379 044416  012700  000002      1$:     MOV     #BIT1,R0
 11383 044422                      IF R0 SET.IN CPUBIT AND R0 OFF.IN CONFIG(R1)
 11384 044436  005037  002114          CLR     ACFLAG
 11385 044442                      END ;OF IF R0
 11390 044442  005737  002114      TST     ACFLAG                  ;ACTIVE MEMORY?
 11391 044446  001415              BEQ     12$                     ;BRANCH IF NOT
 11392 044450  016102  002626      MOV     CONFIG+2(R1),R2
 11393 044454  000302              SWAB    R2
 11394 044456  042702  177770      BIC     #^C7,R2                 ;ISOLATE MEM TYPE BITS
 11395 044462  020227  000002      CMP     R2,#2                   ;IS THIS AN ILLEGAL MEM TYPE?
 11396 044466  003005              BGT     12$                     ;BRANCH IF NOT
 11397 044470                      SET     BMFLAG                  ;SET BAD BANK FLAG
 11398 044476  000137  044740      JMP     ENEXBK                  ;JUMP OVER REST OF FLAG TESTS
 11399 044502  032761  000400  002626  12$:    BIT     #BIT8,CONFIG+2(R1)      ;IS THIS EUB?
 11400 044510  001003              BNE     2$                      ;BRANCH IF NOT
 11401 044512                      SET     EUFLAG                  ;YES - SET EUB FLAG
 11402 044520  032761  001000  002626  2$:     BIT     #BIT9,CONFIG+2(R1)      ;IS THERE ECC THERE?
 11403 044526  001012              BNE     3$                      ;NO - SKIP
 11404 044530                      SET     MKFLAG          ;YES - SET MKFLAG
 11405 044536  032761  000400  002626      BIT     #BIT8,CONFIG+2(R1)      ;IS THIS MF11S-K MEMORY
```

```
11406 044544  001403                          BEQ     3$                      ;NO - IT'S MS11-M
11407 044546                                  SET     KFLAG                   ;YES - SET KFLAG
11408 044554  032761  000200  002624  3$:     BIT     #BIT7,CONFIG(R1)        ;BANK = PROGRAM SPACE?
11409 044562  001406                          BEQ     5$                      ;NO - SKIP
11410 044564                                  SET     PFLAG,RRFLAG
11411 044600  005737  002124          5$:     TST     RLFLAG                  ;IS PROGRAM RELOCATED?
11412 044604  001402                          BEQ     6$                      ;NO - SKIP
11413 044606  005137  002122                  COM     RRFLAG                  ;YES - COMPLEMENT RELOCATION REQUIRED FLAG
11414 044612  032761  000001  002624  6$:     BIT     #BIT0,CONFIG(R1)        ;ERRORS PRESENT IN THIS BANK?
11415 044620  001403                          BEQ     8$                      ;NO - SKIP
11416 044622                                  SET     BMFLAG
11417 044630  005737  002540          8$:     TST     WORST                   ;IS THIS A WORST FIRST PASS?
11418 044634  001002                          BNE     9$                      ;YES - SKIP
11419 044636  005137  002126                  COM     BMFLAG                  ;NO - COMPLEMENT BAD MEMORY FLAG
11420 044642                          9$:     IF SELONLY IS TRUE AND #BIT14 OFF.IN CONFIG+2(R1)
11421 044660                                    SET   RRFLAG
11422 044666                                  END ;OF IF SELONLY
11423 044666  032761  010000  002626          BIT     #BIT12,CONFIG+2(R1)     ;IS THIS BANK INTERLEAVED?
11424 044674  001421                          BEQ     ENEXBK                  ;BRANCH IF IT IS NOT
11425 044676                                  SET     INTFLAG
11426 044704  032761  004000  002626          BIT     #BIT11,CONFIG+2(R1)     ;IS THIS BANK INTERLEAVED WITH 64K BOARDS?
11427 044712  001403                          BEQ     10$                     ;BRANCH IF IT IS NOT
11428 044714                                  SET     INT64K
11429 044722  032761  000040  002624  10$:    BIT     #BIT5,CONFIG(R1)        ;SHOULD THIS BANK BE TESTED?
11430 044730  001403                          BEQ     ENEXBK                  ;BRANCH IF IT SHOULD
11431 044732                                  SET     SKIPMK
11432 044740                          ENEXBK: POP     R2,R1,R0                ;RESTORE REGISTERS
11433 044746  000207                          RETURN
```

```
11436 044750                          BANKOK: SUBTST  <<SUBR  BANK OK?>>
                                      ;*****************************************************************************
                                      ;*SUBTEST       SUBR    BANK OK?
                                      ;*****************************************************************************
11437                                         ;TEST TO INSURE THAT THE TYPE OF MEMORY IN THE PRESENT BANK
11438                                         ;IS OF THE TYPE WE ARE TESTING 'TMFLAG'.
11439                                         ;RESULT IS RETURNED IN THE CONDITION CODES (OK = (=0)).
11440 044750  013700  002132                 MOV     TMFLAG,R0
11441 044754  005100                        COM     R0
11442 044756  013701  002116                 MOV     MKFLAG,R1
11443 044762  074001                        XOR     R0,R1
11444 044764  000207                        RETURN                          ;OK = (=OK)
11445
11446 044766                          INCRPT:
11447 044766                          INCPAT: SUBTST  <<SUBR  INCREMENT PATTERN TESTING >>
                                      ;*****************************************************************************
                                      ;*SUBTEST       SUBR    INCREMENT PATTERN TESTING
                                      ;*****************************************************************************
11448                                         ;INCREMENT THE PATTERN & SET UP THE CONDITION CODES
11449                                         ;RESULT - Z BIT SET INDICATES OVERFLOW
11450 044766  005237  002110                 INC     PATTERN
11451 044772  022737  000030  002110         CMP     #30,PATTERN             ;SET UP CONDITION CODES
11452 045000  000207                        RETURN                          ;NOT EQUAL TO ZERO IS GOOD (NO OVERFLOW)
11453
11454 045002                          SETPAT:
11455 045002                          HIPAT:  SUBTST  <<SUBR  SET HIGHEST PATTERN TESTING TYPE>>
                                      ;*****************************************************************************
                                      ;*SUBTEST       SUBR    SET HIGHEST PATTERN TESTING TYPE
                                      ;*****************************************************************************
11456 045002  012737  000027  002110         MOV     #27,PATTERN             ;SET HIGHEST PATTERN
11457 045010  000207                        RETURN
11458
11459 045012                          INCBNK: SUBTST  <<SUBR  INCREMENT BANK & TEST>>
                                      ;*****************************************************************************
                                      ;*SUBTEST       SUBR    INCREMENT BANK & TEST
                                      ;*****************************************************************************
11460                                         ;RESULTS RETURNED IN CONDITION CODES
11461 045012  005237  002100                 INC     BANK
11462 045016  023737  002526  002100         CMP     LASTBANK,BANK           ;TOO FAR?
11463 045024  000207                        RETURN
```

```
 11466 045026                          BOOT:   SUBTST  <<BOOTSTRAP ROUTINE>>
                                       ;*****************************************************************************
                                       ;*SUBTEST       BOOTSTRAP ROUTINE
                                       ;*****************************************************************************
 11467                                         ;INITIALIZE ALL CSR'S
 11468                                         ;UNRELOCATE IF NECESSARY
 11469                                         ;FLUSH OUT ANY DBE'S
 11470                                         ;TURN OFF MEMORY MANAGEMENT
 11471                                         ;TURN OFF THE UNIBUS MAP
 11472                                         ;BOOT RK0 OR RK1
 11473 045026  104472                        ECCINIT         ;TRAP ON DOUBLE BIT ERRORS (NORMAL)
 11474 045030                                SET4    #BOOT1          ;TRAPS TO 4 GOTO BOOT1
 11475 045036                                IF RLFLAG IS TRUE THEN $CALL UNRELOCATE
 11476 045050  004737  024716                CALL    MT0030          ;FLUSH OUT DBE'S
 11477 045054  104421                        DEENERGIZE              ;TURN OFF MEMORY MANAGEMENT
 11478 045056  005737  002424                TST     N022BIT                 ;IS THIS AN 11/44 OR 11/24?
 11479 045062  001003                        BNE     BOOT1
 11480 045064  042737  000040  172516        BIC     #BIT5,MMR3                      ;TURN OFF THE UNIBUS MAP
 11481 045072  005001                BOOT1:  CLR     R1
 11482 045074  000005                1$:     RESET
 11483 045076  012700  177406                MOV     #177406,R0
 11484 045102  010160  000004                MOV     R1,4(R0)
 11485 045106  012710  177400                MOV     #177400,(R0)
 11486 045112  012740  000005                MOV     #5,-(R0)
 11487 045116  105710                2$:     TSTB    (R0)
 11488 045120  100376                        BPL     2$
 11489 045122  062701  020000                ADD     #BIT13,R1
 11490 045126  005710                        TST     (R0)
 11491 045130  100761                        BMI     1$
 11492 045132  005007                        CLR     PC
```

```
11495 045134                            EXIT:   SUBTST  <<HALT PROGRAM>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       HALT PROGRAM
                                        ;*****************************************************************************
11496 045134  004737  045166                      CALL    SHUTUP
11497 045140                            EXIT2:  IF APTFLAG IS TRUE OR ACTFLAG IS TRUE
11498 045154  000777                                BR    .
11499 045156                                      ELSE
11500 045160  000000                    $EXHALT:    HALT
11501 045162  000137  003630                        JMP   START
11502 045166                                      END ;OF IF APTFLAG
11503
11504 045166                            SHUTUP: SUBTST  <<SHUTDOWN DIAGNOSTIC>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       SHUTDOWN DIAGNOSTIC
                                        ;*****************************************************************************
11505                                             ;INITIALIZE ALL CSR'S
11506                                             ;UNRELOCATE
11507                                             ;FLUSH OUT DBE'S
11508                                             ;RESTORE LOADERS
11509                                             ;TURN OFF MEMORY MANAGEMENT
11510                                             ;UNMAP THE UNIBUS MAP
11514 045166  104472                            ECCINIT         ;TRAP ON DOUBLE BIT ERRORS (NORMAL)
11515 045170                                    IF RLFLAG IS TRUE THEN $CALL UNRELOCATE
11516 045202                                    IF QUICK IS FALSE
11517 045210  004737  024716                      CALL  MT0030          ;FLUSH OUT DBE'S
11518 045214                                    END ;OF IF QUICK
11519 045214  012700  000001                    MOV     #1,R0           ;DESTINATION BANK
11520 045220  013701  002536                    MOV     LOADHOME,R1     ;SOURCE BANK
11521 045224  004737  043750                    CALL    BANKMOV
11522 045230  104421                            DEENERGIZE              ;TURN OFF MEMORY MANAGEMENT
11523 045232  005737  002424                    TST     N022BIT         ;DOES THIS PDP-11 HAVE 22-BIT ADDR?
11524 045236  001003                            BNE     1$              ;BRANCH IF NOT
11525 045240  042737  000040  172516            BIC     #BIT5,MMR3              ;TURN OFF UNIBUS MAP
11529 045246  000207                    1$:     RETURN
11530
11531 045250                            APTDOWN:SUBTST  <<APT SHUTDOWN SEQUENCE>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       APT SHUTDOWN SEQUENCE
                                        ;*****************************************************************************
11532 045250                                    MAP     #0                      ;MAP SUPERVISOR SPACE (TEST AREA) TO BANK #0
11533 045264                                    TESTAREA                ;ENTER TEST MODE
11534 045272  012737  045250  060024            MOV     #APTDOWN,FIRST+24
11535 045300  012737  000340  060026            MOV     #340,FIRST+26
11536 045306  012737  000000  125250            MOV     #0,FIRST+APTDOWN
11537 045314  104417                            KERNEL                  ;ENTER KERNEL MODE
11538 045316  000000                    APTHLT: HALT
```

```
11541 045320                                    SUBTST  <<BLOCK MOVE SUBROUTINE>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       BLOCK MOVE SUBROUTINE
                                        ;*****************************************************************************
11542                                   ;BLOCK3 HAS 3 ARGUEMENTS
11543                                   ;BLOCK2 HAS 2 ARGUEMENTS
11544                                   ;BLOCK1 HAS 1 ARGUEMENTS
11545                                   ;
11546                                   ;ALL ARE CALLED BY THE BMOV MACRO
11547                                   .ENABL  LSB
11548 045320                    BLOCK1: PUSH    R0,R1,R2
11549 045326  012702  177640            MOV     #FASTCITY,R2
11550 045332  012701  000020            MOV     #16.,R1
11551 045336  000413                    BR      3$
11552
11553 045340                    BLOCK2: PUSH    R0,R1,R2
11554 045346  012701  000020            MOV     #16.,R1
11555 045352  000404                    BR      2$
11556
11557 045354                    BLOCK3: PUSH    R0,R1,R2
11558 045362  012501                    MOV     (R5)+,R1
11559 045364  012502            2$:     MOV     (R5)+,R2
11560 045366  012500            3$:     MOV     (R5)+,R0
11561
11562 045370  012022            1$:     MOV     (R0)+,(R2)+
11563 045372  077102                    SOB     R1,1$
11564 045374                            POP     R2,R1,R0
11565 045402  000205                    RTS     R5
11566                                   .DSABL  LSB
```

```
                                        .SBTTL  FIELD SERVICE MODE

                                FIELDSERVICE:SUBTST     <<SUBR  FIELD SERVICE COMMAND MODE>>
                                ;*****************************************************************************
                                ;*SUBTEST        SUBR    FIELD SERVICE COMMAND MODE
                                ;*****************************************************************************
045404  104415                          SAVREG
045406                                  TYPE    MSG020          ;FIELD SERVICE COMMAND MODE
                                
045412                                  IF RLFLAG IS TRUE OR NOFSMODE IS TRUE
045426                                    TYPE  MSG048          ;NOT AVAILABLE NOW - TRY LATER!
045432  104416                            RESREG
045434  000207                            RETURN
045436                                  END ;OF IF RLFLAG
045436  005737  002514                  TST     CACHKN
045442  001402                          BEQ     1$
045444                                  PUSH    CONTRL          ;SAVE CACHE STATUS
045450                          1$:     PUSH    CSRNO,KAMIKAZE  ;SAVE CSR & KAMIKAZE STATUS
045460  104424                          CACHOFF                 ;TURN CACHE OFF
045462                                  SET     KAMIKAZE
045470                          FS1:    TYPE    MSG026          ;COMMAND:
045474  104414                          RDDEC                   ;READ A DECIMAL NUMBER
045476                                  POP     R0              ;COMMAND --> R0
045500  020027  000022                  CMP     R0,#18.
045504  101403                          BLOS    1$
045506                                  TYPE    MSG021
045512  000766                          BR      FS1
045514                          1$:     CASE R0
045524  045602                            FSCMD0                ;EXIT FIELD SERVICE COMMANDS
045526  045704                            FSCMD1                ;READ CSR
045530  046014                            FSCMD2                ;LOAD CSR
045532  046162                            FSCMD3                ;EXAMINE MEMORY
045534  046436                            FSCMD4                ;MODIFY MEMORY
045536  046756                            FSCMD5                ;SELECT BANK & PATTERN
045540  047600                            FSCMD6                ;TYPE CONFIGURATION MAP
045542  047606                            FSCMD7                ;SOB-A-LONG TEST
045544  050100                            FSCMD8                ;ERROR SUMMARY
045546  050304                            FSCMD9                ;REFRESH TEST
045550  050576                            FCMD10                ;SET FILL COUNT
045552  050624                            FCMD11                ;ENTER KAMIKAZE MODE
045554  050646                            FCMD12                ;EXIT KAMIKAZE MODE
045556  050666                            FCMD13                ;TURN CACHE OFF
045560  050710                            FCMD14                ;TURN CACHE ON
045562  050726                            FCMD15                ;TEST ONLY SELECTED BANKS
045564  051012                            FCMD16                ;RESUME TESTING ALL BANKS
045566  051054                            FCMD17                ;ENABLE TRACE
045570  051070                            FCMD18                ;DISABLE TRACE
045572                                  END ;OF CASE
045600  000733                          BR      FS1
```

```
11620 045602                          FSCMD0: SUBTST  <<COMMAND 0     EXIT>>
                                      ;*****************************************************************************
                                      ;*SUBTEST       COMMAND 0       EXIT
                                      ;*****************************************************************************
11621 045602                                  TYPE    MSG103          ;LEAVING FIELD SERVICE MODE
11622 045606  062706  000002                  ADD     #2,SP
11623 045612                                  IF SKIPKAMI IS TRUE
11624 045620  062706  000002                    ADD   #2,SP                   ;THROW AWAY OLD KAMIKAZE FLAG
11625 045624  005037  002006                    CLR   SKIPKAMI
11626 045630                                  ELSE
11627 045632                                    POP   KAMIKAZE                ;RESTORE OLD KAMIKAZE FLAG
11628 045636                                  END ;OF IF SKIPKAMI
11629 045636                                  POP     CSRNO
11630 045642  005737  002514                  TST     CACHKN
11631 045646  001414                          BEQ     RES0
11632 045650                                  IF CACHKN EQ CACHKF           ;IF CACHE IS OFF
11633 045660  062706  000002                    ADD   #2,SP                 ;THROW AWAY CACHE STATUS
11634 045664                                  ELSE
11635 045666  005737  002514                    TST   CACHKN
11636 045672  001402                            BEQ   RES0
11637 045674                                    POP   CONTRL                ;RESTORE CACHE STATUS
11638 045700                                  END ;OF IF CACHKN
11639 045700  104416                    RES0: RESREG
11640 045702  000207                          RETURN
11641
11642 045704                          FSCMD1: SUBTST  <<FS    COMMAND 1       READ CSR>>
                                      ;*****************************************************************************
                                      ;*SUBTEST       FS      COMMAND 1       READ CSR
                                      ;*****************************************************************************
11643 045704  004737  051102                  CALL    WHICHCSR
11644 045710  010637  002266                  MOV     SP,FSSTACK
11645 045714                                  SET4    #RES1           ;TRAPS TO 4 GOTO RES1
11646 045722  104426                          READCSR
11647 045724                                  SET     NOERROR
11648 045732  104026                          ERROR   +26             ;USE ERROR ROUTINE FOR PRINTOUT
11649 045734                                  RES4                    ;RESET TRAPS TO 4 TO DEFAULT
11650 045756  000207                          RETURN
11651 045760                          RES1:   TYPE    MSG025          ;THIS CSR DOES NOT EXIST
11652 045764  013706  002266                  MOV     FSSTACK,SP
11653 045770                                  RES4                    ;RESET TRAPS TO 4 TO DEFAULT
11654 046012  000207                          RETURN
```

```
11657 046014                        FSCMD2: SUBTST  <<FS     COMMAND 2       LOAD CSR>>
                                    ;*****************************************************************************
                                    ;*SUBTEST        FS      COMMAND 2       LOAD CSR
                                    ;*****************************************************************************
11658 046014  004737  051102               CALL    WHICHCSR
11659 046020  010637  002266               MOV     SP,FSSTACK
11660 046024                               SET4    #RES2           ;TRAPS TO 4 GOTO RES2
11661 046032  104426                       READCSR
11662 046034                               TYPE    MSG027
11663 046040                               SET     NOERROR
11664 046046  104026                       ERROR   +26             ;USE ERROR ROUTINE FOR PRINTOUT
11665 046050                               RES4                    ;RESET TRAPS TO 4 TO DEFAULT
11666 046072                               TYPE    MSG023          ;FIRST CSR WORD
11667 046076  104413                       RDOCT                   ;READ AN OCTAL NUMBER
11668 046100                               POP     CSR             ;PUT IN IN LOC ''CSR''
11669 046104  104425                       LOADCSR
11670 046106  104426                       READCSR
11671 046110                               TYPE    MSG028
11672 046114                               SET     NOERROR
11673 046122  104026                       ERROR   +26             ;USE FOR PRINTOUT - NOT AN ERROR
11674 046124  000207                       RETURN
11675 046126                        RES2:  TYPE    MSG025          ;THIS CSR DOES NOT EXIST
11676 046132  013706  002266               MOV     FSSTACK,SP
11677 046136                               RES4                    ;RESET TRAPS TO 4 TO DEFAULT
11678 046160  000207                       RETURN
```

```
11681 046162                                FSCMD3: SUBTST  <<FS     COMMAND 3       EXAMINE MEMORY>>
                                            ;*****************************************************************************
                                            ;*SUBTEST       FS       COMMAND 3       EXAMINE MEMORY
                                            ;*****************************************************************************
11682 046162                                        PUSH    BANK,NOPAR,PARTHERE,4
11683 046202  012737  000002  002074                MOV     #2,NOPAR        ;INDICATE PARITY ACTION
11684 046210                                        TYPE    MSG029          ;EXAMINE MEMORY
11685 046214                                1$:     TYPE    MSG031          ;PHYSICAL ADDRESS (0-17775776)??
11686 046220  104413                              RDOCT                     ;READ OCTAL NUMBER ONTO STACK & $HIOCT
11687 046222  013737  062326  002100              MOV     $HIOCT,BANK       ;PUT MSB'S IN BANK
11688 046230                                      POP     R0                ;PUT LSB'S IN R0
11689 046232  000241                              CLC
11690 046234  006100                              ROL     R0
11691 046236  006137  002100                      ROL     BANK
11692 046242  000241                              CLC
11693 046244  006000                              ROR     R0
11694 046246  023737  002100  002526              CMP     BANK,LASTBANK   ;CHECK FOR BANK TOO HIGH
11695 046254  003357                              BGT     1$              ;BRANCH IF TRUE
11696 046256  062700  060000                      ADD     #FIRST,R0
11697 046262  032700  000001                      BIT     #BIT0,R0        ;CHECK FOR ODD ADDRESS
11698 046266  001352                              BNE     1$              ;BRANCH IF ODD ADDRESS
11699 046270  020027  157776                      CMP     R0,#LAST        ;CHECK FOR ADDRESS OVER 16K
11700 046274  101347                              BHI     1$              ;BRANCH IF OVER 16K
11701 046276  012737  046350  002264              MOV     #3$,PARTHERE    ;INCASE OF ABORTS
11702 046304                                      SET4    #4$             ;TRAPS TO 4 GOTO 4$
11703 046312                                      MAP     BANK                    ;MAP SUPERVISOR SPACE (TEST AREA) TO BANK
11704 046326                                      TESTAREA                ;ENTER TEST MODE
11705 046334  011001                              MOV     (R0),R1
11706 046336  104417                              KERNEL                  ;ENTER KERNEL MODE
11707 046340                                      TYPOCS  R1
11708 046346  000410                              BR      EXCMD3
11709
11710 046350                                3$:     TYPE    MSG032                  ;PARITY ABORT
11711 046354  000405                              BR      EXCMD3
11712
11713 046356  062706  000004                4$:     ADD     #4,SP                   ;FIX STACK
11714 046362                                      TYPE    MSG033                  ;TIMEOUT TRAP
11715 046366  000400                              BR      EXCMD3
11716
11717 046370  104417                      EXCMD3: KERNEL                  ;ENTER KERNEL MODE
11718 046372                                      POP     4,PARTHERE,NOPAR,BANK
11719 046412                                      RES4                    ;RESET TRAPS TO 4 TO DEFAULT
11720 046434  000207                              RETURN
```

```
11723 046436                          FSCMD4: SUBTST  <<FS    COMMAND 4       MODIFY MEMORY>>
                                      ;*****************************************************************************
                                      ;*SUBTEST       FS      COMMAND 4       MODIFY MEMORY
                                      ;*****************************************************************************
11724 046436                                  PUSH    BANK,NOPAR,PARTHERE,4
11725 046456  012737  000003  002074          MOV     #3,NOPAR                ;INDICATE PARITY ACTION
11726 046464                                  TYPE    MSG036                  ;MODIFY MEMORY
11727 046470                          1$:     TYPE    MSG031          ;PHYSICAL ADDRESS (0-17775776)??
11728 046474  104413                          RDOCT                   ;READ OCTAL NUMBER ONTO STACK & $HIOCT
11729 046476  013737  062326  002100          MOV     $HIOCT,BANK     ;PUT MSB'S IN BANK
11730 046504                                  POP     R0              ;PUT LSB'S IN R0
11731 046506  000241                          CLC
11732 046510  006100                          ROL     R0
11733 046512  006137  002100                  ROL     BANK
11734 046516  000241                          CLC
11735 046520  006000                          ROR     R0
11736 046522                                  IF BANK GT LASTBANK THEN GOTO 1$  ;CHECK FOR BANK TOO HIGH
11737 046532  062700  060000                  ADD     #FIRST,R0
11738 046536                                  IF #BIT0 SET.IN R0 THEN GOTO 1$ ;CHECK FOR ODD ADDRESS
11739 046544                                  IF R0 HI #LAST THEN GOTO 1$     ;CHECK FOR ADDRESS OVER 16K
11740 046552  012737  046620  002264          MOV     #3$,PARTHERE            ;INCASE OF ABORTS
11741 046560                                  SET4    #4$             ;TRAPS TO 4 GOTO 4$
11742 046566                                  MAP     BANK                    ;MAP SUPERVISOR SPACE (TEST AREA) TO BANK
11743 046602  104511                          INVALIDATE
11744 046604                                  TESTAREA                ;ENTER TEST MODE
11745 046612  011001                          MOV     (R0),R1
11746                                         ;GETTING HERE MEANS WE GOT LUCKY - NO TRAPS
11747 046614  104417                          KERNEL                  ;ENTER KERNEL MODE
11748 046616  000410                          BR      5$
11749
11750 046620                          3$:     TYPE    MSG032                  ;PARITY ABORT
11751 046624  000431                          BR      EXCMD4                  ;EXIT
11752
11753 046626  062706  000004          4$:     ADD     #4,SP           ;FIX STACK
11754 046632                                  TYPE    MSG033          ;TIMEOUT TRAP
11755 046636  000424                          BR      EXCMD4                  ;EXIT
11756
11757 046640                          5$:     TYPE    MSG037                  ;OLD DATA WAS
11758 046644                                  TYPOCS  R1                      ;PRINT IT
11759 046652                                  TYPE    MSG039                  ;INPUT NEW DATA
11760 046656  104413                          RDOCT                           ;READ ON OCTAL NUMBER ONTO THE STACK
11761 046660                                  POP     R1                      ;GET NEW NUMBER
11762 046662                                  TESTAREA                ;ENTER TEST MODE
11763 046670  010110                          MOV     R1,(R0)                 ;PUT IT IN MEMORY
11764 046672  011001                          MOV     (R0),R1                 ;READ IT AGAIN
11765 046674  104417                          KERNEL                  ;ENTER KERNEL MODE
11766 046676                                  TYPE    MSG038                  ;DATA IS NOW
11767 046702                                  TYPOCS  R1                      ;PRINT IT
11768
11769 046710  104417                  EXCMD4: KERNEL                  ;ENTER KERNEL MODE
11770 046712                                  POP     4,PARTHERE,NOPAR,BANK
11771 046732                                  RES4                    ;RESET TRAPS TO 4 TO DEFAULT
11772 046754  000207                          RETURN
```

```
11775 046756                          FSCMD5: SUBTST  <<FS    COMMAND 5       SELECT BANK & PATTERN>>
                                      ;*****************************************************************************
                                      ;*SUBTEST       FS      COMMAND 5       SELECT BANK & PATTERN
                                      ;*****************************************************************************
11776 046756                                  PUSH    BANK,PATTERN,TESTADD,PCBUMP,TKVEC,TKVEC+2
11777 047006  010637  002266                  MOV     SP,FSSTACK              ;SAVE LAST GOOD STACK POINTER
11778 047012                                  TYPE    MSG040                  ;SELECT BANK & PATTERN TEST
11779 047016                          1$:     TYPE    MSG030                  ;BANK(0-177)?
11780 047022  104413                          RDOCT                           ;READ AN OCTAL NUMBER ONTO THE STACK
11781 047024                                  POP     BANK                    ;PUT IT IN BANK
11782 047030                                  IF BANK GT LASTBANK THEN GOTO 1$  ;CHECK FOR BANK TOO HIGH
11783
11784 047040  013701  002100                  MOV     BANK,R1
11785 047044  006301                          ASL     R1
11786 047046  006301                          ASL     R1
11787 047050                                  IF CPUBIT OFF.IN CONFIG(R1)
11788 047060                                    TYPE  MSG041                  ;BANK NOT ACCESSABLE
11789 047064                                    GOTO 1$
11790 047066                                  END ;OF IF
11791
11792 047066                          2$:     TYPE    MSG042                  ;PATTERN(0-35)?
11793 047072  104413                          RDOCT                           ;READ AN OCTAL NUMBER ONTO THE STACK
11794 047074                                  POP     PATTERN                 ;PUT IT IN PATTERN
11795 047100                                  IF PATTERN GT #35 THEN GOTO 2$  ;CHECK FOR PATTERN TO HIGH
11796 047110                                  IF PATTERN EQ #0
11797 047116                                    TYPE  MSG043                  ;PATTERN 0 DATA IS?
11798 047122  104413                            RDOCT                       ;READ AN OCTAL NUMBER ONTO THE STACK
11799 047124                                    POP   R2                      ;PUT IT IN R2
11800 047126                                  END ;OF IF
11801
11802
11803 047126                                  MAP     BANK                    ;MAP SUPERVISOR SPACE (TEST AREA) TO BANK
11804 047142  104511                          INVALIDATE
11805 047144  004737  044300                  CALL    EXBANK                  ;SET NEW MARGINS
11806 047150                                  IF RRFLAG IS TRUE
11807 047156                                    TYPE  MSG049                  ;BANK REQUIRES RELOCATION
11808 047162                                    GOTO CMD5C
11809 047164                                  END ;OF IF RRFLAG
11810 047164                                  TYPE    MSG046          ;TO ESCAPE TYPE ANY KEY!
11811 047170  012737  047510  000060          MOV     #CMD5C,TKVEC
11812 047176  012737  000340  000062          MOV     #340,TKVEC+2
11813 047204  017700  133374                  MOV     @$TKB,R0                ;KILL ANY OLD INTERRUPT
11814 047210  042737  000200  177776          BIC     #BIT7,PSW               ;LOWER CPU PRIORITY TO 140
11815 047216  052777  000100  133356          BIS     #BIT6,@$TKS             ;ENABLE KEYBOARD INTERRUPTS
11816
11817
11818 047224                                  SET     HEADER,MUT
11819 047240  013701  002100          CMD5B:  MOV     BANK,R1
11820 047244  006301                          ASL     R1
11821 047246  006301                          ASL     R1
11822 047250  005037  002232                  CLR     SPLTCSR
11823 047254  005037  002256                  CLR     PASFLG
11824 047260  012737  060000  002362          MOV     #FIRST,TESTADD
11825 047266  012737  060002  002364          MOV     #FIRST+2,TESTADD+2
11826 047274                                  IF #BIT12 SET.IN CONFIG+2(R1)
11827 047304  005237  002232                    INC   SPLTCSR
11828 047310                                    MAP   BANK
```

FS COMMAND 5 SELECT BANK & PATTERN

```
11829 047324  012737  120000  002364          MOV   #120000,TESTADD+2
11830 047332                                END; OF IF #BIT12
11831 047332                                IF #SW0 SET.IN @SWR
11832 047342  104470                          ECCDIS                    ;DISABLE ERROR CORRECTION
11833 047344                                ELSE
11834 047346                                  PUSH  CSRNO
11835 047352  104502                          CLRCSR                    ;CLEAR CSRS
11836 047354                                  POP   CSRNO
11837 047360                                END ;OF IF
11838 047360  012737  000002  002074          MOV   #2,NOPAR            ;PARITY ACTION
11839 047366  012737  000002  002276          MOV   #2,PCBUMP           ;TRAPS ADD 2 TO PC
11840 047374  013700  002110                  MOV   PATTERN,R0
11841 047400  006300                          ASL   R0
11842 047402  004770  047414                  CALL  @FSPAT(R0)
11843 047406  005037  002074                  CLR   NOPAR
11844 047412  000712                          BR    CMD5B               ;LOOP TILL KEYBOARD INTERRUPT
11845
11846 047414  020122                FSPAT:    MT0000  ;<1 SEC       DATA PATTERN TEST
11847 047416  020202                          MT0001  ;<1 SEC       ADDRESS TEST
11848 047420  020322                          MT0002  ;<1 SEC       COMPLEMENT ADDRESS TEST
11849 047422  020462                          MT0003  ; 1 SEC       3 XOR 9 WORST CASE NOISE TEST
11850 047424  020714                          MT0004  ; 1 SEC       ROTATING ZEROS TEST
11851 047426  021036                          MT0005  ; 1 SEC       ROTATING ONES TEST
11852 047430  021172                          MT0006  ;<1 SEC       INITIAL DATA TEST
11853 047432  021226                          MT0007  ;<1 SEC       ADDRESS BIT TEST
11854 047434  021270                          MT0010  ;<1 SEC       BYTE ADDRESSING TEST
11855 047436  021324                          MT0011  ;<2 SEC       CREATE SINGLE BIT ERROR TEST
11856 047440  021372                          MT0012  ;<1 SEC       WRITE BYTE CLEARS SBE TEST
11857 047442  021466                          MT0013  ; 1 SEC       CREATE DOUBLE BIT ERROR TEST
11858 047444  021542                          MT0014  ; 1 SEC       WRITE INHIBIT DURING DATIP WITH DBE
11859 047446  021620                          MT0015  ; 1 SEC       WRITE INHIBIT OF BYTE WITH DBE
11860 047450  021666                          MT0016  ;<1 SEC       WRITE INHIBIT OF WORD WITH DBE
11861 047452  021734                          MT0017  ;<1 SEC       HOLDING 1'S & 0'S TEST
11862 047454  021756                          MT0020  ;<1 SEC       MARCHING 1'S & 0'S IN CHECK BITS
11863 047456  023046                          MT0021  ; 1 SEC       MARCHING 0'S & 1'S TEST
11864 047460  023320                          MT0022  ;10 SEC       REFRESH & SHIFTING DIAGONAL TEST
11865 047462  023352                          MT0023  ;10 SEC       SHIFTING DIAGONAL TEST
11866 047464  023416                          MT0024  ;20 SEC       FAST GALLOPING PATTERN TEST
11867 047466  023662                          MT0025  ;<1 SEC       INTERRUPT ENABLE TEST
11868 047470  023730                          MT0026  ;<1 SEC       RANDOM DATA TEST
11869 047472  024232                          MT0027  ; 1 SEC       UNIQUE BANK TEST
11870 047474  024716                          MT0030  ; 1 SEC       FLUSH OUT DBE'S TEST
11871 047476  025220                          MT0031  ; 3 SEC       SOB-A-LONG TEST
11872 047500  025410                          MT0032  ;<1 SEC       WRITE RECOVERY TEST
11873 047502  025742                          MT0033  ;35 SEC       BRANCH GOBBLE TEST
11874 047504  026130                          MT0034  ; 1 SEC       SOFT ERROR TEST
11875 047506  026302                          MT0035  ;<1 SEC       WORST CASE NOISE PARITY TEST
11876
11877 047510  013706  002266          CMD5C:  MOV   FSSTACK,SP              ;RECOVER OLD STACK POINTER
11878 047514  042777  000100  133060          BIC   #BIT6,@$TKS
11879 047522                                  POP   TKVEC+2,TKVEC
11880 047532  117700  133046                  MOVB  @$TKB,R0                ;GET CHARACTER TO GET RID OF FLAG
11881 047536                                  POP   PCBUMP,TESTADD
11882 047546                                  POP   PATTERN,BANK
11883 047556                                  MAP   BANK                    ;REMAP OLD BANK
11884 047572  004737  044300                  CALL  EXBANK
11885 047576  000207                          RETURN
```

```
11887 047600                        FSCMD6: SUBTST  <<FS    COMMAND 6       TYPE CONFIGURATION MAP>>
                                    ;*****************************************************************************
                                    ;*SUBTEST        FS      COMMAND 6       TYPE CONFIGURATION MAP
                                    ;*****************************************************************************
11888 047600  004737  036630               CALL    PCONFIG
11889 047604  000207                       RETURN
11890
```

```
11893 047606                                FSCMD7: SUBTST  <<FS    COMMAND 7       SOB-A-LONG TEST>>
                                                ;*****************************************************************************
                                                ;*SUBTEST       FS      COMMAND 7       SOB-A-LONG TEST
                                                ;*****************************************************************************
11894 047606                                        PUSH    BANK,PATTERN,TKVEC,TKVEC+2,NOPAR
11895 047632  010637  002266                      MOV     SP,FSSTACK                  ;SAVE LAST GOOD STACK POINTER
11896 047636                                      TYPE    MSG055                      ;SOB-A-LONG TEST
11897
11898 047642                                      IF #SW0 SET.IN @SWR
11899 047652  104470                                ECCDIS                            ;DISABLE ERROR CORRECTION
11900 047654                                      ELSE
11901 047656  104502                                CLRCSR                            ;CLEAR CSRS
11902 047660                                      END ;OF IF
11903 047660                                      TYPE    MSG056                      ;BELL = EACH PASS COMPLETE
11904
11905 047664                                      TYPE    MSG046                      ;TO ESCAPE TYPE ANY KEY!
11906 047670  012737  050014  000060              MOV     #CMD7C,TKVEC
11907 047676  012737  000340  000062              MOV     #340,TKVEC+2
11908 047704  017700  132674                      MOV     @$TKB,R0                    ;KILL ANY OLD INTERRUPT
11909 047710  042737  000200  177776              BIC     #BIT7,PSW                   ;LOWER CPU PRIORITY TO 140
11910 047716  052777  000100  132656              BIS     #BIT6,@$TKS                 ;ENABLE KEYBOARD INTERRUPTS
11911
11912
11913 047724                                      SET     HEADER,MUT
11914
11915 047740                                CMD7B:  FOR BANK := #0 TO LASTBANK
11916 047744  004737  044300                        CALL EXBANK
11917 047750                                        IF ACFLAG IS TRUE AND RRFLAG IS FALSE
11918 047764  104511                                  INVALIDATE
11919 047766  004737  025220                          CALL MT0031
11920 047772                                        END ;OF IF ACFLAG
11921 047772                                    END ;OF FOR BANK
11922 050006                                    TYPE    $BELL                       ;RING BELL
11923 050012                                    GOTO    CMD7B
11924
11925 050014  013706  002266          CMD7C:    MOV     FSSTACK,SP                  ;RECOVER OLD STACK POINTER
11926 050020  042777  000100  132554              BIC     #BIT6,@$TKS
11927 050026  117700  132552                      MOVB    @$TKB,R0                    ;READ CHAR TO KILL FLAG
11928 050032                                      POP     NOPAR,TKVEC+2,TKVEC,PATTERN,BANK
11929 050056                                      MAP     BANK                        ;MAP SUPERVISOR SPACE (TEST AREA) TO BANK
11930 050072  004737  044300                      CALL    EXBANK
11931 050076  000207                              RETURN
```

```
11934 050100                          FSCMD8: SUBTST  <<FS    COMMAND 8       ERROR SUMMARY>>
                                      ;*****************************************************************************
                                      ;*SUBTEST       FS      COMMAND 8       ERROR SUMMARY
                                      ;*****************************************************************************
11935 050100                                  PUSH    R0,R2,R3,BANK
11936 050112  013737  063034  002404          MOV     $PASS,TEMP
11937 050120  005337  002404                  DEC     TEMP
11938 050124                                  TYPDEC  TEMP
11939 050132                                  TYPE    MSG125                  ;PASSES COMPLETED
11940 050136                                  TYPDEC  $ERTTL
11941 050144                                  TYPE    MSG079                  ;ERROR(S) DETECTED
11942 050150                                  IF $ERTTL NE #0
11943 050156  005037  002304                    CLR   SUCCESS
11944 050162                                    FOR BANK := #0 TO LASTBANK
11945 050166  013703  002100                      MOV BANK,R3
11946 050172  070327  000004                      MUL #4,R3
11947 050176                                      IFB CONFIG+2(R3) NE #0
11948 050204                                        IF SUCCESS IS FALSE
11949 050212                                          TYPE    MSG076        ;BANK  ERRORS
11950 050216                                          SET SUCCESS
11951 050224                                        END ;OF IF SUCCESS
11952 050224                                        TYPOCS    BANK,3
11953 050234  116300  002626                        MOVB      CONFIG+2(R3),R0
11954 050240  042700  177400                        BIC       #^C377,R0
11955 050244                                        TYPDEC    R0
11956 050250                                        TYPE      $CRLF
11957 050254                                      END ;OF IFB CONFIG(R3)
11958 050254                                    END ;OF FOR BANK
11959 050270                                  END ;OF IF $ERTTL
11960 050270                                  POP     BANK,R3,R2,R0
11961 050302  000207                          RETURN
```

```
11964 050304                                FSCMD9: SUBTST  <<FS    COMMAND 9       REFRESH TEST>>
                                            ;*****************************************************************************
                                            ;*SUBTEST       FS      COMMAND 9       REFRESH TEST
                                            ;*****************************************************************************
11965 050304                                        PUSH    BANK,PATTERN,TKVEC,TKVEC+2,NOPAR
11966 050330  010637  002266                  MOV     SP,FSSTACK                  ;SAVE LAST GOOD STACK POINTER
11967 050334                                  TYPE    MSG073                      ;REFRESH TEST
11968
11969 050340                                  IF #SW0 SET.IN @SWR
11970 050350  104470                            ECCDIS                            ;DISABLE ERROR CORRECTION
11971 050352                                  ELSE
11972 050354  104502                            CLRCSR                            ;CLEAR CSRS
11973 050356                                  END ;OF IF
11974 050356                                  TYPE    MSG056                      ;BELL = EACH PASS COMPLETE
11975
11976 050362                                  TYPE    MSG046                      ;TO ESCAPE TYPE ANY KEY!
11977 050366  012737  050512  000060          MOV     #CMD9C,TKVEC
11978 050374  012737  000340  000062          MOV     #340,TKVEC+2
11979 050402  017700  132176                  MOV     @$TKB,R0                    ;KILL ANY OLD INTERRUPT
11980 050406  042737  000200  177776          BIC     #BIT7,PSW                   ;LOWER CPU PRIORITY TO 140
11981 050414  052777  000100  132160          BIS     #BIT6,@$TKS                 ;ENABLE KEYBOARD INTERRUPTS
11982
11983 050422                                  SET     HEADER,MUT
11984
11985 050436                                CMD9B:  FOR BANK := #0 TO LASTBANK
11986 050442  004737  044300                    CALL EXBANK
11987 050446                                    IF ACFLAG IS TRUE AND RRFLAG IS FALSE
11988 050462  104511                              INVALIDATE
11989 050464  004737  023320                      CALL MT0022
11990 050470                                    END ;OF IF ACFLAG
11991 050470                                END ;OF FOR BANK
11992 050504                                TYPE    $BELL                       ;RING BELL
11993 050510                                GOTO    CMD9B
11994
11995 050512  013706  002266          CMD9C:  MOV     FSSTACK,SP                  ;RECOVER OLD STACK POINTER
11996 050516  042777  000100  132056          BIC     #BIT6,@$TKS
11997 050524  117700  132054                  MOVB    @$TKB,R0                    ;READ CHAR TO KILL FLAG
11998 050530                                  POP     NOPAR,TKVEC+2,TKVEC,PATTERN,BANK
11999 050554                                  MAP     BANK                        ;MAP SUPERVISOR SPACE (TEST AREA) TO BANK
12000 050570  004737  044300                  CALL    EXBANK
12001 050574  000207                          RETURN
12002
```

```
12005 050576                          FCMD10: SUBTST  <<FS     COMMAND 10      SET FILL COUNT>>
                                      ;*****************************************************************************
                                      ;*SUBTEST       FS       COMMAND 10      SET FILL COUNT
                                      ;*****************************************************************************
12006 050576                                  PUSH    R0
12007 050600                                  TYPE    MSG085                  ;FILL COUNT(OCTAL)?
12008 050604  104413                          RDOCT
12009 050606                                  POP     R0
12010 050610  042700  177760                  BIC     #^C17,R0
12011 050614  110037  002327                  MOVB    R0,$FILLS
12012 050620                                  POP     R0
12013 050622  000207                          RETURN
12014
12015 050624                          FCMD11: SUBTST  <<FS     COMMAND 11      ENTER KAMIKAZE MODE>>
                                      ;*****************************************************************************
                                      ;*SUBTEST       FS       COMMAND 11      ENTER KAMIKAZE MODE
                                      ;*****************************************************************************
12016 050624                                  TYPE    MSG101          ;ENTERING KAMIKAZE MODE
12017 050630                                  SET     KAMIKAZE,SKIPKAMI
12018 050644  000207                          RETURN
12019
12020 050646                          FCMD12: SUBTST  <<FS     COMMAND 12      EXIT KAMIKAZE MODE>>
                                      ;*****************************************************************************
                                      ;*SUBTEST       FS       COMMAND 12      EXIT KAMIKAZE MODE
                                      ;*****************************************************************************
12021 050646                                  TYPE    MSG102          ;LEAVING KAMIKAZE MODE
12022 050652  005037  002004                  CLR     KAMIKAZE
12023 050656                                  SET     SKIPKAMI
12024 050664  000207                          RETURN
12025
12026 050666                          FCMD13: SUBTST  <<FS     COMMAND 13      TURN CACHE OFF>>
                                      ;*****************************************************************************
                                      ;*SUBTEST       FS       COMMAND 13      TURN CACHE OFF
                                      ;*****************************************************************************
12027 050666                                  TYPE    MSG106                  ;CACHE IS OFF
12028 050672  104424                          CACHOFF                         ;TURN CACHE OFF
12029 050674  013737  002514  002516          MOV     CACHKN,CACHKN+2         ;SAVE OLD CACHE ON STATE
12030 050702  005037  002514                  CLR     CACHKN                  ;KEEP CACHE OFF
12031 050706  000207                          RETURN
12032
12033 050710                          FCMD14: SUBTST  <<FS     COMMAND 14      TURN CACHE ON>>
                                      ;*****************************************************************************
                                      ;*SUBTEST       FS       COMMAND 14      TURN CACHE ON
                                      ;*****************************************************************************
12034 050710                                  TYPE    MSG107                  ;CACHE IS ON (EXCEPT DURING ACTUAL PATTERNS)
12035 050714  013737  002516  002514          MOV     CACHKN+2,CACHKN         ;RESTORE OLD CACHE ON STATE
12036 050722  104423                          CACHON                          ;TURN CACHE ON
12037 050724  000207                          RETURN
12038
```

FS COMMAND 14 TURN CACHE ON

```
12051
12052 050726                          FCMD15: SUBTST  <<FS    COMMAND 15      TEST ONLY SELECTED BANKS>>
                                      ;*****************************************************************************
                                      ;*SUBTEST       FS      COMMAND 15      TEST ONLY SELECTED BANKS
                                      ;*****************************************************************************
12053 050726                                  TYPE    MSG105          ;ENTER BANKS IN OCTAL - USE NUMBER OUTSIDE RANGE TO TERMINAT
12054 050732  004737  051022                  CALL    CMD16A                  ;ERASE OLD SELECTIONS
12055 050736                                  BEGIN CMD16LOOP
12056 050736                                    REPEAT
12057 050736                                      TYPE        MSG030          ;BANK(0-177)?
12058 050742  104413                              RDOCT                       ;READ AN OCTAL NUMBER ONTO THE STACK
12059 050744                                      POP R1                      ;PUT IT IN R1
12060 050746                                      IF R1 GT #177 OR R1 LT #0
12061 050760                                        LEAVE CMD16LOOP
12062 050762                                      END ;OF IF R1
12063 050762  006301                              ASL  R1
12064 050764  006301                              ASL  R1                     ;R1 <- R1 * 4
12065 050766  052761  040000  002626              BIS #BIT14,CONFIG+2(R1)
12066 050774                                    END ;OF REPEAT
12067 050776                                  END CMD16LOOP
12068 050776                                  TYPE    MSG110                  ;ONLY SELECTED BANKS WILL BE TESTED
12069 051002                                  SET     SELONLY
12070 051010  000207                          RETURN
12071
12072 051012                          FCMD16: SUBTST  <<FS    COMMAND 16      RESUME TESTING ALL BANKS>>
                                      ;*****************************************************************************
                                      ;*SUBTEST       FS      COMMAND 16      RESUME TESTING ALL BANKS
                                      ;*****************************************************************************
12073 051012                                  TYPE    MSG111                  ;ALL BANKS WILL BE TESTED
12074 051016  005037  002000                  CLR     SELONLY
12075
12076                                         ;ENTRY POINT FROM CMD15
12077 051022  013702  002526          CMD16A: MOV     LASTBANK,R2
12078 051026  006302                          ASL     R2
12079 051030  006302                          ASL     R2
12080 051032                                  FOR R1 := #0 TO R2 BY #4
12081 051034  042761  040000  002626            BIC   #BIT14,CONFIG+2(R1)
12082 051042                                  END ;OF FOR R1
12083 051052  000207                          RETURN
```

```
12086 051054                          FCMD17: SUBTST  <<FS    COMMAND 17      ENABLE TRACE>>
                                      ;*****************************************************************************
                                      ;*SUBTEST       FS      COMMAND 17      ENABLE TRACE
                                      ;*****************************************************************************
12087 051054                                  TYPE    MSG127
12088 051060  012737  177777  006136          MOV     #-1,TRACE
12089 051066  000207                          RETURN
```

```
 12092 051070                          FCMD18: SUBTST  <<FS    COMMAND 18      DISABLE TRACE>>
                                       ;*****************************************************************************
                                       ;*SUBTEST       FS      COMMAND 18      DISABLE TRACE
                                       ;*****************************************************************************
 12093 051070                                  TYPE    MSG128
 12094 051074  005037  006136                  CLR     TRACE
 12095 051100  000207                          RETURN
```

```
 12098 051102                           WHICHCSR:SUBTST <<SUBR  DETERMINE CORRECT CSR>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       SUBR    DETERMINE CORRECT CSR
                                        ;*****************************************************************************
 12099 051102  013700  002216                  MOV     TOTCSRS,R0      ;GET CSR'S FLAG
 12100 051106  022700  100000                  CMP     #BIT15,R0       ;CSR 0?
 12101 051112  001003                          BNE     1$              ;NO - SKIP
 12102 051114  005037  002146                  CLR     CSRNO           ;YES - SET IT UP
 12103 051120  000207                          RETURN
 12104
 12105 051122                           1$:    TYPE    MSG022          ;WHICH CSR(0-F)
 12106 051126  104412                          RDLIN                   ;GET CHARACTER
 12107 051130                                  POP     R0              ;PUT IN R0
 12108 051132  011000                          MOV     (R0),R0         ;PUT CHAR IN R0
 12109 051134  020027  000106                  CMP     R0,#106         ;CHECK LIMIT
 12110 051140  101370                          BHI     1$              ;IF BAD LOOP TILL HE TYPES IT RIGHT
 12111 051142  022700  000101                  CMP     #'A,R0
 12112 051146  103002                          BHIS    2$
 12113 051150  162700  000007                  SUB     #7,R0
 12114 051154  162700  000060           2$:    SUB     #60,R0
 12115 051160  006300                          ASL     R0
 12116 051162  010037  002146                  MOV     R0,CSRNO
 12117 051166  000207                          RETURN
```

```
12666                                   .SBTTL  ERROR DATA (SUPERVISOR) SETUP STUFF
12667 051170                    $PER25: LET ADDRESS := R1 - #2
12668 051202                            IF ABORTFLAG IS FALSE
12669 051210                              TESTAREA              ;ENTER TEST MODE
12670 051216                              LET BAD := -2(R1)
12671 051224  104417                      KERNEL                ;ENTER KERNEL MODE
12672 051226                            END ;OF IF ABORTFLAG
12673 051226                            IF 177654 EQ #0
12674 051234                              LET GOOD := R2
12675 051240                            ELSE
12676 051242                              LET GOOD := R3
12677 051246                            END ;OF IF
12678 051246  000137  054450              JMP     PERRAW
12679
12680 051252                    PERRA3: SUBTST  <<DATA WAS 3 WORDS>>
                                ;*************************************************************************
                                ;*SUBTEST       DATA WAS 3 WORDS
                                ;*************************************************************************
12681 051252                            IF BADPC EQ #0 THEN $CALL BADSTACK
12682 051264                            PUSH    R0
12683 051266  005037  002144              CLR     CSR             ;MAKE SURE CSR BIT HOLDER IS CLEAR
12684 051272  104505                      CHK1DIS                 ;DISABLE ECC & WRITE CHECKBITS FROM 1 SELECTED CSR
12685 051274                              TESTAREA
12686 051302  005711                      TST     (R1)            ;READ LOCATION TO READ CHECKBITS INTO CSR
12687 051304  104417                      KERNEL
12688 051306  104426                      READCSR                 ;GET CSR CONTENTS
12689 051310  013700  002144              MOV     CSR,R0          ;SAVE CSR CONTENTS IN R0
12690 051314  104503                      CLR1CSR                 ;RETURN CSR TO NORMAL MODE
12691 051316  072027  177773              ASH     #-5,R0          ;MOVE CHECK BITS TO BOTTOM OF WORD
12692 051322  042700  177600              BIC     #^C177,R0       ;CLEAR OFF EXTRANEOUS GARBAGE
12693 051326                              LET ADDRESS := R1       ;SAVE VIRTUAL ADDRESS FOR PRINTOUT
12694 051332  005037  002042              CLR     GOOD            ;FIRST TEST WORD WRITTEN SHOULD ALWAYS BE ZERO
12695 051336                              TESTAREA                ;ENTER TEST MODE
12696 051344  011137  002050              MOV     (R1),BAD        ;GET BAD DATA FROM MUT - FIRST WORD
12697 051350  011437  002052              MOV     (R4),BAD2       ;AND SECOND WORD
12698 051354  104417                      KERNEL                  ;ENTER KERNEL MODE
12699 051356  110037  002054              MOVB    R0,BAD3         ;MOVE BAD CHECKBITS FOR PRINTOUT
12700 051362  105037  002055              CLRB    BAD3+1          ;CLEAR OFF THE OTHER UNUSED BITS
12701 051366  004737  054704              CALL    PERBNK          ;MARK BANK AS BAD IN CONFIG TABLE
12702 051372  104033                      ERROR   +33
12703 051374                              POP     R0              ;RESTORE R0
12704 051376                              IF #SW0 SET.IN @SWR
12705 051406  104506                        ENASBE                ;TRAP ON SINGLE BIT ERRORS
12706 051410                              ELSE
12707 051412  104472                        ECCINIT               ;TRAP ON UNCORRECTABLE ERRORS
12708 051414                              END; OF IF #SW0
12709 051414  000002                      RTI
```

DATA WAS 3 WORDS

```
12712 051416                          $PER30: LET GOOD := R1
12713 051422                                  LET ADDRESS := (SP) - 16
12714 051434                                  IF ABORTFLAG IS FALSE
12715 051442                                    TESTAREA              ;ENTER TEST MODE
12716 051450                                    LET BAD := @ADDRESS
12717 051456  104417                            KERNEL                ;ENTER KERNEL MODE
12718 051460                                  END ;OF IF ABORTFLAG
12719 051460  000137  054450                  JMP PERRAW
12720
12721 051464                          GETDATA:SUBTST  <<GET DATA FROM ABORTED AREA IF POSSIBLE>>
                                      ;*****************************************************************************
                                      ;*SUBTEST       GET DATA FROM ABORTED AREA IF POSSIBLE
                                      ;*****************************************************************************
12722 051464                                  PUSH    R0,4,114
12723 051476  010637  051562                  MOV     SP,GETDA1
12724 051502  012737  051542  000004          MOV     #1$,4
12725 051510  012737  051542  000114          MOV     #1$,114
12726 051516  013700  002032                  MOV     ADDRESS,R0
12727 051522                                  TESTAREA
12728 051530  011037  002050                  MOV     (R0),BAD
12729 051534  104417                          KERNEL
12730 051536  005037  002140                  CLR     ABORTFLAG
12731 051542  013706  051562          1$:     MOV     GETDA1,SP             ;RESTORE KNOWN GOOD STACK POINTER
12732 051546                                  POP     114,4,R0
12733 051560  000207                          RETURN
12734 051562  000000                  GETDA1: 0
```

```
                                                .SBTTL  POWER FAIL AUTO RESTART
                                                .SBTTL  ROUTINE POWER DOWN AND UP
                                        ;*****************************************************************************
                                        ;POWER DOWN ROUTINE
051564                                  $PWRDN:
                                                ;SAVE CACHE STATUS
051564  005737  002514                          TST     CACHKN
051570  001403                                  BEQ     5$
051572                                          PUSH    CONTRL
051576  104423                                  CACHON                          ;TURN CACHE ON
051600  012737  052536  000024          5$:     MOV     #$ILLUP,PWRVEC ;;SET FOR FAST UP
051606  012737  000340  000026                  MOV     #340,PWRVEC+2   ;;PRIO:7
051614                                          PUSH    R0,R1,R2,R3,R4,R5,CSRNO
                                                ;SAVE USER PAR'S & PDR'S
051634  012700  177700                          MOV     #177700,R0
051640  012701  000021                          MOV     #17.,R1
051644                                  1$:     PUSH    -(R0)
051646  077102                                  SOB     R1,1$
                                                ;SAVE SUPERVISOR PAR'S
051650  005737  002426                          TST     NOSUPER
051654  001013                                  BNE     PD1
051656  012700  172300                          MOV     #172300,R0
051662  012701  000020                          MOV     #16.,R1
051666                                  2$:     PUSH    -(R0)
051670  077102                                  SOB     R1,2$
051672                                          IF RLFLAG IS TRUE THEN $CALL WOOPS
                                                ;COPY KERNEL MAP TO USER & SUPERVISOR
051704  012700  172300                  PD1:    MOV     #KIPDR0,R0
051710  012701  177600                          MOV     #UIPDR0,R1
051714  012702  172200                          MOV     #SIPDR0,R2
051720  012703  000040                          MOV     #32.,R3
051724  011021                          3$:     MOV     (R0),(R1)+
051726  012022                                  MOV     (R0)+,(R2)+
051730  077303                                  SOB     R3,3$
```

```
12779                                           ;SAVE USER & SUPERVISOR STACK POINTERS
12780 051732                                    USER
12781 051740  010600                            MOV     USP,R0
12782 051742  104417                            KERNEL                  ;ENTER KERNEL MODE
12783 051744                                    PUSH    R0
12784 051746  005737  002426                    TST     NOSUPER
12785 051752  001006                            BNE     7$
12786 051754                                    SUPERVISOR              ;ENTER SUPERVISOR MODE
12787 051762  010600                            MOV     SSP,R0
12788 051764  104417                            KERNEL                  ;ENTER KERNEL MODE
12789 051766                                    PUSH    R0
12790                                           ;SAVE ECC REGISTERS
12791 051770  013701  002216          7$:       MOV     TOTCSRS,R1      ;GET CSR'S
12792 051774                                    BEGIN   LCSRSAVE
12793 051774                                      FOR CSRNO := #0 TO #36 BY #2
12794 052000  006301                                ASL         R1
12795 052002                                        ON.ERROR
12796 052004  104426                                  READCSR
12797 052006                                          PUSH      CSR
12798 052012                                        END ;OF ON.ERROR
12799 052012                                      IF R1 EQ #0 THEN LEAVE LCSRSAVE
12800 052016                                      END ;OF FOR CSRNO
12801 052034                                    END LCSRSAVE
12802                                           ;SAVE MMR0,1,2,3
12803 052034                                    PUSH    MMR0,MMR1,MMR2
12804 052050  005737  002426                    TST     NOSUPER
12805 052054  001002                            BNE     8$
12806 052056                                    PUSH    MMR3
12807                                           ;SAVE KERNEL PAR'S
12808 052062  012700  172400          8$:       MOV     #172400,R0
12809 052066  012701  000020                    MOV     #16.,R1
12810 052072                          4$:       PUSH    -(R0)
12811 052074  077102                            SOB     R1,4$
12812                                           ;SAVE UNIBUS MAP REGISTERS
12813 052076  022737  000001  003716            CMP     #1,PROTYP               ;IS THIS AN 11/44?
12814 052104  001004                            BNE     9$                      ;BRANCH IF NOT
12815 052106                                    PUSH    MAPH0,MAPL0
12816                                           ;SAVE POSSIBLE SOFTWARE SWITCH REGISTER
12817 052116                          9$:       PUSH    @SWR
12818                                           ;SAVE STACK POINTER
12819 052122  010637  052542                    MOV     SP,$SAVR6       ;;SAVE SP
12820                                           ;NOW SET UP REAL VECTOR
12821 052126  012737  052140  000024            MOV     #$PWRUP,PWRVEC  ;;SET UP VECTOR
12822 052134  000000                  $DOWN:    HALT
12823 052136  000776                            BR      $DOWN           ;;HANG UP
```

```
12826                                   ;*****************************************************************************************
12827                                   ;POWER UP ROUTINE
12828 052140                            $PWRUP:
12832 052140  012737  052536  000024              MOV     #$ILLUP,PWRVEC ;;SET FOR FAST DOWN
12833                                             ;RESTORE STACK POINTER
12834 052146  013706  052542                      MOV     $SAVR6,SP       ;;GET SP
12835 052152  005037  052542                      CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
12836 052156  005237  052542            1$:       INC     $SAVR6          ;;WAIT FOR THE INC
12837 052162  001375                              BNE     1$              ;;OF A WORD
12838                                             ;RESTORE POSSIBLE SOFTWARE SWITCH REGISTER
12839 052164                                      POP     @SWR
12840                                             ;RESTORE UNIBUS MAP
12841 052170  022737  000001  003716              CMP     #1,PROTYP               ;IS THIS AN 11/44?
12842 052176  001006                              BNE     10$
12843 052200                                      POP     MAPLO,MAPHO
12844 052210  004737  043704                      CALL    LOWMAP          ;SETUP LOWER 16K OF UNIBUS MAP
12845                                             ;RESTORE KERNEL PAR'S & PDR'S
12846 052214  012700  172340            10$:      MOV     #172340,R0
12847 052220  012702  172300                      MOV     #KIPDR0,R2
12848 052224  012701  000020                      MOV     #16.,R1
12849 052230                            6$:       POP     (R0)+
12850 052232  012722  077406                      MOV     #77406,(R2)+
12851 052236  077104                              SOB     R1,6$
12852                                             ;RESTORE MMR3,2,1,0
12853 052240  005737  002426                      TST     NOSUPER
12854 052244  001002                              BNE     11$
12855 052246                                      POP     MMR3
12856 052252                            11$:      POP     MMR2,MMR1,MMR0
12857                                             ;RESTORE ECC REGISTERS
12858 052266  013701  002216                      MOV     TOTCSRS,R1      ;GET CSR'S
12859 052272  042701  177400                      BIC     #177400,R1
12860 052276                                      BEGIN LCSRRESTORE
12861 052276                                        FOR CSRNO := #36 DOWNTO #0 BY #2
12862 052304  006201                                  ASR         R1
12863 052306                                          ON.ERROR
12864 052310                                            POP       CSR
12865 052314  104425                                      LOADCSR
12866 052316                                          END ;OF ON.ERROR
12867 052316                                        IF R1 EQ #0 THEN LEAVE LCSRRESTORE
12868 052322                                        END ;OF FOR CSRNO
12869 052340                                      END LCSRRESTORE
12870                                             ;COPY KERNEL MAP TO USER & SUPERVISOR
12871 052340  012700  172300                      MOV     #KIPDR0,R0
12872 052344  012701  177600                      MOV     #UIPDR0,R1
12873 052350  012702  172200                      MOV     #SIPDR0,R2
12874 052354  012703  000040                      MOV     #32.,R3
12875 052360  011021                    3$:       MOV     (R0),(R1)+
12876 052362  012022                              MOV     (R0)+,(R2)+
12877 052364  077303                              SOB     R3,3$
```

```
12879                                          ;RESTORE SUPERVISOR & USER STACK POINTERS
12880 052366  005737  002426                   TST     NOSUPER
12881 052372  001006                           BNE     13$
12882 052374                                   POP     R0
12883 052376                                   SUPERVISOR              ;ENTER SUPERVISOR MODE
12884 052404  010006                           MOV     R0,SSP
12885 052406  104417                           KERNEL                  ;ENTER KERNEL MODE
12886 052410                           13$:    POP     R0
12887 052412                                   USER
12888 052420  010006                           MOV     R0,USP
12889 052422  104417                           KERNEL                  ;ENTER KERNEL MODE
12890                                          ;RESTORE SUPERVISOR PAR'S
12891 052424  012700  172240                   MOV     #172240,R0
12892 052430  012701  000020                   MOV     #16.,R1
12893 052434                           7$:     POP     (R0)+
12894 052436  077102                           SOB     R1,7$
12895                                          ;RESTORE USER PAR'S & PDR7
12896 052440  012700  177636                   MOV     #177636,R0
12897 052444  012701  000021                   MOV     #17.,R1
12898 052450                           8$:     POP     (R0)+
12899 052452  077102                           SOB     R1,8$
12900                                          ;RESTORE POSSIBLE SOFTWARE DISPLAY REGISTER
12901 052454  013777  002010  130116           MOV     $PATMAR,@DISPLAY
12902 052462  013737  002010  000174           MOV     $PATMAR,DISPREG
12903 052470                                   POP     CSRNO,R5,R4,R3,R2,R1,R0
12904 052510  012737  051564  000024           MOV     #$PWRDN,PWRVEC  ;;SET UP THE POWER DOWN VECTOR
12905 052516                                   TYPE    MSG051          ;REPORT THE POWER FAILURE
12906                                          ;RESTORE CACHE STATUS
12907 052522  005737  002514                   TST     CACHKN
12908 052526  001402                           BEQ     9$
12909 052530                                   POP     CONTRL
12910 052534  000002                   9$:     RTI
12911 052536  000000                   $ILLUP: HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
12912 052540  000776                           BR      $ILLUP          ;; BEFORE THE POWER DOWN WAS COMPLETE
12913 052542  000000                   $SAVR6: 0                       ;;PUT THE SP HERE
12914                                          .EVEN
```

```
12926 052544                                WOOPS:  SUBTST  <<POWER FAIL WHILE RELOCATED>>
                                            ;*****************************************************************************
                                            ;*SUBTEST       POWER FAIL WHILE RELOCATED
                                            ;*****************************************************************************
12927 052544                                        PUSH    BANK
12928 052550  005037  002100                  CLR     BANK
12929 052554                                  MAP     BANK                    ;MAP SUPERVISOR SPACE (TEST AREA) TO BANK
12930 052570                                  SUPERVISOR                      ;ENTER SUPERVISOR MODE
12931 052576  013737  060024  053142          MOV     FIRST+PWRVEC,WOOPSAV
12932 052604  013737  060026  053144          MOV     FIRST+PWRVEC+2,WOOPSAV+2
12933 052612                                  BMOV    FIRST+WOOPUP,WOOPSAV+4,WOOPEND-WOOPUP/2+12.
12934 052624  012737  052730  060024          MOV     #WOOPUP,FIRST+PWRVEC
12935 052632  012737  000340  060026          MOV     #340,FIRST+PWRVEC+2
12936 052640                                  BMOV    WOOPUP,FIRST+WOOPUP,WOOPEND-WOOPUP/2
12937 052652  012700  172340                  MOV     #KIPAR0,R0
12938 052656  012701  133112                  MOV     #FIRST+WOOPEND,R1
12939 052662  012702  000010                  MOV     #8.,R2
12940 052666  012021                  1$:     MOV     (R0)+,(R1)+
12941 052670  077202                          SOB     R2,1$
12942 052672  005737  002426                  TST     NOSUPER
12943 052676  001002                          BNE     2$
12944 052700  013721  172516                  MOV     MMR3,(R1)+
12945 052704  013721  177576          2$:     MOV     MMR2,(R1)+
12946 052710  013721  177574                  MOV     MMR1,(R1)+
12947 052714  013721  177572                  MOV     MMR0,(R1)+
12948 052720  104417                          KERNEL                          ;ENTER KERNEL MODE
12949 052722                                  POP     BANK
12950 052726  000207                          RETURN
```

```
12953 052730                          WOOPUP: SUBTST  <<POWER UP FROM BANK 0 TO RELOCATION>>
                                      ;*****************************************************************************
                                      ;*SUBTEST       POWER UP FROM BANK 0 TO RELOCATION
                                      ;*****************************************************************************
12954 052730  012700  053112                  MOV     #WOOPEND,R0
12955 052734  012701  172340                  MOV     #KIPAR0,R1
12956 052740  012703  172300                  MOV     #KIPDR0,R3
12957 052744  012702  000010                  MOV     #8.,R2
12958 052750  012021                  1$:     MOV     (R0)+,(R1)+
12959 052752  012723  077406                  MOV     #77406,(R3)+
12960 052756  077204                          SOB     R2,1$
12961 052760  005737  002426                  TST     NOSUPER
12962 052764  001002                          BNE     3$
12963 052766  012037  172516                  MOV     (R0)+,MMR3
12964 052772  012037  177576          3$:     MOV     (R0)+,MMR2
12965 052776  012037  177574                  MOV     (R0)+,MMR1
12966 053002  012037  177572                  MOV     (R0)+,MMR0
12967 053006  013706  052542                  MOV     $SAVR6,SP
12968 053012                                  PUSH    BANK
12969 053016  005037  002100                  CLR     BANK
12970 053022                                  MAP     BANK                    ;MAP SUPERVISOR SPACE (TEST AREA) TO BANK
12971 053036                                  SUPERVISOR                      ;ENTER SUPERVISOR MODE
12972 053044  013737  053142  060024          MOV     WOOPSAV,FIRST+PWRVEC
12973 053052  013737  053144  060026          MOV     WOOPSAV+2,FIRST+PWRVEC+2
12974                                         ;SIMULATE THE FOLLOWING BLOCK MOV BUT WITH NO STACK ACCESSES
12975                                         ;BMOV   WOOPSAV+4,FIRST+WOOPUP,WOOPEND-WOOPUP/2+12.
12976 053060  012700  053146                  MOV     #WOOPSAV+4,R0
12977 053064  012701  000105                  MOV     #WOOPEND-WOOPUP/2+12.,R1
12978 053070  012702  132730                  MOV     #FIRST+WOOPUP,R2
12979 053074  012022                  2$:     MOV     (R0)+,(R2)+
12980 053076  077102                          SOB     R1,2$
12981
12982 053100  104417                          KERNEL                          ;ENTER KERNEL MODE
12983 053102                                  POP     BANK
12984 053106  000137  052140                  JMP     $PWRUP
12985 053112  000014          WOOPEND:.REPT   12.
12988 053142  000107          WOOPSAV:.REPT   WOOPEND-WOOPUP/2+12.+2
```

```
                                        .SBTTL  IO SUBROUTINES
                                        .SBTTL  ROUTINE TYPE
                                ;*******************************************************************************
                                ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
                                ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
                                ;*NOTE1:         $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
                                ;*NOTE2:         $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
                                ;*NOTE3:         $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
                                ;*
                                ;*CALL:
                                ;*1) USING A TRAP INSTRUCTION
                                ;*      TYPE    MESADR          ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
                                ;*OR
                                ;*      TYPE
                                ;*      MESADR
                                ;*

053360  105737  002330          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
053364  100407                          BMI     6$              ;BR IF NO
053366  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
053370  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
053374  112046                  4$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
053376  001005                          BNE     7$              ;;BR IF IT ISN'T THE TERMINATOR
053400  005726                          TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
053402  012600                  5$:     MOV     (SP)+,R0        ;;RESTORE R0
053404  062716  000002          6$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
053410  000002                          RTI                     ;;RETURN
053412  122716  000011          7$:     CMPB    #HT,(SP)        ;BRANCH IF NOT <HT>
053416  001002                          BNE     11$
053420  112716  000040                  MOVB    #' ,(SP)        ;REPLACE TAB WITH SPACE
053424  122716  000200          11$:    CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
053430  001006                          BNE     8$
053432  005726                          TST     (SP)+           ;;POP  <CR><LF> EQUIV
053434                                  TYPE                    ;;TYPE A CR AND LF
053436  002620                          $CRLF
053440  105037  053672                  CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
053444  000753                          BR      4$              ;;GET NEXT CHARACTER
053446  004737  053506          8$:     CALL    $TYPEC          ;;GO TYPE THIS CHARACTER
053452  123726  002612          9$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
053456  001346                          BNE     4$              ;;IF NO GO GET NEXT CHAR.
053460  013746  002326                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
                                                                ;;AND THE NULL CHAR.
053464  105366  000001          10$:    DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
053470  002770                          BLT     9$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
053472  004737  053506                  CALL    $TYPEC          ;;GO TYPE A NULL
053476  105337  053672                  DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
053502  000770                          BR      10$             ;;LOOP
053504  000000          XOCHAR: .WORD 0
053506          $TYPEC: PUSH    R1
053510  116601  000004                  MOVB    4(SP),R1
053514  005737  002514                  TST     CACHKN
053520  001402                          BEQ     2$
053522                                  PUSH    CONTRL
053526                  2$:     PUSH    R0
053530  104424                          CACHOFF                 ;TURN CACHE OFF
```

ROUTINE TYPE

```
13074 053532  105777  127050          3$:     TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
13075 053536  100375                          BPL     3$
13076 053540  005037  053504                  CLR     XOCHAR
13077 053544  105777  127032                  TSTB    @$TKS           ;;CHECK FOR XOFF
13078 053550  100032                          BPL     NC              ;;SKIP IF NO CHARACTER
13079 053552  117737  127026  053504          MOVB    @$TKB,XOCHAR    ;;SAVE THE CHARACTER
13080 053560  042737  177600  053504          BIC     #^C177,XOCHAR   ;;STRIP OFF ASCII
13081 053566  023727  053504  000023          CMP     XOCHAR,#023     ;;WAS IT A CONTROL S?
13082 053574  001020                          BNE     NC              ;;BRANCH IF NOT
13083 053576  105777  127000          CONTS3: TSTB    @$TKS           ;;WAIT FOR CHARACTER
13084 053602  100375                          BPL     CONTS3
13085 053604  117737  126774  053504          MOVB    @$TKB,XOCHAR    ;;GET CHARACTER
13086 053612  042737  177600  053504          BIC     #^C177,XOCHAR   ;;STRIP OFF ASCII
13087 053620                                  IF XOCHAR EQ #21        ;; IF IT IS A ^Q
13088 053630  000402                            BR    NC
13089 053632                                  ELSE
13090 053634  000760                            BR    CONTS3
13091 053636                                  END ;OF IF XOCHAR
13092 053636  110177  126746          NC:       MOVB  R1,@$TPB        ;;LOAD CHAR TO BE TYPED INTO DATA REG.
13096 053642  122766  000015  000002          CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
13097 053650  001003                          BNE     1$              ;;BRANCH IF NO
13098 053652  105037  053672                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
13099 053656  000406                          BR      $TYPEX          ;;EXIT
13100 053660  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
13101 053666  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
13102 053670  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
13103 053672  000000          $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
13104 053674                  $TYPEX: POP     R0
13105 053676  005737  002514                  TST     CACHKN          ;IS THERE A CACHE?
13106 053702  001402                          BEQ     2$              ;BRANCH IF NOT
13107 053704                                  POP     CONTRL          ;POP CACHE STATUS
13108 053710                  2$:     POP     R1
13109 053712  000207                          RETURN
13110 053714          SUPLIMIT:;!!!!!!!!!!!!!THIS IS THE LIMIT ON SUPERVISOR MAPPED TO MUT SPACE
```

```
13788                                   .SBTTL  ERROR DATA SETUP
13789
13790                           ;       USE THIS      IF THIS CONDITION DISCRIBES THE ERROR
13791                           ;
13792                           ;       PERR01        TRAP
13793                           ;                     BAD DATA IN R0 UNLESS ABORTED
13794                           ;                           THEN BAD DATA IS POINTED TO BY -(R4)
13795                           ;                     GOOD DATA IN R5
13796                           ;
13797                           ;       PERR02        TRAP
13798                           ;                     BAD DATA IN R1 UNLESS ABORTED
13799                           ;                           THEN BAD DATA IS POINTED TO BY -(R4)
13800                           ;                     GOOD DATA IN R2
13801                           ;
13802                           ;       PERR03        TRAP
13803                           ;                     BAD DATA IS POINTED TO BY -(R1)
13804                           ;                     GOOD DATA IN R4
13805                           ;
13806                           ;       PERR04        TRAP
13807                           ;                     BAD DATA IN R4 UNLESS ABORTED
13808                           ;                           THEN BAD DATA IS POINTED TO BY -2(R0)
13809                           ;                     GOOD DATA IN R2
13810                           ;
13811                           ;       PERR05        JSR     PC
13812                           ;                     BAD DATA IS POINTED TO BY -(R0)
13813                           ;                     GOOD DATA IN R2
13814                           ;                     RETURN AFTER SETTING UP GOOD,BAD,ADDRESS
13815                           ;
13816                           ;       PERR06        JSR     PC
13817                           ;                     BAD DATA IS POINTED TO BY -(R0)
13818                           ;                     GOOD DATA IS ZERO
13819                           ;                     RETURN AFTER SETTING UP GOOD,BAD,ADDRESS
13820                           ;
13821                           ;       PERR07        TRAP
13822                           ;                     BAD DATA IN R2 UNLESS ABORTED
13823                           ;                           THEN BAD DATA IS POINTED TO BY (R1)
13824                           ;                     GOOD DATA IN DATBUF
13825                           ;
13826                           ;       PERR10        TRAP
13827                           ;                     BAD DATA IN R2 UNLESS ABORTED
13828                           ;                           THEN BAD DATA IS POINTED TO BY 2(R1)
13829                           ;                     GOOD DATA IN DATBUF+2
13830                           ;
13831                           ;       PERR11        TRAP
13832                           ;                     BYTE TEST
13833                           ;                     BAD DATA IN RIGHT BYTE OF R0 UNLESS ABORTED
13834                           ;                           THEN BAD DATA IS POINTED TO BY (R1)
13835                           ;                     GOOD DATA IS A ZERO BYTE
13836                           ;
13837                           ;       PERR12        TRAP
13838                           ;                     BYTE TEST
13839                           ;                     BAD DATA IN RIGHT BYTE OF R0 UNLESS ABORTED
13840                           ;                           THEN BAD DATA IS POINTED TO BY (R1)
13841                           ;                     GOOD DATA IS A BYTE OF ONES
13842                           ;
13843                           ;       PERR13        TRAP
13844                           ;                     BAD DATA IN R0 UNLESS ABORTED
```

```
13845                                   ;                       THEN BAD DATA IS POINTED TO BY (R1)
13846                                   ;               GOOD DATA IS ZERO
13847                                   ;
13848                                   ; PERR14        TRAP
13849                                   ;               BAD DATA IN R0 UNLESS ABORTED
13850                                   ;                       THEN BAD DATA IS POINTED TO BY (R1)
13851                                   ;               GOOD DATA IS ONES
13852                                   ;
13853                                   ; PERR15        TRAP
13854                                   ;               BAD DATA IN R0 UNLESS ABORTED
13855                                   ;                       THEN BAD DATA IS POINTED TO BY (R1)
13856                                   ;               GOOD DATA IN TSTDAT
13857                                   ;
13858                                   ; PERR16        TRAP
13859                                   ;               BAD DATA IN R0 UNLESS ABORTED
13860                                   ;                       THEN BAD DATA IS POINTED TO BY (R1)
13861                                   ;               GOOD DATA IN TSTDAT+2
13862                                   ;
13863                                   ; PERR17        TRAP
13864                                   ;               BAD DATA IN R0 UNLESS ABORTED
13865                                   ;                       THEN BAD DATA IS POINTED TO BY (R1)
13866                                   ;               GOOD DATA IN R2
13867                                   ;
13868                                   ; PERR20        TRAP
13869                                   ;               BAD DATA IN R0 UNLESS ABORTED
13870                                   ;                       THEN BAD DATA IS POINTED TO BY (R1)
13871                                   ;               GOOD DATA IN R3
13872                                   ;
13873                                   ; PERR21        TRAP
13874                                   ;               7 BIT BYTE TEST
13875                                   ;               BAD DATA IN RIGHT BYTE OF R0 UNLESS ABORTED
13876                                   ;                       THEN BAD DATA IS POINTED TO BY (R1)
13877                                   ;               GOOD DATA IS A 7 BIT BYTE ON ONES
13878                                   ;
13879                                   ; PERR22        TRAP
13880                                   ;               BAD DATA IN R2 UNLESS ABORTED
13881                                   ;                       THEN BAD DATA IS POINTED TO BY (R1)
13882                                   ;               GOOD DATA IN R0
13883                                   ;
13884                                   ; PERR23        TRAP
13885                                   ;               BAD DATA IN R0 UNLESS ABORTED
13886                                   ;                       THEN BAD DATA IS POINTED TO BY (R1)
13887                                   ;               GOOD DATA IN R4
13888                                   ;
13889                                   ; PERR24        TRAP
13890                                   ;               BAD DATA IN R0 UNLESS ABORTED
13891                                   ;                       THEN BAD DATA IS POINTED TO BY (R2)
13892                                   ;               GOOD DATA IN R3
13893                                   ;
13894                                   ; PERR25        TRAP
13895                                   ;               BAD DATA POINTED TO BY -(R1)
13896                                   ;               GOOD DATA IN R2 UNLESS LOC V177654 IS SET
13897                                   ;                       THEN GOOD DATA IS IN R3
13898                                   ;
13899                                   ; PERR26        TRAP
13900                                   ;               BAD DATA IS DOUBLE WORD POINTED TO BY R1 AND IN LOW 7 BITS OF R0
13901                                   ;               GOOD DATA IS 000000,,100000,,100
```

```
13902
13903                   ;PERR27         TRAP
13904                   ;               BAD DATA IS DOUBLE WORD POINTED TO BY R1 AND IN LOW 7 BITS OF R0
13905                   ;               GOOD DATA IS 000000,,000000,,077
13906                   ;
13907                   ;PERR30         TRAP
13908                   ;               BAD DATA IS POINTED TO BY -16(SP)
13909                   ;               GOOD DATA IS IN R1
13910                   ;
13911                   ;PERR31         TRAP
13912                   ;               SPECIAL ECC FAILURE HANDLER
13913                   ;
13914                   ;PERR32         TRAP
13915                   ;               SPECIAL ECC FAILURE HANDLER
13916                   ;
13917                   ;PERR33         TRAP
13918                   ;               SPECIAL ECC FAILURE HANDLER
13919                   ;
13920                   ;PERR34         TRAP
13921                   ;               SPECIAL ECC FAILURE HANDLER
13922                   ;
13923                   ;PERR35         TRAP
13924                   ;               SPECIAL BRANCH GOBBLE FAILURE HANDLER
13925                   ;
13926                   ;       CALLING SEQUENCE FOR TRAP TYPES
13927                   ;       BEQ     2$              ;NO - ERROR,BRANCH FOR CARD
13928                   ;       PERRXX                  ;TRAP TO ERROR ROUTINE
13929                   ;2$:    NEXT    INSTRUCTION     ;CONTINUE TESTING
```

ERROR DATA SETUP

```
13932 053714  010437  002032          $PER01: MOV     R4,ADDRESS
13933 053720  162737  000002  002032          SUB     #2,ADDRESS
13934 053726  010037  002050                  MOV     R0,BAD
13935 053732  010537  002042                  MOV     R5,GOOD
13936 053736  000137  054450                  JMP PERRAW
13937
13938 053742  010437  002032          $PER02: MOV     R4,ADDRESS
13939 053746  162737  000002  002032          SUB     #2,ADDRESS
13940 053754  010137  002050                  MOV     R1,BAD
13941 053760  010237  002042                  MOV     R2,GOOD
13942 053764  000137  054450                  JMP PERRAW
13943
13944 053770  010137  002032          $PER03: MOV     R1,ADDRESS
13945 053774  162737  000002  002032          SUB     #2,ADDRESS
13946 054002  010437  002042                  MOV     R4,GOOD
13947 054006  016137  177776  002050          MOV     -2(R1),BAD
13948 054014  000137  054450                  JMP PERRAW
13949
13950 054020  010037  002032          $PER04: MOV     R0,ADDRESS
13951 054024  162737  000002  002032          SUB     #2,ADDRESS
13952 054032  010437  002050                  MOV     R4,BAD
13953 054036  010237  002042                  MOV     R2,GOOD
13954 054042  000137  054450                  JMP     PERRAW
13955
13956 054046  010237  002042          PERR05: MOV     R2,GOOD
13957 054052  014037  002050          PERA05: MOV     -(R0),BAD
13958 054056  010037  002032                  MOV     R0,ADDRESS
13959 054062  062700  000002                  ADD     #2,R0           ;RESTORE R0
13960 054066  004737  040166                  CALL    BADSTACK
13961 054072  000207                          RETURN
13962
13963 054074  005037  002042          PERR06: CLR     GOOD
13964 054100  000764                          BR      PERA05
13965
13966 054102  010137  002032          $PER07: MOV     R1,ADDRESS
13967 054106  010237  002050                  MOV     R2,BAD
13968 054112  013737  002234  002042          MOV     DATBUF,GOOD
13969 054120  000137  054450                  JMP     PERRAW
13970
13971 054124                          $PER10: LET ADDRESS := R1 + #2
13972 054136                                  LET BAD := R2
13973 054142                                  LET GOOD := DATBUF+2
13974 054150  000137  054450                  JMP     PERRAW
13975
13976 054154                          $PER11: LET ADDRESS := R1
13977 054160                                  LET BAD := R0
13978 054164                                  LET GOOD := #0
13979 054170  000137  054522                  JMP     PERRAB
13980
13981 054174                          $PER12: LET ADDRESS := R1
13982 054200                                  LET BAD := R0
13983 054204                                  LET GOOD := #377
13984 054212  000137  054522                  JMP     PERRAB
```

ERROR DATA SETUP

```
13987 054216                          $PER13: LET ADDRESS := R1
13988 054222                                  LET BAD := R0
13989 054226                                  LET GOOD := #0
13990 054232  000137  054450                  JMP     PERRAW
13991
13992 054236                          $PER14: LET ADDRESS := R1
13993 054242                                  LET BAD := R0
13994 054246                                  LET GOOD := ONES
13995 054254  000137  054450                  JMP     PERRAW
13996
13997 054260                          $PER15: LET ADDRESS := R1
13998 054264                                  LET BAD := R0
13999 054270                                  LET GOOD := TSTDAT
14000 054276  000137  054450                  JMP     PERRAW
14001
14002 054302                          $PER16: LET ADDRESS := R1
14003 054306                                  LET BAD := R0
14004 054312                                  LET GOOD := TSTDAT+2
14005 054320  000453                          BR      PERRAW
14006
14007 054322                          $PER17: LET ADDRESS := R1
14008 054326                                  LET BAD := R0
14009 054332                                  LET GOOD := R2
14010 054336  000444                          BR      PERRAW
14011
14012 054340                          $PER20: LET ADDRESS := R1
14013 054344                                  LET BAD := R0
14014 054350                                  LET GOOD := R3
14015 054354  000435                          BR      PERRAW
14016
14017 054356                          $PER21: LET ADDRESS := R1
14018 054362                                  LET BAD := R0
14019 054366                                  LET GOOD := #177
14020 054374  000477                          BR      PERRA7
14021
14022 054376                          $PER22: LET ADDRESS := R1
14023 054402                                  LET BAD := R2
14024 054406                                  LET GOOD := R0
14025 054412  000416                          BR      PERRAW
14026
14027 054414                          $PER23: LET ADDRESS := R1
14028 054420                                  LET BAD := R0
14029 054424                                  LET GOOD := R4
14030 054430  000407                          BR      PERRAW
14031
14032 054432                          $PER24: LET ADDRESS := R2
14033 054436                                  LET BAD := R0
14034 054442                                  LET GOOD := R3
14035 054446  000400                          BR      PERRAW
```

```
14037 054450                        PERRAW: SUBTST  <<DATA WAS A WORD>>
                                    ;*****************************************************************************
                                    ;*SUBTEST       DATA WAS A WORD
                                    ;*****************************************************************************
14038 054450  004737  054704              CALL    PERBNK
14039 054454                              IF ABORTFLAG IS TRUE THEN $CALL GETDATA
14040 054466                              IF BADPC EQ #0 THEN $CALL BADSTACK
14041 054500  004737  054660              CALL    PERXOR
14042 054504                              IF ABORTFLAG IS FALSE
14043 054512  104011                        ERROR +11
14044 054514                              ELSE
14045 054516  104012                        ERROR +12
14046 054520                              END ;OF IF ABORTFLAG
14047 054520  000002                      RTI
14048
14049 054522                        PERRAB: SUBTST  <<DATA WAS A BYTE>>
                                    ;*****************************************************************************
                                    ;*SUBTEST       DATA WAS A BYTE
                                    ;*****************************************************************************
14050 054522  004737  054704              CALL    PERBNK
14051 054526                              IF ABORTFLAG IS TRUE THEN $CALL GETDATA
14052 054540                              IF BADPC EQ #0 THEN $CALL BADSTACK
14053 054552  004737  054660              CALL    PERXOR
14054 054556                              IF ABORTFLAG IS FALSE
14055 054564  104014                        ERROR +14
14056 054566                              ELSE
14057 054570  104015                        ERROR +15
14058 054572                              END ;OF IF ABORTFLAG
14059 054572  000002                      RTI
```

DATA WAS A BYTE

```
14062 054574                                PERRA7: SUBTST  <<DATA WAS A 7 BIT BYTE>>
                                            ;*****************************************************************************
                                            ;*SUBTEST       DATA WAS A 7 BIT BYTE
                                            ;*****************************************************************************
14063 054574                                        IF BADPC EQ #0 THEN $CALL BADSTACK
14064 054606  004737  054660                        CALL    PERXOR
14065 054612  004737  054704                        CALL    PERBNK
14066 054616  104022                                ERROR   +22
14067 054620  000002                                RTI
14068
14069 054622                                $PER26: LET GOOD2 := #100000
14070 054630                                        LET GOOD3 := #100
14071 054636  000137  051252                        JMP PERRA3
14072
14073 054642  005037  002044                $PER27: CLR     GOOD2
14074 054646                                        LET GOOD3 := #077
14075 054654  000137  051252                        JMP PERRA3
14076
14077 054660                                PERXOR: SUBTST  <<DETERMINE XOR OF GOOD & BAD>>
                                            ;*****************************************************************************
                                            ;*SUBTEST       DETERMINE XOR OF GOOD & BAD
                                            ;*****************************************************************************
14078 054660                                        PUSH    R0
14079 054662  013700  002042                        MOV     GOOD,R0
14080 054666  013737  002050  002056                MOV     BAD,BADXOR
14081 054674  074037  002056                        XOR     R0,BADXOR
14082 054700                                        POP     R0
14083 054702  000207                                RETURN
```

```
14086 054704                          PERBNK: SUBTST<<LOG ERROR ON BAD BANK>>
                                      ;*****************************************************************************
                                      ;*SUBTEST       LOG ERROR ON BAD BANK
                                      ;*****************************************************************************
14087                                         ;WHILE WE'RE HERE LET'S MARK THE BAD BANK IN THE CONFIGURATION TABLE
14088 054704                                  PUSH    R0,R1
14089 054710  013701  002100                  MOV     BANK,R1
14090 054714  006301                          ASL     R1
14091 054716  006301                          ASL     R1
14092 054720  052761  000001  002624          BIS     #BIT0,CONFIG(R1)
14093 054726  105261  002626                  INCB    CONFIG+2(R1)            ;BUMP BANK COUNTER
14094 054732  001002                          BNE     12$                     ;NO OVERFLOW - SKIP
14095 054734  105361  002626                  DECB    CONFIG+2(R1)            ;SET BACK TO 255.
14096 054740  126137  002626  002524  12$:    CMPB    CONFIG+2(R1),ERRMAX     ;IS IT PAST MAX?
14097 054746  101403                          BLOS    11$                     ;NO - SKIP
14098 054750                                  SET     TOOMANY                 ;YES
14099 054756                          11$:    POP     R1,R0
14100 054762  000207                          RETURN
14101
14102 054764  010037  002050          PERECC: MOV     R0,BAD
14103 054770                                  IF ADDRESS EQ TESTADD
14104 055000  013737  002240  002042            MOV TSTDAT,GOOD
14105 055006                                  ELSE
14106 055010  013737  002242  002042            MOV   TSTDAT+2,GOOD
14107 055016                                  END ;OF IF (R1)
14108 055016  004737  054660                  CALL    PERXOR
14109 055022                                  SET     HEADER
14110 055030  000207                          RETURN
14111
14112 055032                          $PER31: IF BADPC EQ #0 THEN $CALL BADSTACK
14113 055044  004737  054764                  CALL    PERECC
14114 055050                                  IF REALPAT EQ #11
14115 055060  104037                            ERROR +37
14116 055062                                  END ;OF IF REALPAT
14117 055062                                  IF REALPAT EQ #14
14118 055072  104042                            ERROR +42
14119 055074                                  END ;OF IF REALPAT
14120 055074                                  IF REALPAT EQ #15
14121 055104  104043                            ERROR +43
14122 055106                                  END ;OF IF REALPAT
14123 055106                                  IF REALPAT EQ #16
14124 055116  104044                            ERROR +44
14125 055120                                  END ;OF IF REALPAT
14126 055120                                  SET HEADER
14127 055126  000002                          RTI
```

LOG ERROR ON BAD BANK

```
14130 055130                                $PER32: IF BADPC EQ #0 THEN $CALL BADSTACK
14131 055142  010137  002032                        MOV     R1,ADDRESS
14132 055146  010037  002050                        MOV     R0,BAD
14133 055152  010237  002042                        MOV     R2,GOOD
14134 055156                                        SET     HEADER
14135 055164  104040                                ERROR   +40
14136 055166                                        SET     HEADER
14137 055174  000002                                RTI
14138
14139 055176                                $PER33: IF BADPC EQ #0 THEN $CALL BADSTACK
14140 055210  010137  002032                        MOV     R1,ADDRESS
14141 055214  010037  002050                        MOV     R0,BAD
14142 055220  105037  002051                        CLRB    BAD+1
14143 055224  012737  000377  002042                MOV     #377,GOOD
14144 055232  004737  054660                        CALL    PERXOR
14145 055236                                        SET     HEADER
14146 055244  104041                                ERROR   +41
14147 055246                                        SET     HEADER
14148 055254  000002                                RTI
14149
14150 055256                                $PER34:IF BADPC EQ #0 THEN $CALL BADSTACK
14151 055270                                        IF #BIT15!BIT4 OFF.IN CSR
14152 055300  104016                                  ERROR +16                     ;NO SBE OR DBE
14153 055302                                        ELSE
14154 055304  104001                                  ERROR +1                      ;EXPECTED SBE SO DBE MUST HAVE GOTTEN SET
14155 055306                                        END ;OF IF #BIT15!BIT4
14156 055306  000002                                RTI
14157
14158                                               ;DURING BRANCH GOBBLE THE CONDITION CODES WERE WRONG
14159 055310  004737  054704          $PER35:       CALL    PERBNK
14160 055314  004737  040166                        CALL    BADSTACK
14161 055320  013737  002030  002050                MOV     BADPSW,BAD
14162 055326  012737  000012  002042                MOV     #12,GOOD
14163 055334  104047                                ERROR   +47
14164 055336  062706  000004                        ADD     #4,SP                 ;FIX STACK FROM TRAP
14165 055342  000207                                RETURN                        ;ABORTING TEST
14166
14167 055344  010037  002042          $PER36:       MOV     R0,GOOD
14168 055350  010137  002050                        MOV     R1,BAD
14169 055354                                        SET     HEADER
14170 055362  104023                                ERROR   +23
14171 055364                                        SET     HEADER
14172 055372  000002                                RTI
```

```
                                        .SBTTL  ROUTINE SCOPE HANDLER
                                ;*******************************************************************************
                                ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
                                ;*AND LOAD THE DISPLAY DATA INTO THE DISPLAY REGISTER
                                ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
                                ;*SW14=1         LOOP ON TEST
                                ;*SW9=1 LOOP ON ERROR
                                ;*CALL
                                ;*      SCOPE           ;;SCOPE=IOT
055374  005237  063036        $SCOPE: INC     $DEVCT          ;TELL APT WE ARE ALIVE
055400                                IF RESULT IS LT
055402  005037  063036                  CLR   $DEVCT
055406  105237  063040                  INCB  $UNIT
055412                                END ;OF IF RESULT
055412  104410                        CKSWR                   ;;TEST  FOR CHANGE IN SOFT-SWR
055414  005737  006136                TST     TRACE
055420  001402                        BEQ     NOTRCE
055422  004737  061332                CALL    CONTT           ;TRACE
055426                          NOTRCE:
055426  005737  056510                TST     CPERRF          ;IS THERE A CPU ERROR REGISTER?         ;R-C
055432  001410                        BEQ     SKJ             ;BRANCH IF NOT                          ;R-C
055434  013737  177766  056506        MOV     @#177766,CPSAVE ;GET CONTENTS OF ERROR REGISTER         ;R-C
055442  032737  000001  056506        BIT     #BIT0,CPSAVE    ;IS THE POWER FAIL MONITOR BIT SET?     ;R-C
055450  001401                        BEQ     SKJ             ;BRANCH IF NOT                          ;R-C
055452  104177                        ERROR   +177            ;REPORT IF SO                           ;R-C
055454                          SKJ:    IF STOPOK IS TRUE AND #SW8 SET.IN @SWR                          ;R-C
055472  005037  002370                  CLR   STOPOK
055476  000137  045134                  JMP   EXIT
055502                                END ;OF IF STOPOK
055502                                IF NOSCOPE IS TRUE
055510  000002                          RTI
055512                                END ;OF IF NOSCOPE
055512                          1$:     IF #SW14 SET.IN @SWR THEN GOTO $OVER
                                ;#####START OF CODE FOR THE XOR TESTER#####
055522  000425                $XTSTR: BR      2$              ;;IF RUNNING ON THE ''XOR'' TESTER CHANGE
                                                            ;;THIS INSTRUCTION TO A ''NOP'' (NOP=240)
055524  013746  000004                MOV     ERRVEC,-(SP)    ;;SAVE THE CONTENTS OF THE ERROR VECTOR
055530  012737  055550  000004        MOV     #1$,ERRVEC      ;;SET FOR TIMEOUT
055536  005737  177060                TST     177060  ;;TIME OUT ON XOR?
055542  012637  000004                MOV     (SP)+,ERRVEC    ;;RESTORE THE ERROR VECTOR
055546  000430                        BR      $SVLAD          ;;GO TO THE NEXT TEST
055550  062706  000004          1$:   ADD     #4,SP           ;FIX STACK FROM TRAP
055554  022737  000001  003716        CMP     #1,PROTYP       ;IS THIS AN 11/44?
055562  001002                        BNE     6$              ;BRANCH IF NOT
055564  005037  177766                CLR     CPUERR          ;RESET CPU ERROR REGISTER
055570  012637  000004          6$:   MOV     (SP)+,ERRVEC    ;;RESTORE THE ERROR VECTOR
055574  000407                        BR      4$              ;;LOOP ON THE PRESENT TEST
055576                          2$:;#####END OF CODE FOR THE XOR TESTER#####
055576  105737  002012          3$:   TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
055602  001412                        BEQ     $SVLAD          ;;BR IF NO
055604  032777  001000  124764        BIT     #SW9,@SWR       ;;LOOP ON ERROR?
055612  001404                        BEQ     5$              ;;BR IF NO
055614  013737  002564  002562  4$:   MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
055622  000410                        BR      $OVER
055624  105037  002012          5$:   CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
055630  011637  002562          $SVLAD: MOV   (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
055634  011637  002564                MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
```

```
14240 055640  005037  002332                CLR     $ESCAPE          ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
14241 055644  004737  055656        $OVER:  CALL    GETDIS
14242 055650  013716  002562                MOV     $LPADR,(SP)      ;;FUDGE RETURN ADDRESS
14243 055654  000002                        RTI                      ;;FIXES PS
```

```
14245 055656                            GETDIS: SUBTST  <<SUBR  DISPLAY>>
                                        ;*****************************************************************************
                                        ;*SUBTEST        SUBR    DISPLAY
                                        ;*****************************************************************************
14246 055656  113737  002100  002011            MOVB    BANK,$BANK
14247 055664  113737  002260  002010            MOVB    REALPAT,$PATMAR
14248 055672                                    PUSH    R0
14249 055674  005737  002124                    TST     RLFLAG                  ;ARE WE RELOCATED?
14250 055700  001403                            BEQ     1$                      ;NO - SKIP
14251 055702  052737  100000  002010            BIS     #BIT15,$PATMAR          ;YES - SET MSB
14252 055710                            1$:
14256 055710  013777  002010  124662            MOV     $PATMAR,@DISPLAY
14257 055716  013737  002010  000174            MOV     $PATMAR,DISPREG         ;SOFTWARE DISPLAY REGISTER
14258 055724                                    POP     R0
14259 055726  000207                            RETURN
```

```
                                        .SBTTL  ROUTINE ERROR HANDLER
                                ;*******************************************************************************
                                ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
                                ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
                                ;*AND GO TO $ERRTYP ON ERROR
                                ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
                                ;*SW15=1         HALT ON ERROR
                                ;*SW13=1         INHIBIT ERROR TYPEOUTS
                                ;*SW10=1         BELL ON ERROR
                                ;*SW9=1 LOOP ON ERROR
                                ;*CALL
                                ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER

                                        .ENABL  LSB
055730  105037  056504        $ERROR: CLRB    IBSAVE                                          ;R-C
055734                                  IF NOERROR IS FALSE
055742  104410                            CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
055744                          BACK:                                                           ;R-C
055744  105237  002012        1$:       INCB  $ERFLG              ;;SET THE ERROR FLAG
055750  001775                          BEQ   1$                  ;;DON'T LET THE FLAG GO TO ZERO
055752  004737  055656                  CALL  GETDIS              ;SETUP DISPLAY STUFF
055756  013737  002010  063032        MOV   $PATMAR,$TESTN      ;FOR APT
055764  032777  002000  124604        BIT   #SW10,@SWR          ;;BELL ON ERROR?
055772  001404                          BEQ   2$                  ;;NO - SKIP
055774                                  TYPE  $BELL               ;;RING BELL
056000                                  TYPE  MSG014              ;CONTROL Z
056004  005237  002570        2$:       INC   $ERTTL              ;;COUNT THE NUMBER OF ERRORS
056010                                  IF RESULT IS MI
056012  012737  077777  002570          MOV #77777,$ERTTL
056020                                  END ;OF IF RESULT
056020                          END ;OF IF NOERROR
056020  011637  002016        MOV     (SP),ERRPC      ;;GET ADDRESS OF ERROR INSTRUCTION
056024  162737  000002  002016  SUB     #2,ERRPC
056032  010637  002022        MOV     SP,ERRSP
056036  016637  000002  002026  MOV     2(SP),ERRPSW
056044  117737  123746  002013  MOVB    @ERRPC,$ITEMB   ;;STRIP AND SAVE THE ERROR ITEM CODE
056052  122737  000177  002013  CMPB    #177,$ITEMB     ;IS THIS THE POWER FAIL CALL?           ;R-C
056060  001431                  BEQ     1001$           ;BRANCH IF SO                           ;R-C
056062  105737  056504          TSTB    IBSAVE          ;2ND ERROR CALL?                        ;R-C
056066  001024                  BNE     1000$           ;BRANCH IF SO                           ;R-C
056070  005737  056510          TST     CPERRF          ;IS THERE A CPU ERROR REGISTER?         ;R-C
056074  001423                  BEQ     1001$           ;BRANCH IF NOT                          ;R-C
056076  013737  177766  056506  MOV     177766,CPSAVE   ;SAVE CONTENTS                          ;R-C
056104  032737  000001  056506  BIT     #BIT0,CPSAVE    ;POWER MONITOR BIT SET?                 ;R-C
056112  001414                  BEQ     1001$           ;BRANCH IF NOT                          ;R-C
056114  042737  000001  177766  BIC     #BIT0,177766    ;CLEAR THE BIT                          ;R-C
056122  112737  002013  056504  MOVB    #$ITEMB,IBSAVE  ;MAKE IBSAVE NON-ZERO FOR DUAL CALL     ;R-C
056130  112737  000177  002013  MOVB    #177,$ITEMB     ;SET $ITEMB TO POWER FAIL POINTER       ;R-C
056136  000402                  BR      1001$                                                   ;R-C
056140  105037  056504  1000$:  CLRB    IBSAVE                                                  ;R-C
056144                  1001$:                                                                  ;R-C
056144                          IF NOERROR IS FALSE
056152                            IF BADPC NE #0
056160  013737  002020  002016        MOV BADPC,ERRPC
056166  162737  000002  002016        SUB #2,ERRPC
056174  013737  002024  002022        MOV BADSP,ERRSP
```

```
14319 056202  013737  002030  002026          MOV BADPSW,ERRPSW
14320 056210  005037  002020                  CLR BADPC
14321 056214                                END ;IF
14322 056214  013737  002016  063030        MOV   ERRPC,$FATAL    ;FOR APT
14323 056222                                IF #SW13 SET.IN @SWR
14324 056232  000412                          BR  3$
14325 056234                                END ;OF IF #SW13
14331 056234                                IF #SW5 SET.IN @SWR AND TOOMANY IS TRUE
14332 056252                                  GOTO 3$
14333 056254                                END ;OF IF #SW5
14334 056254                              END ;OF IF NOERROR
14335 056254  004737  056512              CALL    $ERRTYP         ;;GO TO USER ERROR ROUTINE
```

```
14337 056260                                3$:     IF NOERROR IS FALSE
14338 056266  005777  124304                        TST   @SWR                ;;HALT ON ERROR
14339 056272  100002                        $HALT:  BPL   7$                 ;;SKIP IF CONTINUE
14340 056274  000000                                HALT                     ;;HALT ON ERROR!
14341 056276  104410                                CKSWR                    ;;TEST FOR CHANGE IN SOFT-SWR
14342 056300                                7$:     IF NOSCOPE IS FALSE AND #SW9 SET.IN @SWR
14343 056316  013716  002564                          MOV $LPERR,(SP)        ;;FUDGE RETURN FOR LOOPING
14344 056322                                        END ;OF IF NOSCOPE
14345 056322  005737  002332                        TST   $ESCAPE            ;;CHECK FOR AN ESCAPE ADDRESS
14346 056326  001402                                BEQ   9$                 ;;BR IF NONE
14347 056330  013716  002332                        MOV   $ESCAPE,(SP)       ;;FUDGE RETURN ADDRESS FOR ESCAPE
14348 056334                                9$:     IF DETFLAG IS FALSE
14349 056342  022737  000001  003716                  CMP #1,PROTYP                  ;IS THIS AN 11/44?
14350 056350  001002                                  BNE 11$
14351 056352  005037  177766                          CLR CPUERR
14352 056356                                      11$: IF ACTFLAG IS TRUE OR APTFLAG IS TRUE OR FATAL$ IS TRUE
14353 056400  012737  000001  063026                    MOV     #1,$MSGTY      ;FOR APT
14354 056406  000137  045134                            JMP     EXIT
14355 056412                                          END ;OF IF ACTFLAG
14356 056412                                          IF XXDPCHAIN IS TRUE AND $ERTTL HI #20
14357 056430                                            TYPE    MSG066         ;ERROR COUNT EXCEEDED 20 - ABORTING FOR XXDP CHAIN
14358 056434  013700  000042                            MOV     42,R0
14359 056440  005037  000042                            CLR     42
14360 056444  000137  014450                            JMP     $ZAP42
14361 056450                                          END ;OF IF XXDPCHAIN
14362 056450                                        END ;OF IF DETFLAG
14363 056450                                       ELSE
14364 056452                                         SET   HEADER
14365 056460                                        END ;OF IF NOERROR
14366 056460                                10$:    CLEAR   TOOMANY,NOERROR
14367 056470  105737  056504                        TSTB    IBSAVE                 ;POWER FAIL ERROR CALL? ;R-C
14368 056474  001402                                BEQ     213$                   ;R-C
14369 056476  000137  055744                        JMP     BACK                   ;JUMP IF SO             ;R-C
14370 056502  000002                        213$:   RTI                         ;;RETURN
14371 056504  000000                        IBSAVE: .WORD 0                                             ;R-C
14372 056506  000000                        CPSAVE: .WORD 0                                             ;R-C
14373 056510  000000                        CPERRF: .WORD 0                                             ;R-C
14374                                               .DSABL  LSB
```

```
14377                                   .SBTTL  ROUTINE ERROR MESSAGE TYPEOUT
14378
14379                           ;*******************************************************************************
14380                           ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
14381                           ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
14382                           ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
14383
14384 056512  104415          $ERRTYP:SAVREG
14385 056514                          TYPE    $CRLF           ;;"CARRIAGE RETURN" & "LINE FEED"
14386 056520  005000                  CLR     R0              ;;PICKUP THE ITEM INDEX
14387 056522  153700  002013          BISB    $ITEMB,R0
14388 056526  001004                  BNE     1$              ;;IF ITEM NUMBER IS ZERO, JUST
14389                                                         ;;TYPE THE PC OF THE ERROR
14390 056530                          TYPOCT  ERRPC,<ERROR ADDRESS>
14391 056536  000511                  BR      11$             ;;GET OUT
14392 056540  122700  000177  1$:     CMPB    #177,R0         ;POWER MONITOR CALL?                    ;R-C
14393 056544  001003                  BNE     100$            ;BRANCH IF NOT                          ;R-C
14394 056546  012700  057022          MOV     #PFECWS,R0      ;MOV ADDRESS OF PFE BIT ERROR TO R0     ;R-C
14395 056552  000406                  BR      110$                                                    ;R-C
14396 056554  005300  100$:   DEC     R0              ;;ADJUST THE INDEX SO THAT IT WILL      ;R-C
14397 056556  006300                  ASL     R0              ;;      WORK FOR THE ERROR TABLE
14398 056560  006300                  ASL     R0
14399 056562  006300                  ASL     R0
14400 056564  062700  063450          ADD     #$ERRTB,R0      ;;FORM TABLE POINTER
14401 056570  012037  056626  110$:    MOV     (R0)+,3$        ;;PICKUP "ERROR MESSAGE" POINTER        ;R-C
14402 056574  001417                  BEQ     4$              ;;SKIP TYPEOUT IF NO POINTER
14403 056576  005737  002400          TST     NOERROR         ;IS THIS REALLY AN ERROR?
14404 056602  001003                  BNE     12$             ;YES - SKIP
14405 056604  005737  002552          TST     HEADER          ;TYPE HEADER?
14406 056610  100011                  BPL     4$              ;NO - SKIP
14407 056612  005737  002062  12$:    TST     FATAL$          ;WAS IT A FATAL ERROR?
14408 056616  001402                  BEQ     2$              ;NO - SKIP
14409 056620                          TYPE    MSG067          ;FATAL
14410 056624                  2$:     TYPE                    ;;TYPE THE "ERROR MESSAGE"
14411 056626  000000          3$:     .WORD   0               ;;"ERROR MESSAGE" POINTER GOES HERE
14412 056630                          TYPE    $CRLF           ;;"CARRIAGE RETURN" & "LINE FEED"
14413 056634  012037  056660  4$:     MOV     (R0)+,5$        ;;PICKUP "DATA HEADER" POINTER
14414 056640  001412                  BEQ     6$              ;;SKIP TYPEOUT IF 0
14415 056642  005737  002400          TST     NOERROR         ;IS THIS REALLY AN ERROR?
14416 056646  001003                  BNE     13$             ;YES - SKIP
14417 056650  005737  002552          TST     HEADER          ;TYPE HEADER?
14418 056654  100004                  BPL     6$              ;NO - SKIP
14419 056656                  13$:    TYPE                    ;;TYPE THE "DATA HEADER"
14420 056660  000000          5$:     .WORD   0               ;;"DATA HEADER" POINTER GOES HERE
14421 056662                          TYPE    $CRLF           ;;"CARRIAGE RETURN" & "LINE FEED"
14422 056666  012001          6$:     MOV     (R0)+,R1        ;;PICKUP "DATA TABLE" POINTER
14423 056670  001427                  BEQ     10$             ;;BR IF NO DATA TO BE TYPED
14424 056672  012002                  MOV     (R0)+,R2        ;;PICKUP "DATA FORMAT" POINTER
```

```
14427 056674  112203                  7$:     MOVB    (R2)+,R3
14428 056676  006303                          ASL     R3              ;MAKE IT A WORD ADDRESS
14429 056700  004773  056706                   CALL    @8$(R3)
14430 056704  000412                          BR      9$
14431 056706  057132                  8$:     TAG70$
14432 056710  057142                          TAG71$
14433 056712  057152                          TAG72$
14434 056714  057222                          TAG73$
14435 056716  057262                          TAG74$
14436 056720  057274                          TAG75$
14437 056722  057306                          TAG76$
14438 056724  057352                          TAG77$
14439 056726  057360                          TAG78$
14440 056730  057440                          TAG79$
14445 056732  062701  000002          9$:     ADD     #2,R1           ;UPDATE DATA TABLE POINTER
14446 056736  005711                          TST     (R1)            ;;IS THERE ANOTHER NUMBER?
14447 056740  001403                          BEQ     10$             ;;BR IF NO
14448 056742                                  TYPE    MSG018          ;TYPE 2 SPACES
14449 056746  000752                          BR      7$              ;;LOOP
14450
14451 056750  005737  002106          10$:    TST     MUT             ;IS THERE A MEMORY UNDER TEST
14452 056754  001402                          BEQ     11$             ;NO - SKIP
14453 056756  005237  002552                  INC     HEADER          ;YES - BUMP HEADER FLAG
14454 056762  104416                  11$:    RESREG
14455 056764                                  IF #SW7 SET.IN @SWR AND DETFLAG IS FALSE AND NOERROR IS FALSE
14456 057010  004737  057462                    CALL  DETAIL
14457 057014                                  END ;OF IF #SW7
14458 057014                                  TYPE    MSG104          ;CONTROL Z
14459 057020  000207                          RETURN
14460                                         .EVEN                                   ;R-C
14461 057022  057032  057066  057116  PFECWS: .WORD   PFECEM,PFECDH,PFECDT,PFECDF     ;R-C
      057030  057126
14462 057032    120     117     127  PFECEM: .ASCIZ  'POWER MONITOR BIT FOUND SET'   ;R-C
      057035    105     122     040
      057040    115     117     116
      057043    111     124     117
      057046    122     040     102
      057051    111     124     040
      057054    106     117     125
      057057    116     104     040
      057062    123     105     124
      057065    000
14463 057066    124     105     123  PFECDH: .ASCIZ  'TESTNO  ERR PC  CPUERR'       ;R-C
      057071    124     116     117
      057074    040     040     105
      057077    122     122     040
      057102    120     103     040
      057105    040     103     120
      057110    125     105     122
      057113    122     000
14464                                         .EVEN                                   ;R-C
14465 057116  063032  002016  056506  PFECDT: .WORD   $TESTN,ERRPC,CPSAVE,0          ;R-C
      057124  000000
14466 057126    000     000     000  PFECDF: .BYTE   0,0,0,0                         ;R-C
      057131    000
14467
```

```
14470                                   ;*****************************************************************************
14471                                   ;*** OCTAL ***
14472                                   ;*****************************************************************************
14473 057132                            TAG70$: TYPOCT  @(R1)           ;;TYPE AN OCTAL NUMBER
14474 057140  000207                            RETURN
14475
14476                                   ;*****************************************************************************
14477                                   ;*** DECIMAL ***
14478                                   ;*****************************************************************************
14479 057142                            TAG71$: TYPDEC  @(R1)           ;;TYPE A DECIMAL NUMBER
14480 057150  000207                            RETURN
14481
14482                                   ;*****************************************************************************
14483                                   ;*** INTERLEAVE ***
14484                                   ;*****************************************************************************
14485 057152                            TAG72$: PUSH    R1,R5
14486 057156  013701  002100                    MOV     BANK,R1
14487 057162  070127  000004                    MUL     #4,R1
14488 057166                                    SET     NOTAB                   ;INDICATE NO TABLE TO BE PRINTED - NOW
14489 057174                                    TYPE    MSG014
14490 057200  004737  037254                    CALL    TCFIG1
14491 057204  005037  002342                    CLR     NOTAB
14492 057210                                    POP     R5,R1
14493 057214                                    TYPE    MSG014                  ;1 SPACE
14494 057220  000207                            RETURN
14495
14496                                   ;*****************************************************************************
14497                                   ;*** CSR ***
14498                                   ;*****************************************************************************
14499 057222                            TAG73$: PUSH    R1,R5
14500 057226  013701  002100                    MOV     BANK,R1
14501 057232  070127  000004                    MUL     #4,R1
14502 057236                                    SET     NOTAB
14503 057244  004737  037574                    CALL    TCFIG3
14504 057250  005037  002342                    CLR     NOTAB
14505 057254                                    POP     R5,R1
14506 057260  000207                            RETURN
14507
14508                                   ;*****************************************************************************
14509                                   ;*** PATTERN ***
14510                                   ;*****************************************************************************
14511 057262                            TAG74$: TYPOCS  REALPAT,<TYPE (0-77)>,2,Z
14512 057272  000207                            RETURN
14513
14514                                   ;*****************************************************************************
14515                                   ;*** BANK ***
14516                                   ;*****************************************************************************
14517 057274                            TAG75$: TYPOCS  BANK,<TYPE (0-167)>,3
14518 057304  000207                            RETURN
```

```
14520                                   ;*****************************************************************************
14521                                   ;*** MTYPE ***
14522                                   ;*****************************************************************************
14523 057306                            TAG76$: PUSH    R1,R5
14524 057312  013701  002100                    MOV     BANK,R1
14525 057316  070127  000004                    MUL     #4,R1
14526 057322                                    SET     NOTAB
14527 057330                                    TYPE    MSG019
14528 057334  004737  037406                    CALL    TCFIG2
14529 057340  005037  002342                    CLR     NOTAB
14530 057344                                    POP     R5,R1
14531 057350  000207                            RETURN
14532
14533                                   ;*****************************************************************************
14534                                   ;*** UNKNOWN DATA ***
14535                                   ;*****************************************************************************
14536 057352                            TAG77$: TYPE    MSG061
14537 057356  000207                            RETURN
14538
14539                                   ;*****************************************************************************
14540                                   ;*** PHYSICAL ADDRESS ***
14541                                   ;*****************************************************************************
14542 057360  013737  002032  002036  TAG78$: MOV     ADDRESS,PHYADD
14543 057366  162737  060000  002036          SUB     #FIRST,PHYADD
14544 057374  013737  002100  002040          MOV     BANK,PHYADD+2
14545 057402  006237  002040                  ASR     PHYADD+2
14546 057406  103003                          BCC     1$
14547 057410  052737  100000  002036          BIS     #BIT15,PHYADD
14548 057416  012746  002036          1$:     MOV     #PHYADD,-(SP)           ;POINTER TO DOUBLE WORD ON STACK
14549 057422  004737  062706                  CALL    $DB2O                   ;CALL DOUBLE PRECISION CONVERSION ROUTINE
14550 057426  062706  000002                  ADD     #2,SP                   ;FIX STACK
14551 057432                                    TYPE    $OCT8
14552 057436  000207                            RETURN
14553
14554                                   ;*****************************************************************************
14555                                   ;*** OCTAL BYTE ***
14556                                   ;*****************************************************************************
14557 057440                            TAG79$: TYPE    MSG018          ;2 SPACES
14558 057444                                    TYPOCS  @(R1),<TYPE BYTE>,3,Z
14559 057454                                    TYPE    MSG014          ;SPACE
14560 057460  000207                            RETURN
```

```
14604 057462                          DETAIL: SUBTST  <<SUBR  DETAILED ERROR REPORT>>
                                      ;*****************************************************************************
                                      ;*SUBTEST       SUBR    DETAILED ERROR REPORT
                                      ;*****************************************************************************
14605 057462  005237  002212                    INC     DETFLAG
14606 057466  022737  000003  002212            CMP     #3,DETFLAG
14607 057474  101473                            BLOS    4$
14608 057476  022737  000002  002212            CMP     #2,DETFLAG
14609 057504  001435                            BEQ     2$
14610 057506                                    PUSH    HEADER,MUT
14611 057516                                    SET     HEADER
14612 057524  005037  002106                    CLR     MUT
14613 057530  010037  002172                    MOV     R0,DETR0
14614 057534  012700  002174                    MOV     #DETR1,R0
14615 057540  010120                            MOV     R1,(R0)+
14616 057542  010220                            MOV     R2,(R0)+
14617 057544  010320                            MOV     R3,(R0)+
14618 057546  010420                            MOV     R4,(R0)+
14619 057550  010520                            MOV     R5,(R0)+
14620 057552  013720  002022                    MOV     ERRSP,(R0)+
14621 057556  013720  002026                    MOV     ERRPSW,(R0)+
14622 057562  013700  002172                    MOV     DETR0,R0
14623 057566                                    SET     NOERROR
14624 057574  104013                            ERROR   +13
14625 057576  000423                            BR      1$
14626 057600                          2$:       PUSH    HEADER,MUT
14627 057610                                    SET     HEADER
14628 057616  005037  002106                    CLR     MUT
14629 057622                                    SET     NOERROR
14630 057630  104031                            ERROR   +31
14631 057632  022737  000001  003716            CMP     #1,PROTYP               ;IS THIS AN 11/44?
14632 057640  001002                            BNE     1$
14633 057642  005037  177766                    CLR     CPUERR
14634 057646                          1$:       POP     MUT,HEADER
14635                                           ;WARNING RECURSIVE
14636 057656  004737  057462                    CALL    DETAIL
14637 057662  000207                            RETURN
```

```
14640
14641 057664  004737  061332          4$:     CALL    CONTT                   ;DISPLAY 'DISPLAY'' INFO
14642
14643                                         ;TYPE CONTENTS OF ALL CSR'S
14644 057670                                  PUSH    CSR,CSRNO,R1
14645 057702                                  TYPE    MSG058
14646 057706                                  TYPE    $CRLF
14647 057712  013701  002216                 MOV     TOTCSRS,R1
14648 057716                                  BEGIN DUMPCSRLOOP
14649 057716                                    FOR CSRNO := #0 TO #36 BY #2
14650 057722  006301                              ASL         R1
14651 057724                                      ON.ERROR
14652 057726  104426                                READCSR
14653 057730                                        TYPOCT    CSR
14654 057736                                        TYPE      MSG018          ;2 SPACES
14655 057742                                      END ;OF ON.ERROR
14656 057742                                      IF R1 EQ #0 THEN LEAVE DUMPCSRLOOP
14657 057746                                    END ;OF FOR CSRNO
14658 057764                                  END DUMPCSRLOOP
14659 057764                                  POP     R1,CSRNO,CSR
14660
14661                                         ;TYPE STACKS
14662 057776                                  PUSH    R0,R1
14663 060002                                  TYPE    MSG088          ;KERNEL STACK
14664 060006  013701  002534                 MOV     KSTACK,R1
14665 060012  162701  000002                 SUB     #2,R1
14666 060016                                  FOR     R0 := SP TO R1 BY #2
14667 060020                                    TYPE  $CRLF
14668 060024                                    TYPOCT        R0
14669 060030                                    TYPE          MSG018          ;2 SPACES
14670 060034                                    TYPOCT        (R0)
14671 060040                                  END ;OF FOR R0
14672                                         ;SET PREVIOUS MODE TO SUPERVISOR
14673 060050  005737  002426                 TST     NOSUPER
14674 060054  001036                         BNE     DET1
14675 060056  042737  030000  177776         BIC     #BIT13!BIT12,PSW
14676 060064  052737  010000  177776         BIS     #BIT12,PSW
14677 060072  006506                         MFPI    SSP
14678 060074                                  POP     R1,R0
14679 060100                                  TYPE    MSG089          ;SUPERVISOR STACK
14680 060104                                  IF R0 LT #SUPSTK
14681 060112                                    FOR R0 := R0 TO #SUPSTK-2 BY #2
14682 060112                                      TYPE        $CRLF
14683 060116                                      TYPOCT      R0
14684 060122                                      TYPE        MSG018          ;2 SPACES
14685 060126                                      TYPOCT      (R0)
14686 060132                                    END ;OF FOR R0
14687 060144                                  ELSE
14688 060146                                    TYPE  MSG091          ;IS EMPTY
14689 060152                                  END ;OF IF R0
14690                                         ;SET PREVIOUS MODE TO USER
14691 060152  052737  030000  177776  DET1:   BIS     #BIT13!BIT12,PSW
14692 060160  006506                         MFPI    USP
14693 060162                                  POP     R0
14694 060164                                  TYPE    MSG090          ;USER STACK
14695 060170                                  IF R0 LT #USESTK
14696 060176                                    FOR R0 := R0 TO #USESTK-2 BY #2
```

```
14697 060176                                        TYPE        $CRLF
14698 060202                                        TYPOCT      R0
14699 060206                                        TYPE        MSG018          ;2 SPACES
14700 060212                                        TYPOCT      (R0)
14701 060216                                      END ;OF FOR R0
14702 060230                                    ELSE
14703 060232                                      TYPE  MSG091          ;IS EMPTY
14704 060236                                    END ;OF IF R0
14705 060236                                    TYPE    $CRLF
14706 060242  005037  002212                    CLR     DETFLAG
14707 060246                                    POP     R0
14708 060250  000207                            RETURN
```

```
14746                                   .SBTTL  ROUTINE BINARY TO OCTAL (ASCII) AND TYPE
14747
14748                           ;*************************************************************************
14749                           ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
14750                           ;*OCTAL (ASCII) NUMBER AND TYPE IT.
14751                           ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
14752                           ;*CALL:
14753                           ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
14754                           ;*      TYPOS                   ;;CALL FOR TYPEOUT
14755                           ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
14756                           ;*      .BYTE   M               ;;M=1 OR 0
14757                           ;*                                      ;;1=TYPE LEADING ZEROS
14758                           ;*                                      ;;0=SUPPRESS LEADING ZEROS
14759                           ;*
14760                           ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
14761                           ;*$TYPOS OR $TYPOC
14762                           ;*CALL:
14763                           ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
14764                           ;*      TYPON                   ;;CALL FOR TYPEOUT
14765                           ;*
14766                           ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
14767                           ;*CALL:
14768                           ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
14769                           ;*      TYPOC                   ;;CALL FOR TYPEOUT
14770
14771 060252  017646  000000    $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
14772 060256  116637  000001  060475        MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
14773 060264  112637  060477            MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
14774 060270  062716  000002            ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
14775 060274  000406                    BR      $TYPON
14776 060276  112737  000001  060475  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
14777 060304  112737  000006  060477        MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
14778 060312  112737  000005  060474  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
14779 060320  010346                    MOV     R3,-(SP)        ;;SAVE R3
14780 060322  010446                    MOV     R4,-(SP)        ;;SAVE R4
14781 060324  010546                    MOV     R5,-(SP)        ;;SAVE R5
14782 060326  113704  060477            MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
14783 060332  005404                    NEG     R4
14784 060334  062704  000006            ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
14785 060340  110437  060476            MOVB    R4,$OMODE       ;;SAVE IT FOR USE
14786 060344  113704  060475            MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
14787 060350  016605  000012            MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
14788 060354  005003                    CLR     R3              ;;CLEAR THE OUTPUT WORD
14789 060356  006105                1$: ROL     R5              ;;ROTATE MSB INTO "C"
14790 060360  000404                    BR      3$              ;;GO DO MSB
14791 060362  006105                2$: ROL     R5              ;;FORM THIS DIGIT
14792 060364  006105                    ROL     R5
14793 060366  006105                    ROL     R5
14794 060370  010503                    MOV     R5,R3
14795 060372  006103                3$: ROL     R3              ;;GET LSB OF THIS DIGIT
14796 060374  105337  060476            DECB    $OMODE          ;;TYPE THIS DIGIT?
14797 060400  100016                    BPL     6$              ;;BR IF NO
14798 060402  042703  177770            BIC     #177770,R3      ;;GET RID OF JUNK
14799 060406  001002                    BNE     4$              ;;TEST FOR 0
14800 060410  005704                    TST     R4              ;;SUPPRESS THIS 0?
14801 060412  001403                    BEQ     5$              ;;BR IF YES
14802 060414  005204                4$: INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
```

```
14803 060416  052703  000060              BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
14804 060422  052703  000040        5$:   BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
14805 060426  110337  060472              MOVB    R3,8$           ;;SAVE FOR TYPING
14806 060432                              TYPE    8$              ;;GO TYPE THIS DIGIT
14807 060436  105337  060474        6$:   DECB    $OCNT           ;;COUNT BY 1
14808 060442  003347                      BGT     2$              ;;BR IF MORE TO DO
14809 060444  002402                      BLT     7$              ;;BR IF DONE
14810 060446  005204                      INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
14811 060450  000744                      BR      2$              ;;GO DO THE LAST DIGIT
14812 060452  012605                7$:   MOV     (SP)+,R5        ;;RESTORE R5
14813 060454  012604                      MOV     (SP)+,R4        ;;RESTORE R4
14814 060456  012603                      MOV     (SP)+,R3        ;;RESTORE R3
14815 060460  016666  000002  000004      MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
14816 060466  012616                      MOV     (SP)+,(SP)
14817 060470  000002                      RTI                     ;;RETURN
14818 060472     000                8$:   .BYTE   0               ;;STORAGE FOR ASCII DIGIT
14819 060473     000                      .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
14820 060474     000          $OCNT:   .BYTE   0               ;;OCTAL DIGIT COUNTER
14821 060475     000          $OFILL:  .BYTE   0               ;;ZERO FILL SWITCH
14822 060476  000000          $OMODE:  .WORD   0               ;;NUMBER OF DIGITS TO TYPE
```

ROUTINE CONVERT BINARY TO DECIMAL AND TYPE

```
                                        .SBTTL  ROUTINE CONVERT BINARY TO DECIMAL AND TYPE
                                ;*******************************************************************************
                                ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
                                ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
                                ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
                                ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
                                ;*REPLACED WITH SPACES.
                                ;*CALL:
                                ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
                                ;*      TYPDS                   ;;GO TO THE ROUTINE
060500                  $TYPDS: PUSH    R0,R1,R2,R3,R5
060512  012746  020200          MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
060516  016605  000020          MOV     20(SP),R5       ;;GET THE INPUT NUMBER
060522  100004                  BPL     1$              ;;BR IF INPUT IS POS.
060524  005405                  NEG     R5              ;;MAKE THE BINARY NUMBER POS.
060526  112766  000055  000001      MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
060534  005000              1$: CLR     R0              ;;ZERO THE CONSTANTS INDEX
060536  012703  060714          MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
060542  112723  000040          MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
060546  005002              2$: CLR     R2              ;;CLEAR THE BCD NUMBER
060550  016001  060704          MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
060554  160105              3$: SUB     R1,R5           ;;FORM THIS BCD DIGIT
060556  002402                  BLT     4$              ;;BR IF DONE
060560  005202                  INC     R2              ;;INCREASE THE BCD DIGIT BY 1
060562  000774                  BR      3$
060564  060105              4$: ADD     R1,R5           ;;ADD BACK THE CONSTANT
060566  005702                  TST     R2              ;;CHECK IF BCD DIGIT=0
060570  001002                  BNE     5$              ;;FALL THROUGH IF 0
060572  105716                  TSTB    (SP)            ;;STILL DOING LEADING 0'S?
060574  100407                  BMI     7$              ;;BR IF YES
060576  106316              5$: ASLB    (SP)            ;;MSD?
060600  103003                  BCC     6$              ;;BR IF NO
060602  116663  000001  177777      MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
060610  052702  000060      6$: BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
060614  052702  000040      7$: BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
060620  110223                  MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
060622  005720                  TST     (R0)+           ;;JUST INCREMENTING
060624  020027  000010          CMP     R0,#10          ;;CHECK THE TABLE INDEX
060630  002746                  BLT     2$              ;;GO DO THE NEXT DIGIT
060632  003002                  BGT     8$              ;;GO TO EXIT
060634  010502                  MOV     R5,R2           ;;GET THE LSD
060636  000764                  BR      6$              ;;GO CHANGE TO ASCII
060640  105726              8$: TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
060642  100003                  BPL     9$              ;;BR IF NO
060644  116663  177777  177776      MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
060652  105013              9$: CLRB    (R3)            ;;SET THE TERMINATOR
060654                          POP     R5,R3,R2,R1,R0
060666                          TYPE    $DBLK           ;;NOW TYPE THE NUMBER
060672  016666  000002  000004      MOV     2(SP),4(SP)     ;;ADJUST THE STACK
060700  012616                  MOV     (SP)+,(SP)
060702  000002                  RTI                     ;;RETURN TO USER
060704  023420              $DTBL:  10000.
060706  001750                  1000.
060710  000144                  100.
060712  000012                  10.
060714  000000  000000  000000 $DBLK:  .WORD   0,0,0,0
060722  000000
```

```
14881                                           .SBTTL  ROUTINE TTY INPUT
14882                                   ;*************************************************************************************
14883                                   ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
14884                                   ;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
14885                                   ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
14886                                   ;*WHEN OPERATING IN TTY FLAG MODE.
14887                                           .ENABLE LSB
14888 060724                           $CKSWR:
14894 060724  005737  053504                     TST     XOCHAR          ;;SOMETHING THERE?
14895 060730  001406                             BEQ     NOCH            ;; GO ON IF NOT
14896 060732  013746  053504                     MOV     XOCHAR,-(SP)    ;; USE IT
14897 060736  005037  053504                     CLR     XOCHAR
14898 060742  000137  060764                     JMP     CONTS1
14899 060746  105777  121630           NOCH:     TSTB    @$TKS           ;;CHAR THERE?
14900 060752  100130                             BPL     12$             ;;IF NO, DON'T WAIT AROUND
14901 060754  117746  121624                     MOVB    @$TKB,-(SP)     ;;SAVE THE CHAR
14902 060760  042716  177600                     BIC     #^C177,(SP)     ;;STRIP-OFF THE ASCII
14903 060764  022716  000006           CONTS1:   CMP     #6,(SP)         ;IS IT CONTROL F?
14904 060770  001002                             BNE     1$              ;NO SKIP
14905 060772  004737  045404                     CALL    FIELDSERVICE
14906 060776  022716  000024           1$:       CMP     #24,(SP)        ;IS IT CONTROL T?
14907 061002  001002                             BNE     16$             ;NO - SKIP
14908 061004  004737  061332                     CALL    CONTT           ;YES - CALL CONTROL T ROUTINE
14909 061010  022716  000003           16$:      CMP     #3,(SP)         ;IS IT CONTROL C?
14910 061014  001454                             BEQ     5$              ;YES EXIT *****NOTE***** STACK IS SCREWED UP!
14911 061016  022716  000023           2$:       CMP     #23,(SP)        ;IS IT CONTRGL S?
14912 061022  001002                             BNE     17$             ;NO - SKIP
14913 061024  004737  061406                     CALL    CONTS           ;YES - CALL CONTROL S ROUTINE
14914 061030  022716  000013           17$:      CMP     #13,(SP)        ;IS IT CONTROL K?
14915 061034  001005                             BNE     6$              ;NO - SKIP
14916 061036                                     TYPE    $CNTLK          ;TYPE A ^K
14917 061042  013706  002142                     MOV     CTLKVEC,SP      ;RESET KSP TO AFTER PATTERN EXEC ROUTINE
14918 061046  000207                             RETURN                  ;RETURN TO PATTERN EXEC ROUTINE
14919 061050  022737  000176  002576  6$:       CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED?
14920 061056  001067                             BNE     CKEND           ;;BRANCH IF NO
14921 061060  022716  000007                     CMP     #7,(SP)         ;;IS IT A CONTROL G?
14922 061064  001064                             BNE     CKEND           ;;NO, RETURN TO USER
14923 061066  005737  002060                     TST     $AUTO           ;ARE WE RUNNING IN AUTO-MODE?
14924 061072  001061                             BNE     CKEND           ;BRANCH IF YES
14925 061074                                     TYPE    $CNTLG          ;;ECHO THE CONTROL-G (^G)
14926 061100                           $GTSWR:   TYPE    $MSWR           ;;TYPE CURRENT CONTENTS
14927 061104                                     TYPOCT  @SWR            ;;OF THE SWR
14928 061112                                     TYPE    $MNEW           ;;PROMPT FOR NEW SWR
14929 061116  005046                     3$:     CLR     -(SP)           ;;CLEAR COUNTER
14930 061120  005046                             CLR     -(SP)           ;;THE NEW SWR
14931 061122  105777  121454             4$:     TSTB    @$TKS           ;;CHAR THERE?
14932 061126  100375                             BPL     4$              ;;IF NOT TRY AGAIN
14933 061130  117746  121450                     MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
14934 061134  042716  177600                     BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
14935 061140  021627  000003                     CMP     (SP),#3         ;;IS IT A CONTROL-C?
14936 061144  001006                             BNE     7$              ;;BRANCH IF NOT
14937 061146                             5$:     TYPE    $CNTLC          ;;YES, ECHO CONTROL-C (^C)
14938 061152  062706  000006                     ADD     #6,SP           ;;CLEAN UP STACK
14939 061156  000137  045026                     JMP     BOOT            ;;CONTROL-C RESTART
14940 061162  021627  000025             7$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
14941 061166  001005                             BNE     9$              ;;BRANCH IF NOT
14942 061170                                     TYPE    $CNTLU          ;;YES, ECHO CONTROL-U (^U)
```

```
14943 061174  062706  000006          8$:     ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
14944 061200  000746                  BR      3$              ;;LET'S TRY IT AGAIN
14945 061202  021627  000015          9$:     CMP     (SP),#15        ;;IS IT A <CR>?
14946 061206  001016                  BNE     13$             ;;BRANCH IF NO
14947 061210  005766  000004                  TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
14948 061214  001403                  BEQ     10$             ;;BRANCH IF YES
14949 061216  016677  000002 121352          MOV     2(SP),@SWR      ;;SAVE NEW SWR
14950 061224  062706  000006          10$:    ADD     #6,SP           ;;CLEAR UP STACK
14951 061230                          TYPE    $CRLF           ;;ECHO <CR> AND <LF>
14952 061234  000002                  12$:    RTI                     ;;RETURN
14953 061236  062706  000002          CKEND:  ADD     #2,SP           ;FIX STACK
14954 061242  000002                  RTI                     ;RETURN
14955 061244  004737  053506          13$:    CALL    $TYPEC          ;;ECHO CHAR
14956 061250  021627  000060                  CMP     (SP),#60        ;;CHAR < 0?
14957 061254  002420                  BLT     15$             ;;BRANCH IF YES
14958 061256  021627  000067                  CMP     (SP),#67        ;;CHAR > 7?
14959 061262  003015                  BGT     15$             ;;BRANCH IF YES
14960 061264  042726  000060                  BIC     #60,(SP)+       ;;STRIP-OFF ASCII
14961 061270  005766  000002                  TST     2(SP)           ;;IS THIS THE FIRST CHAR
14962 061274  001403                  BEQ     14$             ;;BRANCH IF YES
14963 061276  006316                  ASL     (SP)            ;;NO, SHIFT PRESENT
14964 061300  006316                  ASL     (SP)            ;;    CHAR OVER TO MAKE
14965 061302  006316                  ASL     (SP)            ;;    ROOM FOR NEW ONE.
14966 061304  005266  000002          14$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
14967 061310  056616  177776                  BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
14968 061314  000702                  BR      4$              ;;GET THE NEXT ONE
14969 061316                          15$:    TYPE    $QUES           ;;TYPE ?<CR><LF>
14970 061322  000724                  BR      8$              ;;SIMULATE CONTROL-U
14971 061324     136     113     015  $CNTLK: .ASCIZ   /^K/<15><12>    ;CONTROL K ASCII STRING
      061327     012     000
14972                                 .EVEN
14973                                 .DSABL  LSB
```

```
14976 061332                            CONTT:  SUBTST  <<CONTROL T>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       CONTROL T
                                        ;*****************************************************************************
14977 061332                                    PUSH    R0
14978 061334                                    TYPE    $CRLF
14988 061340                                      IF RLFLAG IS TRUE
14989 061346                                        TYPE        MSG092          ;RELOCATED
14990 061352                                      END ;OF IF RLFLAG
14991 061352                                      TYPE  MSG093                  ;BANK=
14992 061356                                      TYPOCS        BANK,,3         ;TYPE 3 DIGITS
14993 061366                                      TYPE  MSG095                  ;PAT=
14994 061372                                      TYPOCS        REALPAT,,2      ;TYPE 2 DIGITS
14998 061402                                    POP     R0
14999 061404  000207                            RETURN
15000
15001 061406                            CONTS:  SUBTST  <<CONTROL S & CONTROL Q>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       CONTROL S & CONTROL Q
                                        ;*****************************************************************************
15002 061406                                    POP     R0              ;GET RID OF RETURN ADDRESS FROM STACK
15003 061410  105777  121166            CONTS2: TSTB    @$TKS           ;WAIT FOR CHARACTER
15004 061414  100375                            BPL     CONTS2
15005 061416  117716  121162                    MOVB    @$TKB,(SP)      ;REPLACE OVER OLD CHARACTER ON STACK
15006 061422  042716  177600                    BIC     #^C177,(SP)     ;STRIP ALL BUT ASCII
15007 061426                                    IF (SP) EQ #21          ;IF IT IS A CONTROL Q
15008 061434  000137  060764                      JMP   CONTS1
15009 061440                                    ELSE
15010 061442  000762                              BR    CONTS2
15011 061444                                    END ;OF IF (SP)
```

```
15013                                   ;*****************************************************************************
15014                                   ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
15015                                   ;*CALL:
15016                                   ;*      RDCHR                   ;;INPUT A SINGLE CHARACTER FROM THE TTY
15017                                   ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
15018                                   ;*                              ;;WITH PARITY BIT STRIPPED OFF
15019                                   ;
15020
15021 061444  011646                   $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
15022 061446  016666  000004  000002           MOV     4(SP),2(SP)     ;;SAVE THE PS
15023 061454  105777  121122           1$:     TSTB    @$TKS           ;;WAIT FOR
15024 061460  100375                           BPL     1$              ;;A CHARACTER
15025 061462  117766  121116  000004           MOVB    @$TKB,4(SP)     ;;READ THE TTY
15026 061470  042766  177600  000004           BIC     #^C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
15027 061476  026627  000004  000023           CMP     4(SP),#23       ;;IS IT A CONTROL-S?
15028 061504  001013                           BNE     3$              ;;BRANCH IF NO
15029 061506  105777  121070           2$:     TSTB    @$TKS           ;;WAIT FOR A CHARACTER
15030 061512  100375                           BPL     2$              ;;LOOP UNTIL ITS THERE
15031 061514  117746  121064                   MOVB    @$TKB,-(SP)     ;;GET CHARACTER
15032 061520  042716  177600                   BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
15033 061524  022627  000021                   CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
15034 061530  001366                           BNE     2$              ;;IF NOT DISCARD IT
15035 061532  000750                           BR      1$              ;;YES, RESUME
15036 061534  026627  000004  000021  3$:     CMP     4(SP),#21       ;;IS IT A RANDOM CONTROL-Q?     ;R-C
15037 061542  001744                           BEQ     1$              ;;BRANCH BACK IF SO             ;R-C
15038 061544  026627  000004  000140           CMP     4(SP),#140      ;;IS IT UPPER CASE?
15039 061552  002407                           BLT     4$              ;;BRANCH IF YES
15040 061554  026627  000004  000175           CMP     4(SP),#175      ;;IS IT A SPECIAL CHAR?
15041 061562  003003                           BGT     4$              ;;BRANCH IF YES
15042 061564  042766  000040  000004           BIC     #40,4(SP)       ;;MAKE IT UPPER CASE
15043 061572  000002                   4$:     RTI                     ;;GO BACK TO USER
15044                                   ;*****************************************************************************
15045                                   ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
15046                                   ;*CALL:
15047                                   ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
15048                                   ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
15049                                   ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
15050 061574  010346                   $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
15051 061576  005046                           CLR     -(SP)           ;;CLEAR THE RUBOUT KEY
15052 061600  012703  062072           1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
15053 061604  022703  062116           2$:     CMP     #$TTYIN+20.,R3  ;;BUFFER FULL?
15054 061610  101477                           BLOS    8$              ;;BR IF YES
15055 061612  104411                           RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
15056 061614  112613                           MOVB    (SP)+,(R3)      ;;GET CHARACTER
15057 061616  122713  000003                   CMPB    #3,(R3)         ;;IS IT A CONTROL-C?
15058 061622  001016                           BNE     3$              ;;BRANCH IF NO
15059 061624                                   TYPE    $CNTLC          ;;TYPE A CONTROL-C (^C)
15060 061630  005726                           TST     (SP)+           ;;CLEAN RUBOUT KEY OFF OF THE STACK
15061 061632  012603                           MOV     (SP)+,R3        ;;RESTORE R3
15062 061634  032777  000400  120734           BIT     #BIT8,@SWR      ;;IS THERE A HALT FLAG SET IN THE SWR?
15063 061642  001404                           BEQ     11$             ;;BRANCH IF NOT TO BOOT ROUTINE
15064 061644  005037  002370                   CLR     STOPOK          ;;GET READY TO HALT PROGRAM
15065 061650  000137  045134                   JMP     EXIT            ;;GO HALT PROGRAM
15066 061654  000137  045026           11$:    JMP     BOOT            ;;GOTO CONTROL-C RESTART
15067 061660  122713  000177           3$:     CMPB    #177,(R3)       ;;IS IT A RUBOUT
15068 061664  001022                           BNE     5$              ;;BR IF NO
15069 061666  005716                           TST     (SP)            ;;IS THIS THE FIRST RUBOUT?
```

```
15070 061670  001007                        BNE     4$              ;;BR IF NO
15071 061672  112737  000134  062070        MOVB    #'\,10$         ;;TYPE A BACK SLASH
15072 061700                                TYPE    10$
15073 061704  012716  177777                MOV     #-1,(SP)        ;;SET THE RUBOUT KEY
15074 061710  005303                4$:     DEC     R3              ;;BACKUP BY ONE
15075 061712  020327  062072                CMP     R3,#$TTYIN      ;;STACK EMPTY?
15076 061716  103434                        BLO     8$              ;;BR IF YES
15077 061720  111337  062070                MOVB    (R3),10$                ;;SETUP TO TYPEOUT THE DELETED CHAR.
15078 061724                                TYPE    10$             ;;GO TYPE
15079 061730  000725                        BR      2$              ;;GO READ ANOTHER CHAR.
15080 061732  005716                5$:     TST     (SP)            ;;RUBOUT KEY SET?
15081 061734  001406                        BEQ     6$              ;;BR IF NO
15082 061736  112737  000134  062070        MOVB    #'\,10$         ;;TYPE A BACK SLASH
15083 061744                                TYPE    10$
15084 061750  005016                        CLR     (SP)            ;;CLEAR THE RUBOUT KEY
15085 061752  122713  000025          6$:   CMPB    #25,(R3)        ;;IS CHARACTER A CTRL U?
15086 061756  001003                        BNE     7$              ;;BR IF NO
15087 061760                                TYPE    $CNTLU          ;;TYPE A CONTROL 'U''
15088 061764  000705                        BR      1$              ;;GO START OVER
15089 061766  122713  000022          7$:   CMPB    #22,(R3)        ;;IS CHARACTER A ''^R''?
15090 061772  001011                        BNE     9$              ;;BRANCH IF NO
15091 061774  105013                        CLRB    (R3)            ;;CLEAR THE CHARACTER
15092 061776                                TYPE    $CRLF           ;;TYPE A ''CR'' & ''LF''
15093 062002                                TYPE    $TTYIN          ;;TYPE THE INPUT STRING
15094 062006  000676                        BR      2$              ;;GO PICKUP ANOTHER CHACTER
15095 062010                          8$:   TYPE    $QUES           ;;TYPE A '?'
15096 062014  000671                        BR      1$              ;;CLEAR THE BUFFER AND LOOP
15097 062016  111337  062070          9$:   MOVB    (R3),10$                ;;ECHO THE CHARACTER
15098 062022                                TYPE    10$
15099 062026  122723  000015                CMPB    #15,(R3)+       ;;CHECK FOR RETURN
15100 062032  001264                        BNE     2$              ;;LOOP IF NOT RETURN
15101 062034  105063  177777                CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
15102 062040                                TYPE    $LF             ;;TYPE A LINE FEED
15103 062044  005726                        TST     (SP)+           ;;CLEAN RUBOUT KEY FROM THE STACK
15104 062046  012603                        MOV     (SP)+,R3        ;;RESTORE R3
15105 062050  011646                        MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
15106 062052  016666  000004  000002        MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
15107 062060  012766  062072  000004        MOV     #$TTYIN,4(SP)
15108 062066  000002                        RTI                     ;;RETURN
15109 062070     000                10$:    .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
15110 062071     000                        .BYTE   0               ;;TERMINATOR
15111 062072  000024                $TTYIN: .REPT   20.             ;;RESERVE SIZE BYTES FOR TTY INPUT
15114 062116     136     103     015 $CNTLC: .ASCIZ  /^C/<15><12>    ;;CONTROL ''C''
      062121     012     000
15115 062123     136     125     015 $CNTLU: .ASCIZ  /^U/<15><12>    ;;CONTROL 'U''
      062126     012     000
15116 062130     136     107     015 $CNTLG: .ASCIZ  /^G/<15><12>    ;;CONTROL ''G''
      062133     012     000
15117 062135     015     012     123 $MSWR:  .ASCIZ  <15><12>/SWR = /
      062140     127     122     040
      062143     075     040     000
15118 062146     040     040     116 $MNEW:  .ASCIZ  /  NEW = /
      062151     105     127     040
      062154     075     040     000
15119                                       .EVEN
```

```
15121                                           .SBTTL  ROUTINE READ AN OCTAL NUMBER FROM THE TTY
15122                                   ;*******************************************************************************
15123                                   ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
15124                                   ;*CHANGE IT TO BINARY.
15125                                   ;*THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL
15126                                   ;*OCTAL DIGITS. IF AN ILLEGAL CHARACTER IS READ A "?" WILL BE TYPED
15127                                   ;*FOLLOWED BY A CARRIAGE RETURN-LINE FEED. THE COMPLETE NUMBER MUST
15128                                   ;*THEN BE RETYPED. THE INPUT IS TERMINATED BY TYPING A CARRIAGE RETURN.
15129                                   ;*CALL:
15130                                   ;*      RDOCT                   ;;READ AN OCTAL NUMBER
15131                                   ;*      RETURN HERE             ;;LOW ORDER BITS ARE ON TOP OF THE STACK
15132                                   ;*                              ;;HIGH ORDER BITS ARE IN $HIOCT
15133 062160  011646                  $RDOCT: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR THE
15134 062162  016666  000004  000002          MOV     4(SP),2(SP)     ;;INPUT NUMBER
15135 062170                                  PUSH    R0,R1,R2
15136 062176  104412                  1$:     RDLIN                   ;;READ AN ASCIZ LINE
15137 062200  012600                          MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
15138 062202  010037  062306                  MOV     R0,5$           ;;AND SAVE IT
15139 062206  005001                          CLR     R1              ;;CLEAR DATA WORD
15140 062210  005002                          CLR     R2
15141 062212  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
15142 062214  001420                          BEQ     3$              ;;IF ZERO GET OUT
15143 062216  122716  000060                  CMPB    #'0,(SP)        ;;MAKE SURE THIS CHARACTER
15144 062222  003026                          BGT     4$              ;;IS AN OCTAL DIGIT
15145 062224  122716  000067                  CMPB    #'7,(SP)
15146 062230  002423                          BLT     4$
15147 062232  006301                          ASL     R1              ;;*2
15148 062234  006102                          ROL     R2
15149 062236  006301                          ASL     R1              ;;*4
15150 062240  006102                          ROL     R2
15151 062242  006301                          ASL     R1              ;;*8
15152 062244  006102                          ROL     R2
15153 062246  042716  177770                  BIC     #^C7,(SP)       ;;STRIP THE ASCII JUNK
15154 062252  062601                          ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
15155 062254  000756                          BR      2$              ;;LOOP
15156 062256  005726                  3$:     TST     (SP)+           ;;CLEAN TERMINATOR FROM STACK
15157 062260  010166  000012                  MOV     R1,12(SP)       ;;SAVE THE RESULT
15158 062264  010237  062326                  MOV     R2,$HIOCT
15159 062270                                  POP     R2,R1,R0
15160 062276  000002                          RTI                     ;;RETURN
15161 062300  005726                  4$:     TST     (SP)+           ;;CLEAN PARTIAL FROM STACK
15162 062302  105010                          CLRB    (R0)            ;;SET A TERMINATOR
15163 062304                                  TYPE                    ;;TYPE UP THRU THE BAD CHAR.
15164 062306  000000                  5$:     .WORD   0
15165 062310                                  TYPE    MSG062          ;INPUT MUST BE A
15166 062314                                  TYPE    MSG063          ;N OCTAL
15167 062320                                  TYPE    MSG064          ;NUMBER
15168 062324  000724                          BR      1$              ;;TRY AGAIN
15169 062326  000000                  $HIOCT: .WORD   0               ;;HIGH ORDER BITS GO HERE
15170                                           .SBTTL  ROUTINE READ A DECIMAL NUMBER FROM THE TTY
15171
15172                                   ;*******************************************************************************
15173                                   ;*THIS ROUTINE WILL READ A DECIMAL (ASCII) NUMBER FROM THE TTY AND
15174                                   ;*CHANGE IT TO BINARY. IF TOO MANY CHARACTERS OR ANY ILLEGAL CHARACTERS
15175                                   ;*ARE READ A "?" FOLLOWED BY A CARRIAGE RETURN-LINE FEED WILL BE TYPED.
15176                                   ;*THE COMPLETE NUMBER MUST BE RETYPED. THE INPUT IS TERMINATED BY THE
15177                                   ;*USER TYPING A CARRIAGE RETURN. THE RANGE OF THE INPUT NUMBER IS
```

```
15178                                   ;*POSITIVE 32767 TO NEGATIVE 32768.
15179                                   ;*CALL:
15180                                   ;*      RDDEC                   ;;READ A DELIMAL NUMBER
15181                                   ;*      RETURN HERE             ;;NUMBER IS ON TOP OF THE STACK
15182                                   ;
15183
15184 062330  011646                    $RDDEC: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR
15185 062332  016666  000004  000002            MOV     4(SP),2(SP)     ;;THE INPUT NUMBER
15186 062340                                    PUSH    R0,R1,R2
15187 062346  104412                    1$:     RDLIN                   ;;READ AN ASCIZ LINE
15188 062350  012600                            MOV     (SP)+,R0        ;;ADDRESS OF 1ST CHAR.
15189 062352  010037  062476                    MOV     R0,6$           ;;SAVE INCASE OF BAD INPUT
15190 062356  005046                            CLR     -(SP)           ;;CLEAR DATA WORD
15191 062360  005002                            CLR     R2              ;;SIGN SET POSITIVE
15192 062362  122710  000055                    CMPB    #'-,(R0)        ;;SEE IF A MINUS SIGN WAS TYPED
15193 062366  001001                            BNE     2$              ;;BR IF NO MINUS SIGN
15194 062370  112002                            MOVB    (R0)+,R2        ;;SAVE FOR LATER USE
15195 062372  112001                    2$:     MOVB    (R0)+,R1        ;;PICKUP THIS CHARACTER
15196 062374  001424                            BEQ     3$              ;;GET OUT IF ZERO
15197 062376  122701  000060                    CMPB    #'0,R1          ;;MAKE SURE THIS CHARACTER
15198 062402  003032                            BGT     5$              ;;IS A DIGIT BETWEEN 0 & 9
15199 062404  122701  000071                    CMPB    #'9,R1
15200 062410  002427                            BLT     5$
15201 062412  032716  170000                    BIT     #^C7777,(SP)    ;;DON'T LET NUMBER GET TO BIG
15202 062416  001024                            BNE     5$              ;;BR IF NUMBER WOULD OVERFLOW
15203 062420  006316                            ASL     (SP)            ;;*2
15204 062422  011646                            MOV     (SP),-(SP)      ;;SAVE FOR LATER
15205 062424  006316                            ASL     (SP)            ;;*4
15206 062426  006316                            ASL     (SP)            ;;*8.
15207 062430  062616                            ADD     (SP)+,(SP)      ;;*10.
15208 062432  102416                            BVS     5$              ;;OVERFLOW ISN'T ALLOWED
15209 062434  162701  000060                    SUB     #'0,R1          ;;STRIP AWAY THE ASCII JUNK
15210 062440  060116                            ADD     R1,(SP)         ;;ADD IN THIS DIGIT
15211 062442  102412                            BVS     5$              ;;OVERFLOW ISN'T ALLOWED
15212 062444  000752                            BR      2$              ;;LOOP
15213 062446  005702                    3$:     TST     R2              ;;CHECK IF NUMBER IS NEG
15214 062450  001401                            BEQ     4$              ;;BR IF NO
15215 062452  005416                            NEG     (SP)            ;;YES--NEGATE THE NUMBER
15216 062454  012666  000012            4$:     MOV     (SP)+,12(SP)    ;;SAVE THE RESULT
15217 062460                                    POP     R2,R1,R0
15218 062466  000002                            RTI                     ;;RETURN
15219
15220 062470  005726                    5$:     TST     (SP)+           ;;CLEAN PARTIAL NUMBER FROM STACK
15221 062472  105010                            CLRB    (R0)            ;;SET A TERMINATOR
15222 062474                                    TYPE                    ;;TYPE THE INPUT UP TO BAD CHAR.
15223 062476  000000                    6$:     .WORD   0               ;;POINTER GOES HERE
15224 062500                                    TYPE    MSG062          ;INPUT MUSST BE A
15225 062504                                    TYPE    MSG065          ;DECIMAL
15226 062510                                    TYPE    MSG064          ;NUMBER
15227 062514  000714                            BR      1$              ;;TRY AGAIN
```

```
15229                                   .SBTTL  ROUTINE SAVE AND RESTORE R0-R5
15230
15231                                   ;*******************************************************************************
15232                                   ;*SAVE R0-R5
15233                                   ;*CALL:
15234                                   ;*      SAVREG
15235                                   ;*UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
15236                                   ;*
15237                                   ;*TOP---(+16)
15238                                   ;* +2---(+18)
15239                                   ;* +4---R5
15240                                   ;* +6---R4
15241                                   ;* +8---R3
15242                                   ;*+10---R2
15243                                   ;*+12---R1
15244                                   ;*+14---R0
15245
15246 062516                            $SAVREG:
15247 062516                                    PUSH    R0,R1,R2,R3,R4,R5
15248 062532  016646  000022                    MOV     22(SP),-(SP)    ;;SAVE PS OF MAIN FLOW
15249 062536  016646  000022                    MOV     22(SP),-(SP)    ;;SAVE PC OF MAIN FLOW
15250 062542  016646  000022                    MOV     22(SP),-(SP)    ;;SAVE PS OF CALL
15251 062546  016646  000022                    MOV     22(SP),-(SP)    ;;SAVE PC OF CALL
15252 062552  000002                            RTI
15253
15254                                   ;*RESTORE R0-R5
15255                                   ;*CALL:
15256                                   ;*      RESREG
15257 062554                            $RESREG:
15258 062554  012666  000022                    MOV     (SP)+,22(SP)    ;;RESTORE PC OF CALL
15259 062560  012666  000022                    MOV     (SP)+,22(SP)    ;;RESTORE PS OF CALL
15260 062564  012666  000022                    MOV     (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
15261 062570  012666  000022                    MOV     (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
15262 062574                                    POP     R5,R4,R3,R2,R1,R0
15263 062610  000002                            RTI
```

```
                                        .SBTTL  ROUTINE RANDOM NUMBER GENERATOR

                                ;*******************************************************************************
                                ;*THIS ROUTINE IS A DOUBLE PRECISION PSEUDO RANDOM NUMBER GENERATOR
                                ;*WITH A RANGE OF 0 TO 2**(+33)-1.
                                ;*CALL:
                                ;*      CALL    $RAND           ;;CALL THE ROUTINE
                                ;*      RETURN                  ;;RETURN HERE THE RANDOM
                                ;*                              ;;NUMBER WILL BE IN
                                ;*                              ;;$HINUM,$LONUM

062612                          $RAND.  PUSH    R0,R1,R2
062620  013700  002544                  MOV     SEEDLO,R0       ;SET R0 WITH LOW
062624  013701  002542                  MOV     SEEDHI,R1       ;SET R1 WITH HIGH
062630  012702  000007                  MOV     #7,R2           ;SET SHIFT COUNT
062634  006300                  1$:     ASL     R0              ;;SHIFT R0 LEFT AND
062636  006101                          ROL     R1              ;;ROTATE CARRY INTO R1 AND
062640  077203                          SOB     R2,1$
062642  063700  002544                  ADD     SEEDLO,R0       ;ADD NUMBER TO MAKE X 129
062646  005501                          ADC     R1              ;;PROPOGATE CARRY
062650  063701  002542                  ADD     SEEDHI,R1       ;ADD NUMBER TO MAKE X 129
062654  062700  001057                  ADD     #1057,R0        ;;ADD LOW CONSTANT
062660  005501                          ADC     R1              ;;PROPOGATE CARRY
062662  062701  047401                  ADD     #47401,R1       ;;ADD HIGH CONSTANT
062666  010037  002544                  MOV     R0,SEEDLO       ;SAVE R0
062672  010137  002542                  MOV     R1,SEEDHI       ;SAVE R1
062676                                  POP     R2,R1,R0
062704  000207                          RETURN
```

```
15295                                   .SBTTL  ROUTINE DOUBLE LENGTH BINARY TO OCTAL ASCII CONVERT
15296                                   ;*************************************************************************
15297                                   ;*THIS ROUTINE WILL CONVERT A 32-BIT UNSIGNED BINARY NUMBER TO AN
15298                                   ;*UNSIGNED OCTAL ASCIZ NUMBER.
15299                                   ;*CALL
15300                                   ;*      MOV     #PNTR,-(SP)     ;;POINTER TO LOW WORD OF BINARY NUMBER
15301                                   ;*      CALL    $DB20           ;;CALL THE ROUTINE
15302                                   ;*      RETURN                  ;;THE ADDRESS OF THE FIRST ASCIZ CHAR. IS ON THE STACK
15303
15304
15305 062706  104415                   $DB20:  SAVREG                  ;;SAVE ALL REGISTERS
15306 062710  016601  000002                   MOV     2(SP),R1        ;;PICKUP THE POINTER TO LOW WORD
15307 062714  012705  063025                   MOV     #$OCTVL+13.,R5  ;;POINTER TO DATA TABLE
15308 062720  012704  000014                   MOV     #12.,R4         ;;DO ELEVEN CHARACTERS
15309 062724  012703  177770                   MOV     #^C7,R3         ;;MASK
15310 062730  012100                          MOV     (R1)+,R0        ;;LOWER WORD
15311 062732  012101                          MOV     (R1)+,R1        ;;HIGH WORD
15312 062734  005002                          CLR     R2              ;;TERMINATOR
15313 062736  110245                   1$:    MOVB    R2,-(R5)        ;;PUT CHARACTER IN DATA TABLE
15314 062740  010002                          MOV     R0,R2           ;;GET THIS DIGIT
15315 062742  005304                          DEC     R4              ;;COUNT THIS CHARACTER
15316 062744  003007                          BGT     3$              ;;BR IF NOT THE LAST DIGIT
15317 062746  001405                          BEQ     2$              ;;BR IF IT IS THE LAST DIGIT
15318 062750  005205                          INC     R5              ;;ALL DIGITS DONE-ADJUST POINTER FOR FIRST
15319 062752  010566  000002                   MOV     R5,2(SP)        ;;ASCIZ CHAR. & PUT IT ON THE STACK
15320 062756  104416                          RESREG                  ;;RESTORE ALL REGISTERS
15321 062760  000207                          RETURN                  ;;RETURN TO USER
15322 062762  006203                   2$:    ASR     R3              ;;POSITION THE MASK FOR THE LAST DIGIT
15323 062764  006001                   3$:    ROR     R1              ;;POSITION THE BINARY NUMBER FOR
15324 062766  006000                          ROR     R0              ;;      THE NEXT OCTAL DIGIT
15325 062770  006001                          ROR     R1
15326 062772  006000                          ROR     R0
15327 062774  006001                          ROR     R1
15328 062776  006000                          ROR     R0
15329 063000  040302                          BIC     R3,R2           ;;MASK OUT ALL JUNK
15330 063002  062702  000060                   ADD     #'0,R2          ;;MAKE THIS CHAR. ASCII
15331 063006  000753                          BR      1$              ;;GO PUT IT IN THE DATA TABLE
15332 063010  000016                   $OCTVL: .REPT   14.             ;;RESERVE DATA TABLE
15335         063014                   $OCT8=$OCTVL+4                  ;POINTER TO 11 DIGIT NUMBER
```

```
                                        .SBTTL  TABLES

                                        .SBTTL  APT MAILBOX-ETABLE
063026                          $MAIL:
063026  000000                  $MSGTY: .WORD   0       ;;MESSAGE TYPE CODE
063030  000000                  $FATAL: .WORD   0       ;;FATAL ERROR NUMBER (ERROR PC)
063032  000000                  $TESTN: .WORD   0       ;;TEST PATTERN NUMBER
063034  000000                  $PASS:  .WORD   0       ;;PASS COUNT
063036  000000                  $DEVCT: .WORD   0       ;;DEVICE COUNT
063040  000000                  $UNIT:  .WORD   0       ;;I/O UNIT NUMBER
063042  000000                  $MSGAD: .WORD   0       ;;MESSAGE ADDRESS
063044  000000                  $MSGLG: .WORD   0       ;;MESSAGE LENGTH
063046                          $ETABLE:                ;;APT ENVIRONMENT TABLE
063046      000                 $ENV:   .BYTE   0       ;;ENVIRONMENT BYTE      ;SET TO A 1 FOR APT AUTO MODE
                                ;NOTE:  IF BIT #7 IS SET IN $ENVM THE TABLE BELOW (BEGINNING AT $MAMS1 AND
                                ;       ENDING AT $MADR4) MUST BE FILLED IN TO INDICATE THE PROPER AMOUNT OF
                                ;       EACH TYPE OF MEMORY.
063047      000                 $ENVM:  .BYTE   0       ;ENVIRONMENT MODE
                                                        ;BIT7(200)=USE APT SIZE INFO ;BIT5(40)=NO CONSOLE
063050  000101                 $SWREG: .WORD   101     ;;APT SWITCH REGISTER
063052  000000                 $USWR:  .WORD   0       ;USED TO LIMIT THE NUMBER OF PASSES
063054  000000                 $CPUOP: .WORD   0       ;;CPU TYPE,OPTIONS
                                ;*                      BITS 15-11=CPU TYPE
                                ;*                              11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
                                ;*                              11/70=06,PDQ=07,Q=10
                                ;*                      BIT 10=REAL TIME CLOCK
                                ;*                      BIT  9=FLOATING POINT PROCESSOR
                                ;*                      BIT  8=MEMORY MANAGEMENT
063056      001                 $MAMS1: .BYTE   1       ;;HIGH ADDRESS,M.S. BYTE ;DEFAULT = 64K
063057      004                 $MTYP1: .BYTE   4       ;;MEM. TYPE,BLK#1
                                ;*                      MEM.TYPE BYTE   --  (HIGH BYTE)
                                ;*                              900 NSEC CORE=001
                                ;*                              300 NSEC BIPOLAR=002
                                ;*                              PARITY MOS=003
                                ;*                              ERROR CORRECTING MOS=004
063060  177776                 $MADR1: .WORD   177776  ;;HIGH ADDRESS,BLK#1
                                ;*                        MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF ''TYPE'' ABOVE
063062      000                 $MAMS2: .BYTE   0       ;;HIGH ADDRESS,M.S. BYTE
063063      000                 $MTYP2: .BYTE   0       ;;MEM.TYPE,BLK#2
063064  000000                  $MADR2: .WORD   0       ;;MEM.LAST ADDRESS,BLK#2
063066      000                 $MAMS3: .BYTE   0       ;;HIGH ADDRESS,M.S.BYTE
063067      000                 $MTYP3: .BYTE   0       ;;MEM.TYPE,BLK#3
063070  000000                  $MADR3: .WORD   0       ;;MEM.LAST ADDRESS,BLK#3
063072      000                 $MAMS4: .BYTE   0       ;;HIGH ADDRESS,M.S.BYTE
063073      000                 $MTYP4: .BYTE   0       ;;MEM.TYPE,BLK#4
063074  000000                  $MADR4: .WORD   0       ;;MEM.LAST ADDRESS,BLK#4
063076  000000                  $VECT1: .WORD   0       ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
063100  000000                  $VECT2: .WORD   0       ;;INTERRUPT VECTOR#2BUS PRIORITY#2
063102  000000                  $BASE:  .WORD   0       ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
063104  000000                  $DEVM:  .WORD   0       ;;DEVICE MAP

063106  000000                  $CDW1:  .WORD   0
063110  000000                  $CDW2:  .WORD   0
```

```
 15391                                  ;THE FOLLOWING LOCATIONS SPECIFY WHICH PATTERNS
 15392                                  ;ARE TO BE RUN FOR PARTICULAR MEMORIES
 15393                                  ;
 15394                                  ;REFERENCE THE TABLE LISTED BELOW TO RELATE BITS TO PATTERNS.
 15395                                  ;BIT0 SET WILL RUN THE FIRST ENTRY IN THE TABLE, BIT0 SET
 15396                                  ;IN THE SECOND WORD WILL RUN THE 17TH ENTRY IN THE TABLE ...
 15397                                  ;
 15398                                  ;NOTE** NULL TESTS DO NOT TAKE ANY TIME
 15399                                  ;                                        ;FIELD SERVICE VALUE
 15400 063112  177777                   $DDW0:  .WORD   177777  ;ECC CSR TESTS          177777          TABLE = MKCSRT:
 15401 063114  177777                   $DDW1:  .WORD   177777  ;ECC CSR TESTS          177777          TABLE = MKCSRT:
 15402 063116  177777                   $DDW2:  .WORD   177777  ;ECC PATTERNS          103777          TABLE = MKPAT:
 15403 063120  177777                   $DDW3:  .WORD   177777  ;ECC PATTERNS          177777          TABLE = MKPAT:
 15404 063122  177777                   $DDW4:  .WORD   177777  ;PARITY PATTERNS       003777          TABLE = MJPAT:
 15405 063124  177777                   $DDW5:  .WORD   177777  ;PARITY PATTERNS       177774          TABLE = MJPAT:
 15409 063126                           $ETEND:
 15410                                  ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
 15411                                  ;INTERFACE SPEC.
 15412
 15413 063126                           $APTHD:
 15414 063126  000000                   $HIBTS: .WORD   0               ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
 15415 063130  063026                   $MBADR: .WORD   $MAIL           ;;ADDRESS OF APT MAILBOX (BITS 0-15)
 15416 063132  000043                   $TSTM:  .WORD   35.             ;;RUN TIM OF LONGEST TEST
 15417 063134  001274                   $PASTM: .WORD   700.            ;;RUN TIME IN SECS. OF 1ST PASS ON 128K (QUICK VERIFY)
 15418 063136  000000                   $UNITM: .WORD   0.              ;;EXTRA RUN TIME OF A PASS FOR EACH ADDITIONAL 128K (QV)
 15419 063140  000040                           .WORD   $ETEND-$MAIL/2  ;;LENGTH MAILBOX-ETABLE(WORDS)
```

```
15421                                   .SBTTL  ROUTINE TRAP DECODER
15422
15423                                   ;*************************************************************************************
15424                                   ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
15425                                   ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
15426                                   ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
15427                                   ;*GO TO THAT ROUTINE.
15428
15429 063142  010046                   $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
15430 063144  016600  000002                   MOV     2(SP),R0        ;;GET TRAP ADDRESS
15431 063150  005740                           TST     -(R0)           ;;BACKUP BY 2
15432 063152  111000                           MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
15433 063154  006300                           ASL     R0              ;;POSITION FOR INDEXING
15434 063156  016000  063204                   MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
15435 063162  000200                           RTS     R0              ;;GO TO POUTINE
15436
15437
15438                                   ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
15439
15440 063164  011646                   $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
15441 063166  016666  000004  000002           MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
15442 063174  000002                           RTI                     ;;RESTORE THE PSW
15443
15444 063176                           $NOTRAP:TYPE    MSG006          ;UNDEFINED TRAP INSTRUCTION
15445 063202  000000                   $HALT2: HALT
```

```
                                                        .SBTTL  TRAP TABLE

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

                                                ;       ROUTINE
                                                ;       -------
063204  063164                                  $TRPAD: .WORD   $TRAP2
063206  053360                                          $TYPE   ;CALL=TYPEIT    TRAP+1(104401)  TTY TYPEOUT ROUTINE
063210  060276                                          $TYPOC  ;CALL=TYPOC     TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
063212  060252                                          $TYPOS  ;CALL=TYPOS     TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
063214  063176                                          $NOTRAP;$TYPON  ;CALL=TYPON     TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
063216  060500                                          $TYPDS  ;CALL=TYPDS     TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
063220  063176                                          $NOTRAP;$TYPBN  ;CALL=TYPBN     TRAP+6(104406)  TYPE BINARY (ASCII) NUMBER

063222  061100                                          $GTSWR  ;CALL=GTSWR     TRAP+7(104407)  GET SOFT-SWR SETTING
063224  060724                                          $CKSWR  ;CALL=CKSWR     TRAP+10(104410) TEST FOR CHANGE IN SOFT-SWR

063226  061444                                          $RDCHR  ;CALL=RDCHR     TRAP+11(104411) TTY TYPEIN CHARACTER ROUTINE
063230  061574                                          $RDLIN  ;CALL=RDLIN     TRAP+12(104412) TTY TYPEIN STRING ROUTINE
063232  062160                                          $RDOCT  ;CALL=RDOCT     TRAP+13(104413) READ AN OCTAL NUMBER FROM TTY
063234  062330                                          $RDDEC  ;CALL=RDDEC     TRAP+14(104414) READ A DECIMAL NUMBER FROM TTY

063236  062516                                          $SAVREG ;CALL=SAVREG    TRAP+15(104415) SAVE R0-R5 ROUTINE
063240  062554                                          $RESREG ;CALL=RESREG    TRAP+16(104406) RESTORE R0-R5 ROUTINE

063242  040216                                          $KERNEL  ;CALL=KERNEL   TRAP+17(104417) ENTER KERNEL MODE
063244  040226                                          $ENERGIZE;CALL=ENERGIZETRAP+20(104420) TURN ON MEMORY MANAGEMENT & TRAPS
063246  040236                                          $DEENERGI;CALL=DEENERGITRAP+21(104421) TURN OFF MEMORY MENAGEMENT & TRAPS

063250  042434                                          $KMAP    ;CALL=KMAP     TRAP+22(104422) MAP KERNEL 1 TO 1

063252  040246                                          $CACHN  ;CALL=CACHON    TRAP+23(104423) TURN CACHE ON
063254  040272                                          $CACHF  ;CALL=CACHOFF   TRAP+24(104424) TURN CACHE OFF

063256  040310                                          $LOADC  ;CALL=LOADCSR   TRAP+25(104425) LOAD CORRECT CSR
063260  040404                                          $READC  ;CALL=READCSR   TRAP+26(104426) READ CORRECT CSR

063262  053714                                          $PER01  ;CALL=PERR01    TRAP+27(104427) PROGRAM DETECTED ERROR
063264  053742                                          $PER02  ;CALL=PERR02    TRAP+30(104430) PROGRAM DETECTED ERROR
063266  053770                                          $PER03  ;CALL=PERR03    TRAP+31(104431) PROGRAM DETECTED ERROR
063270  054020                                          $PER04  ;CALL=PERR04    TRAP+32(104432) PROGRAM DETECTED ERROR
063272  054102                                          $PER07  ;CALL=PERR07    TRAP+33(104433) PROGRAM DETECTED ERROR
063274  054124                                          $PER10  ;CALL=PERR10    TRAP+34(104434) PROGRAM DETECTED ERROR
063276  054154                                          $PER11  ;CALL=PERR11    TRAP+35(104435) PROGRAM DETECTED ERROR
063300  054174                                          $PER12  ;CALL=PERR12    TRAP+36(104436) PROGRAM DETECTED ERROR
063302  054216                                          $PER13  ;CALL=PERR13    TRAP+37(104437) PROGRAM DETECTED ERROR
063304  054236                                          $PER14  ;CALL=PERR14    TRAP+40(104440) PROGRAM DETECTED ERROR
063306  054260                                          $PER15  ;CALL=PERR15    TRAP+41(104441) PROGRAM DETECTED ERROR
063310  054302                                          $PER16  ;CALL=PERR16    TRAP+42(104442) PROGRAM DETECTED ERROR
063312  054322                                          $PER17  ;CALL=PERR17    TRAP+43(104443) PROGRAM DETECTED ERROR
063314  054340                                          $PER20  ;CALL=PERR20    TRAP+44(104444) PROGRAM DETECTED ERROR
063316  054356                                          $PER21  ;CALL=PERR21    TRAP+45(104445) PROGRAM DETECTED ERROR
063320  054376                                          $PER22  ;CALL=PERR22    TRAP+46(104446) PROGRAM DETECTED ERROR
063322  054414                                          $PER23  ;CALL=PERR23    TRAP+47(104447) PROGRAM DETECTED ERROR
063324  054432                                          $PER24  ;CALL=PERR24    TRAP+50(104450) PROGRAM DETECTED ERROR
063326  051170                                          $PER25  ;CALL=PERR25    TRAP+51(104451) PROGRAM DETECTED ERROR
```

TRAP TABLE

```
 15505 063330  054622                $PER26  ;CALL=PERR26    TRAP+52(104452) PROGRAM DETECTED ERROR
 15506 063332  054642                $PER27  ;CALL=PERR27    TRAP+53(104453) PROGRAM DETECTED ERROR
 15507 063334  051416                $PER30  ;CALL=PERR30    TRAP+54(104454) PROGRAM DETECTED ERROR
 15508 063336  055032                $PER31  ;CALL=PERR31    TRAP+55(104455) PROGRAM DETECTED ERROR
 15509 063340  055130                $PER32  ;CALL=PERR32    TRAP+56(104456) PROGRAM DETECTED ERROR
 15510 063342  055176                $PER33  ;CALL=PERR33    TRAP+57(104457) PROGRAM DETECTED ERROR
 15511 063344  055256                $PER34  ;CALL=PERR34    TRAP+60(104460) PROGRAM DETECTED ERROR
 15512 063346  055310                $PER35  ;CALL=PERR35    TRAP+61(104461) PROGRAM DETECTED ERROR
 15513 063350  055344                $PER36  ;CALL=PERR36    TRAP+62(104462) PROGRAM DETECTED ERROR
 15514 063352  063176                $NOTRAP ;CALL=PERR37    TRAP+63(104463) PROGRAM DETECTED ERROR
 15515 063354  063176                $NOTRAP ;CALL=PERR40    TRAP+64(104464) PROGRAM DETECTED ERROR
 15516 063356  063176                $NOTRAP ;CALL=PERR41    TRAP+65(104465) PROGRAM DETECTED ERROR
 15517 063360  063176                $NOTRAP ;CALL=PERR42    TRAP+66(104466) PROGRAM DETECTED ERROR
 15518 063362  063176                $NOTRAP ;CALL=PERR43    TRAP+67(104467) PROGRAM DETECTED ERROR
 15519
 15520 063364  040630                $ECCDIS ;CALL=ECCDIS    TRAP+70(104470) DISABLE ECC ON ALL CSR'S
 15521 063366  040644                $ECC1DIS;CALL=ECC1DIS   TRAP+71(104471) DISABLE ECC ON 1 SELECTED CSR
 15522 063370  040656                $ECCINIT;CALL=ECCINIT   TRAP+72(104472) INITIALIZE ALL MK11 CSR'S
 15523 063372  040672                $ECC1INIT;CALL=ECC1INIT TRAP+73(104473) INITIALIZE 1 SELECTED MK11 CSR
 15524 063374  040732                $CBCSR  ;CALL=CBCSR     TRAP+74(104474) WRITE GENERATED CHECKBITS IN ALL CSR'S
 15525 063376  040754                $CB1CSR ;CALL=CB1CSR    TRAP+75(104475) WRITE GENERATED CHECKBITS IN 1 SELECTED CSR
 15526 063400  040774                $WASSBE ;CALL=WASSBE    TRAP+76(104476) WAS THERE A SBE ON ANY CSR?
 15527 063402  041110                $WAS1SBE;CALL=WAS1SBE   TRAP+77(104477) WAS THERE A SBE ON 1 SELECTED CSR?
 15528 063404  041140                $WASDBE ;CALL=WASDBE    TRAP+100(104500)WAS THERE A DBE ON ANY CSR?
 15529 063406  041254                $WAS1DBE;CALL=WAS1DBE   TRAP+101(104501)WAS THERE A DBE ON 1 SELECTED CSR?
 15530 063410  041304                $CLRCSR ;CALL=CLRCSR    TRAP+102(104502)CLEAR ALL CSR'S
 15531 063412  041316                $CLR1CSR;CALL=CLR1CSR   TRAP+103(104503)CLEAR 1 SELECTED CSR
 15532 063414  041326                $CHKDIS ;CALL=CHKDIS    TRAP+104(104504)DISABLE ECC & WRITE CKBITS FROM ALL CSR'S
 15533 063416  041342                $CHK1DIS;CALL=CHK1DIS   TRAP+105(104505)DISABLE ECC & WRITE CKBITS FROM 1 CSR
 15534 063420  040704                $ENASBE ;CALL=ENASBE    TRAP+106(104506)ENABLE TRAPS ON SBE'S FROM ALL CSR'S
 15535 063422  040720                $ENA1SBE;CALL=ENA1SBE   TRAP+107(104507)ENABLE TRAPS ON SBE'S FROM 1 SELECTED CSR
 15536 063424  040424                $TSTRD  ;CALL=TSTREAD   TRAP+110(104510)TEST LOC (R1) & TST FOR SBE (WITHOUT FETCHES
 15537 063426  041422                $INVALID;CALL=INVALID   TRAP+111(104511)INVALIDATE BACKGROUND PATTERN ON BANK
 15538 063430  041452                $ERRGEN ;CALL=ERRGEN    TRAP+114(104512)TEST ERROR ADDRESS
 15539 063432  063176                $NOTRAP
 1554C 063434  063176                $NOTRAP
 15541 063436  063176                $NOTRAP
 15542 063440  063176                $NOTRAP
 15543 063442  063176                $NOTRAP
 15544 063444  063176                $NOTRAP
 15545 063446  063176                $NOTRAP
```

```
 15548          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


063450                          $ERRTB: ;ERROR  1
063450  065741                          EM24
063452  070066                          DH13
063454  064320                          DT13
063456  064656                          DF11
                                        ;ERROR  2
063460  064715                          EM2
063462  067375                          DH1
063464  064150                          DT1
063466  064534                          DF2
                                        ;ERROR  3
063470  064753                          EM3
063472  067455                          DH3
063474  064162                          DT3
063476  064651                          DF9
                                        ;ERROR  4
063500  065005                          EM4
063502  067455                          DH3
063504  064172                          DT4
063506  064651                          DF9
                                        ;ERROR  5
063510  065053                          EM5
063512  067511                          DH5
063514  064202                          DT5
063516  064534                          DF2
                                        ;ERROR  6
063520  065130                          EM6
063522  067511                          DH5
063524  064202                          DT5
063526  064534                          DF2
                                        ;ERROR  7
063530  065155                          EM7
063532  067511                          DH5
063534  064202                          DT5
063536  064534                          DF2
                                        ;ERROR  10
063540  067273                          EM53
063542  070531                          DH25
063544  064476                          DT25
063546  064534                          DF2
```

```
 15607                                  ;ERROR  11
 15608 063550  065215                   EM11
 15609 063552  067635                   DH7
 15610 063554  064234                   DT7
 15611 063556  064560                   DF3
 15612                                  ;ERROR  12
 15613 063560  065215                   EM11
 15614 063562  067635                   DH7
 15615 063564  064234                   DT7
 15616 063566  064573                   DF4
 15617                                  ;ERROR  13
 15618 063570  065237                   EM12
 15619 063572  067745                   DH10
 15620 063574  064264                   DT10
 15621 063576  064534                   DF2
 15622                                  ;ERROR  14
 15623 063600  065215                   EM11
 15624 063602  067635                   DH7
 15625 063604  064234                   DT7
 15626 063606  064606                   DF5
 15627                                  ;ERROR  15
 15628 063610  065215                   EM11
 15629 063612  067635                   DH7
 15630 063614  064234                   DT7
 15631 063616  064621                   DF6
 15632                                  ;ERROR  16
 15633 063620  065263                   EM13
 15634 063622  070066                   DH13
 15635 063624  064320                   DT13
 15636 063626  064656                   DF11
 15637                                  ;ERROR  17
 15638 063630  065315                   EM14
 15639 063632  070066                   DH13
 15640 063634  064320                   DT13
 15641 063636  064656                   DF11
 15642                                  ;ERROR  20
 15643 063640  065361                   EM15
 15644 063642  070066                   DH13
 15645 063644  064320                   DT13
 15646 063646  064656                   DF11
 15647                                  ;ERROR  21
 15648 063650  067322                   EM55
 15649 063652  070555                   DH26
 15650 063654  064506                   DT26
 15651 063656  064534                   DF2
 15652                                  ;ERROR  22
 15653 063660  065427                   EM17
 15654 063662  067635                   DH7
 15655 063664  064234                   DT7
 15656 063666  064606                   DF5
 15657                                  ;ERROR  23
 15658 063670  067130                   EM50
 15659 063672  070403                   DH23
 15660 063674  064434                   DT23
 15661 063676  064667                   DF13
```

```
 15664                                  ;ERROR  24
 15665 063700  065467                   EM19
 15666 063702  070066                   DH13
 15667 063704  064320                   DT13
 15668 063706  064656                   DF11
 15669                                  ;ERROR  25
 15670 063710  065544                   EM20
 15671 063712  070066                   DH13
 15672 063714  064320                   DT13
 15673 063716  064656                   DF11
 15674                                  ;ERROR  26
 15675 063720  000000                   0                       ;NO MESSAGE
 15676 063722  070061                   DH12
 15677 063724  064314                   DT12
 15678 063726  064534                   DF2
 15679                                  ;ERROR  27
 15680 063730  065626                   EM21
 15681 063732  070043                   DH11
 15682 063734  064306                   DT11
 15683 063736  064534                   DF2
 15684                                  ;ERROR  30
 15685 063740  065665                   EM22
 15686 063742  070066                   DH13
 15687 063744  064320                   DT13
 15688 063746  064656                   DF11
 15689                                  ;ERROR  31
 15690 063750  000000                   0                       ;NO MESSAGE
 15691 063752  070163                   DH14
 15692 063754  064342                   DT14
 15693 063756  064534                   DF2
 15694                                  ;ERROR  32
 15695 063760  065712                   EM23
 15696 063762  067511                   DH5
 15697 063764  064202                   DT5
 15698 063766  064534                   DF2
 15706                                  ;ERROR  33
 15707 063770  066020                   EM25
 15708 063772  070242                   DH15
 15709 063774  064360                   DT16
 15710 063776  064634                   DF7
 15711                                  ;ERROR  34
 15712 064000  066045                   EM26
 15713 064002  070361                   DH16
 15714 064004  064410                   DT17
 15715 064006  064560                   DF3
```

```
 15725                                          ;ERROR  35
 15726 064010  067246                           EM52
 15727 064012  070531                           DH25
 15728 064014  064476                           DT25
 15729 064016  064534                           DF2
 15730                                          ;ERROR  36
 15731 064020  066116                           EM27
 15732 064022  070361                           DH16
 15733 064024  064410                           DT17
 15734 064026  064647                           DF8
 15742                                          ;ERROR  37
 15743 064030  066722                           EM35
 15744 064032  067635                           DH7
 15745 064034  064234                           DT7
 15746 064036  064560                           DF3
 15747                                          ;ERROR  40
 15748 064040  066206                           EM29
 15749 064042  067635                           DH7
 15750 064044  064234                           DT7
 15751 064046  064560                           DF3
 15752                                          ;ERROR  41
 15753 064050  066270                           EM30
 15754 064052  067635                           DH7
 15755 064054  064234                           DT7
 15756 064056  064606                           DF5
 15757                                          ;ERROR  42
 15758 064060  066407                           EM31
 15759 064062  067635                           DH7
 15760 064064  064234                           DT7
 15761 064066  064560                           DF3
 15762                                          ;ERROR  43
 15763 064070  066507                           EM32
 15764 064072  067635                           DH7
 15765 064074  064234                           DT7
 15766 064076  064560                           DF3
 15767                                          ;ERROR  44
 15768 064100  066614                           EM33
 15769 064102  067635                           DH7
 15770 064104  064234                           DT7
 15771 064106  064560                           DF3
 15772                                          ;ERROR  45
 15773 064110  067164                           EM51
 15774 064112  070462                           DH24
 15775 064114  064456                           DT24
 15776 064116  064677                           DF14
 15777                                          ;ERROR  46
 15778 064120  067007                           EM36
 15779 064122  067570                           DH6
 15780 064124  064220                           DT6
 15781 064126  064534                           DF2
```

```
 15796                                  ;ERROR  47
 15797 064130  067056                   EM40
 15798 064132  067432                   DH2
 15799 064134  064416                   DT20
 15800 064136  064534                   DF2
 15838                                  ;ERROR  50
 15839 064140  067343                   EM56
 15840 064142  070573                   DH27
 15841 064144  064514                   DT27
 15842 064146  064706                   DF15
```

ERROR DATA TAGS (DT)

```
                                                        .SBTTL  ERROR DATA TAGS (DT)
15853 064150  002016  002032  002042  DT1:    .WORD   ERRPC,ADDRESS,GOOD,BAD,0
      064156  002050  000000
15857 064162  002016  002034  002070  DT3:    .WORD   ERRPC,PADDRESS,PARCNT,0
      064170  000000
15858 064172  002016  002032  002066  DT4:    .WORD   ERRPC,ADDRESS,NEMCNT,0
      064200  000000
15859 064202  002016  177572  177574  DT5:    .WORD   ERRPC,MMR0,MMR1,MMR2,MMR3,CPUERR,0
      064210  177576  172516  177766
      064216  000000
15860 064220  002016  002372  002350  DT6:    .WORD   ERRPC,APTPAR,LSIZE,APTECC,MSIZE,0
      064226  002374  002352  000000
15861 064234  002016  002170  002032  DT7:    .WORD   ERRPC,DUMMY,ADDRESS,DUMMY,GOOD,BAD,BADXOR
      064242  002170  002042  002050
      064250  002056
15862 064252  002170  002170  002170          .WORD   DUMMY,DUMMY,DUMMY,DUMMY,0
      064260  002170  000000
15863 064264  002172  002174  002176  DT10:   .WORD   DETR0,DETR1,DETR2,DETR3,DETR4,DETR5,DETSP,DETPSW,0
      064272  002200  002202  002204
      064300  002206  002210  000000
15864 064306  002016  002144  000000  DT11:   .WORD   ERRPC,CSR,0
15865 064314  002144  000000          DT12:   .WORD   CSR,0
15866 064320  002016  002170  002032  DT13:   .WORD   ERRPC,DUMMY,ADDRESS,DUMMY,TSTDAT,TSTDAT+2,CHECK,CSR,0
      064326  002170  002240  002242
      064334  002274  002144  000000
15867 064342  177746  177572  177574  DT14:   .WORD   CONTRL,MMR0,MMR1,MMR2,MMR3,CPUERR,0
      064350  177576  172516  177766
      064356  000000
15868 064360  002016  002170  002170  DT16:   .WORD   ERRPC,DUMMY,DUMMY,GOOD,GOOD2,GOOD3
      064366  002042  002044  002046
15869 064374  002050  002052  002054          .WORD   BAD,BAD2,BAD3,DUMMY,DUMMY,0
      064402  002170  002170  000000
15870 064410  002016  002170  000000  DT17:   .WORD   ERRPC,DUMMY,0
15875 064416  002016  002042  002050  DT20:   .WORD   ERRPC,GOOD,BAD,0
      064424  000000
15879 064426  002016  002170  000000  DT22:   .WORD   ERRPC,DUMMY,0
15880 064434  002016  002170  002042  DT23:   .WORD   ERRPC,DUMMY,GOOD,BAD,DUMMY,DUMMY,DUMMY,DUMMY,0
      064442  002050  002170  002170
      064450  002170  002170  000000
15881 064456  002016  002170  002144  DT24:   .WORD   ERRPC,DUMMY,CSR,DUMMY,DUMMY,DUMMY,DUMMY,0
      064464  002170  002170  002170
      064472  002170  000000
15882 064476  002016  002042  002144  DT25:   .WORD   ERRPC,GOOD,CSR,0
      064504  000000
15883 064506  002016  002050  000000  DT26:   .WORD   ERRPC,BAD,0
15884 064514  002016  002170  002032  DT27:   .WORD   ERRPC,DUMMY,ADDRESS,DUMMY,DUMMY,DUMMY,DUMMY,0
      064522  002170  002170  002170
      064530  002170  000000
```

```
 15891                                          .SBTTL  ERROR DATA FORMATS (DF)
 15892 064534   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
       064537   000    000    000
       064542   000    000    000
       064545   000    000    000
       064550   000    000    000
       064553   000    000    0C0
       064556   000    000
 15893 064560   000    005    000   DF3:     .BYTE   0,5,0,8.,0,0,0,3,6,2,4
       064563   010    000    000
       064566   000    003    006
       064571   002    004
 15894 064573   000    005    000   DF4:     .BYTE   0,5,0,8.,0,8.,8.,3,6,2,4
       064576   010    000    010
       064601   010    003    006
       064604   002    004
 15895 064606   000    005    000   DF5:     .BYTE   0,5,0,8.,9.,9.,9.,3,6,2,4
       064611   010    011    011
       064614   011    003    006
       064617   002    004
 15896 064621   000    005    000   DF6:     .BYTE   0,5,0,8.,9.,8.,8.,3,6,2,4
       064624   010    011    010
       064627   010    003    006
       064632   002    004
 15897 064634   000    005    010   DF7:     .BYTE   0,5,8.,0,0,9.,0,0,9.,2,4
       064637   000    000    011
       064642   000    000    011
       064645   002    004
 15898 064647   000    005           DF8:     .BYTE   0,5
 15899 064651   000    001    001   DF9:     .BYTE   0,1,1,1,1
       064654   001    001
 15900 064656   000    005    000   DF11:    .BYTE   0,5,0,8.,0,0,0,0,0
       064661   010    000    000
       064664   000    000    000
 15901 064667   000    005    000   DF13:    .BYTE   0,5,0,0,3,6,2,4
       064672   000    003    006
       064675   002    004
 15902 064677   000    005    000   DF14:    .BYTE   0,5,0,3,6,2,4
       064702   003    006    002
       064705   004
 15903 064706   000    005    000   DF15:    .BYTE   0,5,0,8.,3,6,4
       064711   010    003    006
       064714   004
```

```
15909                                           .SBTTL  ERROR MESSAGES (EM)
15918 064715    103     101     116  EM2:      .ASCIZ  /CAN'T SET 22 BIT MODE IN MMR3/
15919 064753    120     101     122  EM3:      .ASCIZ  /PARITY ERROR(S) IN BANK 0/
15920 065005    116     117     116  EM4:      .ASCIZ  /NON-EXISTANT MEMORY (HOLES) IN BANK 0/
15921 065053    111     114     114  EM5:      .ASCIZ  /ILLEGAL OR RESERVED INSTRUCTION (TRAP TO 10)/
15922 065130    125     116     105  EM6:      .ASCIZ  /UNEXPECTED TRAP TO 4/
15923 065155    115     105     115  EM7:      .ASCIZ  /MEMORY MANAGEMENT (TRAP TO 250)/
15927
15928 065215    115     105     115  EM11:     .ASCIZ  /MEMORY DATA ERROR/
15929 065237    104     105     124  EM12:     .ASCIZ  /DETAILED ERROR DUMP/
15930 065263    115     111     123  EM13:     .ASCIZ  /MISSING EXPECTED SBE FLAG/
15931 065315    127     122     111  EM14:     .ASCIZ  /WRITE BYTE FAILED TO CLEAR SBE FLAG/
15932 065361    106     101     111  EM15:     .ASCIZ  /FAILED TO GET INTERRUPT WITH DBE FLAG/
15933 065427    115     105     115  EM17:     .ASCIZ  /MEMORY DATA ERROR IN CHECK BITS/
15934 065467    123     102     105  EM19:     .ASCIZ  /SBE OR DBE CAUSED PARITY TRAP WHEN INHIBITED/
15935 065544    123     102     105  EM20:     .ASCIZ  /SBE OR DBE DID NOT CAUSE PARITY TRAP WHEN ENABLED/
15936 065626    123     102     105  EM21:     .ASCIZ  /SBE OR DBE ON MASTER TEST WORD/
15937 065665    115     111     123  EM22:     .ASCIZ  /MISSING EXPECTED DBE/
15938 065712    125     116     105  EM23:     .ASCIZ  /UNEXPECTED PARITY TRAP/
15939 065741    122     105     103  EM24:     .ASCIZ  /RECEIVED DBE FLAG WHEN EXPECTING ONLY SBE FLAG/
15940 066020    103     110     105  EM25:     .ASCIZ  /CHECK BIT DATA ERROR/
15941 066045    101     104     104  EM26:     .ASCIZ  /ADDRESS PARITY ERROR DID NOT CAUSE ABORT/
15942 066116    105     103     103  EM27:     .ASCIZ  /ECC INHIBIT MODE POINTER FAILURE - DID NOT PROTECT BANK/
15946 066206    103     117     122  EM29:     .ASCIZ  /CORRECTION FAILURE WITH ECC ENABLED ON FORCED SBE/
15947 066270    127     122     111  EM30:     .ASCII  /WRITE BYTE (MOVB) WITH ECC ENABLED FAILE^ TO CLEAR DATA AT/<CRLF>
15948 066363    106     117     122               .ASCIZ  /FORCED SBE LOCATION/
15949 066407    101     123     122  EM31:     .ASCIZ  /ASRB (R3)+ WITH ECC ENABLED CHANGED DATA AT FORCED DBE LOCATION/
15950 066507    115     117     126  EM32:     .ASCIZ  /MOVB #360,(R2)+ WITH ECC ENABLED CHANGED DATA AT FORCED DBE LOCATION/
15951 066614    115     117     126  EM33:     .ASCIZ  /MOV #177400,(R1) WITH ECC ENABLED CHANGED DATA AT FORCED DBE LOCATION/
15952 066722    125     116     105  EM35:     .ASCIZ  /UNEXPECTED CORRECTION WITH ECC DISABLE ON FORCED SBE/
15953 067007    101     120     124  EM36:     .ASCIZ  /APT SIZE DISAGREES WITH PROGRAM SIZING/
15959 067056    102     122     101  EM40:     .ASCIZ  /BRANCH GOBBLE FAILED CONDITION CODES TEST/
15968 067130    102     101     104  EM50:     .ASCIZ  /BAD ERROR ADDRESS GENERATED/
15969 067164    123     102     105  EM51:     .ASCIZ  /SBE & DBE FLAGS NOT SET ON FORCED UNCORRECTED SBE/
15970 067246    102     111     124  EM52:     .ASCIZ  /BIT SET ERROR IN CSR/
15971 067273    102     111     124  EM53:     .ASCIZ  /BIT CLEAR ERROR IN CSR/
15972 067322    111     114     114  EM55:     .ASCIZ  /ILLEGAL CSR TYPE/
15973 067343    102     101     104  EM56:     .ASCIZ  /BAD PARITY TR.P GENERATED/
```

```
 15979                                          .SBTTL  ERROR DATA HEADERS (DH)
 15980 067375  040   040   120   DH1:    .ASCIZ  /  PC    DEV ADD   GOOD    BAD/
 15981 067432  040   040   120   DH2:    .ASCIZ  /  PC    GD-CC BD-CC/
 15982 067455  040   040   120   DH3:    .ASCIZ  /  PC    1ST ADD  # OF ERRORS/
 15983 067511  040   040   120   DH5:    .ASCIZ  /  PC      MMR0     MMR1     MMR2     MMR3   CPUERR/
 15984 067570  040   040   120   DH6:    .ASCIZ  /  PC    APTPAR   LSIZE  APTECC  MSIZE/
 15985 067635  040   040   120   DH7:    .ASCII  /  PC     BANK  VADD      PADD      GOOD/
 15986 067701  040   040   040           .ASCIZ  /      BAD      XOR  CSR  MTYP INT PAT/
 15987 067745  040   040   122   DH10:   .ASCIZ  /  R0       R1       R2       R3       R4       R5       SP       PSW/
 15988 070043  040   040   120   DH11:   .ASCIZ  /  PC       CSR/
 15989 070061  040   103   123   DH12:   .ASCIZ  / CSR/
 15990 070066  040   040   120   DH13:   .ASCII  /  PC     BANK  VADD      PADD     WROTE1  WROTE2/
 15991 070143  040   103   110           .ASCIZ  / CHKBITS     CSR/
 15992 070163  103   117   116   DH14:   .ASCIZ  /CONTRL    MMR0     MMR1     MMR2     MMR3   CPUERR/
 15993 070242  040   040   120   DH15:   .ASCII  /  PC     BANK     PADD   GD-WD1  GD-WD2  GD-CHK/
 15994 070317  040   102   101           .ASCIZ  / BAD-WD1 BAD-WD2  BAD-CHK INT PAT/
 15995 070361  040   040   120   DH16:   .ASCIZ  /  PC     BANK/
 16000 070376  040   040   120   DH19:   .ASCIZ  /  PC/
 16006 070403  040   040   120   DH23:   .ASCIZ  /  PC     BANK GD-ERR BAD-ERR CSR  MTYP INT  PAT/
 16007 070462  040   040   120   DH24:   .ASCIZ  /  PC     BANK  (CSR) CSR  MTYP INT  PAT/
 16008 070531  040   040   120   DH25:   .ASCIZ  /  PC   GD-DAT  (CSR)/
 16009 070555  040   040   120   DH26:   .ASCIZ  /  PC   BADCODE/
 16010 070573  040   040   120   DH27:   .ASCIZ  /  PC     BANK  VADD      PADD    CSR  MTYP PAT/
```

```
16013                                           .SBTTL  MESSAGES
16014 070647    200     040     103     MSG000: .ASCIZ  <CRLF>" CZMSDC - MS11L/M MEMORY DIAGNOSTIC"
16015 070714    200     040     040     MSG001: .ASCIZ  <CRLF>/                         MEMORY CONFIGURATION MAP/
16016 070776    200     040     040     MSG002: .ASCIZ  <CRLF>/                             16K WORD BANKS/
16017 071053    200     040     040     MSG003: .ASCII  <CRLF>/                 1       2       3/
16018 071115    040     040     040             .ASCIZ  /       4       5       6       7   /
16019 071160    200     040     040     MSG004: .ASCII  <CRLF>/         01234567012345670123456701234567/
16020 071221    060     061     062             .ASCIZ  /0123456701234567012345670123456701234567/
16021 071266    200     105     122     MSG005: .ASCIZ  <CRLF>/ERRORS  /
16022 071300    200     125     116     MSG006: .ASCIZ  <CRLF>/UNDEFINED TRAP INSTRUCTION/<32>
16023 071335    200     111     116     MSG007: .ASCIZ  <CRLF>/INTRLV  /                    ;INTERLEAVED CSR #
16024 071347    200     103     120     MSG008: .ASCIZ  <CRLF>/CPU MAP /                    ;CPU ACCESSED BANK
16025 071361    200     115     105     MSG009: .ASCIZ  <CRLF>/MEMTYPE /                    ;MEMORY TYPE
16026 071373    200     120     122     MSG010: .ASCIZ  <CRLF>/PROTECT /                    ;MEMORY PROTECTED
16027 071405    040     040     040     MSG011: .ASCIZ  /       0       1       2       3       4       5       6/
16028 071473    064     065     066     MSG012: .ASCIZ  /4567012345670123456701234567012345670123456701234567012345670123456701234567/
16029 071570    130     000             MSG013: .ASCIZ  /X/
16030 071572    040     000             MSG014: .ASCIZ  / /                 ;SPACE
16031 071574    000     000             MSG015: .BYTE   0,0                 ;FOR SINGLE ASCII CHARACTERS & TERMINATOR
16032 071576    200     103     123     MSG016: .ASCIZ  <CRLF>/CSR      /
16033 071610    040     040     040     MSG017: .ASCIZ  /       /                   ;8 SPACES
16034 071621    040     040     000     MSG018: .ASCIZ  /  /                        ;2 SPACES
16035 071624    040     040     040     MSG019: .ASCIZ  /   /                       ;3 SPACES
16036 071630    200     106     123     MSG020: .ASCIZ  <CRLF>/FS COMMAND MODE/
16037 071651    200     103     117     MSG021: .ASCII  <CRLF>/COMMANDS AVAILABLE:/
16038 071675    200     060     040             .ASCII  <CRLF>/0 = EXIT/
16039 071706    200     061     040             .ASCII  <CRLF>/1 = READ CSR/
16040 071723    200     062     040             .ASCII  <CRLF>/2 = LOAD CSR/
16041 071740    200     063     040             .ASCII  <CRLF>/3 = EXAMINE MEMORY/
16042 071763    200     064     040             .ASCII  <CRLF>/4 = MODIFY MEMORY/
16043 072005    200     065     040             .ASCII  <CRLF>/5 = SELECT BANK & PATTERN/
16044 072037    200     066     040             .ASCII  <CRLF>/6 = TYPE CONFIG MAP/
16045 072063    200     067     040             .ASCII  <CRLF>/7 = SOB-A-LONG TEST/
16046 072107    200     070     040             .ASCII  <CRLF>/8 = ERROR SUMMARY/
16047 072131    200     071     075             .ASCII  <CRLF>/9= REFRESH TEST/
16048 072151    200     061     060             .ASCII  <CRLF>/10= SET FILL COUNT/
16049 072174    200     061     061             .ASCII  <CRLF>/11= ENTER KAMIKAZE MODE/
16050 072224    200     061     062             .ASCII  <CRLF>/12= EXIT KAMIKAZE MODE/
16051 072253    200     061     063             .ASCII  <CRLF>/13= TURN CACHE OFF/
16052 072276    200     061     064             .ASCII  <CRLF>/14= TURN CACHE ON/
16057 072320    200     061     065             .ASCII  <CRLF>/15= TEST SELECTED BANKS/
16058 072350    200     061     066             .ASCII  <CRLF>/16= TEST ALL BANKS/
16059 072373    200     061     067             .ASCII  <CRLF>/17= ENABLE TRACE/
16060 072414    200     061     070             .ASCII  <CRLF>/18= DISABLE TRACE/
16061 072436    015     012     000             .BYTE   15,12,0
16062 072441    200     127     110     MSG022: .ASCIZ  <CRLF>/WHICH CSR(0-F)? /
16063 072463    200     103     123     MSG023: .ASCIZ  <CRLF>/CSR WORD? /
16064 072477    200     103     123     MSG025: .ASCIZ  <CRLF>/CSR DOES NOT EXIST/
16065 072523    200     103     117     MSG026: .ASCIZ  <CRLF>/COMMAND:/
16066 072535    200     117     114     MSG027: .ASCIZ  <CRLF>/OLD CSR WAS/
16067 072552    200     103     123     MSG028: .ASCIZ  <CRLF>/CSR IS NOW/
16068 072566    200     105     130     MSG029: .ASCIZ  <CRLF>/EXAMINE MEMORY/
16069 072606    200     102     101     MSG030: .ASCIZ  <CRLF>/BANK(0-177)? /
16070 072625    200     120     110     MSG031: .ASCIZ  <CRLF>/PHYSICAL ADDRESS(0-17757776)? /
16071 072665    200     120     101     MSG032: .ASCIZ  <CRLF>/PARITY ABORT/<32>
16072 072704    200     124     111     MSG033: .ASCIZ  <CRLF>/TIMEOUT TRAP/<32>
16073 072723    200     102     131     MSGA34: .ASCIZ  <CRLF>/BYPASSING ECC LOGIC TESTS ON BANK /
```

```
16074 072767 040 104 125 MSGB34: .ASCIZ / DUE TO LACK OF SBE FREE LOCATIONS/
16075 073032 121 126 000 MSG035: .ASCIZ /QV/
16076 073035 200 115 117 MSG036: .ASCIZ <CRLF>/MODIFY MEMORY/
16077 073054 200 117 114 MSG037: .ASCIZ <CRLF>/OLD DATA WAS /
16078 073073 200 104 101 MSG038: .ASCIZ <CRLF>/DATA IS NOW /
16079 073111 200 111 116 MSG039: .ASCIZ <CRLF>/INPUT NEW DATA? /
16080 073133 200 123 105 MSG040: .ASCIZ <CRLF>/SELECT BANK & PATTERN TEST/
16081 073167 200 102 101 MSG041: .ASCIZ <CRLF>/BANK NOT ACCESSABLE/
16082 073214 200 120 101 MSG042: .ASCIZ <CRLF>/PATTERN(0-35)? /
16083 073235 200 120 101 MSG043: .ASCIZ <CRLF>/PATTERN 0 DATA IS? /
16084 073262 200 124 117 MSG046: .ASCIZ <CRLF>/TO ESCAPE TYPE ANY KEY/<CRLF><12><12>
16085 073315 200 124 105 MSG047: .ASCIZ <CRLF>/TEST COMPLETE/
16086 073334 040 116 117 MSG048: .ASCIZ / NOT AVAILABLE NOW - TRY LATER!/
16087 073374 200 102 101 MSG049: .ASCIZ <CRLF>/BANK REQUIRES RELOCATION/
16088 073426 200 102 101 MSG050: .ASCII <CRLF>/BATTERY BACKUP TEST/
16089 073452 200 127 122         .ASCIZ <CRLF>/WRITING & CHECKING ADDRESS PATTERN AS BACKGROUND/
16090 073534 200 120 117 MSG051: .ASCIZ <CRLF>/POWER RECOVERY/
16091 073554 200 122 105 MSG052: .ASCIZ <CRLF>/REMOVE SYSTEM POWER FOR/
16092 073605 040 123 105 MSG053: .ASCIZ / SECONDS MAX!/
16093 073623 200 116 117 MSG054: .ASCIZ <CRLF>/NOW STARTING READ TEST OF MEMORY BANKS/
16094 073673 200 123 117 MSG055: .ASCIZ <CRLF>/SOB-A-LONG TEST/
16095 073714 200 102 105 MSG056: .ASCIZ <CRLF>/BELL = EACH PASS COMPLETE/
16096 073747 200 040 040 MSG058: .ASCIZ <CRLF>/  CSR    CSR .../
16097 073771 077 077 077 MSG061: .ASCIZ /??????/
16098 074000 111 116 120 MSG062: .ASCIZ /INPUT MUST BE A/
16099 074020 116 040 117 MSG063: .ASCIZ /N OCTAL /
16100 074031 116 125 115 MSG064: .ASCIZ /NUMBER/<CRLF>
16101 074041 040 104 105 MSG065: .ASCIZ / DECIMAL /
16102 074053 200 105 122 MSG066: .ASCIZ <CRLF>/ERROR COUNT EXCEEDED 20 - ABORTING FOR XXDP CHAIN/
16103 074136 106 101 124 MSG067: .ASCIZ /FATAL /
16104 074145 113 040 127 MSG070: .ASCIZ /K WORDS OF MEMORY TOTAL/<CRLF>
16105 074176 113 040 117 MSG071: .ASCIZ /K OF BIPOLAR/<CRLF>
16106 074214 113 040 117 MSG072: .ASCIZ /K OF MF11S-K/<CRLF>
16107 074232 200 122 105 MSG073: .ASCIZ <CRLF>/REFRESH TEST/
16108 074250 200 122 105 MSG075: .ASCIZ <CRLF>/RELOCATION NOT POSSIBLE/<32>
16109 074302 200 040 040 MSG076: .ASCIZ <CRLF>/  BANK  ERRORS/<CRLF>
16110 074323 200 105 116 MSG077: .ASCIZ <CRLF>/END PASS #/
16111 074337 040 105 122 MSG079: .ASCIZ / ERROR(S) DETECTED/<CRLF>
16118 074363 200 106 111 MSG085: .ASCIZ <CRLF>/FILL COUNT(OCTAL)? /
16126 074410 200 113 105 MSG088: .ASCIZ <CRLF>/KERNEL STACK/
16127 074426 200 123 125 MSG089: .ASCIZ <CRLF>/SUPERVISOR STACK/
16128 074450 200 125 123 MSG090: .ASCIZ <CRLF>/USER STACK/
16129 074464 040 111 123 MSG091: .ASCIZ / IS EMPTY/
16130 074476 122 105 114 MSG092: .ASCIZ /RELOCATED  /
16131 074512 102 101 116 MSG093: .ASCIZ /BANK=/
16132 074520 040 040 120 MSG095: .ASCIZ /  PAT=/
16140 074527 200 105 116 MSG101: .ASCIZ <CRLF>/ENTERING KAMIKAZE MODE/
16141 074557 200 114 105 MSG102: .ASCIZ <CRLF>/LEAVING KAMIKAZE MODE/
16142 074606 200 114 105 MSG103: .ASCIZ <CRLF>/LEAVING FS MODE/<CRLF>
16143 074630 032 000     MSG104: .BYTE  32,0                                ;CONTROL Z
16144 074632 200 105 116 MSG105: .ASCIZ <CRLF>/ENTER BANKS IN OCTAL - USE NUMBER OUTSIDE RANGE TO TERMINATE (177)/
16145 074736 200 103 101 MSG106: .ASCIZ <CRLF>/CACHE IS OFF/
16146 074754 200 103 101 MSG107: .ASCIZ <CRLF>/CACHE IS ON (EXCEPT DURING ACTUAL PATTERNS)/
16151 075031 200 117 116 MSG110: .ASCIZ <CRLF>/ONLY SELECTED BANKS WILL BE TESTED/
16152 075075 200 101 114 MSG111: .ASCIZ <CRLF>/ALL BANKS WILL BE TESTED/
16153 075127 113 040 117 MSG112: .ASCIZ /K OF MS11-L/<CRLF>
16154 075144 113 040 117 MSG113: .ASCIZ /K OF MS11-M/<CRLF>
```

MESSAGES

```
 16155 075161      113      040      117  MSG114: .ASCIZ  /K OF UNIBUS PARITY/<CRLF>
 16156 075205      200      040      040  MSG116: .ASCIZ  <CRLF>"   11/34"
 16157 075217      200      040      040  MSG117: .ASCIZ  <CRLF>"   11/44"
 16158 075231      200      040      040  MSG118: .ASCIZ  <CRLF>"   11/60"
 16159 075243      200      040      040  MSG119: .ASCIZ  <CRLF>/   NO/
 16160 075252      040      103      101  MSG120: .ASCIZ  / CACHE AVAILABLE/
 16161 075273      040      103      101  MSG121: .ASCIZ  / CACHE BYPASSED/
 16162 075313      200      103      123  MSG122: .ASCII  <CRLF>/CSR NUMBER /
 16163 075327      000                   MSGA122:.BYTE   0
 16164 075330      040      103      117          .ASCIZ  / CONTROLS TOO MANY BANKS/
 16165 075361      200      120      122  MSG123: .ASCIZ  <CRLF>/PROGRAM RELOCATED - ECC TESTS INHIBITED/
 16166 075432      200      116      125  MSG124: .ASCIZ  <CRLF>/NUMBER OF CSR'S IS WRONG IN BANK /
 16167 075475      040      120      101  MSG125: .ASCIZ  / PASSES COMPLETED/
 16168 075517      200      120      122  MSG126: .ASCIZ  <CRLF>/PROGRAM CSR COULD NOT BE DETERMINED/
 16169 075564      200      124      122  MSG127: .ASCIZ  <CRLF>/TRACE ENABLED/
 16170 075603      200      124      122  MSG128: .ASCIZ  <CRLF>/TRACE DISABLED/
 16171                                           .EVEN
 16177 075624                               $$END
 16178 075624                           END:
 16179 075624  004064                             .PRINT  60000-SUPLIMIT  ;SUPERVISOR ADDRESSES LEFT
 16191 075624  002154                             .PRINT  100000-END      ;ADDRESSES LEFT IN 16K
 16195         000200                             .END START3
```

SYMBOL TABLE

```
ABORTF  002140        B1      011732        CACHVE= 000114        DATBUF  002234        DT14    064342
ACFLAG  002114        B10     014620        CBCSR = 104474        DBEMSK  002250        DT16    064360
ACTFLA  002320        B11     016236        CB1CSR= 104475        DDISP = 177570        DT17    064410
ADDRES  002032        B12     016504        CHECK   002274        DEENER= 104421        DT20    064416
ANA2    007166        B13     016512        CHKDIS= 104504        DETAIL  057462        DT22    064426
APTDOW  045250        B14     016530        CHKGEN  041722        DETFLA  002212        DT23    064434
APTECC  002374        B15     016572        CHKTAB  042030        DETPSW  002210        DT24    064456
APTFLA  002322        B16     016644        CHK1DI= 104505        DETR0   002172        DT25    064476
APTHAN  014554        B17     022040        CKEND   061236        DETR1   002174        DT26    064506
APTHLT  045316        B2      012512        CKSWR = 104410        DETR2   002176        DT27    064514
APTPAR  002372        B20     022300        CLRCSR= 104502        DETR3   002200        DT3     064162
APTSIZ  002414        B21     024320        CLREX   007136        DETR4   002202        DT4     064172
BACK    055744        B22     024324        CLRMEM  007036        DETR5   002204        DT5     064202
BACKGN  036530        B23     024524        CLR1CS= 104503        DETSP   002206        DT6     064220
BAD     002050        B24     024532        CMD16A  051022        DET1    060152        DT7     064234
BADPC   002020        B25     025030        CMD16L= 000052        DF11    064656        DUMMY   002170
BADPSW  002030        B26     034450        CMD5B   047240        DF13    064667        DUMPCS= 000061
BADSP   002024        B27     034506        CMD5C   047510        DF14    064677        ECCDIS= 104470
BADSTA  040166        B3      012554        CMD7B   047740        DF15    064706        ECCINI= 104472
BADXOR  002056        B30     034522        CMD7C   050014        DF2     064534        ECC1DI= 104471
BAD2    002052        B31     034642        CMD9B   050436        DF3     064560        ECC1IN= 104473
BAD3    002054        B32     035022        CMD9C   050512        DF4     064573        EMTVEC= 000030
BAFPAF  014710        B33     035044        CONFGE  002420        DF5     064606        EM11    065215
BAFPAR  015016        B34     036240        CONFIE  003630        DF6     064621        EM12    065237
BAKPAT  002572        B35     041006        CONFIG  002624        DF7     064634        EM13    065263
BANK    002100        B36     041012        CONTFL  002214        DF8     064647        EM14    065315
BANKIN  002102        B37     041152        CONTRL= 177746        DF9     064651        EM15    065361
BANKMO  043750        B4      013356        CONTS   061406        DH1     067375        EM17    065427
BANKOK  044750        B40     041156        CONTS1  060764        DH10    067745        EM19    065467
BAWPAF  015124        B41     041362        CONTS2  061410        DH11    070043        EM2     064715
BAWPAR  015254        B42     041366        CONTS3  053576        DH12    070061        EM20    065544
BGTEST  036106        B43     042570        CONTT   061332        DH13    070066        EM21    065626
BIT0  = 000001        B44     042576        COUNT   002340        DH14    070163        EM22    065665
BIT1  = 000002        B45     042726        CPERRF  056510        DH15    070242        EM23    065712
BIT10 = 002000        B46     042742        CPSAVE  056506        DH16    070361        EM24    065741
BIT11 = 004000        B47     047744        CPUBIT  002104        DH19    070376        EM25    066020
BIT12 = 010000        B5      013520        CPUERR= 177766        DH2     067432        EM26    066045
BIT13 = 020000        B50     050166        CR    = 000015        DH23    070403        EM27    066116
BIT14 = 040000        B51     050442        CRLF  = 000200        DH24    070462        EM29    066206
BIT15 = 100000        B52     050736        CSR     002144        DH25    070531        EM3     064753
BIT2  = 000004        B53     050736        CSRADD= 172100        DH26    070555        EM30    066270
BIT3  = 000010        B54     051034        CSRCAS  017416        DH27    070573        EM31    066407
BIT4  = 000020        B55     051774        CSRFBA  002224        DH3     067455        EM32    066507
BIT5  = 000040        B56     052000        CSRFIR  002220        DH5     067511        EM33    066614
BIT6  = 000100        B57     052276        CSRHOL  002476        DH6     067570        EM35    066722
BIT7  = 000200        B6      013562        CSRINC  002300        DH7     067635        EM36    067007
BIT8  = 000400        B60     052304        CSRINF  002432        DIAGFL  002002        EM4     065005
BIT9  = 001000        B61     057716        CSRINT  002230        DISPLA  002600        EM40    067056
BLOCK1  045320        B62     057722        CSRLAS  002222        DISPRE= 000174        EM5     065053
BLOCK2  045340        B63     060020        CSRLBA  002226        DISPTB  014236        EM50    067130
BLOCK3  045354        B64     060112        CSRLOO  002302        DOBACK  014610        EM51    067164
BMFLAG  002126        B65     060176        CSRNO   002146        DSWR  = 177570        EM52    067246
BOOT    045026        B7      014046        CSROUT  041354        DT1     064150        EM53    067273
BOOT1   045072        CACHKF  002520        CSRSTU= 000014        DT10    064264        EM55    067322
BRGOBB  036110        CACHKN  002514        CTEST   006140        DT11    064306        EM56    067343
BSIZE   002344        CACHOF= 104424        CTLKVE  002142        DT12    064314        EM6     065130
B0      004554        CACHON= 104423        DATARG= 177754        DT13    064320        EM7     065155
```

SYMBOL TABLE

```
ENASBE= 104506          E5      013560          GOOD3   002046          LKS    = 177546          L15     010720
ENA1SB= 104507          E50     050270          GTSWR = 104407          LOADBA  002402          L150    021452
END     075624          E51     050504          HEADER  002552          LOADCS= 104425          L151    021502
ENERGI= 104420          E52     050776          HIPAT   045002          LOADER= 000043          L152    021512
ENEXBK  044740          E53     050776          HOLDLO= 000011          LOADHO  002536          L153    021512
ERRADD  002430          E54     051052          HT     = 000011          LOOP    014210          L154    021556
ERRGEN= 104512          E55     052034          I       002422          LOWMAP  043704          L155    021566
ERRMAX  002524          E56     052034          IBSAVE  056504          LSIZE   002350          L156    021566
ERROR = 104000          E57     052340          IIII   = 177777          LWDBE = 000037          L157    021576
ERRPC   002016          E6      013610          ILLCSR  013256          LWSBE = 000035          L16     010724
ERRPSW  002026          E60     052340          IMPTES  012222          L0      004564          L160    021634
ERRSP   002022          E61     057764          INCBNK  045012          L1      004626          L161    021644
ERRVEC= 000004          E62     057764          INCPAT  044766          L10     005124          L162    021644
EUFLAG  002130          E63     060050          INCRPT  044766          L100    014210          L163    021702
EVEN    002334          E64     060144          INHBAN  002510          L101    014324          L164    021712
EXBANK  044300          E65     060230          INHECC  002506          L102    014424          L165    021712
EXCMD3  046370          E7      014166          INTFLA  002134          L103    014434          L166    021772
EXCMD4  046710          FASTCI= 177640          INT64K  002136          L104    014604          L167    022002
EXIT    045134          FATAL$  002062          INVALI= 104511          L105    014604          L17     010726
EXIT2   045140          FCMD10  050576          IOTVEC= 000020          L106    014672          L170    022002
E0      004602          FCMD11  050624          JMPRLI  043300          L107    015004          L171    022144
E1      012150          FCMD12  050646          KAMIKA  002004          L11     005132          L172    022072
E10     014706          FCMD13  050666          KAMITE  026430          L110    015112          L173    022072
E11     016256          FCMD14  050710          KDIAG = 000010          L111    015242          L174    022124
E12     017030          FCMD15  050726          KDPAR0= 172360          L112    015372          L175    022124
E13     017014          FCMD16  051012          KDPAR6= 172374          L113    015522          L176    022202
E14     016750          FCMD17  051054          KDPAR7= 172376          L114    015674          L177    022236
E15     016750          FCMD18  051070          KERNEL= 104417          L115    016024          L2      004734
E16     016720          FIELDS  045404          KERSTK= 002000          L116    016176          L20     011540
E17     022136          FINDBA= 000045          KFLAG   002500          L117    016256          L200    022240
E2      012714          FINT    006436          KIPAR0= 172340          L12     005162          L201    022266
E20     022400          FIRST = 060000          KIPAR4= 172350          L120    016314          L202    022270
E21     024502          FLIPLO  002556          KIPAR5= 172352          L121    016324          L203    022354
E22     024466          FLIPWA  036400          KIPAR6= 172354          L122    016750          L204    022466
E23     024710          FLUSH   014330          KIPDR0= 172300          L123    016732          L205    022470
E24     024674          FSCMD0  045602          KMAP   = 104422          L124    016776          L206    023330
E25     025106          FSCMD1  045704          KPFLAG  002112          L125    017240          L207    023362
E26     035014          FSCMD2  046014          KSIZE   002346          L126    017350          L21     011554
E27     035000          FSCMD3  046162          KSTACK  002534          L127    017506          L210    023426
E3      012700          FSCMD4  046436          LAST   = 157776          L13     005414          L211    023676
E30     034614          FSCMD5  046756          LASTBA  002526          L130    017534          L212    023706
E31     034764          FSCMD6  047600          LASTBL  002530          L131    017540          L213    023706
E32     035040          FSCMD7  047606          LASTER  002014          L132    017542          L214    024452
E33     035064          FSCMD8  050100          LBLS0 = 000455          L133    017606          L215    024420
E34     036362          FSCMD9  050304          LBLS1 = 000065          L134    017642          L216    024452
E35     041060          FSINFL  002412          LBLS2 = 000447          L135    017646          L217    024516
E36     041060          FSPAT   047414          LBLS3 = 000442          L136    017650          L22     011700
E37     041224          FSSTAC  002266          LBLS4 = 000315          L137    020016          L220    024660
E4      013512          FS1     045470          LBLS5 = 000317          L14     005414          L221    024626
E40     041224          FS7FLA  002416          LBLS6 = 000016          L140    020472          L222    024660
E41     041416          FULLRE  002512          LCSROU= 000041          L141    021340          L223    025072
E42     041416          GBLENG= 000076          LCSRRE= 000057          L142    021350          L224    025072
E43     042726          GETDAT  051464          LCSRSA= 000055          L143    021350          L225    025200
E44     042706          GETDA1  051562          LEGALC= 000003          L144    021406          L226    025152
E45     043146          GETDIS  055656          LF     = 000012          L145    021416          L227    025230
E46     043054          GOOD    002042          LINK1   002472          L146    021416          L23     011656
E47     050006          GOOD2   002044          LINK2   002474          L147    021446          L230    025420
```

SYMBOL TABLE

L231	025752	L313	043044	L376	053634	L5	005052	MSG003	071053
L232	026242	L314	043044	L377	053636	L50	012626	MSG004	071160
L233	026300	L315	043044	L4	004774	L51	012636	MSG005	071266
L234	026400	L316	043024	L40	012214	L52	012656	MSG006	071300
L235	026452	L317	043044	L400	054466	L53	013502	MSG007	071335
L236	026460	L32	012016	L401	054500	L54	013456	MSG008	071347
L237	026464	L320	043112	L402	054516	L55	013432	MSG009	071361
L24	011670	L321	043112	L403	054520	L56	013424	MSG010	071373
L240	030302	L322	043142	L404	054540	L57	013430	MSG011	071405
L241	030300	L323	043262	L405	054552	L6	005132	MSG012	071473
L242	030302	L324	044442	L406	054570	L60	013454	MSG013	071570
L243	031730	L325	044666	L407	054572	L61	013450	MSG014	071572
L244	034472	L326	045050	L41	012416	L62	013454	MSG015	071574
L245	034502	L327	045154	L410	054606	L63	013502	MSG016	071576
L246	034614	L33	012134	L411	055010	L64	013502	MSG017	071610
L247	034544	L330	045160	L412	055016	L65	013502	MSG018	071621
L25	011704	L331	045166	L413	055044	L66	013576	MSG019	071624
L250	034556	L332	045202	L414	055062	L67	013646	MSG020	071630
L251	034574	L333	045214	L415	055074	L7	005132	MSG021	071651
L252	034602	L334	045426	L416	055106	L70	014020	MSG022	072441
L253	034626	L335	045436	L417	055120	L71	014210	MSG023	072463
L254	034764	L336	045572	L42	012416	L72	014156	MSG025	072477
L255	034664	L337	045632	L420	055142	L73	014074	MSG026	072523
L256	034676	L34	012066	L421	055210	L74	014076	MSG027	072535
L257	034730	L340	045636	L422	055270	L75	014136	MSG028	072552
L26	012134	L341	045666	L423	055304	L76	014152	MSG029	072566
L260	034714	L342	045700	L424	055306	L77	014206	MSG030	072606
L261	034726	L343	047066	L425	055412	MAINT =	177750	MSG031	072625
L262	034752	L344	047126	L426	055502	MAPH0 =	170202	MSG032	072665
L263	034740	L345	047164	L427	055512	MAPL0 =	170200	MSG033	072704
L264	034752	L346	047332	L43	012416	MAPL1 =	170204	MSG035	073032
L265	035064	L347	047346	L430	056020	MAPPER	042070	MSG036	073035
L266	036742	L35	012134	L431	056020	MBERR	013134	MSG037	073054
L267	037370	L350	047360	L432	056254	MEMDON	014256	MSG038	073073
L27	012134	L351	047656	L433	056214	MFPT =	000007	MSG039	073111
L270	037556	L352	047660	L434	056234	MJPAT	020042	MSG040	073133
L271	037660	L353	047772	L435	056254	MJTEST	017736	MSG041	073167
L272	040572	L354	050270	L436	056452	MKCONT	016304	MSG042	073214
L273	040600	L355	050254	L437	056322	MKCSRT	017426	MSG043	073235
L274	041036	L356	050224	L44	012422	MKFLAG	002116	MSG046	073262
L275	041036	L357	050354	L440	056450	MKLOOP	016466	MSG047	073315
L276	041100	L36	012206	L441	056400	MKPAT	017656	MSG048	073334
L277	041106	L360	050356	L442	056412	MKTEST	017516	MSG049	073374
L3	004756	L361	050470	L443	056450	MMR0 =	177572	MSG050	073426
L30	012134	L362	050760	L444	056460	MMR1 =	177574	MSG051	073534
L300	041202	L363	050762	L445	057014	MMR2 =	177576	MSG052	073554
L301	041202	L364	051226	L446	057742	MMR3 =	172516	MSG053	073605
L302	041244	L365	051242	L447	060146	MMTRAP	040142	MSG054	073623
L303	041252	L366	051246	L45	012700	MMVEC =	000250	MSG055	073673
L304	041374	L367	051264	L450	060152	MSEEDH	002546	MSG056	073714
L305	042542	L37	012212	L451	060232	MSEEDL	002550	MSG058	073747
L306	042556	L370	051412	L452	060236	MSGA12	075327	MSG061	073771
L307	042570	L371	051414	L453	061352	MSGA34	072723	MSG062	074000
L31	012012	L372	051460	L454	061442	MSGB34	072767	MSG063	074020
L310	042570	L373	051704	L455	061444	MSG000	070647	MSG064	074031
L311	042672	L374	052012	L46	012700	MSG001	070714	MSG065	074041
L312	042722	L375	052316	L47	012654	MSG002	070776	MSG066	074053

SYMBOL TABLE

```
MSG067  074136        MTPA26  035656        MT0013  021466        PATERR  002072        PFECWS  057022
MSG070  074145        MTPB03  027204        MT0014  021542        PATPLU  004546        PFLAG   002120
MSG071  074176        MTPB04  027336        MT0015  021620        PATTER  002110        PGMCSR  002502
MSG072  074214        MTPB20  034136        MT0016  021666        PCBUMP  002276        PHEBE   013136
MSG073  074232        MTPB21  034322        MT0017  021734        PCONFI  036630        PHYADD  002036
MSG075  074250        MTPB24  035156        MT0020  021756        PCONFS  037130        PROTYP  003716
MSG076  074302        MTPB25  035550        MT0021  023046        PCONF1  037040        PSIZE   002354
MSG077  074323        MTPB26  035672        MT0022  023320        PCONF2  037076        PSW    = 177776
MSG079  074337        MTPC03  027244        MT0023  023352        PDP110  040154        PWRVEC= 000024
MSG085  074363        MTPC20  034166        MT0024  023416        PD1     051704        QUICK   002406
MSG088  074410        MTPC21  034356        MT0025  023662        PERA05  054052        QVFLAG  002316
MSG089  074426        MTPC24  035172        MT0026  023730        PERBNK  054704        RANODD  035706
MSG090  074450        MTPC25  035610        MT0027  024232        PERECC  054764        RDCHR = 104411
MSG091  074464        MTPC26  035726        MT0030  024716        PERRAB  054522        RDDEC = 104414
MSG092  074476        MTPD03  027262        MT0031  025220        PERRAW  054450        RDLIN = 104412
MSG093  074512        MTPD20  034216        MT0032  025410        PERRA3  051252        RDOCT = 104413
MSG095  074520        MTPD21  034412        MT0033  025742        PERRA7  054574        READCS= 104426
MSG101  074527        MTPD25  035454        MT0034  026130        PERR01= 104427        READON  002360
MSG102  074557        MTPD26  035746        MT0035  026302        PERR02= 104430        REALPA  002260
MSG103  074606        MTPE20  034246        MT020Y  022614        PERR03= 104431        REFRES  035016
MSG104  074630        MTPE25  035476        MT020Z  022430        PERR04= 104432        REFSUB  035066
MSG105  074632        MTP000  027034        MT0999  026414        PERR05  054046        REGCOP  036370
MSG106  074736        MTP001  027060        MT1     016264        PERR06  054074        RELENT  043152
MSG107  074754        MTP002  027112        MT2     016270        PERR07= 104433        RELOCA  042526
MSG110  075031        MTP005  027356        MUT     002106        PERR10= 104434        RELOC1  043166
MSG111  075075        MTP006  027412        NC      053636        PERR11= 104435        RESREG= 104416
MSG112  075127        MTP007  027612        NEMCNT  002066        PERR12= 104436        RESTAR  002566
MSG113  075144        MTP010  027712        NEWBAN  002270        PERR13= 104437        RESVEC= 000010
MSG114  075161        MTP011  030020        NEWKER  044200        PERR14= 104440        RES0    045700
MSG116  075205        MTP012  030616        NEWLOA  044246        PERR15= 104441        RES1    045760
MSG117  075217        MTP013  031204        NOCH    060746        PERR16= 104442        RES2    046126
MSG118  075231        MTP014  031720        NOERRO  002400        PERR17= 104443        RLFLAG  002124
MSG119  075243        MTP015  032502        NOFSMO  002376        PERR20= 104444        RRFLAG  002122
MSG120  075252        MTP016  033246        NONEM   002076        PERR21= 104445        RTNVAL=%000000
MSG121  075273        MTP017  034030        NONEXI  040064        PERR22= 104446        SAVREG= 104415
MSG122  075313        MTP020  034106        NOOJ    036742        PERR23= 104447        SBEMSK  002244
MSG123  075361        MTP022  034442        NOPAR   002074        PERR24= 104450        SBENT   017370
MSG124  075432        MTP025  035210        NORES   003640        PERR25= 104451        SBETES  017112
MSG125  075475        MTP030  035764        NOSCOP  002410        PERR26= 104452        SCOPE  = 000004
MSG126  075517        MTP031  035774        NOSUPE  002426        PERR27= 104453        SDPAR0= 172260
MSG127  075564        MTP032  036052        NOTAB   002342        PERR30= 104454        SDPAR5= 172272
MSG128  075603        MTP033  036104        NOTRCE  055426        PERR31= 104455        SDPAR6= 172274
MSIZE   002352        MTP034  036202        NO22BI  002424        PERR32= 104456        SDPAR7= 172276
MTA030  024730        MTP035  036226        NULLFL  002314        PERR33= 104457        SEEDHI  002542
MTB020= 000017        MTST3   011560        OLDCAC  002262        PERR34= 104460        SEEDLO  002544
MTEST   016210        MTV020  022426        OLDCSR  002150        PERR35= 104461        SELONL  002000
MTLA11  030046        MT0000  020122        ONES    002554        PERR36= 104462        SETPAT  045002
MTLB11  030060        MT0001  020202        PADDRE  002034        PERR37= 104463        SHADL1  011610
MTLC11  030072        MT0002  020322        PAFBAF  015404        PERR40= 104464        SHUTUP  045166
MTLD11  030166        MT0003  020462        PAFBAW  015534        PERR41= 104465        SIPAR0= 172240
MTL020  022032        MT0004  020714        PARBAF  015706        PERR42= 104466        SIPAR3= 172246
MTPA03  027144        MT0005  021036        PARBAW  016036        PERR43= 104467        SIPAR5= 172252
MTPA04  027302        MT0006  021172        PARCNT  002070        PERXOR  054660        SIPAR6= 172254
MTPA20  034106        MT0007  021226        PARITY  037760        PFECDF  057126        SIPDR0= 172200
MTPA21  034272        MT0010  021270        PARTHE  002264        PFECDH  057066        SIZE   = 040000
MTPA24  035116        MT0011  021324        PARVEC= 000114        PFECDT  057116        SKIPKA  002006
MTPA25  035526        MT0012  021372        PASFLG  002256        PFECEM  057032        SKIPMK  002312
```

SYMBOL TABLE

```
SKJ      055454    SW8    = 000400    UDPAR0= 177660    $CNTLC  062116    $KERNE  040216
SKPERR   002064    SW9    = 001000    UDPAR7= 177676    $CNTLG  062130    $KMAP   042434
SKUB     043142    SYSSIZ   003720    UIPAR0= 177640    $CNTLK  061324    $K$     = 000061
SKUJ     013140    TAG2$    011200    UIPAR1= 177642    $CNTLU  062123    $L      = 000066
SOBK     002532    TAG3$    011234    UIPAR2= 177644    $CPUOP  063054    $LF       002621
SOBLEN=  000056    TAG4$    026560    UIPAR3= 177646    $CRLF   002620    $LL     = 000064
SOFTPA   002560    TAG70$   057132    UIPAR4= 177650    $DBLK   060714    $LOADC  040310
SOURCE   002272    TAG71$   057142    UIPAR5= 177652    $DB20   062706    $LPADR  002562
SPLTCS   002232    TAG72$   057152    UIPAR6= 177654    $DDW0   063112    $LPERR  002564
SSP    =%000006    TAG73$   057222    UIPDR0= 177600    $DDW1   063114    $L$     = 000000
ST     = 177776    TAG74$   057262    UNITOP   002366    $DDW2   063116    $MADR1  063060
STACK  = 002000    TAG75$   057274    UNRELO   043416    $DDW3   063120    $MADR2  063064
START    003630    TAG76$   057306    UPPFLG   002257    $DDW4   063122    $MADR3  063070
START1   000300    TAG77$   057352    USERMA   044116    $DDW5   063124    $MADR4  063074
START2   000310    TAG78$   057360    USESTK=  000700    $DEENE  040236    $MAIL   063026
START3   000200    TAG79$   057440    USP    =%000006    $DEVCT  063036    $MAMS1  063056
STAR27   024312    TAG9$    011026    WARN1    011114    $DEVM   063104    $MAMS2  063062
STOPOK   002370    TBG4$    026736    WARN2    027156    $DIDDO= 000000    $MAMS3  063066
STRIPE   002336    TCFIG1   037254    WARN3    027172    $DOAGA  014604    $MAMS4  063072
SUBAAA   004604    TCFIG2   037406    WARN4    027216    $DOAGN  014500    $MBADR  063130
SUBAAB   004734    TCFIG3   037574    WARN5    027232    $DOWN   052134    $MNEW   062146
SUBAAI   011604    TCONFI   037132    WARN6    036612    $DTBL   060704    $MSGAD  063042
SUBAAP   013320    TEMP     002404    WARN6A   036552    $ECCDI  040630    $MSGLG  063044
SUBAAR   012464    TEST     006042    WARN6B   036604    $ECCIN  040656    $MSGTY  063026
SUBAAS   010536    TESTAD   002362    WARN7    024270    $ECC1D  040644    $MSWR   062135
SUCCES   002304    TESTMO   002522    WASDBE=  104500    $ECC1I  040672    $MTYP1  063057
SUPDOA   002254    TIME     002310    WASSBE=  104476    $ENASB  040704    $MTYP2  063063
SUPDO1   026464    TIMEOU   040130    WAS1DB=  104501    $ENA1S  040720    $MTYP3  063067
SUPDO2   026500    TKVEC  = 000060    WAS1SB=  104477    $ENDAD  014470    $MTYP4  063073
SUPDO3   026642    TMFLAG   002132    WHICHC   051102    $ENERG  040226    $NOTRA  063176
SUPDO4   026656    TOOMAN   002356    WOOPEN   053112    $ENV    063046    $NULL   002326
SUPDR0   002152    TOTCSR   002216    WOOPS    052544    $ENVM   063047    $NWTST= 000001
SUPDR1   002154    TRACE    006136    WOOPSA   053142    $EOP    014334    $OCNT   060474
SUPDR2   002156    TRAPVE=  000034    WOOPUP   052730    $ERFLG  002012    $OCTVL  063010
SUPDR3   002160    TSTBAN   011446    WORST    002540    $ERRGE  041452    $OCT8 = 063014
SUPDR4   002162    TSTDAT   002240    XOCHAR   053504    $ERROR  055730    $OMODE  060476
SUPDR5   002164    TSTRD1   040602    XXDPCH   002324    $ERRTB  063450    $OVER   055644
SUPDR6   002166    TSTREA=  104510    ZEROS    002306    $ERRTY  056512    $O$     = 000000
SUPLIM   053714    TST1     005434    $APTHD   063126    $ERTTL  002570    $PASS   063034
SUPSTK=  000740    TST2     010542    $AUTO    002060    $ESCAP  002332    $PASTM  063134
SWAPAT   002574    TST3     010726    $BANK    002011    $ETABL  063046    $PATMA  002010
SWR      002576    TST4     011714    $BASE    063102    $ETEND  063126    $PER01  053714
SWREG  = 000176    TST5     014210    $BELL    002613    $EXHAL  045160    $PER02  053742
SW0    = 000001    TST6     014262    $CACHF   040272    $E$     = 000001  $PER03  053770
SW1    = 000002    TYPDS  = 104405    $CACHN   040246    $FATAL  063030    $PER04  054020
SW10   = 002000    TYPEIT=  104401    $CBCSR   040732    $FILLC  002612    $PER07  054102
SW11   = 004000    TYPOC  = 104402    $CB1CS   040754    $FILLS  002327    $PER10  054124
SW12   = 010000    TYPOS  = 104403    $CDW1    063106    $F$     = 000000  $PER11  054154
SW13   = 020000    TYPS0  = 000000    $CDW2    063110    $GTSWR  061100    $PER12  054174
SW14   = 040000    TYPS1  = 000002    $CHARC   053672    $HALT   056274    $PER13  054216
SW15   = 100000    TYPS2  = 000000    $CHKDI   041326    $HALT2  063202    $PER14  054236
SW2    = 000004    TYPS3  = 000000    $CHK1D   041342    $HIBTS  063126    $PER15  054260
SW3    = 000010    TYPS4  = 000000    $CKSWR   060724    $HIOCT  062326    $PER16  054302
SW4    = 000020    TYPS5  = 000000    $CLRCS   041304    $ILLUP  052536    $PER17  054322
SW5    = 000040    TYPS6  = 000002    $CLR1C   041316    $INVAL  041422    $PER20  054340
SW6    = 000100    T12A     033246    $CMTAG   002000    $ITEMB  002013    $PER21  054356
SW7    = 000200    T12B     033270    $CMTGE   002514    $I$     = 000001  $PER22  054376
```

SYMBOL TABLE

```
$PER23  054414          $R     = 177777          $SWR   = 163000          $TSTR'  040424          $VECT1  063076
$PER24  054432          $RAND    062612          $SWREG   063050          $TTYIN  062072          $VECT2  063100
$PER25  051170          $RDCHR   061444          $T     = 000456          $TYPDS  060500          $WASDB  041140
$PER26  054622          $RDDEC   062330          $TESTN   063032          $TYPE   053360          $WASSB  040774
$PER27  054642          $RDLIN   061574          $TKB     002604          $TYPEC  053506          $WAS1D  041254
$PER30  051416          $RDOCT   062160          $TKS     002602          $TYPEX  053674          $WAS1S  041110
$PER31  055032          $READC   040404          $TN    = 000007          $TYPOC  060276          $XTSTR  055522
$PER32  055130          $RESRE   062554          $TPB     002610          $TYPON  060312          $Y$    = 000000
$PER33  055176          $SAVRE   062516          $TPFLG   002330          $TYPOS  060252          $ZAP42  014450
$PER34  055256          $SAVR6   052542          $TPS     002606          $T1    = 000000          $Z$    = 000000
$PER35  055310          $SCOPE   055374          $TRAP    063142          $T2    = 000455          $$S    = 000000
$PER36  055344          $STN   = 000001          $TRAP2   063164          $UNIT    063040          $$T    = 000441
$PWRDN  051564          $SVLAD   055630          $TRPAD   063204          $UNITM   063136          $$TT   = 000447
$PWRUP  052140          $SV$   = 000000          $TSTM    063132          $USWR    063052          $OFILL  060475
$QUES   002617

. ABS.  075624      000
        000000      001
ERRORS DETECTED:  0

VIRTUAL MEMORY USED:  26160 WORDS  ( 103 PAGES)
DYNAMIC MEMORY:  20034 WORDS  ( 77 PAGES)
ELAPSED TIME:  03:25:39
CZMSDC.BIN,CZMSDC/CR/-SP/NL:TOC=CZMSDC.SML,CZMSDC.P11
```

CZMSDC CREATED BY MACRO ON 22-APR-81 AT 15:24 PAGE 1
SEQUENCE 366
CREF V01

SYMBOL CROSS REFERENCE

SYMBOL	VALUE	REFERENCES
ABORTF	002140	#139-5040 *322-10552 406-12668 408-12714 *408-12730 439-14039 439-14042 439-14051 439-14054
ACFLAG	002114	#139-5030 167-6293 169-6333 179-6826 188-7167 190-7183 192-7213 194-7243 196-7280 198-7321 200-7359 202-7404 204-7442 208-7536 236-8383 236-8412 238-8455 348-11096 *356-11369 *356-11384 356-11390 378-11917 382-11987
ACTFLA	002320	#139-5096 *153-5477 164-6154 170-6369 214-7721 214-7736 223-7994 223-8002 223-8016 223-8025 225-8036 225-8044 227-8059 227-8059 232-8311 246-8664 348-11090 362-11497 451-14352
ADDRES	002032	#139-5006 *164-6148 *266-9063 *266-9069 *266-9090 *266-9103 *268-9179 *268-9193 *270-9267 *274-9367 *274-9374 *276-9469 *276-9475 *278-9567 *278-9574 *302-10034 *302-10046 *312-10243 *312-10258 *324-10578 *406-12667 *406-12667 *406-12693 *408-12713 *408-12713 408-12716 408-12726 *436-13932 *436-13933 *436-13938 *436-13939 *436-13944 *436-13945 *436-13950 *436-13951 *436-13958 *436-13966 *436-13971 *436-13971 *436-13976 *436-13981 *438-13987 *438-13992 *438-13997 *438-14002 *438-14007 *438-14012 *438-14017 *438-14022 *438-14027 *438-14032 443-14103 *445-14131 *445-14140 458-14542 495-15853 495-15858 495-15861 495-15866 495-15884
ANA2	007166	#163-5885
APTDOW	045250	153-5471 #362-11531 362-11534 *362-11536
APTECC	002374	#139-5120 *183-7045 183-7050 495-15860
APTFLA	002322	#139-5097 *153-5470 183-7021 186-7149 223-7994 223-8002 223-8016 223-8025 225-8036 225-8044 232-8311 246-8664 348-11090 362-11497 451-14352
APTHAN	014554	#186-7151 186-7157
APTHLT	045316	#362-11538
APTPAR	002372	#139-5119 *183-7042 183-7050 495-15860
APTSIZ	002414	#139-5128 *153-5467 183-7021
BACK	055744	#450-14279 451-14369
BACKGN	036530	219-7931 219-7949 221-7977 227-8121 230-8207 232-8274 244-8648 244-8653 #314-10301
BAD	002050	#139-5012 *158-5683 *158-5691 *406-12670 *406-12696 *408-12716 *408-12728 *436-13934 *436-13940 *436-13947 *436-13952 *436-13957 *436-13967 *436-13972 *436-13977 *436-13982 *438-13988 *438-13993 *438-13998 *438-14003 *438-14008 *438-14013 *438-14018 *438-14023 *438-14028 *438-14033 441-14080 *443-14102 *445-14132 *445-14141 *445-14142 *445-14161 *445-14168 495-15853 495-15861 495-15869 495-15875 495-15880 495-15883
BADPC	002020	#139-5001 *324-10605 406-12681 439-14040 439-14052 441-14063 443-14112 445-14130 445-14139 445-14150 450-14315 450-14316 *450-14320
BADPSW	002030	#139-5005 *324-10606 445-14161 450-14319
BADSP	002024	#139-5003 *324-10603 *324-10604 450-14318
BADSTA	040166	322-10553 322-10561 324-10586 324-10590 324-10593 #324-10602 406-12681 436-13960 439-14040 439-14052 441-14063 443-14112 445-14130 445-14139 445-14150 445-14160
BADXOR	002056	#139-5015 *441-14080 *441-14081 495-15861
BAD2	002052	#139-5013 *406-12697 495-15869
BAD3	002054	#139-5014 *406-12699 *406-12700 495-15869
BAFPAF	014710	184-7072 #190-7179 190-7204
BAFPAR	015016	184-7073 #192-7209 192-7234
BAKPAT	002572	#141-5217 *147-5359 230-8206 232-8273
BANK	002100	#139-5024 *166-6169 *166-6181 166-6182 166-6184 166-6191 *166-6210 *167-6230 *167-6296 *167-6299 *167-6303 167-6305 *169-6329 *170-6366 170-6366 *170-6368 171-6378 *179-6801 179-6856 *179-6861 179-6861 *188-7165 *188-7173 188-7173 *190-7180 *192-7210 *194-7240 *196-7277 *198-7316 *200-7354 *202-7399 *204-7437 208-7505 208-7532 208-7569 *208-7584 210-7639 210-7664 227-8071 227-8078 227-8122 230-8217 230-8231 232-8284 234-8333 234-8346 *236-8381 236-8385 *236-8403 236-8403 *236-8410 236-8413 *236-8432 236-8432 *238-8452 *238-8461 238-8461 *238-8471 240-8489 242-8518 244-8582 248-8672 250-8705 339-10849 346-11053 *348-11094 348-11097 *348-11107 348-11107 350-11201 *350-11202 *350-11206 *350-11210 356-11372 *358-11461 358-11462 371-11682 *371-11687 *371-11691 371-11694 371-11703 *371-11718 373-11724 *373-11729 *373-11733 373-11736 373-11742 *373-11770 375-11776 *375-11781

SYMBOL CROSS REFERENCE

SYMBOL	VALUE	REFERENCES								
		375-11782	375-11784	375-11803	375-11819	375-11828	*375-11882	375-11883	378-11894	*378-11915
		*378-11921	378-11921	*378-11928	378-11929	380-11935	*380-11944	380-11945	380-11952	*380-11958
		380-11958	*380-11960	382-11965	*382-11985	*382-11991	382-11991	*382-11998	382-11999	415-12927
		*415-12928	415-12929	*415-12949	417-12968	*417-12969	417-12970	*417-12983	443-14089	448-14246
		457-14486	457-14500	457-14517	458-14524	458-14544	470-14992			
BANKIN	002102	#139-5025	*166-6187	166-6243	167-6292	169-6332	179-6803	208-7511	223-8006	227-8067
		244-8612	244-8638	340-10858	346-11031	348-11102	350-11204	350-11208	*356-11375	
BANKMO	043750	186-7148	348-11100	350-11199	#352-11252	362-11521				
BANKOK	044750	198-7319	200-7357	202-7402	204-7440	#358-11436				
BAWPAF	015124	184-7074	#194-7239	194-7263	194-7271					
BAWPAR	015254	184-7075	#196-7276	196-7300	196-7308					
BGTEST	036106	#310-10179	310-10200							
BIT0	= 000001	#111-4170	157-5643	157-5647	158-5673	158-5679	158-5689	163-6007	163-6028	163-6038
		163-6074	163-6082	166-6250	166-6269	312-10238	312-10253	318-10387	320-10469	320-10475
		326-10619	326-10623	326-10630	329-10691	329-10693	331-10714	331-10718	331-10722	331-10726
		333-10761	333-10763	333-10769	333-10772	335-10796	335-10798	335-10806	335-10808	348-11118
		348-11172	348-11181	350-11220	350-11223	356-11414	371-11697	373-11738	443-14092	447-14197
		450-14306	450-14308							
BIT1	= 000002	#111-4169	151-5440	157-5652	157-5658	158-5671	158-5687	159-5752	159-5765	163-5957
		163-5977	163-6082	179-6874	179-6900	179-6912	320-10429	320-10467	320-10496	331-10706
		331-10710	331-10722	331-10726	331-10731	331-10736	337-10822	337-10827	356-11379	
BIT10	= 002000	#111-4160	181-6950	312-10246						
BIT11	= 004000	#111-4159	163-6009	179-6841	272-9304	274-9412	276-9510	280-9612	356-11426	
BIT12	= 010000	#111-4158	155-5572	155-5573	155-5579	163-6005	179-6841	179-6880	179-6914	223-8007
		320-10427	348-11183	356-11423	375-11826	463-14675	463-14676	463-14691		
BIT13	= 020000	#111-4157	155-5572	155-5579	157-5649	157-5650	157-5653	157-5656	157-5659	171-6380
		171-6396	186-7110	244-8618	244-8632	266-9132	272-9301	274-9409	276-9507	280-9609
		328-10655	329-10681	339-10852	348-11104	348-11152	350-11209	350-11214	350-11232	350-11245
		360-11489	463-14675	463-14691						
BIT14	= 040000	#111-4156	157-5644	157-5645	163-6059	163-6061	248-8694	293-9879	293-9880	329-10700
		340-10884	340-10886	348-11158	350-11217	352-11276	352-11292	356-11420	385-12065	385-12081
		411-12780	411-12786	414-12883	414-12887	415-12930	417-12971			
BIT15	= 100000	#111-4155	157-5643	158-5705	158-5715	158-5724	160-5844	163-6022	163-6048	210-7633
		210-7651	210-7654	210-7658	228-8143	266-9074	329-10690	329-10700	335-10785	348-11103
		348-11119	348-11158	348-11167	348-11188	350-11205	350-11217	352-11276	352-11292	391-12100
		411-12780	414-12887	445-14151	448-14251	458-14547				
BIT2	= 000004	#111-4168	154-5506	154-5507	154-5508	158-5662	158-5670	158-5685	159-5754	163-5959
		163-5995	163-6084	312-10238	320-10463	322-10555	331-10731	331-10736	337-10822	337-10827
BIT3	= 000010	#111-4167	154-5506	154-5507	154-5510	157-5641	157-5655	158-5718	158-5727	
BIT4	= 000020	#111-4166	137-4960	145-5316	145-5317	157-5641	157-5658	158-5715	158-5724	163-6022
		163-6048	186-7144	210-7633	210-7658	266-9074	329-10690	333-10750	333-10770	445-14151
BIT5	= 000040	#111-4165	153-5463	163-5950	163-6013	163-6043	186-7144	348-11163	348-11171	350-11227
		350-11230	356-11429	360-11480	362-11525					
BIT6	= 000100	#111-4164	169-6349	316-10328	316-10365	320-10520	348-11122	356-11376	375-11815	375-11878
		378-11910	378-11926	382-11981	382-11996					
BIT7	= 000200	#111-4163	153-5466	154-5485	154-5486	167-6301	266-9123	272-9290	274-9398	276-9496
		280-9598	316-10327	329-10676	348-11118	348-11122	356-11408	375-11814	378-11909	382-11980
BIT8	= 000400	#111-4162	181-6951	181-6966	356-11399	356-11405	441-15062			
BIT9	= 001000	#111-4161	181-6952	181-6958	181-6965	348-11121	356-11402			
BLOCK1	045320	166-6178	217-7838	217-7860	217-7886	219-7910	219-7913	219-7939	219-7959	228-8153
		228-8162	228-8178	228-8187	228-8196	230-8235	230-8238	230-8242	230-8245	232-8294
		234-8350	234-8358	236-8376	238-8445	242-8547	244-8616	314-10314	329-10686	#364-11548

SYMBOL CROSS REFERENCE

SYMBOL	VALUE	REFERENCES
BLOCK2	045340	#364-11553
BLOCK3	045354	219-7914 219-7915 219-7940 219-7960 232-8295 232-8296 234-8351 234-8353 240-8491 244-8585 348-11159 350-11218 352-11277 352-11293 #364-11557 415-12933 415-12936
BMFLAG	002126	#139-5035 194-7245 196-7282 200-7361 204-7444 348-11096 *356-11370 *356-11397 *356-11416 *356-11419
BOOT	045026	#360-11466 468-14939 471-15066
BOOT1	045072	360-11474 360-11479 #360-11481
BRGOBB	036110	*310-10180 310-10184 310-10218
BSIZE	002344	#139-5109 *181-6944 *181-6967 *181-6976 *181-6977 *181-6978 *181-6979 181-6980 181-6993 181-6995
CACHKF	002520	#141-5191 *145-5309 326-10637 367-11632
CACHKN	002514	#141-5190 *145-5311 154-5514 *154-5514 *154-5515 326-10627 326-10629 326-10634 365-11579 367-11630 367-11632 367-11635 384-12029 *384-12029 *384-12030 384-12035 *384-12035 410-12750 414-12907 419-13045 419-13105
CACHOF	= 104424	#110-4047 154-5513 159-5733 163-5946 164-6138 166-6200 166-6217 166-6234 166-6260 169-6327 208-7548 210-7617 248-8696 250-8733 292-9841 365-11583 384-12028 419-13049
CACHON	= 104423	#110-4046 147-5386 160-5852 163-6116 164-6122 164-6141 166-6208 166-6221 166-6242 166-6265 170-6374 208-7560 210-7669 210-7677 248-8698 250-8735 292-9819 384-12036 410-12753
CACHVE	= 000114	#111-4185 111-4186
CBCSR	= 104474	#110-4090
CB1CSR	= 104475	#110-4091 163-5967 266-9054 268-9164 270-9259 274-9358 276-9458 278-9557 302-10023
CHECK	002274	#139-5086 *163-5917 *266-9129 *266-9130 *272-9295 *272-9297 *272-9299 *274-9403 *274-9405 *274-9407 *276-9501 *276-9503 *276-9505 *280-9603 *280-9605 *280-9607 331-10730 331-10735 *342-10937 495-15866
CHKDIS	= 104504	#110-4098
CHKGEN	041722	266-9042 268-9155 270-9248 274-9346 276-9445 278-9545 300-9963 #342-10911
CHKTAB	042030	342-10921 #344-10942
CHK1DI	= 104505	#110-4099 171-6385 171-6397 171-6407 171-6417 406-12684
CKEND	061236	468-14920 468-14922 468-14924 #468-14953
CKSWR	= 104410	#110-4030 447-14189 450-14278 451-14341
CLRCSR	= 104502	#110-4096 159-5777 160-5845 #214-7724 236-8371 238-8464 375-11835 #378-11901 #382-11972
CLREX	007136	161-5864 #161-5880
CLRMEM	007036	159-5782 160-5853 #161-5864
CLR1CS	= 104503	#110-4097 163-5966 163-5975 163-6106 171-6389 171-6402 171-6412 171-6421 171-6435 210-7625 210-7649 210-7673 228-8199 266-9020 266-9081 266-9094 266-9140 268-9147 268-9167 268-9222 270-9228 270-9242 270-9261 272-9312 274-9340 274-9362 274-9420 276-9439 276-9460 276-9519 278-9561 280-9621 300-9998 312-10251 406-12690
CMD16A	051022	385-12054 #385-12077
CMD5B	047240	#375-11819 375-11844
CMD5C	047510	375-11808 375-11811 #375-11877
CMD7B	047740	#378-11915 378-11923
CMD7C	050014	378-11906 #378-11925
CMD9B	050436	#382-11985 382-11993
CMD9C	050512	382-11977 #382-11995
CONFGE	002420	#139-5134 *163-6030 *163-6076 *166-6251 *166-6270 *179-6857 *179-6939
CONFIE	003630	#143-5260 151-5436
CONFIG	002624	#143-5257 151-5434 *154-5485 *154-5486 163-5950 *163-5998 *163-6002 *163-6005 *163-6009 *163-6013 163-6014 *163-6014 163-6015 *163-6015 *163-6028 *163-6029 163-6031 *163-6031 163-6032 *163-6032 163-6041 *163-6041 163-6042 *163-6042 *163-6043 *163-6074 *163-6075 *164-6150 166-6192 *166-6250 *166-6252 *166-6269 *166-6273 *167-6301 169-6342 *169-6349 170-6354 *171-6438 *171-6439 179-6827 179-6835 *179-6841 *179-6842 179-6873 179-6880

CZMSDC CREATED BY MACRO ON 22-APR-81 AT 15:24 PAGE 4
SEQUENCE 369

SYMBOL CROSS REFERENCE CREF V01

SYMBOL	VALUE	REFERENCES
		179-6900 179-6912 179-6914 181-6949 181-6950 181-6951 181-6952 181-6958 181-6965
		181-6966 181-6983 186-7109 208-7512 208-7523 223-8007 227-8099 244-8618 *244-8632
		244-8639 312-10246 318-10387 318-10401 320-10427 320-10429 320-10435 320-10456 320-10458
		320-10495 320-10516 320-10520 *339-10852 *348-11103 *348-11104 348-11118 348-11119 348-11120
		348-11121 348-11122 348-11122 *348-11152 348-11177 348-11183 348-11185 *350-11205 *350-11209
		*350-11214 350-11231 356-11376 356-11383 356-11392 356-11399 356-11402 356-11405 356-11408
		356-11414 356-11420 356-11423 356-11426 356-11429 375-11787 375-11826 380-11947 380-11953
		*385-12065 *385-12081 *443-14092 *443-14093 *443-14095 443-14096
CONTFL	002214	#139-5064 *184-7067 *190-7194 *192-7224 *194-7256 *196-7293 *198-7327 *200-7367 *202-7410
		*204-7450 206-7486 *208-7582
CONTRL	= 177746	#111-4191 145-5305 154-5496 *154-5506 *154-5507 154-5508 154-5510 *326-10629 *326-10630
		*326-10637 365-11581 *367-11637 410-12752 *414-12909 419-13047 *419-13107 495-15867
CONTS	061406	468-14913 #470-15001
CONTS1	060764	468-14898 #468-14903 470-15008
CONTS2	061410	#470-15003 470-15004 470-15010
CONTS3	053576	#419-13083 419-13084 419-13090
CONTT	061332	447-14192 463-14641 468-14908 #470-14976
COUNT	002340	#139-5107 *292-9820 292-9823 *292-9823 *292-9824 292-9825 *292-9832 *292-9839 293-9844
		*293-9844 *293-9845 293-9846 *293-9865
CPERRF	056510	*145-5345 *145-5347 447-14194 450-14303 #451-14373
CPSAVE	056506	*447-14196 447-14197 *450-14305 450-14306 #451-14372 455-14465
CPUBIT	002104	#139-5026 *151-5440 164-6150 166-6252 166-6273 181-6949 181-6983 320-10456 348-11120
		356-11383 375-11787
CPUERR	= 177766	#111-4197 *145-5348 *151-5452 *161-5882 *164-6151 *166-6211 *367-11649 *367-11653 *369-11665
		*369-11677 *371-11719 *373-11771 *447-14227 *451-14351 *461-14633 495-15859 495-15867
CR	= 000015	#111-4125 419-13096
CRLF	= 000200	#111-4126 419-13025 499-15947 503-16014 503-16015 503-16016 503-16017 503-16019 503-16021
		503-16022 503-16023 503-16024 503-16025 503-16026 503-16032 503-16036 503-16037 503-16038
		503-16039 503-16040 503-16041 503-16042 503-16043 503-16044 503-16045 503-16046 503-16047
		503-16048 503-16049 503-16050 503-16051 503-16052 503-16057 503-16058 503-16059 503-16060
		503-16062 503-16063 503-16064 503-16065 503-16066 503-16067 503-16068 503-16069 503-16070
		503-16071 503-16072 503-16073 503-16076 503-16077 503-16078 503-16079 503-16080 503-16081
		503-16082 503-16083 503-16084 503-16084 503-16085 503-16087 503-16088 503-16089 503-16090
		503-16091 503-16093 503-16094 503-16095 503-16096 503-16100 503-16102 503-16104 503-16105
		503-16106 503-16107 503-16108 503-16109 503-16109 503-16110 503-16111 503-16118 503-16126
		503-16127 503-16128 503-16140 503-16141 503-16142 503-16142 503-16144 503-16145 503-16146
		503-16151 503-16152 503-16153 503-16154 503-16155 503-16156 503-16157 503-16158 503-16159
		503-16162 503-16165 503-16166 503-16168 503-16169 503-16170
CSR	002144	#139-5042 *158-5713 *158-5722 163-5972 163-6048 *171-6380 *171-6396 *171-6406 *171-6416
		210-7633 210-7658 266-9074 266-9074 *312-10238 *312-10253 *328-10655 328-10656 *328-10664
		329-10690 329-10690 *329-10701 *331-10706 *331-10710 *331-10714 *331-10718 *331-10722 *331-10726
		*331-10730 *331-10731 *331-10735 *331-10736 333-10750 333-10770 335-10785 335-10804 *337-10813
		*337-10817 *337-10822 *337-10827 340-10875 *369-11668 *406-12683 406-12689 411-12797 *413-12864
		445-14151 463-14644 463-14653 *463-14659 495-15864 495-15865 495-15866 495-15881 495-15882
CSRADD	= 172100	#113-4402 157-5634 *159-5754 *159-5757 *159-5759 *159-5762 159-5764 *160-5802 160-5812
		*160-5829 160-5837 *163-5959 *163-5962 *163-6052 163-6054 *163-6059 163-6060 *163-6061
		*328-10656 328-10664 329-10670 *340-10884 340-10885 *340-10886
CSRCAS	017416	208-7557 #212-7683
CSRFBA	002224	#139-5069 *208-7506 208-7531 *208-7575
CSRFIR	002220	#139-5067 *208-7531 208-7537 208-7539 *208-7572 208-7572
CSRHOL	002476	#139-5143 *208-7516 *208-7526 208-7528
CSRINC	002300	#139-5088

```
SYMBOL  VALUE          REFERENCES
CSRINF     002432      #139-5139  *157-5640  *157-5641  *157-5647  *157-5652  *158-5670   158-5671   158-5673  *158-5679
                        158-5681   158-5683   158-5685   158-5687   158-5689   158-5691  *158-5693  *158-5718  *158-5727
                        159-5752   159-5765   163-5957   163-5977   163-5999   163-6082   163-6084   179-6795  *179-6830
                       *179-6850  *179-6851   179-6867   179-6869
CSRINT     002230      #139-5071  *208-7508  *208-7522   208-7574
CSRLAS     002222      #139-5068  *208-7537  *208-7538   208-7564
CSRLBA     002226      #139-5070  *208-7507  *208-7520   208-7572
CSRLOO     002302      #139-5089  *208-7510  *208-7521   208-7578
CSRNO      002146      #139-5043  *159-5740  *159-5746  *160-5796  *160-5809  *160-5823  *160-5834  *163-5938  *163-5945
                        163-6051  *169-6345   169-6346   169-6347  *170-6357  *208-7528  *208-7529  *227-8111  *244-8643
                        244-8644   328-10645  328-10651  328-10653  328-10663   329-10671  329-10677  329-10679 *333-10746
                       *333-10756  333-10756 *335-10781 *335-10791  335-10791  *339-10836 *339-10842   339-10842  340-10883
                        365-11582 *367-11629  375-11834 *375-11836 *391-12102  *391-12116  410-12756 *411-12793 *411-12800
                        411-12800 *413-12861 *413-12868   413-12868 *414-12903   463-14644 *463-14649 *463-14657   463-14657
                       *463-14659
CSROUT     041354       331-10707  331-10715  331-10723  331-10732  337-10814  337-10823 #339-10832
CTEST      006140       158-5707  #159-5733
CTLKVE     002142      #139-5041  *208-7555  *208-7556  *214-7733  *214-7734  *216-7788  *216-7789   468-14917
DATARG  =  177754      #111-4194   154-5500
DATBUF     002234      #139-5073  *260-8887  *260-8888   260-8889   260-8890   260-8892   260-8897   260-8901  *260-8903
                       *260-8903  *260-8906  *260-8907   260-8908   260-8909   260-8911   260-8916   260-8920  *260-8922
                       *260-8922  *266-9027  *266-9028   266-9033   266-9034   266-9115  *266-9117  *266-9117  *268-9148
                       *268-9149   268-9152   268-9153   268-9212  *268-9214  *268-9214  *270-9230  *270-9231   270-9236
                        270-9237  *274-9328  *274-9329   274-9334   274-9335  *276-9427  *276-9428   276-9433   276-9434
                       *278-9528  *278-9529   278-9534   278-9535   436-13968   436-13973
DBEMSK     002250      #139-5076  *270-9234  *270-9235   270-9243   270-9245   270-9253   270-9255   272-9275  *272-9277
                       *272-9277  *272-9294   272-9298  *272-9300   272-9301  *272-9306  *272-9307  *274-9332  *274-9333
                        274-9341   274-9343   274-9351   274-9353   274-9383  *274-9385  *274-9385  *274-9402   274-9406
                       *274-9408   274-9409  *274-9414  *274-9415  *276-9431  *276-9432   276-9440   276-9442   276-9450
                        276-9452   276-9482  *276-9484  *276-9484  *276-9500   276-9504  *276-9506   276-9507  *276-9512
                       *276-9513  *278-9532  *278-9533   278-9540   278-9542   278-9550   278-9552   280-9584  *280-9586
                       *280-9586  *280-9602   280-9606  *280-9608   280-9609  *280-9614  *280-9615
DDISP   =  177570      #111-4119   141-5222   151-5447
DEENER  =  104421      #110-4043   155-5567   360-11477   362-11522
DETAIL     057462       455-14456 #461-14604  461-14636
DETFLA     002212      #139-5063   451-14348  455-14455  *461-14605   461-14606   461-14608 *463-14706
DETPSW     002210      #139-5062   495-15863
DETR0      002172      #139-5055  *461-14613  461-14622   495-15863
DETR1      002174      #139-5056   461-14614  495-15863
DETR2      002176      #139-5057   495-15863
DETR3      002200      #139-5058   495-15863
DETR4      002202      #139-5059   495-15863
DETR5      002204      #139-5060   495-15863
DETSP      002206      #139-5061   495-15863
DET1       060152       463-14674 #463-14691
DF11       064656       485-15569   487-15636  487-15641   487-15646  489-15668   489-15673  489-15688 #497-15900
DF13       064667       487-15661  #497-15901
DF14       064677       491-15776  #497-15902
DF15       064706       493-15842  #497-15903
DF2        064534       485-15574   485-15589  485-15594   485-15599  485-15604   487-15621  487-15651   489-15678  489-15683
                        489-15693   489-15698  491-15729   491-15781  493-15800  #497-15892
DF3        064560       487-15611   489-15715  491-15746   491-15751  491-15761   491-15766  491-15771  #497-15893
```

SYMBOL CROSS REFERENCE CREF V01

SYMBOL		VALUE	REFERENCES
DF4		064573	487-15616 #497-15894
DF5		064606	487-15626 487-15656 491-15756 #497-15895
DF6		064621	487-15631 #497-15896
DF7		064634	489-15710 #497-15897
DF8		064647	491-15734 #497-15898
DF9		064651	485-15579 485-15584 #497-15899
DH1		067375	485-15572 #501-15980
DH10		067745	487-15619 #501-15987
DH11		070043	489-15681 #501-15988
DH12		070061	489-15676 #501-15989
DH13		070066	485-15567 487-15634 487-15639 487-15644 489-15666 489-15671 489-15686 #501-15990
DH14		070163	489-15691 #501-15992
DH15		070242	489-15708 #501-15993
DH16		070361	489-15713 491-15732 #501-15995
DH19		070376	#501-16000
DH2		067432	493-15798 #501-15981
DH23		070403	487-15659 #501-16006
DH24		070462	491-15774 #501-16007
DH25		070531	485-15602 491-15727 #501-16008
DH26		070555	487-15649 #501-16009
DH27		070573	493-15840 #501-16010
DH3		067455	485-15577 485-15582 #501-15982
DH5		067511	485-15587 485-15592 485-15597 489-15696 #501-15983
DH6		067570	491-15779 #501-15984
DH7		067635	487-15609 487-15614 487-15624 487-15629 487-15654 491-15744 491-15749 491-15754 491-15759 491-15764 491-15769 #501-15985
DIAGFL		002002	#139-4991 *231-8263 *231-8265 292-9837
DISPLA		002600	#141-5222 *151-5447 *151-5454 414-12901 448-14256
DISPRE	=	000174	#113-4398 151-5454 *414-12902 *448-14257
DISPTB		014236	184-7070 #184-7072
DOBACK		014610	184-7081 184-7092 #188-7163
DSWR	=	177570	#111-4118 141-5221 151-5446
DT1		064150	485-15573 #495-15853
DT10		064264	487-15620 #495-15863
DT11		064306	489-15682 #495-15864
DT12		064314	489-15677 #495-15865
DT13		064320	485-15568 487-15635 487-15640 487-15645 489-15667 489-15672 489-15687 #495-15866
DT14		064342	*145-5312 *145-5324 *145-5329 489-15692 #495-15867
DT16		064360	489-15709 #495-15868
DT17		064410	489-15714 491-15733 #495-15870
DT20		064416	493-15799 #495-15875
DT22		064426	#495-15879
DT23		064434	487-15660 #495-15880
DT24		064456	491-15775 #495-15881
DT25		064476	485-15603 491-15728 #495-15882
DT26		064506	487-15650 #495-15883
DT27		064514	493-15841 #495-15884
DT3		064162	485-15578 #495-15857
DT4		064172	485-15583 #495-15858
DT5		064202	*145-5323 *145-5328 485-15588 485-15593 485-15598 489-15697 #495-15859
DT6		064220	491-15780 #495-15860
DT7		064234	487-15610 487-15615 487-15625 487-15630 487-15655 491-15745 491-15750 491-15755 491-15760

SYMBOL	VALUE	REFERENCES								
		491-15765	491-15770	#495-15861						
DUMMY	002170	#139-5053	495-15861	495-15861	495-15862	495-15862	495-15862	495-15862	495-15866	495-15866
		495-15868	495-15868	495-15869	495-15869	495-15870	495-15879	495-15880	495-15880	495-15880
		495-15880	495-15880	495-15881	495-15881	495-15881	495-15881	495-15881	495-15884	495-15884
		495-15884	495-15884	495-15884						
ECCDIS	= 104470	#110-4086	214-7722	238-8450	375-11832	378-11899	382-11970			
ECCINI	= 104472	#110-4088	164-6123	#164-6157	#170-6372	208-7581	#214-7739	#227-8115	238-8467	238-8474
		238-8478	360-11473	362-11514	#406-12707					
ECC1DI	= 104471	#110-4087	171-6431	210-7618	210-7641	266-9052	266-9059	266-9136	268-9162	268-9217
		270-9257	272-9309	274-9356	274-9417	276-9455	276-9515	278-9555	280-9617	300-10003
		302-10011	302-10021							
ECC1IN	= 104473	#110-4089	210-7668	210-7676	300-9968	302-10014				
EMTVEC	= 000030	#111-4180	*147-5368	*147-5369						
EM11	065215	487-15608	487-15613	487-15623	487-15628	#499-15928				
EM12	065237	487-15618	#499-15929							
EM13	065263	487-15633	#499-159[illegible]							
EM14	065315	487-15638	#499-15931							
EM15	065361	487-15643	#499-15932							
EM17	065427	487-15653	#499-15933							
EM19	065467	489-15665	#499-15934							
EM2	064715	485-15571	#499-15918							
EM20	065544	489-15670	#499-15935							
EM21	065626	489-15680	#499-15936							
EM22	065665	489-15685	#499-15937							
EM23	065712	489-15695	#499-15938							
EM24	065741	485-15566	#499-15939							
EM25	066020	489-15707	#499-15940							
EM26	066045	489-15712	#499-15941							
EM27	066116	491-15731	#499-15942							
EM29	066206	491-15748	#499-15946							
EM3	064753	485-15576	#499-15919							
EM30	066270	491-15753	#499-15947							
EM31	066407	491-15758	#499-15949							
EM32	066507	491-15763	#499-15950							
EM33	066614	491-15768	#499-15951							
EM35	066722	491-15743	#499-15952							
EM36	067007	491-15778	#499-15953							
EM4	065005	485-15581	#499-15920							
EM40	067056	493-15797	#499-15959							
EM5	065053	485-15586	#499-15921							
EM50	067130	487-15658	#499-15968							
EM51	067164	491-15773	#499-15969							
EM52	067246	491-15726	#499-15970							
EM53	067273	485-15601	#499-15971							
EM55	067322	487-15648	#499-15972							
EM56	067343	493-15839	#499-15973							
EM6	065130	485-15591	#499-15922							
EM7	065155	485-15596	#499-15923							
ENASBE	= 104506	#110-4100	164-6155	170-6370	214-7737	227-8113	406-12705			
ENA1SB	= 104507	#110-4101	300-9972	300-10005	302-10016					
END	075624	#503-16178	503-16190	503-16191						
ENERGI	= 104420	#110-4042	155-5606	186-7145						

```
SYMBOL  VALUE          REFERENCES
ENEXBK    044740        356-11398  356-11424  356-11430 #356-11432
ERRADD    002430       #139-5138   210-7636   210-7661  *340-10905
ERRGEN  = 104512       #110-4104   210-7635   210-7660   266-9080   266-9106   272-9272   312-10249
ERRMAX    002524       #141-5193   443-14096
ERROR   = 104000       #111-4113   158-5684   158-5692   158-5717   158-5726   164-6145   164-6149   171-6440   183-7051
                        266-9075   268-9180   270-9268   300-9977   302-10035  302-10047  312-10244  312-10259  322-10562
                        324-10587  324-10591  324-10594  367-11648  369-11664  369-11673  406-12702  439-14043 #439-14045
                        439-14055 #439-14057  441-14066  443-14115  443-14118  443-14121  443-14124  445-14135  445-14146
                        445-14152 #445-14154  445-14163  445-14170  447-14199  461-14624  461-14630
ERRPC     002016       #139-5000  *450-14293 *450-14294  450-14297 *450-14316 *450-14317  450-14322  453-14390  455-14465
                        495-15853  495-15857  495-15858  495-15859  495-15860  495-15861  495-15864  495-15866  495-15868
                        495-15870  495-15875  495-15879  495-15880  495-15881  495-15882  495-15883  495-15884
ERRPSW    002026       #139-5004  *450-14296 *450-14319  461-14621
ERRSP     002022       #139-5002  *450-14295 *450-14318  461-14620
ERRVEC  = 000004       #111-4173  *147-5378  *147-5379   447-14219 *447-14220 *447-14222 *447-14228
EUFLAG    002130       #139-5036   219-7892   340-10865  340-10880 *356-11368 *356-11401
EVEN      002334       #139-5105  *292-9808   292-9809  *293-9871   293-9871
EXBANK    044300        167-6291   169-6331   179-6802   188-7166   190-7182   192-7212   194-7242   196-7279   198-7318
                        200-7356   202-7401   204-7439   236-8382   236-8411   238-8453   238-8472   348-11095  350-11203
                        350-11207 #356-11345  375-11805  375-11884  378-11916  378-11930  382-11986  382-12000
EXCMD3    046370        371-11708  371-11711  371-11715 #371-11717
EXCMD4    046710        373-11751  373-11755 #373-11769
EXIT      045134       #362-11495   447-14210  451-14354  471-15065
EXIT2     045140       #362-11497
FASTCI  = 177640       #111-4227   166-6206   166-6240   248-8697   329-10687  364-11549
FATAL$    002062       #139-5017  *164-6145  *164-6149  *300-9977  *322-10562 *324-10587 *324-10591 *324-10594  451-14352
                        453-14407
FCMD10    050576        365-11603 #384-12005
FCMD11    050624        365-11604 #384-12015
FCMD12    050646        365-11605 #384-12020
FCMD13    050666        365-11606 #384-12026
FCMD14    050710        365-11607 #384-12033
FCMD15    050726        365-11612 #385-12052
FCMD16    051012        365-11613 #385-12072
FCMD17    051054        365-11614 #387-12086
FCMD18    051070        365-11615 #389-12092
FIELDS    045404       #365-11570   468-14905
FINT      006436        159-5781  #160-5788
FIRST   = 060000       #113-4405   166-6195   166-6228   166-6258   169-6330   208-7506   216-7781   216-7782   217-7830
                        217-7843   217-7867   219-7896   219-7932   219-7950   221-7978   227-8066   228-8170   228-8171
                        230-8212   232-8278   234-8327   236-8386   236-8415   238-8457   240-8491   240-8496   240-8497
                        242-8523   242-8553   242-8560   244-8585   244-8588   244-8589   244-8590   244-8591   244-8608
                        282-9632   292-9821   292-9840   293-9876   293-9878   293-9879   312-10237  314-10304 *362-11534
                       *362-11535 *362-11536  371-11696  373-11737   375-11824  375-11825  415-12931  415-12932   415-12933
                       *415-12934 *415-12935  415-12936  415-12938 *417-12972 *417-12973  417-12978  458-14543
FLIPLO    002556       #141-5206  *147-5353   219-7922  *313-10276 *313-10277  313-10278  313-10280  313-10282
FLIPWA    036400        219-7895  #313-10274
FLUSH     014330       #184-7097
FSCMD0    045602       #365-11593 #367-11620
FSCMD1    045704        365-11594 #367-11642
FSCMD2    046014        365-11595 #369-11657
FSCMD3    046162        365-11596 #371-11681
```

```
SYMBOL   VALUE          REFERENCES
FSCMD4     046436        365-11597 #373-11723
FSCMD5     046756        365-11598 #375-11775
FSCMD6     047600        365-11599 #376-11887
FSCMD7     047606        365-11600 #378-11893
FSCMD8     050100        365-11601 #380-11934
FSCMD9     050304        365-11602 #382-11964
FSINFL     002412       #139-5127  *186-7108
FSPAT      047414        375-11842 #375-11846
FSSTAC     002266       #139-5083  *367-11644  367-11652 *369-11659  369-11676 *375-11777  375-11877 *378-11895  378-11925
                        *382-11966  382-11995
FS1        045470       #365-11585  365-11591  365-11617
FS7FLA     002416       #139-5129   236-8405
FULLRE     002512       #139-5148  *238-8440  *238-8468  *238-8479   348-11134
GBLENG   = 000076        244-8585   244-8588  #310-10219
GETDAT     051464       #408-12721  439-14039  439-14051
GETDA1     051562       *408-12723  408-12731 #408-12734
GETDIS     055656        248-8673   250-8706   447-14241 #448-14245  450-14282
GOOD       002042       #139-5009  *158-5714  *158-5723  *406-12674 *406-12676 *406-12694 *408-12712 *436-13935 *436-13941
                        *436-13946 *436-13953 *436-13956 *436-13963 *436-13968 *436-13973 *436-13978 *436-13983 *438-13989
                        *438-13994 *438-13999 *438-14004 *438-14009 *438-14014 *438-14019 *438-14024 *438-14029 *438-14054
                         441-14079 *443-14104 *443-14106 *445-14133 *445-14143 *445-14162 *445-14167  495-15853  495-15861
                         495-15868  495-15875  495-15880  495-15882
GOOD2      002044       #139-5010  *441-14069 *441-14073  495-15868
GOOD3      002046       #139-5011  *441-14070 *441-14074  495-15868
GTSWR    = 104407       #110-4029   155-5590
HEADER     002552       #141-5204  *147-5354  *184-7087  *184-7089  *188-7168  *188-7171  *206-7480  *206-7495  *208-7551
                        *266-9089  *266-9092  *266-9102  *266-9105  *268-9178  *268-9181  *268-9192  *268-9195  *270-9266
                        *270-9269  *302-10036 *302-10048 *375-11818 *378-11913 *382-11983 *443-14109 *443-14126 *445-14134
                        *445-14136 *445-14145 *445-14147 *445-14169 *445-14171 *451-14364  453-14405  453-14417 *455-14453
                         461-14610 *461-14611  461-14626 *461-14627 *461-14634
HIPAT      045002        202-7397   204-7435  #358-11455
HT       = 000011       #111-4123   419-13022
I          002422       #139-5135  *149-5419  *149-5425   149-5425  *163-5953  *163-5955   163-5995   163-6003   163-6007
                         163-6034   163-6038   163-6086   163-6091   163-6093   163-6097  *181-6974  *181-6980   181-6991
                        *181-6991   181-7013  *208-7550   208-7553  *208-7559   208-7559  *236-8380   236-8390   236-8396
                        *236-8404   236-8404  *236-8409   236-8419   236-8425  *236-8433   236-8433
IBSAVE     056504       *450-14276  450-14301 *450-14309 *450-14312  451-14367 #451-14371
ILLCSR     013256        179-6871   179-6888  #179-6931
IMPTES     012222        169-6351   170-6359  #171-6377
INCBNK     045012        190-7195   192-7225   194-7257   196-7294   198-7328   200-7368   202-7411   204-7451  #358-11459
INCPAT     044766        190-7191   194-7253  #358-11447
INCRPT     044766        198-7331   200-7371  #358-11446
INHBAN     002510       #139-5147  *348-11128  348-11136
INHECC     002506       #139-5146   208-7504   328-10646  348-11126 *348-11127  348-11135 *348-11143 *348-11151 *350-11211
INTFLA     002134       #139-5038   167-6293   169-6336   170-6353   179-6834   208-7517   227-8068   228-8165   340-10894
                        *356-11371 *356-11425
INT64K     002136       #139-5039   167-6294   340-10897  346-11032 *356-11371 *356-11428
INVALI   = 104511       #110-4103   208-7533   214-7731   216-7786   236-8384   373-11743  375-11804  378-11918  382-11988
IOTVEC   = 000020       #111-4178  *147-5366  *147-5367
JMPRL1     043300        348-11162 #348-11172
KAMIKA     002004       #139-4992   246-8664   365-11582 *365-11584 *367-11627 *384-12017 *384-12022
KAMITE     026430        231-8251   231-8259   232-8269   240-8484   242-8513   244-8577  #246-8663
```

SYMBOL CROSS REFERENCE CREF V01

```
SYMBOL  VALUE          REFERENCES
KDIAG   = 000010       #292-9807   292-9823   293-9844   293-9870
KDPAR0  = 172360       #111-4318   219-7914   219-7916   219-7940   219-7941   219-7960   219-7961   232-8296   232-8299
                        234-8351   234-8354   234-8359
KDPAR6  = 172374       #111-4324  *219-7917  *232-8300
KDPAR7  = 172376       #111-4325  *219-7942  *219-7962  *234-8352
KERNEL  = 104417       #110-4040   166-6207   166-6220   166-6241   166-6264   171-6444   210-7667   210-7675   240-8492
                        242-8548   244-8586   248-8699   250-8736   348-11160  350-11219  352-11278  352-11294  362-11537
                        371-11706  371-11717  373-11747  373-11765  373-11769  406-12671  406-12687  406-12698  408-12717
                        408-12729  411-12782  411-12788  414-12885  414-12889  415-12948  417-12982
KERSTK  = 002000       #111-4110
KFLAG     002500       #139-5144   179-6845   223-8028   266-9072   268-9172   274-9320  *356-11368 *356-11407
KIPAR0  = 172340       #111-4308   346-10992  346-11068  354-11302  354-11322  415-12937   417-12955
KIPAR4  = 172350       #111-4312  *159-5735  *160-5791  *160-5806  *160-5846  *160-5850  *161-5867  *161-5876  *163-5933
                       *163-5941   163-6019   163-6069  *163-6108   163-6109   163-6111   354-11336
KIPAR5  = 172352       #111-4313  *163-5934  *163-5942   163-6072  *163-6089  *163-6095  *163-6101  *163-6109
KIPAR6  = 172354       #111-4314
KIPDR0  = 172300       #111-4288   346-11070  354-11304  410-12771  413-12847  413-12871  417-12956
KMAP    = 104422       #110-4044   155-5601
KPFLAG    002112       #139-5029  *356-11368 *356-11378
KSIZE     002346       #139-5110  *181-6944  *181-6955   181-6997   181-6999
KSTACK    002534       #141-5197   144-5270   163-5943   186-7141   463-14664
LAST    = 157776       #113-4406   208-7507   217-7865   227-8084   227-8085   227-8093   230-8210   232-8279   240-8498
                        242-8524   242-8552   244-8592   282-9634   292-9822   293-9843   293-9880   312-10261  371-11699
                        373-11739
LASTBA    002526       #141-5194  *144-5280  *145-5327   163-5926   163-5928   166-6182   167-6305   170-6366   179-6861
                       *179-6906   181-6945   188-7173   232-8290   236-8403   236-8410   238-8461   316-10333  316-10342
                        346-11049  346-11056  348-11107  348-11114  358-11462  371-11694  373-11736  375-11782  378-11921
                        380-11958  382-11991  385-12077
LASTBL    002530       #141-5195  *163-5924  *163-5931   163-6111
LASTER    002014       #139-4999  *186-7114
LF      = 000012       #111-4124   419-13100
LINK1     002472       #139-5141  *166-6175  *166-6179   166-6194   166-6233  *236-8374  *236-8378   236-8394   236-8423
                       *238-8446  *238-8448   238-8458  *240-8498  *240-8506  *242-8522  *242-8535  *244-8592  *244-8600
                        306-10120  308-10167  310-10210
LINK2     002474       #139-5142  *166-6176  *166-6180   166-6219  *242-8525  *242-8536   242-8554   242-8561
LKS     = 177546       #111-4120
LOADBA    002402       #139-5123  *348-11098  350-11197  350-11202
LOADCS  = 104425       #110-4049   312-10239  312-10254  331-10711  331-10719  331-10727  331-10737  337-10818  337-10828
                        339-10839  369-11669  413-12865
LOADHO    002536       #141-5198   186-7146   348-11099  350-11198  350-11206  362-11520
LOOP      014210       #184-7066   186-7160
LOWMAP    043704        348-11170  350-11229 #350-11239  413-12844
LSIZE     002350       #139-5111  *181-6944  *181-6959   181-7001   181-7003   183-7050   495-15860
MAINT   = 177750       #111-4192   145-5307   154-5498
MAPH0   = 170202       #113-4333  *348-11169 *350-11228  411-12815 *413-12843
MAPL0   = 170200       #113-4332  *348-11167 *350-11228  350-11241  411-12815 *413-12843
MAPL1   = 170204       #113-4334   350-11242
MAPPER    042070        155-5605   166-6191   170-6367   171-6378   208-7532   240-8489   242-8518   244-8582   248-8672
                        250-8705  #346-10991  362-11532  371-11703  373-11742  375-11803  375-11828  375-11883  378-11929
                        382-11999  415-12929  417-12970
MBERR     013134       *179-6866  *179-6882   179-6886  *179-6890  *179-6895 #179-6908
MEMDON    014256        184-7071  #184-7081
```

SYMBOL CROSS REFERENCE CREF V01

SYMBOL	VALUE	REFERENCES
MFPT	= 000007	#111-4127 145-5333
MJPAT	020042	149-5410 216-7785 216-7785 216-7790 #216-7798
MJTEST	017736	206-7493 #216-7775
MKCONT	016304	206-7488 #208-7499
MKCSRT	017426	149-5396 #212-7689
MKFLAG	002116	#139-5031 169-6334 179-6831 206-7483 238-8454 *356-11368 *356-11404 358-11442
MKLOOP	016466	#208-7528 208-7580
MKPAT	017656	149-5403 214-7730 214-7730 214-7735 #214-7747
MKTEST	017516	188-7169 206-7491 #214-7717
MMR0	= 177572	#111-4201 *326-10619 *326-10623 *348-11172 *348-11181 *350-11220 *350-11223 411-12803 *413-12856 415-12947 *417-12966 495-15859 495-15867
MMR1	= 177574	#111-4202 411-12803 *413-12856 415-12946 *417-12965 495-15859 495-15867
MMR2	= 177576	#111-4203 411-12803 *413-12856 415-12945 *417-12964 495-15859 495-15867
MMR3	= 172516	#111-4204 145-5314 *145-5315 *145-5316 145-5317 *186-7144 *348-11163 *348-11171 *350-11227 *350-11230 *360-11480 *362-11525 411-12806 *413-12855 415-12944 *417-12963 495-15859 495-15867
MMTRAP	040142	147-5380 #324-10590
MMVEC	= 000250	#111-4188 *147-5380 *147-5381
MPT	= ******	113-4408 137-4971 137-4979 139-5130 139-5149 141-5212 144-5264 144-5271 144-5293 147-5361 147-5382 154-5520 155-5585 155-5593 155-5598 155-5602 172-6447 173-6490 179-6798 179-6804 179-6896 183-7053 184-7093 214-7718 216-7776 274-9322 308-10131 318-10408 324-10596 346-11078 356-11380 356-11386 362-11511 362-11526 365-11608 384-12039 391-12118 410-12742 413-12829 415-12916 419-13050 419-13093 420-13112 447-14200 448-14253 450-14326 455-14441 458-14561 464-14710 468-14889 470-14979 470-14995 478-15406 489-15699 489-15716 491-15735 493-15784 495-15801 493-15843 495-15854 495-15871 495-15876 495-15885 497-15904 499-15915 499-15924 499-15943 499-15954 499-15960 499-15974 501-15996 501-16001 503-16053 503-16112 503-16119 503-16133 503-16147 503-16183
MSEEDH	002546	#141-5202 *147-5355 147-5357
MSEEDL	002550	#141-5203 *147-5356 147-5358
MSGA12	075327	*179-6937 #503-16163
MSGA34	072723	208-7568 #503-16073
MSGB34	072767	208-7570 #503-16074
MSG000	070647	154-5488 #503-16014
MSG001	070714	316-10330 #503-16015
MSG002	070776	316-10331 #503-16016
MSG003	071053	316-10332 #503-16017
MSG004	071160	316-10340 316-10360 #503-16019
MSG005	071266	318-10386 #503-16021
MSG006	071300	479-15444 #503-16022
MSG007	071335	320-10425 #503-16023
MSG008	071347	318-10400 #503-16024
MSG009	071361	320-10455 #503-16025
MSG010	071373	320-10515 #503-16026
MSG011	071405	316-10346 316-10359 #503-16027
MSG012	071473	316-10349 316-10361 #503-16028
MSG013	071570	318-10389 #503-16029
MSG014	071572	316-10358 318-10391 320-10525 450-14287 457-14489 457-14493 458-14559 #503-16030
MSG015	071574	*318-10414 318-10415 *320-10431 *320-10433 *320-10443 320-10444 *320-10465 *320-10471 *320-10473 *320-10477 *320-10479 *320-10481 320-10482 *320-10494 *320-10504 320-10505 *320-10518 *320-10522 320-10523 #503-16031
MSG016	071576	320-10493 #503-16032
MSG017	071610	316-10345 316-10348 #503-16033
MSG018	071621	455-14448 458-14557 463-14654 463-14669 463-14684 463-14699 #503-16034

SYMBOL CROSS REFERENCE CREF V01

SYMBOL	VALUE	REFERENCES
MSG019	071624	458-14527 #503-16035
MSG020	071630	365-11572 #503-16036
MSG021	071651	365-11590 #503-16037
MSG022	072441	391-12105 #503-16062
MSG023	072463	369-11666 #503-16063
MSG025	072477	367-11651 369-11675 #503-16064
MSG026	072523	365-11585 #503-16065
MSG027	072535	369-11662 #503-16066
MSG028	072552	369-11671 #503-16067
MSG029	072566	371-11684 #503-16068
MSG030	072606	375-11779 385-12057 #503-16069
MSG031	072625	371-11685 373-11727 #503-16070
MSG032	072665	371-11710 373-11750 #503-16071
MSG033	072704	371-11714 373-11754 #503-16072
MSG035	073032	186-7119 #503-16075
MSG036	073035	373-11726 #503-16076
MSG037	073054	373-11757 #503-16077
MSG038	073073	373-11766 #503-16078
MSG039	073111	373-11759 #503-16079
MSG040	073133	375-11778 #503-16080
MSG041	073167	375-11788 #503-16081
MSG042	073214	375-11792 #503-16082
MSG043	073235	375-11797 #503-16083
MSG046	073262	375-11810 378-11905 382-11976 #503-16084
MSG047	073315	#503-16085
MSG048	073334	365-11575 #503-16086
MSG049	073374	375-11807 #503-16087
MSG050	073426	#503-16088
MSG051	073534	414-12905 #503-16090
MSG052	073554	#503-16091
MSG053	073605	#503-16092
MSG054	073623	#503-16093
MSG055	073673	378-11896 #503-16094
MSG056	073714	378-11903 382-11974 #503-16095
MSG058	073747	463-14645 #503-16096
MSG061	073771	458-14536 #503-16097
MSG062	074000	472-15165 472-15224 #503-16098
MSG063	074020	472-15166 #503-16099
MSG064	074031	472-15167 472-15226 #503-16100
MSG065	074041	472-15225 #503-16101
MSG066	074053	451-14357 #503-16102
MSG067	074136	453-14409 #503-16103
MSG070	074145	181-7014 #503-16104
MSG071	074176	181-6996 #503-16105
MSG072	074214	181-7000 #503-16106
MSG073	074232	382-11967 #503-16107
MSG075	074250	348-11109 348-11147 #503-16108
MSG076	074302	380-11949 #503-16109
MSG077	074323	186-7117 #503-16110
MSG079	074337	380-11941 #503-16111
MSG085	074363	384-12007 #503-16118
MSG088	074410	463-14663 #503-16126

SYMBOL CROSS REFERENCE

SYMBOL	VALUE	REFERENCES								
MSG089	074426	463-14679	#503-16127							
MSG090	074450	463-14694	#503-16128							
MSG091	074464	463-14688	463-14703	#503-16129						
MSG092	074476	470-14989	#503-16130							
MSG093	074512	470-14991	#503-16131							
MSG095	074520	470-14993	#503-16132							
MSG101	074527	384-12016	#503-16140							
MSG102	074557	384-12021	#503-16141							
MSG103	074606	367-11621	#503-16142							
MSG104	074630	455-14458	#503-16143							
MSG105	074632	385-12053	#503-16144							
MSG106	074736	384-12027	#503-16145							
MSG107	074754	384-12034	#503-16146							
MSG110	075031	385-12068	#503-16151							
MSG111	075075	385-12073	#503-16152							
MSG112	075127	181-7004	#503-16153							
MSG113	075144	181-7008	#503-16154							
MSG114	075161	181-7012	#503-16155							
MSG116	075205	154-5503	#503-16156							
MSG117	075217	154-5501	#503-16157							
MSG118	075231	154-5505	#503-16158							
MSG119	075243	154-5517	#503-16159							
MSG120	075252	154-5518	#503-16160							
MSG121	075273	154-5512	#503-16161							
MSG122	075313	179-6938	#503-16162							
MSG123	075361	348-11139	#503-16165							
MSG124	075432	179-6855	#503-16166							
MSG125	075475	380-11939	#503-16167							
MSG126	075517	160-5854	#503-16168							
MSG127	075564	387-12087	#503-16169							
MSG128	075603	389-12093	#503-16170							
MSIZE	002352	#139-5112	*181-6944	*181-6961	181-7005	181-7007	183-7050	495-15860		
MTA030	024730	#238-8441	238-8473							
MTEST	016210	190-7189	192-7219	194-7251	196-7288	198-7325	200-7365	202-7408	204-7448	#206-7479
MTLA11	030046	266-9022	#266-9027	266-9122						
MTLB11	030060	#266-9030	266-9118							
MTLC11	030072	#266-9033	266-9112							
MTLD11	030166	#266-9050	266-9134							
MTL020	022032	#227-8068	227-8135							
MTPA03	027144	219-7904	219-7910	#254-8789						
MTPA04	027302	219-7936	219-7939	219-7957	#258-8848	258-8868				
MTPA20	034106	228-8149	228-8150	228-8153	#284-9663	284-9673				
MTPA21	034272	230-8221	230-8235	#288-9732						
MTPA24	035116	232-8294	#296-9897	298-9956						
MTPA25	035526	300-9967	300-10002	302-10010	#302-10021					
MTPA26	035656	234-8337	234-8338	234-8350	#304-10053	304-10096				
MTPB03	027204	219-7907	219-7913	#254-8812	256-8845					
MTPB04	027336	219-7940	*219-7955	*219-7957	219-7960	258-8852	#258-8861	258-8874		
MTPB20	034136	228-8158	228-8159	228-8162	#284-9677	284-9687				
MTPB21	034322	230-8223	230-8238	#288-9749						
MTPB24	035156	232-8288	232-8295	296-9938	#298-9941					
MTPB25	035550	300-10004	302-10012	#302-10028						

SYMBOL CROSS REFERENCE CREF V01

SYMBOL	VALUE	REFERENCES
MTPB26	035672	234-8341 234-8342 234-8358 #304-10060
MTPC03	027244	219-7914 254-8814 #256-8830 256-8844
MTPC20	034166	228-8174 228-8175 228-8178 #284-9691 284-9701
MTPC21	034356	230-8226 230-8242 #288-9766
MTPC24	035172	232-8296 298-9947 #298-9949
MTPC25	035610	300-10006 302-10015 302-10017 #302-10040
MTPC26	035726	234-8351 304-10054 304-10063 #304-10078
MTPD03	027262	219-7915 256-8836 #256-8838
MTPD20	034216	228-8183 228-8184 228-8187 #286-9706 286-9716
MTPD21	034412	230-8228 230-8245 #290-9784
MTPD25	035454	300-9979 300-9981 300-9984 300-9986 #300-10002
MTPD26	035746	*234-8337 *234-8341 234-8353 #304-10091
MTPE20	034246	228-8192 228-8193 228-8196 #286-9720 286-9728
MTPE25	035476	300-9990 300-9992 300-9995 300-9997 #302-10010
MTP000	027034	217-7835 217-7838 #252-8752 252-8754 *314-10308 314-10309 *314-10311 314-10314
MTP001	027060	217-7857 217-7860 #252-8763
MTP002	027112	217-7883 217-7886 #252-8775
MTP005	027356	219-7954 219-7955 219-7959 #258-8870
MTP006	027412	221-7971 #260-8884
MTP007	027612	221-7981 #262-8927
MTP010	027712	221-7988 #264-8967
MTP011	030020	223-7998 #266-9003
MTP012	030616	223-8012 #268-9144
MTP013	031204	223-8020 #270-9226
MTP014	031720	223-8030 #274-9316
MTP015	032502	225-8040 #276-9424
MTP016	033246	225-8048 #278-9523
MTP017	034030	225-8053 #282-9625
MTP020	034106	#284-9658
MTP022	034442	231-8254 231-8262 #292-9802
MTP025	035210	232-8315 #300-9959
MTP030	035764	238-8445 238-8449 #306-10099
MTP031	035774	240-8491 #306-10105 306-10127
MTP032	036052	242-8547 242-8566 #308-10156
MTP033	036104	244-8585 #310-10176 310-10219
MTP034	036202	236-8376 236-8393 236-8399 236-8422 236-8428 244-8616 244-8622 244-8628 #311-10222 311-10224
MTP035	036226	244-8649 #312-10234
MTST3	011560	166-6175 166-6176 166-6178 166-6204 166-6238 #166-6276 166-6277
MTV020	022426	*227-8118 227-8122 *227-8130 227-8130 #227-8138
MT0000	020122	#217-7828 375-11846
MT0001	020202	214-7751 216-7803 #217-7841 375-11847
MT0002	020322	214-7752 216-7804 #217-7863 375-11848
MT0003	020462	216-7805 #219-7891 375-11849
MT0004	020714	214-7753 216-7806 #219-7927 375-11850
MT0005	021036	214-7754 216-7807 #219-7945 375-11851
MT0006	021172	212-7689 216-7800 #221-7967 375-11852
MT0007	021226	214-7750 216-7802 #221-7974 375-11853
MT0010	021270	212-7690 #221-7984 375-11854
MT0011	021324	212-7692 #223-7993 375-11855
MT0012	021372	212-7693 #223-8001 375-11856
MT0013	021466	212-7694 #223-8015 375-11857

SYMBOL CROSS REFERENCE

SYMBOL	VALUE	REFERENCES								
MT0014	021542	212-7695	#223-8024	375-11858						
MT0015	021620	212-7696	#225-8035	375-11859						
MT0016	021666	212-7697	#225-8043	375-11860						
MT0017	021734	214-7749	216-7801	#225-8051	375-11861					
MT0020	021756	214-7756	#227-8058	375-11862						
MT0021	023046	214-7755	216-7808	#230-8203	375-11863					
MT0022	023320	214-7757	216-7810	#231-8250	375-11864	382-11989				
MT0023	023352	216-7811	#231-8258	375-11865						
MT0024	023416	214-7759	216-7813	#232-8268	375-11866					
MT0025	023662	212-7691	#232-8310	375-11867						
MT0026	023730	214-7758	216-7812	#234-8320	375-11868					
MT0027	024232	184-7088	#236-8367	375-11869						
MT0030	024716	184-7098	#238-8438	360-11476	362-11517	375-11870				
MT0031	025220	214-7760	216-7814	#240-8483	375-11871	378-11919				
MT0032	025410	214-7761	216-7815	#242-8512	375-11872					
MT0033	025742	214-7762	216-7816	#244-8576	375-11873					
MT0034	026130	214-7730	214-7748	214-7763	216-7785	216-7799	216-7817	#244-8606	375-11874	
MT0035	026302	216-7809	#244-8636	375-11875						
MT020Y	022614	228-8145	228-8161	#228-8165						
MT020Z	022430	227-8127	#228-8140							
MT0999	026414	149-5422	212-7698	212-7699	212-7700	212-7701	212-7702	212-7703	212-7704	212-7705
		212-7706	212-7707	212-7708	212-7709	212-7710	212-7711	212-7712	214-7730	214-7765
		214-7766	214-7767	214-7768	214-7769	214-7770	214-7771	214-7772	216-7785	216-7819
		216-7820	216-7821	216-7822	216-7823	#246-8658				
MT1	016264	206-7484	#206-7493							
MT2	016270	206-7492	#206-7494							
MUT	002106	#139-5027	*184-7087	*184-7090	*188-7168	*188-7170	*206-7481	*206-7494	*375-11818	*378-11913
		*382-11983	455-14451	461-14610	*461-14612	461-14626	*461-14628	*461-14634		
NC	C53636	419-13078	419-13082	419-13088	#419-13092					
NEMCNT	002066	#139-5019	*164-6133	164-6146	*166-6190	166-6214	166-6226	166-6248	166-6272	*324-10575
		324-10576	*324-10580	495-15858						
NEWBAN	002270	#139-5084	238-8471	*348-11155	348-11164	348-11174	*350-11215	354-11310	354-11323	
NEWKER	044200	348-11173	350-11221	#354-11320						
NEWLOA	044246	348-11101	350-11200	#354-11333						
NOCH	060746	468-14895	#468-14899							
NOERRO	002400	#139-5122	*367-11647	*369-11663	*369-11672	450-14277	450-14314	451-14337	*451-14366	453-14403
		453-14415	455-14455	*461-14623	*461-14629					
NOFSMO	002376	#139-5121	*236-8379	*236-8406	*236-8434	*238-8451	*238-8468	*238-8479	365-11574	
NONEM	002076	#139-5023	*164-6134	*166-6171	*166-6257	*166-6266	324-10573			
NONEXI	040064	164-6135	166-6172	#324-10573						
NOOJ	036742	316-10334	#316-10337							
NOPAR	002074	#139-5022	*144-5285	*144-5291	*157-5635	*161-5866	*164-6132	*166-6170	*166-6212	*208-7549
		*214-7726	*214-7741	*216-7779	*216-7791	*223-8021	*227-8063	*227-8136	*238-8442	*240-8488
		*242-8517	*244-8581	*300-9964	*300-9999	*312-10235	*312-10263	322-10546	322-10550	322-10557
		371-11682	*371-11683	*371-11718	373-11724	*373-11725	*373-11770	*375-11838	*375-11843	378-11894
		*378-11928	382-11965	*382-11998						
NORES	003640	144-5268	#144-5270							
NOSCOP	002410	#139-5126	*230-8204	*230-8247	*232-8271	*232-8308	*238-8451	*238-8468	*238-8479	*240-8486
		*240-8508	*242-8515	*242-8570	*244-8579	*244-8603	447-14212	451-14342		
NOSUPE	002426	#139-5137	*145-5322	*145-5341	155-5568	217-7845	217-7869	240-8499	242-8526	244-8593
		250-8728	340-10859	346-10995	346-11012	346-11037	346-11058	410-12763	411-12784	411-12804
		413-12853	414-12880	415-12942	417-12961	463-14673				

SYMBOL CROSS REFERENCE

SYMBOL	VALUE	REFERENCES
NOTAB	002342	#139-5108 320-10445 320-10483 320-10506 *457-14488 *457-14491 *457-14502 *457-14504 *458-14526 *458-14529
NOTRCE	055426	447-14191 #447-14193
NO22BI	002424	#139-5136 *145-5326 163-5922 163-6057 186-7142 227-8070 227-8077 227-8122 340-10878 346-11051 360-11478 362-11523
NULLFL	002314	#139-5094 *246-8660
OLDCAC	002262	#139-5081
OLDCSR	002150	#139-5044 *169-6328 169-6346 *169-6347
ONES	002554	#141-5205 166-6232 171-6398 171-6400 210-7644 210-7646 219-7930 292-9811 292-9813 313-10285 313-10290 438-13994
PADDRE	002034	#139-5007 495-15857
PAFBAF	015404	184-7076 #198-7313 198-7338 198-7346
PAFBAW	015534	184-7077 #200-7351 200-7378 200-7383 200-7391
PARBAF	015706	184-7078 #202-7396 202-7421 202-7429
PARBAW	016036	184-7079 #204-7434 204-7461 204-7466 204-7474
PARCNT	002070	#139-5020 *164-6131 164-6143 *166-6189 166-6224 166-6246 166-6268 *322-10548 495-15857
PARITY	037760	147-5374 #322-10546
PARTHE	002264	#139-5082 *228-8150 *228-8154 *228-8159 *228-8163 *228-8175 *228-8179 *228-8184 *228-8188 *228-8193 *228-8197 *300-9966 *302-10028 *302-10040 *302-10043 *312-10240 *312-10255 322-10559 371-11682 *371-11701 *371-11718 373-11724 *373-11740 *373-11770
PARVEC	= 000114	#111-4186 *147-5374 *147-5375
PASFLG	002256	#139-5078 *206-7482 *208-7552 *238-8439 238-8462 *238-8463 266-9035 266-9107 266-9119 *266-9121 *266-9123 270-9238 272-9273 272-9285 *272-9287 *272-9290 274-9336 274-9381 274-9393 *274-9395 *274-9398 276-9435 276-9480 276-9492 *276-9494 *276-9496 278-9536 280-9582 280-9594 *280-9596 *280-9598 *375-11823
PATERR	002072	#139-5021 *166-6188 166-6222 166-6244 *166-6282
PATPLU	004546	149-5398 149-5401 149-5405 149-5408 149-5412 149-5415 #149-5418
PATTER	002110	#139-5028 *188-7164 *190-7187 *192-7221 *194-7249 *196-7290 *198-7314 *200-7352 *202-7414 *204-7454 214-7728 216-7783 *358-11450 358-11451 *358-11456 375-11776 *375-11794 375-11795 375-11796 375-11840 *375-11882 378-11894 *378-11928 382-11965 *382-11998
PCBUMP	002276	#139-5087 *214-7727 *216-7780 *219-7894 *219-7929 *219-7947 *221-7969 *221-7986 *234-8322 322-10554 375-11776 *375-11839 *375-11881
PCONFI	036630	181-7016 #316-10321 376-11888
PCONFS	037130	*316-10323 316-10364 #316-10370
PCONF1	037040	316-10336 #316-10355
PCONF2	037076	316-10324 316-10343 316-10353 #316-10364
PDP110	040154	147-5376 #324-10593
PD1	051704	410-12764 #410-12771
PERA05	054052	#436-13957 436-13964
PERBNK	054704	312-10245 312-10260 406-12701 439-14038 439-14050 441-14065 #443-14086 445-14159
PERECC	054764	#443-14102 443-14113
PERRAB	054522	436-13979 436-13984 #439-14049
PERRAW	054450	406-12678 408-12719 436-13936 436-13942 436-13948 436-13954 436-13969 436-13974 438-13990 438-13995 438-14000 438-14005 438-14010 438-14015 438-14025 438-14030 438-14035 #439-14037
PERRA3	051252	#406-12680 441-14071 441-14075
PERRA7	054574	438-14020 #441-14062
PERR01	= 104427	#110-4052 252-8771 308-10171
PERR02	= 104430	#110-4053 252-8759 252-8784 308-10165 311-10229
PERR03	= 104431	#110-4054 256-8833 256-8841
PERR04	= 104432	#110-4055 258-8857 258-8879
PERR05	054046	#436-13956
PERR06	054074	#436-13963

SYMBOL CROSS REFERENCE

SYMBOL	VALUE	REFERENCES							
PERR07	= 104433	#110-4056	260-8894	260-8913					
PERR10	= 104434	#110-4057	260-8899	260-8918					
PERR11	= 104435	#110-4058	262-8935	264-8985					
PERR12	= 104436	#110-4059	262-8941	264-8992					
PERR13	= 104437	#110-4060	262-8949						
PERR14	= 104440	#110-4061	262-8955						
PERR15	= 104441	#110-4062							
PERR16	= 104442	#110-4063							
PERR17	= 104443	#110-4064	288-9737	288-9759	288-9770	290-9794	293-9849	293-9853	
PERR20	= 104444	#110-4065	288-9743	288-9753	288-9776	290-9788	293-9858	293-9862	
PERR21	= 104445	#110-4066							
PERR22	= 104446	#110-4067	282-9647						
PERR23	= 104447	#110-4068	296-9926	298-9955					
PERR24	= 104450	#110-4069	296-9931						
PERR25	= 104451	#110-4070	304-10066	304-10071					
PERR26	= 104452	#110-4071	284-9681	286-9712					
PERR27	= 104453	#110-4072	284-9669	284-9695	286-9724				
PERR30	= 104454	#110-4073	306-10112						
PERR31	= 104455	#110-4074	266-9064	266-9070	274-9368	274-9375	276-9470	276-9476	278-9568	278-9575
PERR32	= 104456	#110-4075	266-9085	266-9098					
PERR33	= 104457	#110-4076	268-9186						
PERR34	= 104460	#110-4077	266-9091	266-9104	268-9194				
PERR35	= 104461	#110-4078	310-10188	310-10195					
PERR36	= 104462	#110-4079	340-10904						
PERR37	= 104463	#110-4080							
PERR40	= 104464	#110-4081							
PERR41	= 104465	#110-4082							
PERR42	= 104466	#110-4083							
PERR43	= 104467	#110-4084							
PERXOR	054660	439-14041	439-14053	441-14064	#441-14077	443-14108	445-14144		
PFECDF	057126	455-14461	#455-14466						
PFECDH	057066	455-14461	#455-14463						
PFECDT	057116	455-14461	#455-14465						
PFECEM	057032	455-14461	#455-14462						
PFECWS	057022	453-14394	#455-14461						
PFLAG	002120	#139-5032	169-6348	348-11096	*356-11370	*356-11410			
PGMCSR	002502	#139-5145	*159-5734	*159-5775	159-5780	*160-5818	*160-5843	*160-5844	*160-5855	244-8644
		328-10647	328-10649	328-10653	329-10673	329-10675	329-10679	348-11156	*348-11156	*348-11182
		*348-11189	350-11222	*350-11222						
PHEBE	013136	#179-6909	*179-6911	*179-6916	179-6919	179-6921	179-6923	179-6925	*179-6927	*179-6929
		266-9021	266-9078							
PHYADD	002036	#139-5008	*458-14542	*458-14543	*458-14544	*458-14545	*458-14547	458-14548		
PROTYP	003716	#145-5303	*145-5338	145-5339	145-5348	151-5452	161-5882	164-6151	166-6173	166-6202
		166-6211	166-6236	217-7833	217-7855	217-7881	219-7901	219-7934	219-7952	228-8147
		228-8156	228-8172	228-8181	228-8190	230-8213	230-8215	232-8280	232-8282	234-8329
		234-8331	236-8372	236-8391	236-8397	236-8420	236-8426	238-8443	242-8544	242-8564
		244-8614	244-8620	244-8626	314-10306	329-10682	348-11161	350-11225	367-11649	367-11653
		369-11665	369-11677	371-11719	373-11771	411-12813	413-12841	447-14225	451-14349	461-14631
PSIZE	002354	#139-5113	*181-6944	*181-6953	181-7009	181-7011				
PSW	= 177776	#111-4115	*155-5572	*155-5573	*155-5579	*166-6201	*166-6218	*166-6235	*166-6261	*171-6381
		*210-7616	*240-8490	*242-8538	*244-8584	*248-8694	*250-8727	*316-10327	*329-10698	*329-10700
		*348-11158	*350-11217	*352-11276	*352-11292	*362-11533	*371-11704	*373-11744	*373-11762	*375-11814

SYMBOL CROSS REFERENCE

SYMBOL	VALUE	REFERENCES
		*378-11909 *382-11980 *406-12669 *406-12685 *406-12695 *408-12715 *408-12727 *411-12780 *411-12786
		*414-12883 *414-12887 *415-12930 *417-12971 *463-14675 *463-14676 *463-14691
PWRVEC	= 000024	#111-4179 *147-5372 *147-5373 *153-5471 *410-12754 *410-12755 *411-12821 *413-12832 *414-12904
		415-12931 415-12932 *415-12934 *415-12935 *417-12972 *417-12973
QUICK	002406	#139-5125 *153-5470 362-11516
QVFLAG	002316	#139-5095 *153-5470 *153-5475 186-7118 *186-7120 228-8144 244-8651 266-9114 268-9211
		272-9280 274-9388 276-9487 280-9589
RANODD	035706	*234-8340 *234-8357 #304-10068 *304-10072
RDCHR	= 104411	#110-4032 471-15055
RDDEC	= 104414	#110-4035 365-11586
RDLIN	= 104412	#110-4033 391-12106 472-15136 472-15187
RDOCT	= 104413	#110-4034 369-11667 371-11686 373-11728 373-11760 375-11780 375-11793 375-11798 384-12008
		385-12058
READCS	= 104426	#110-4050 163-5970 266-9073 300-9976 302-10033 302-10045 312-10248 333-10749 333-10768
		335-10784 335-10803 367-11646 369-11661 369-11670 406-12688 411-12796 463-14652
READON	002360	#139-5115
REALPA	002260	#139-5080 *217-7829 *217-7842 *217-7864 *219-7893 *219-7928 *219-7946 *221-7968 *221-7975
		*221-7985 *223-7997 *223-8005 *223-8019 *223-8029 *225-8039 *225-8047 *225-8052 *227-8062
		*230-8205 *231-8253 *231-8261 *232-8272 *232-8314 *234-8321 *236-8370 *238-8441 *240-8487
		*242-8516 *244-8580 *244-8607 *244-8637 *246-8659 348-11137 348-11145 443-14114 443-14117
		443-14120 443-14123 448-14247 457-14511 470-14994
REFRES	035016	292-9837 #293-9874
REFSUB	035066	293-9877 293-9881 #293-9885
REGCOP	036370	217-7832 217-7854 #313-10268
RELENT	043152	348-11138 348-11140 #348-11152
RELOCA	042526	190-7201 192-7231 194-7268 196-7305 198-7343 200-7388 202-7426 204-7471 238-8465
		#348-11088
RELOC1	043166	#348-11155
RESREG	= 104416	#110-4038 219-7906 219-7920 232-8304 234-8343 234-8360 314-10312 314-10317 352-11259
		352-11296 365-11576 367-11639 455-14454 476-15320
RESTAR	002566	*137-4975 *137-4977 #141-5210 151-5433
RESVEC	= 000010	#111-4174 *147-5376 *147-5377
RES0	045700	367-11631 367-11636 #367-11639
RES1	045760	367-11645 #367-11651
RES2	046126	369-11660 #369-11675
RLFLAG	002124	#139-5034 190-7198 192-7228 194-7265 196-7302 198-7340 200-7385 202-7423 204-7468
		*348-11190 *350-11224 356-11411 360-11475 362-11515 365-11574 410-12769 448-14249 470-14988
RRFLAG	002122	#139-5033 188-7167 190-7185 192-7215 194-7247 196-7284 198-7323 200-7363 202-7406
		204-7446 208-7536 236-8383 236-8412 238-8455 *356-11370 *356-11410 *356-11413 *356-11421
		375-11806 378-11917 382-11987
SAVREG	= 104415	#110-4037 219-7903 219-7911 232-8287 232-8293 234-8336 234-8349 314-10303 352-11257
		352-11260 365-11571 453-14384 476-15305
SBEMSK	002244	#139-5075 *266-9030 *266-9031 266-9046 266-9048 266-9109 *266-9111 *266-9111 *268-9150
		*268-9151 268-9156 268-9158 268-9169 268-9206 *268-9208 *268-9208 *270-9232 *270-9233
		270-9243 270-9245 270-9249 270-9251 272-9281 *272-9283 *272-9283 *272-9293 272-9296
		*272-9303 272-9304 272-9306 *274-9330 *274-9331 274-9341 274-9343 274-9347 274-9349
		274-9389 *274-9391 *274-9391 *274-9401 274-9404 *274-9411 274-9412 274-9414 *276-9429
		*276-9430 276-9440 276-9442 276-9446 276-9448 276-9488 *276-9490 *276-9490 *276-9499
		276-9502 *276-9509 276-9510 276-9512 *278-9530 *278-9531 278-9540 278-9542 278-9546
		278-9548 280-9590 *280-9592 *280-9592 *280-9601 280-9604 *280-9611 280-9612 280-9614
SBENT	017370	210-7621 210-7623 210-7628 210-7630 210-7639 210-7645 210-7647 210-7653 210-7655
		210-7664 #210-7673

SYMBOL CROSS REFERENCE
SYMBOL VALUE REFERENCES

```
SBETES     017112  208-7546  #210-7588
SCOPE    = 000004  #111-4114  157-5609  164-6124  166-6161  169-6309  184-7066  184-7083  248-8700  250-8737
SDPAR0   = 172260  #111-4278  219-7915  219-7917  232-8295  232-8297  232-8298  234-8353
SDPAR5   = 172272  #111-4283  *219-7919  *232-8299
SDPAR6   = 172274  #111-4284  *234-8355
SDPAR7   = 172276  #111-4285  *219-7918
SEEDHI     002542  #141-5200  *147-5357  234-8324  *234-8363  474-15278  474-15285  *474-15290
SEEDLO     002544  #141-5201  *147-5358  234-8323  *234-8362  474-15277  474-15283  *474-15289
SELONL     002000  #139-4990  184-7086  208-7503  356-11420  *385-12069  *385-12074
SETPAT     045002  192-7217  196-7286  #358-11454
SHADL1     011610  #167-6291  167-6306
SHUTUP     045166  186-7129  362-11496  #362-11504
SIPAR0   = 172240  #111-4268  346-10993
SIPAR3   = 172246  #311-4271  266-9023  *266-9024  340-10861  346-11011
SIPAR5   = 172252  #111-4273  217-7847  217-7871  240-8501  242-8528  244-8595  266-9024  *266-9025  346-11041
                   346-11062
SIPAR6   = 172254  #111-4274  217-7847  217-7871  240-8501  242-8528  244-8595  *346-11062
SIPDR0   = 172200  #111-4248  346-10994  410-12773  413-12873
SIZE     = 040000  #113-4407  164-6137  166-6197  166-6230  217-7831  217-7844  217-7866  219-7933  219-7951
                   234-8328  236-8388  236-8417  238-8456  242-8521  242-8558  244-8609  314-10305  348-11159
                   350-11218  352-11277  352-11293
SKIPKA     002006  #139-4993  367-11623  *367-11625  *384-12017  *384-12023
SKIPMK     002312  #139-5093  169-6335  206-7487  227-8089  *356-11371  *356-11431
SKJ        055454  447-14195  447-14198  #447-14208
SKPERR     002064  #139-5018  *210-7634  *210-7659  340-10902  *340-10906
SKUB       043142  348-11146  #348-11149
SKUJ       013140  179-6907  #179-6910
SOBK       002532  #141-5196  240-8493
SOBLEN   = 000056  240-8491  240-8496  #306-10127
SOFTPA     002560  #141-5207  244-8610
SOURCE     002272  #139-5085  *266-9041  *268-9154  *270-9247  *274-9345  *276-9444  *278-9544  *300-9962  342-10922
SPLTCS     002232  #139-5072  *169-6338  *170-6360  *208-7509  *208-7519  208-7541  *208-7573  *208-7576  *208-7583
                   *227-8095  *227-8128  *227-8133  346-11019  346-11024  *375-11822  *375-11827
SSP      =%000006  #111-4132  *155-5576  *248-8695  *250-8732  411-12787  *414-12884  463-14677
ST       = 177776  #483-15548
STACK    = 002000  #111-4109  111-4110  137-4965  137-4983  141-5197  153-5474  186-7125
START      003630  137-4976  137-4978  #144-5263  362-11501
START1     000300  137-4969  #137-4975
START2     000310  137-4970  #137-4977
START3     000200  #137-4969  503-16195
STAR27     024312  236-8375  #236-8380
STOPOK     002370  #139-5118  *177-6790  447-14208  *447-14209  *471-15064
STRIPE     002336  #139-5106  *292-9816  292-9820  292-9839  *293-9870  293-9870
SUBAAA     004604  149-5416  #151-5429
SUBAAB     004734  #153-5459
SUBAAI     011604  166-6213  #167-6286
SUBAAP     013320  179-6930  #181-6943
SUBAAR     012464  170-6375  #177-6790
SUBAAS     010536  163-5944  #164-6122
SUCCES     002304  #139-5090  *208-7535  *208-7561  208-7567  *380-11943  380-11948  *380-11950
SUPDOA     002254  #139-5077  *217-7835  *217-7857  *217-7883  *219-7904  *219-7907  *219-7936  *219-7954  *221-7971
                   *221-7981  *221-7988  *223-7998  *223-8012  *223-8020  *223-8030  *225-8040  *225-8048  *225-8053
```

SYMBOL CROSS REFERENCE
SYMBOL VALUE REFERENCES

*228-8149 *228-8158 *228-8174 *228-8183 *228-8192 *230-8221 *230-8223 *230-8226 *230-8228
*231-8254 *231-8262 *232-8288 *232-8297 *232-8315 *234-8338 *234-8342 *236-8377 *236-8393
*236-8399 *236-8422 *236-8428 *238-8449 *240-8497 *242-8553 *242-8566 *244-8589 *244-8617
*244-8622 *244-8628 *244-8649 250-8734 *314-10309

SUPD01 026464 217-7839 217-7861 217-7887 219-7912 228-8155 228-8180 230-8236 234-8356 234-8361
236-8378 238-8446 244-8631 #248-8671 314-10316

SUPD02 026500 219-7921 219-7943 219-7963 228-8164 228-8189 228-8198 230-8239 230-8243 230-8246
242-8569 #248-8673

SUPD03 026642 217-7836 217-7858 217-7884 219-7905 221-7972 221-7982 221-7989 223-7999 223-8013
223-8022 223-8031 225-8041 225-8049 225-8054 228-8151 228-8176 230-8222 231-8255
231-8264 232-8316 234-8339 234-8344 236-8374 236-8400 236-8429 238-8448 244-8623
244-8629 244-8650 #250-8705 314-10310

SUPD04 026656 219-7908 219-7937 219-7956 228-8160 228-8185 228-8194 230-8224 230-8227 230-8229
232-8301 232-8307 240-8507 242-8557 242-8567 244-8602 244-8654 #250-8706

SUPDR0 002152 #139-5046 *248-8675 248-8683 *250-8708 250-8716

SUPDR1 002154 #139-5047 248-8676 250-8709

SUPDR2 002156 #139-5048

SUPDR3 002160 #139-5049

SUPDR4 002162 #139-5050

SUPDR5 002164 #139-5051

SUPDR6 002166 #139-5052 248-8686 250-8719

SUPLIM 053714 #419-13110 503-16178 503-16179

SUPSTK = 000740 #111-4111 155-5575 248-8695 250-8732 463-14680 463-14686

SWAPAT 002574 #141-5218 *147-5360

SWR 002576 #141-5221 *151-5446 151-5448 *151-5453 *153-5472 155-5589 164-6154 170-6369 181-7015
184-7068 186-7118 214-7721 214-7736 227-8112 228-8144 266-9113 268-9210 272-9279
274-9387 276-9486 280-9588 316-10336 348-11089 348-11108 348-11144 375-11831 378-11898
382-11969 406-12704 411-12817 413-12839 447-14208 447-14215 447-14233 450-14284 450-14323
450-14331 451-14338 451-14342 455-14455 468-14919 468-14927 468-14949 471-15062

SWREG = 000176 #113-4399 151-5453 155-5589 468-14919

SW0 = 000001 #111-4152 164-6154 170-6369 214-7721 214-7736 227-8112 375-11831 378-11898 382-11969
406-12704

SW1 = 000002 #111-4151

SW10 = 002000 #111-4142 450-14284

SW11 = 004000 #111-4141 186-7118 228-8144 266-9113 268-9210 272-9279 274-9387 276-9486 280-9588

SW12 = 010000 #111-4140 348-11089

SW13 = 020000 #111-4139 348-11108 348-11144 450-14323

SW14 = 040000 #111-4138 447-14215

SW15 = 100000 #111-4137

SW2 = 000004 #111-4150

SW3 = 000010 #111-4149

SW4 = 000020 #111-4148 316-10336

SW5 = 000040 #111-4147 450-14331

SW6 = 000100 #111-4146 181-7015

SW7 = 000200 #111-4145 455-14455

SW8 = 000400 #111-4144 447-14208

SW9 = 001000 #111-4143 447-14233 451-14342

SYSSIZ 003720 145-5302 #145-5304

TAG2$ 011200 166-6183 #166-6210

TAG3$ 011234 166-6209 #166-6214

TAG4$ 026560 248-8684 #248-8686

TAG70$ 057132 455-14431 #457-14473

SYMBOL CROSS REFERENCE

SYMBOL	VALUE	REFERENCES
TAG71$	057142	455-14432 #457-14479
TAG72$	057152	455-14433 #457-14485
TAG73$	057222	455-14434 #457-14499
TAG74$	057262	455-14435 #457-14511
TAG75$	057274	455-14436 #457-14517
TAG76$	057306	455-14437 #458-14523
TAG77$	057352	455-14438 #458-14536
TAG78$	057360	455-14439 #458-14542
TAG79$	057440	455-14440 #458-14557
TAG9$	011026	166-6177 #166-6181 166-6216 166-6253 166-6275
TBG4$	026736	250-8717 #250-8719
TCFIG1	037254	#320-10427 320-10447 457-14490
TCFIG2	037406	#320-10456 320-10485 458-14528
TCFIG3	037574	#320-10494 320-10508 457-14503
TCONFI	037132	316-10341 316-10352 316-10362 #318-10386
TEMP	002404	#139-5124 *163-5974 163-6022 163-6024 *183-7022 183-7039 *183-7040 *380-11936 *380-11937 380-11938
TEST	006042	157-5642 157-5654 157-5657 158-5661 158-5667 #158-5709
TESTAD	002362	#139-5116 159-5736 *159-5737 *159-5738 163-5918 *163-5948 *163-5949 *163-6088 *163-6099 *163-6100 *208-7539 208-7540 *208-7540 *208-7543 *208-7545 *208-7564 208-7564 210-7614 210-7615 *216-7781 *216-7782 221-7970 221-7987 *227-8072 *227-8079 *227-8084 *227-8087 *227-8092 *227-8094 *227-8101 *227-8102 *227-8123 *227-8125 228-8141 228-8142 266-9050 266-9051 266-9137 266-9138 268-9160 268-9201 268-9218 270-9229 272-9310 274-9355 274-9361 274-9418 276-9454 276-9461 276-9465 276-9516 278-9554 278-9563 280-9618 300-9965 375-11776 *375-11824 *375-11825 *375-11829 *375-11881 443-14103
TESTMO	002522	#141-5192 *145-5321 *145-5342 166-6201 166-6218 166-6235 166-6261 171-6381 210-7616 240-8490 242-8538 244-8584 250-8727 329-10698 362-11533 371-11704 373-11744 373-11762 406-12669 406-12685 406-12695 408-12715 408-12727
TIME	002310	#139-5092
TIMEOU	040130	145-5348 147-5378 151-5452 161-5882 164-6151 166-6211 #324-10586 367-11649 367-11653 369-11665 369-11677 371-11719 373-11771
TKVEC	= 000060	#111-4182 316-10322 316-10322 *316-10324 *316-10325 *316-10367 *316-10367 375-11776 375-11776 *375-11811 *375-11812 *375-11879 *375-11879 378-11894 378-11894 *378-11906 *378-11907 *378-11928 *378-11928 382-11965 382-11965 *382-11977 *382-11978 *382-11998 *382-11998
TMFLAG	002132	#139-5037 *198-7334 *200-7374 *202-7417 *204-7457 358-11440
TOOMAN	002356	#139-5114 *443-14098 450-14331 *451-14366
TOTCSR	002216	#139-5065 *158-5706 159-5741 160-5794 160-5821 163-5919 333-10743 335-10778 339-10834 391-12099 411-12791 413-12858 463-14647
TRACE	006136	#158-5729 *161-5865 *161-5871 161-5873 *161-5877 *161-5881 *387-12088 *389-12094 447-14190
TRAPVE	= 000034	#111-4181 *147-5370 *147-5371
TSTBAN	011446	166-6193 #166-6256
TSTDAT	002240	#139-5074 *266-9033 *266-9034 *266-9037 *266-9038 266-9039 266-9040 266-9041 *266-9047 *266-9049 266-9053 266-9055 266-9061 266-9067 *266-9126 *266-9127 *268-9152 *268-9153 268-9154 *268-9157 *268-9159 268-9163 268-9166 *270-9236 *270-9237 *270-9240 *270-9241 270-9247 *270-9250 *270-9252 *270-9254 *270-9256 270-9258 270-9260 *272-9291 *272-9292 *274-9334 *274-9335 *274-9338 *274-9339 274-9345 *274-9348 *274-9350 *274-9352 *274-9354 274-9357 274-9359 274-9365 274-9372 *274-9399 *274-9400 *276-9433 *276-9434 *276-9437 *276-9438 276-9444 *276-9447 *276-9449 *276-9451 *276-9453 276-9456 276-9459 276-9467 276-9473 *276-9497 *276-9498 *278-9534 *278-9535 *278-9538 *278-9539 278-9544 *278-9547 *278-9549 *278-9551 *278-9553 278-9556 278-9558 278-9565 278-9572 *280-9599 *280-9600 *300-9960 *300-9961 300-9962 *300-9978 *300-9980 *300-9982 *300-9983 *300-9985 *300-9988 *300-9989 *300-9991 *300-9993 *300-9994 *300-9996 302-10022 302-10024 438-13999 438-14004

SYMBOL CROSS REFERENCE

SYMBOL	VALUE	REFERENCES								
		443-14104	443-14106	495-15866	495-15866					
TSTRD1	040602	329-10684	329-10686	#329-10697						
TSTREA	= 104510	#110-4102	210-7632	210-7657	266-9087	266-9100	268-9176	268-9188		
TST1	005434	#157-5609								
TST2	010542	#164-6124								
TST3	010726	#166-6161								
TST4	011714	#169-6309								
TST5	014210	#184-7066								
TST6	014262	#184-7083								
TYPDS	= 104405	#110-4026	181-6995	181-6999	181-7003	181-7007	181-7011	181-7013	186-7122	380-11938
		380-11940	380-11955	457-14479						
TYPEIT	= 104401	#110-4022	154-5488	154-5501	154-5503	154-5505	154-5512	154-5517	154-5518	160-5854
		179-6855	179-6938	181-6992	181-6996	181-7000	181-7004	181-7008	181-7012	181-7014
		186-7117	186-7119	208-7568	208-7570	316-10330	316-10331	316-10332	316-10340	316-10344
		316-10345	316-10346	316-10347	316-10348	316-10349	316-10358	316-10359	316-10360	316-10361
		318-10386	318-10389	318-10391	318-10400	318-10415	320-10425	320-10444	320-10455	320-10482
		320-10493	320-10505	320-10515	320-10523	320-10525	348-11109	348-11139	348-11147	365-11572
		365-11575	365-11585	365-11590	367-11621	367-11651	369-11662	369-11666	369-11671	369-11675
		371-11684	371-11685	371-11710	371-11714	373-11726	373-11727	373-11750	373-11754	373-11757
		373-11759	373-11766	375-11778	375-11779	375-11788	375-11792	375-11797	375-11807	375-11810
		378-11896	378-11903	378-11905	378-11922	380-11939	380-11941	380-11949	380-11956	382-11967
		382-11974	382-11976	382-11992	384-12007	384-12016	384-12021	384-12027	384-12034	385-12053
		385-12057	385-12068	385-12073	387-12087	389-12093	391-12105	414-12905	419-13028	450-14286
		450-14287	451-14357	453-14385	453-14409	453-14410	453-14412	453-14419	453-14421	455-14448
		455-14458	457-14489	457-14493	458-14527	458-14536	458-14551	458-14557	458-14559	463-14645
		463-14646	463-14654	463-14663	463-14667	463-14669	463-14679	463-14682	463-14684	463-14688
		463-14694	463-14697	463-14699	463-14703	463-14705	466-14806	467-14871	468-14916	468-14925
		468-14926	468-14928	468-14937	468-14942	468-14951	468-14969	470-14978	470-14989	470-14991
		470-14993	471-15059	471-15072	471-15078	471-15083	471-15087	471-15092	471-15093	471-15095
		471-15098	471-15102	472-15163	472-15165	472-15166	472-15167	472-15222	472-15224	472-15225
		472-15226	479-15444							
TYPOC	= 104402	#110-4023	453-14390	457-14473	463-14653	463-14668	463-14670	463-14683	463-14685	463-14698
		463-14700	468-14927							
TYPOS	= 104403	#110-4024	179-6856	208-7569	371-11707	373-11758	373-11767	380-11952	457-14511	457-14517
		458-14558	470-14992	470-14994						
T12A	033246	#278-9528	280-9597							
T12B	033270	#278-9532	280-9593							
UDPAR0	= 177660	#111-4238	232-8300							
UDPAR7	= 177676	#111-4245	*232-8298							
UIPAR0	= 177640	111-4227	#111-4228	346-10997	354-11301					
UIPAR1	= 177642	#111-4229	*219-7916	*234-8354	*234-8359					
UIPAR2	= 177644	#111-4230	166-6179	219-7918	234-8355	*314-10315				
UIPAR3	= 177646	#111-4231	166-6180	236-8377	244-8617	340-10863	346-11014			
UIPAR4	= 177650	#111-4232	228-8163	228-8179	228-8197	352-11263	352-11280			
UIPAR5	= 177652	#111-4233	217-7850	217-7874	*219-7941	*219-7961	240-8504	242-8531	244-8598	346-11039
		346-11060								
UIPAR6	= 177654	#111-4234	217-7850	217-7874	219-7942	219-7962	228-8154	228-8188	240-8504	242-8531
		244-8598	*346-11060							
UIPDR0	= 177600	#111-4207	346-10998	354-11303	410-12772	413-12872				
UNITOP	002366	#139-5117	*181-6984	*181-6987	*181-6988	*181-6989	*181-6990	181-6991	181-6991	
UNRELO	043416	190-7205	192-7235	194-7272	196-7309	198-7347	200-7392	202-7430	204-7475	238-8475
		#350-11194	360-11475	362-11515						

SYMBOL CROSS REFERENCE CREF V01

SYMBOL	VALUE	REFERENCES
UPPFLG	002257	#139-5079 *274-9360 274-9377 *274-9379 *278-9559 280-9578 *280-9580
USERMA	044116	348-11157 350-11216 352-11258 #354-11300
USESTK	= 000700	#111-4112 155-5581 250-8730 463-14695 463-14701
USP	=%000006	#111-4133 *155-5582 *250-8730 411-12781 *414-12888 463-14692
WARN1	011114	#166-6194
WARN2	027156	#254-8801 *313-10294
WARN3	027172	#254-8806 *313-10295
WARN4	027216	#254-8818 *313-10296
WARN5	027232	#254-8823 *313-10297
WARN6	036612	#314-10315
WARN6A	036552	#314-10308
WARN6B	036604	314-10307 #314-10314
WARN7	024270	#236-8377
WASDBE	= 104500	#110-4094
WASSBE	= 104476	#110-4092
WAS1DB	= 104501	#110-4095 171-6391 171-6404 171-6414 171-6424 270-9264
WAS1SB	= 104477	#110-4093
WHICHC	051102	367-11643 369-11658 #391-12098
WOOPEN	053112	415-12933 415-12936 415-12938 417-12954 417-12977 #417-12985 417-12988
WOOPS	052544	410-12769 #415-12926
WOOPSA	053142	*415-12931 *415-12932 415-12933 417-12972 417-12973 417-12976 #417-12988
WOOPUP	052730	415-12933 415-12933 415-12934 415-12936 415-12936 415-12936 #417-12953 417-12977 417-12978 417-12988
WORST	002540	#141-5199 *147-5352 *194-7260 *196-7297 *200-7380 *204-7463 356-11417
XOCHAR	053504	#419-13042 *419-13076 *419-13079 *419-13080 419-13081 *419-13085 *419-13086 419-13087 468-14894 468-14896 *468-14897
XXDPCH	002324	#139-5098 *153-5479 451-14356
ZEROS	002306	#139-5091 145-5312 292-9810 292-9814
$APTHD	063126	137-4967 #478-15413
$AUTO	002060	#139-5016 *153-5470 *153-5475 155-5588 468-14923
$BANK	002011	#139-4996 *448-14246
$BASE	063102	#477-15385
$BELL	002613	#141-5228 378-11922 382-11992 450-14286
$CACHF	040272	#326-10634 481-15481
$CACHN	040246	#326-10627 481-15480
$CBCSR	040732	#331-10730 481-15524
$CB1CS	040754	#331-10735 481-15525
$CDW1	063106	#477-15388
$CDW2	063110	#477-15389
$CHARC	053672	*419-13030 *419-13040 *419-13098 #419-13103
$CHKDI	041326	#337-10822 481-15532
$CHK1D	041342	#337-10827 481-15533
$CKSWR	060724	#468-14888 481-15464
$CLRCS	041304	#337-10813 481-15530
$CLR1C	041316	#337-10817 481-15531
$CMTAG	002000	#139-4989 144-5276
$CMTGE	002514	#139-5186 144-5278
$CNTLC	062116	468-14937 471-15059 #471-15114
$CNTLG	062130	468-14925 #471-15116
$CNTLK	061324	468-14916 #468-14971
$CNTLU	062123	468-14942 471-15087 #471-15115
$CPUOP	063054	#477-15358

SYMBOL	VALUE	REFERENCES
$CRLF	002620	#141-5230 181-6992 316-10344 316-10347 380-11956 419-13029 453-14385 453-14412 453-14421 463-14646 463-14667 463-14682 463-14697 463-14705 468-14951 470-14978 471-15092
$DBLK	060714	467-14841 467-14871 #467-14879
$DB20	062706	458-14549 #476-15305
$DDW0	063112	149-5394 #478-15400
$DDW1	063114	#478-15401
$DDW2	063116	#478-15402
$DDW3	063120	#478-15403
$DDW4	063122	#478-15404
$DDW5	063124	#478-15405
$DEENE	040236	#326-10623 481-15476
$DEVCT	063036	*186-7153 *447-14184 *447-14186 #477-15345
$DEVM	063104	#477-15386
$DOAGA	014604	186-7124 186-7126 #186-7160
$DOAGN	014500	#186-7136
$DOWN	052134	#411-12822 411-12823
$DTBL	060704	467-14844 #467-14875
$ECCDI	040630	#331-10706 481-15520
$ECCIN	040656	#331-10714 481-15522
$ECC1D	040644	#331-10710 481-15521
$ECC1I	040672	#331-10718 481-15523
$ENASB	040704	#331-10722 481-15534
$ENA1S	040720	#331-10726 481-15535
$ENDAD	014470	137-4958 153-5476 154-5487 #186-7132
$ENERG	040226	#326-10619 481-15475
$ENV	063046	144-5267 153-5469 #477-15350
$ENVM	063047	153-5463 153-5466 #477-15354
$EOP	014334	#186-7108
$ERFLG	002012	#139-4997 447-14231 *447-14237 *450-14280
$ERRGE	041452	#340-10856 481-15538
$ERROR	055730	147-5368 #450-14276
$ERRTB	063450	453-14400 #485-15565
$ERRTY	056512	450-14335 #453-14384
$ERTTL	002570	#141-5211 186-7114 380-11940 380-11942 *450-14288 *450-14290 451-14356
$ESCAP	002332	#139-5104 *447-14240 451-14345 451-14347
$ETABL	063046	#477-15349
$ETEND	063126	#478-15409 478-15419
$EXHAL	045160	#362-11500
$FATAL	063030	*450-14322 #477-15342
$FILLC	002612	#141-5227 419-13033
$FILLS	002327	#139-5101 *384-12011
$GTSWR	061100	#468-14926 481-15463
$HALT	056274	#451-14340
$HALT2	063202	#479-15445
$HIBTS	063126	#478-15414
$HIOCT	062326	371-11687 373-11729 *472-15158 #472-15169
$ILLUP	052536	410-12754 413-12832 #414-12911 414-12912
$INVAL	041422	#339-10847 481-15537
$ITEMB	002013	#139-4998 *450-14297 450-14299 450-14309 *450-14310 453-14387
$KERNE	040216	#326-10615 481-15474
$KMAP	042434	#346-11066 481-15478
$LF	002621	#141-5231 471-15102

SYMBOL CROSS REFERENCE

SYMBOL		VALUE	REFERENCES
$LOADC		040310	#328-10644 481-15483
$LPADR		002562	#141-5208 248-8674 *248-8684 248-8685 *248-8701 250-8707 *250-8717 250-8718 *250-8738 *447-14235 *447-14238 447-14242
$LPERR		002564	#141-5209 248-8674 *248-8685 *248-8701 250-8707 *250-8718 *250-8738 447-14235 *447-14239 451-14343
$MADR1		063060	#477-15372
$MADR2		063064	#477-15376
$MADR3		063070	#477-15379
$MADR4		063074	#477-15382
$MAIL		063026	#477-15340 478-15415 478-15419
$MAMS1		063056	183-7023 #477-15365
$MAMS2		063062	#477-15374
$MAMS3		063066	#477-15377
$MAMS4		063072	#477-15380
$MBADR		063130	#478-15415
$MNEW		062146	468-14928 #471-15118
$MSGAD		063042	#477-15347
$MSGLG		063044	#477-15348
$MSGTY		063026	*451-14353 #477-15341
$MSWR		062135	468-14926 #471-15117
$MTYP1		063057	#477-15366
$MTYP2		063063	#477-15375
$MTYP3		063067	#477-15378
$MTYP4		063073	#477-15381
$NOTRA		063176	#479-15444 481-15459 481-15461 481-15514 481-15515 481-15516 481-15517 481-15518 481-15539 481-15540 481-15541 481-15542 481-15543 481-15544 481-15545
$NULL		002326	#139-5100 419-13035
$NWTST	=	000001	#157-5609 157-5609 #157-5609 #164-6124 164-6124 #164-6124 #166-6161 166-6161 #166-6161 #169-6309 169-6309 #169-6309 #184-7066 184-7066 #184-7066 #184-7083 184-7083 #184-7083
$OCNT		060474	*466-14778 *466-14807 #466-14820
$OCTVL		063010	476-15307 #476-15332 476-15335
$OCT8	=	063014	458-14551 #476-15335
$OMODE		060476	*466-14773 *466-14777 466-14782 *466-14785 *466-14796 #466-14822
$OVER		055644	447-14215 447-14236 #447-14241
$PASS		063034	*153-5462 *186-7115 *186-7116 186-7122 186-7150 *186-7156 223-7995 223-8003 223-8017 223-8026 225-8037 225-8045 227-8060 232-8312 348-11091 380-11936 #477-15344
$PASTM		063134	#478-15417
$PATMA		002010	#139-4995 414-12901 414-12902 *448-14247 *448-14251 448-14256 448-14257 450-14283
$PER01		053714	#436-13932 481-15486
$PER02		053742	#436-13938 481-15487
$PER03		053770	#436-13944 481-15488
$PER04		054020	#436-13950 481-15489
$PER07		054102	#436-13966 481-15490
$PER10		054124	#436-13971 481-15491
$PER11		054154	#436-13976 481-15492
$PER12		054174	#436-13981 481-15493
$PER13		054216	#438-13987 481-15494
$PER14		054236	#438-13992 481-15495
$PER15		054260	#438-13997 481-15496
$PER16		054302	#438-14002 481-15497
$PER17		054322	#438-14007 481-15498
$PER20		054340	#438-14012 481-15499

SYMBOL		VALUE	REFERENCES								
$PER21		054356	#438-14017	481-15500							
$PER22		054376	#438-14022	481-15501							
$PER23		054414	#438-14027	481-15502							
$PER24		054432	#438-14032	481-15503							
$PER25		051170	#406-12667	481-15504							
$PER26		054622	#441-14069	481-15505							
$PER27		054642	#441-14073	481-15506							
$PER30		051416	#408-12712	481-15507							
$PER31		055032	#443-14112	481-15508							
$PER32		055130	#445-14130	481-15509							
$PER33		055176	#445-14139	481-15510							
$PER34		055256	#445-14150	481-15511							
$PER35		055310	#445-14159	481-15512							
$PER36		055344	#445-14167	481-15513							
$PWRDN		051564	147-5372	#410-12741	414-12904						
$PWRUP		052140	411-12821	#413-12828	417-12984						
$QUES		002617	#141-5229	468-14969	471-15095						
$RAND		062612	#474-15276								
$RDCHR		061444	#471-15021	481-15466							
$RDDEC		062330	#472-15184	481-15469							
$RDLIN		061574	#471-15050	481-15467							
$RDOCT		062160	#472-15133	481-15468							
$READC		040404	#328-10662	481-15484							
$RESRE		062554	#473-15257	481-15472							
$SAVRE		062516	#473-15246	481-15471							
$SAVR6		052542	*411-12819	413-12834	*413-12835	*413-12836	#414-12913	417-12967			
$SCOPE		055374	147-5366	#447-14184							
$STN	=	000001	#157-5609	#164-6124	#166-6161	#169-6309	#184-7066	#184-7083			
$SVLAD		055630	447-14223	447-14232	#447-14238						
$SWR	=	163000	#108-3986	157-5609	164-6124	166-6161	169-6309	184-7066	184-7083		
$SWREG		063050	153-5472	#477-15356							
$TESTN		063032	*450-14283	455-14465	#477-15343						
$TKB		002604	#141-5224	316-10326	316-10366	375-11813	375-11880	378-11908	378-11927	382-11979	382-11997
			419-13079	419-13085	468-14901	468-14933	470-15005	471-15025	471-15031		
$TKS		002602	#141-5223	316-10328	316-10365	375-11815	375-11878	378-11910	378-11926	382-11981	382-11996
			419-13077	419-13083	468-14899	468-14931	470-15003	471-15023	471-15029		
$TN	=	000007	#108-3987	157-5609	157-5609	#157-5609	164-6124	164-6124	#164-6124	166-6161	166-6161
			#166-6161	169-6309	169-6309	#169-6309	184-7066	184-7066	#184-7066	184-7083	184-7083
			#184-7083								
$TPB		002610	#141-5226	419-13092							
$TPFLG		002330	#139-5102	*153-5464	419-13012						
$TPS		002606	#141-5225	419-13074							
$TRAP		063142	147-5370	#479-15429							
$TRAP2		063164	#479-15440	481-15455							
$TRPAD		063204	479-15434	#481-15455							
$TSTM		063132	#478-15416								
$TSTRD		040424	#329-10669	481-15536							
$TTYIN		062072	471-15052	471-15053	471-15075	471-15093	471-15107	#471-15111			
$TYPDS		060500	#467-14834	481-15460							
$TYPE		053360	#419-13012	481-15456							
$TYPEC		053506	419-13032	419-13039	#419-13043	468-14955					
$TYPEX		053674	419-13099	419-13101	#419-13104						

SYMBOL CROSS REFERENCE CREF V01

SYMBOL	VALUE	REFERENCES
$TYPOC	060276	#466-14776 481-15457
$TYPON	060312	466-14775 #466-14778
$TYPOS	060252	#466-14771 481-15458
$UNIT	063040	*186-7154 *447-14187 #477-15346
$UNITM	063136	#478-15418
$USWR	063052	186-7150 #477-15357
$VECT1	063076	#477-15383
$VECT2	063100	#477-15384
$WASDB	041140	#335-10777 481-15528
$WASSB	040774	#333-10742 481-15526
$WAS1D	041254	#335-10803 481-15529
$WAS1S	041110	#333-10768 481-15[illegible]
$XTSTR	055522	#447-14217
$ZAP42	014450	#186-7125 451-14360
$OFILL	060475	*466-14772 *466-14776 466-14786 #466-14821

MACRO CROSS REFERENCE CREF V01

MACRO NAME	REFERENCES
AND	#6-4500 #108-3976
ANDB	#18-600 #108-3979
BEGIN	#12-6300 #108-3977 179-6833 206-7485 208-7534 227-8069 333-10745 335-10780 339-10835 348-11093 348-11113 385-12055 411-12792 413-12860 463-14648
BMOV	#126-4717 #166-6178 #217-7838 #217-7860 #217-7886 #219-7910 #219-7913 #219-7914 #219-7915 #219-7939 #219-7940 #219-7959 #219-7960 #228-8153 #228-8162 #228-8178 #228-8187 #228-8196 #230-8235 #230-8238 #230-8242 #230-8245 #232-8294 #232-8295 #232-8296 #234-8350 #234-8351 #234-8353 #234-8358 #236-8376 #238-8445 #240-8491 #242-8547 #244-8585 #244-8616 #314-10314 #329-10686 #348-11159 #350-11218 #352-11277 #352-11293 #415-12933 #415-12936
CASE	#9-100 #108-3976 212-7684 365-11592
CLEAR	#19-100 #108-3974 #181-6944 #210-7619 #210-7626 #210-7674 #238-8468 #238-8479 #266-9139 #272-9311 #274-9419 #276-9517 #313-10287 #316-10339 #316-10357 #337-10813 #337-10817 #348-11151 #350-11211 #350-11228 #356-11368 #356-11370 #356-11371 #451-14366
CLEARB	#19-1300 #108-3974
DLEFT	#135-4919 #260-8903 #260-8922 #266-9111 #266-9117 #268-9208 #268-9214 #272-9277 #272-9283 #274-9385 #274-9391 #276-9484 #276-9490 #280-9586 #280-9592
DOWNTO	#10-3800 #108-3977
ELSE	#7-100 #108-3976 #153-5473 #153-5478 #164-6156 #167-6297 #167-6302 #169-6339 #170-6371 #179-6852 #181-6954 #181-6957 #181-6960 #181-6964 #183-7030 #214-7723 #214-7738 #223-8009 #227-8088 #227-8106 #227-8114 #244-8624 #246-8666 #292-9812 #292-9828 #293-9855 #329-10692 #333-10762 #335-10797 #362-11499 #367-11626 #367-11634 #375-11833 #378-11900 #382-11971 #406-12675 #406-12706 #419-13089 #439-14044 #439-14056 #443-14105 #445-14153 #451-14363 #463-14687 #463-14702 #470-15009
END	#12-100 #108-3977 #149-5423 #149-5425 #151-5438 #151-5455 #153-5465 #153-5468 #153-5480 #153-5481 #153-5482 #154-5489 #155-5591 #155-5592 #164-6158 #166-6271 #166-6274 #167-6300 #167-6304 #169-6341 #169-6350 #170-6361 #170-6362 #170-6363 #170-6364 #170-6365 #170-6366 #170-6373 #171-6425 #171-6426 #171-6427 #171-6430 #179-6844 #179-6847 #179-6854 #179-6858 #179-6859 #179-6860 #179-6861 #181-6956 #181-6962 #181-6963 #181-6968 #181-6969 #181-6970 #181-6971 #181-6972 #181-6981 #181-6985 #181-6986 #181-7017 #183-7032 #183-7043 #183-7046 #183-7048 #183-7049 #183-7052 #183-7064 #184-7091 #186-7121 #186-7158 #186-7159 #188-7172 #188-7173 #206-7489 #206-7490 #208-7559 #208-7563 #208-7564 #208-7565 #208-7566 #208-7571 #208-7572 #208-7574 #210-7640 #210-7665 #212-7713 #214-7725 #214-7732 #214-7740 #216-7787 #223-7996 #223-8004 #223-8011 #223-8018 #223-8027 #225-8038 #225-8046 #227-8061 #227-8075 #227-8076 #227-8082 #227-8083 #227-8086 #227-8098 #227-8108 #227-8116 #227-8126 #227-8130 #228-8146 #232-8313 #236-8395 #236-8401 #236-8402 #236-8403 #236-8404 #236-8408 #236-8424 #236-8430 #236-8431 #236-8432 #236-8433 #238-8459 #238-8460 #238-8461 #238-8470 #238-8477 #244-8633 #246-8668 #266-9076 #266-9077 #292-9815 #292-9831 #292-9834 #293-9850 #293-9854 #293-9859 #293-9863 #293-9864 #293-9867 #293-9870 #293-9871 #293-9878 #293-9883 #312-10261 #329-10694 #333-10753 #333-10754 #333-10756 #333-10757 #333-10764 #335-10788 #335-10789 #335-10791 #335-10792 #335-10799 #339-10840 #339-10842 #339-10843 #348-11092 #348-11106 #348-11107 #348-11110 #348-11112 #348-11125 #348-11129 #348-11130 #348-11131 #348-11132 #348-11133 #348-11141 #348-11142 #348-11148 #348-11150 #348-11168 #356-11385 #356-11422 #362-11502 #362-11518 #365-11578 #365-11616 #367-11628 #367-11638 #375-11790 #375-11800 #375-11809 #375-11830 #375-11837 #378-11902 #378-11920 #378-11921 #380-11951 #380-11957 #380-11958 #380-11959 #382-11973 #382-11990 #382-11991 #385-12062 #385-12066 #385-12067 #385-12082 #406-12672 #406-12677 #406-12708 #408-12718 #411-12798 #411-12800 #411-12801 #413-12866 #413-12868 #413-12869 #419-13091 #439-14046 #439-14058 #443-14107 #443-14116 #443-14119 #443-14122 #443-14125 #445-14155 #447-14188 #447-14211 #447-14214 #450-14291 #450-14292 #450-14321 #450-14325 #450-14333 #450-14334 #451-14344 #451-14355 #451-14361 #451-14362 #451-14365 #455-14457 #463-14655 #463-14657 #463-14658 #463-14671 #463-14686 #463-14689 #463-14701 #463-14704 #470-14990 #470-15011
ENDPRO	#18-5000
FATAL	#115-4421 #164-6145 #164-6149 #300-9977 #322-10562 #324-10587 #324-10591 #324-10594
FI	#18-3600
FOR	#10-100 #108-3977 149-5419 169-6329 179-6801 181-6948 181-6975 181-6982 183-7024 188-7165 208-7530 208-7531 208-7539 208-7550 227-8118 236-8380 236-8381 236-8409 236-8410 238-8452 292-9808 292-9816 293-9876 312-10237 #333-10746 #335-10781 #339-10836 #348-11094 348-11117 378-11915

MACRO CROSS REFERENCE CREF V01

MACRO NAME	REFERENCES									
	380-11944	382-11985	385-12080	#411-12793	#413-12861	#463-14649	463-14666	463-14681	463-14696	
GOTO	#13-2000	#108-3978	#228-8145	#375-11789	#375-11808	#378-11923	#382-11993	#450-14332		
IF	#4-100	#108-3976	151-5433	151-5448	#153-5474	153-5476	154-5487	155-5588	155-5589	164-6154
	166-6268	166-6272	167-6293	167-6294	169-6333	169-6334	169-6335	169-6336	169-6346	169-6348
	170-6353	170-6369	179-6826	179-6831	#179-6834	179-6840	179-6845	#181-6949	181-6950	181-6951
	181-6952	#181-6958	#181-6965	181-6966	#181-6983	181-6991	181-7015	183-7021	183-7028	183-7050
	184-7086	186-7118	186-7149	186-7150	188-7167	#206-7486	206-7487	208-7503	#208-7504	208-7536
	208-7567	210-7633	210-7639	210-7658	210-7664	214-7721	214-7730	214-7736	216-7785	219-7892
	223-7994	223-7995	223-8002	223-8003	223-8007	223-8016	223-8017	223-8025	223-8026	223-8028
	225-8036	225-8037	225-8044	225-8045	227-8059	227-8060	227-8068	#227-8070	227-8071	227-8077
	227-8078	227-8100	227-8112	227-8122	228-8144	232-8311	232-8312	236-8383	236-8390	236-8396
	236-8405	236-8412	236-8419	236-8425	238-8454	238-8455	238-8462	244-8618	244-8651	246-8664
	266-9072	266-9074	266-9113	266-9114	268-9210	268-9211	272-9279	272-9280	274-9320	274-9387
	274-9388	276-9486	276-9487	280-9588	280-9589	292-9809	#292-9823	#292-9824	#292-9825	292-9837
	#293-9844	#293-9845	#293-9846	293-9848	293-9852	293-9857	293-9861	316-10336	320-10445	320-10483
	320-10506	328-10646	329-10690	333-10750	333-10755	335-10785	335-10790	339-10841	348-11089	#348-11090
	348-11091	348-11096	348-11108	#348-11118	348-11119	348-11120	348-11121	348-11122	348-11126	348-11134
	348-11135	348-11144	356-11383	356-11420	360-11475	362-11497	362-11515	#362-11516	365-11574	367-11623
	367-11632	373-11736	373-11738	373-11739	375-11782	375-11787	375-11795	375-11796	375-11806	375-11826
	375-11831	378-11898	378-11917	380-11942	380-11948	382-11969	382-11987	385-12060	406-12668	406-12673
	406-12681	406-12704	408-12714	410-12769	411-12799	413-12867	419-13087	439-14039	#439-14040	439-14042
	439-14051	#439-14052	439-14054	441-14063	443-14103	443-14112	443-14114	443-14117	443-14120	443-14123
	445-14130	445-14139	445-14150	#445-14151	447-14185	447-14208	447-14212	447-14215	450-14277	450-14289
	450-14314	450-14315	450-14323	450-14331	451-14337	451-14342	451-14348	451-14352	451-14356	455-14455
	463-14656	463-14680	463-14695	470-14988	470-15007					
IFB	#18-1100	#108-3979	#153-5463	#153-5466	#153-5469	#183-7025	#183-7041	#183-7044	#380-11947	
JUMPTO	#13-1500	#108-3978								
LEAVE	#13-100	#108-3978	#179-6843	#179-6846	#179-6853	#208-7562	#227-8074	#227-8081	#333-10752	#335-10787
	#348-11105	#348-11124	#385-12061							
LET	#16-100	#108-3978	#181-6953	#181-6955	#181-6959	#181-6961	#181-6967	#181-6984	#208-7553	#236-8385
	#236-8413	#292-9810	#292-9811	#292-9813	#292-9814	#292-9820	#292-9821	#292-9826	#292-9827	#292-9829
	#292-9830	#292-9832	#292-9833	#292-9839	#292-9840	#293-9847	#293-9851	#293-9856	#293-9860	#293-9865
	#293-9866	#293-9879	#293-9882	#348-11155	#350-11215	#406-12667	#406-12670	#406-12674	#406-12676	#406-12693
	#408-12712	#408-12713	#408-12715	#436-13971	#436-13972	#436-13973	#436-13976	#436-13977	#436-13978	#436-13981
	#436-13982	#436-13983	#438-13987	#438-13988	#438-13989	#438-13992	#438-13993	#438-13994	#438-13997	#438-13998
	#438-13999	#438-14002	#438-14003	#438-14004	#438-14007	#438-14008	#438-14009	#438-14012	#438-14013	#438-14014
	#438-14017	#438-14018	#438-14019	#438-14022	#438-14023	#438-14024	#438-14027	#438-14028	#438-14029	#438-14032
	#438-14033	#438-14034	#441-14069	#441-14070	#441-14074					
MAP	#128-4780	155-5605	166-6191	170-6367	171-6378	#208-7532	240-8489	242-8518	244-8582	248-8672
	250-8705	362-11532	371-11703	373-11742	375-11803	375-11828	375-11883	378-11929	382-11999	415-12929
	417-12970									
NEWTST	#117-4471	157-5609	164-6124	166-6161	169-6309	184-7066	184-7083			
ON.ERR	#18-3100	#108-3979	#171-6428	#190-7202	#192-7232	#194-7269	#196-7306	#198-7344	#200-7389	#202-7427
	#204-7472	#231-8252	#231-8260	#232-8270	#238-8466	#240-8485	#242-8514	#244-8578	#333-10748	#333-10760
	#335-10783	#335-10795	#339-10838	#348-11166	#411-12795	#413-12863	#463-14651			
ON.NOE	#18-2600	#108-3979	149-5421	171-6395	171-6405	171-6415	208-7547			
OR	#6-100	#108-3976								
ORB	#18-100	#108-3979								
POP	#14-900	#108-3978	155-5605	163-5976	163-6067	166-6191	166-6267	170-6367	171-6378	171-6441
	171-6443	179-6894	186-7130	208-7532	208-7558	208-7584	210-7670	210-7678	227-8129	227-8131
	240-8489	242-8518	244-8582	248-8672	248-8701	250-8705	250-8738	313-10298	316-10367	328-10657
	328-10665	329-10689	333-10759	335-10794	339-10844	339-10853	340-10891	340-10907	342-10938	346-11063

CZMSDC CREATED BY MACRO ON 22-APR-81 AT 15:24 PAGE 30
SEQUENCE 395

MACRO CROSS REFERENCE CREF VG1

MACRO NAME	REFERENCES									
	346-11084	350-11210	350-11236	350-11248	354-11330	354-11341	356-11432	362-11532	364-11564	365-11587
	#367-11627	367-11629	367-11637	369-11668	371-11688	371-11703	371-11718	373-11730	373-11742	373-11761
	373-11770	375-11781	375-11794	375-11799	375-11803	375-11828	375-11836	375-11879	375-11881	375-11882
	375-11883	378-11928	378-11929	380-11960	382-11998	382-11999	384-12009	384-12012	385-12059	391-12107
	406-12703	408-12732	413-12839	413-12843	413-12849	413-12855	413-12856	413-12864	414-12882	414-12886
	414-12893	414-12898	414-12903	414-12909	415-12929	415-12949	417-12970	417-12983	419-13104	419-13107
	419-13108	441-14082	443-14099	448-14258	457-14492	457-14505	458-14530	461-14634	463-14659	463-14678
	463-14693	463-14707	467-14870	470-14998	470-15002	472-15159	472-15217	473-15262	474-15291	
PROCED	#18-4100									
PUSH	#14-100	#108-3978	#155-5575	#155-5581	#155-5605	#163-5956	#163-6053	#166-6191	#166-6256	#170-6367
	#171-6378	#171-6382	#171-6384	#179-6862	#186-7128	#208-7505	#208-7532	#208-7554	#210-7613	#227-8117
	#227-8119	#240-8489	#242-8518	#244-8582	#248-8672	#248-8674	#250-8705	#250-8707	#313-10275	#316-10322
	#328-10644	#328-10662	#329-10669	#333-10742	#335-10777	#339-10833	#339-10848	#340-10857	#340-10882	#342-10919
	#346-10991	#346-11066	#350-11196	#350-11201	#350-11240	#354-11321	#354-11335	#356-11367	#362-11532	#364-11548
	#364-11553	#364-11557	#365-11581	#365-11582	#371-11682	#371-11703	#373-11724	#373-11742	#375-11776	#375-11803
	#375-11828	#375-11834	#375-11883	#378-11894	#378-11929	#380-11935	#382-11965	#382-11999	#384-12006	#406-12682
	#408-12722	#410-12752	#410-12756	#410-12760	#410-12767	#411-12783	#411-12789	#411-12797	#411-12803	#411-12806
	#411-12810	#411-12815	#411-12817	#415-12927	#415-12929	#417-12968	#417-12970	#419-13043	#419-13047	#419-13048
	#441-14078	#443-14088	#448-14248	#457-14485	#457-14499	#458-14523	#461-14610	#461-14626	#463-14644	#463-14662
	#467-14834	#470-14977	#472-15135	#472-15186	#473-15247	#474-15276				
RCC	#3-4000	#108-3975								
RCS	#3-3700	#108-3975								
REPEAT	#11-100	#108-3977	#385-12056							
REQ	#3-400	#108-3975								
RES4	#133-4892	145-5348	151-5452	161-5882	164-6151	166-6211	367-11649	367-11653	369-11665	369-11677
	371-11719	373-11771								
RGE	#3-1000	#108-3975								
RGT	#3-1300	#108-3975								
RHI	#3-2500	#108-3975								
RHIS	#3-3100	#108-3975								
RLE	#3-1600	#108-3975								
RLO	#3-3400	#108-3975								
RLOS	#3-2800	#108-3975								
RLT	#3-700	#108-3975								
RMI	#3-2200	#108-3975								
RNE	#3-100	#108-3975								
RPL	#3-1900	#108-3975								
SET	#19-700	#108-3974	137-4977	147-5352	147-5354	153-5464	153-5467	153-5470	153-5475	153-5477
	#153-5479	163-6030	163-6076	166-6251	166-6270	177-6790	179-6857	179-6939	184-7087	184-7089
	188-7168	188-7171	206-7480	206-7481	206-7495	208-7551	208-7561	208-7582	210-7634	210-7659
	230-8204	231-8263	232-8271	236-8379	238-8440	238-8451	238-8463	240-8486	242-8515	244-8579
	246-8660	266-9089	266-9092	266-9102	266-9105	268-9178	268-9181	268-9192	268-9195	270-9266
	270-9269	302-10036	302-10048	322-10552	333-10751	335-10786	348-11127	348-11190	356-11369	356-11378
	356-11397	356-11401	356-11404	356-11407	356-11410	356-11416	356-11421	356-11425	356-11428	356-11431
	365-11584	367-11647	369-11663	369-11672	375-11818	378-11913	380-11950	382-11983	384-12017	384-12023
	385-12069	443-14098	443-14109	443-14126	445-14134	445-14136	445-14145	445-14147	445-14169	445-14171
	#451-14364	457-14488	457-14502	458-14526	461-14611	461-14623	461-14627	461-14629		
SETB	#19-1900	#108-3974								
SET4	#133-4880	#145-5304	#145-5306	#145-5313	#145-5332	#145-5344	#151-5445	#154-5495	#154-5497	#154-5499
	#157-5636	#160-5788	#160-5804	#160-5819	#160-5847	#161-5864	#163-5916	#164-6135	#166-6172	#360-11474
	#367-11645	#369-11660	#371-11702	#373-11741						
SMACIT	#1-300	#108-3973	108-3988							

MACRO CROSS REFERENCE

MACRO NAME	REFERENCES									
SUBTST	#119-4536	#144-5263	#144-5281	#145-5301	#147-5351	#147-5365	#149-5389	#149-5418	#151-5429	#151-5441
	#153-5459	#154-5484	#154-5491	#155-5566	#155-5584	#155-5597	#161-5857	#163-5885	#164-6153	#167-6286
	#179-6793	#181-6943	#183-7020	#184-7097	#188-7163	#190-7179	#192-7209	#194-7239	#196-7276	#198-7313
	#200-7351	#202-7396	#204-7434	#206-7479	#208-7499	#210-7588	#212-7683	#214-7717	#216-7775	#217-7828
	#217-7841	#217-7863	#219-7891	#219-7927	#219-7945	#221-7967	#221-7974	#221-7984	#223-7993	#223-8001
	#223-8015	#223-8024	#225-8035	#225-8043	#225-8051	#227-8058	#230-8203	#231-8250	#231-8258	#232-8268
	#232-8310	#234-8320	#236-8367	#238-8438	#240-8483	#242-8512	#244-8576	#244-8606	#244-8636	#246-8658
	#246-8663	#248-8671	#252-8752	#252-8763	#252-8775	#254-8789	#254-8812	#256-8830	#256-8838	#258-8848
	#258-8861	#258-8870	#260-8884	#262-8927	#264-8967	#266-9003	#268-9144	#270-9226	#274-9316	#276-9424
	#278-9523	#282-9625	#284-9658	#288-9732	#292-9802	#293-9874	#296-9897	#298-9941	#298-9949	#300-9959
	#304-10053	#304-10060	#304-10078	#304-10091	#306-10099	#306-10105	#308-10156	#310-10176	#311-10222	#312-10234
	#313-10268	#313-10274	#314-10301	#316-10321	#318-10373	#324-10602	#339-10832	#339-10847	#340-10856	#342-10911
	#346-10980	#348-11088	#350-11194	#350-11239	#352-11252	#354-11300	#354-11320	#354-11333	#356-11345	#358-11436
	#358-11447	#358-11455	#358-11459	#360-11466	#362-11495	#362-11504	#362-11531	#364-11541	#365-11570	#367-11620
	#367-11642	#369-11657	#371-11681	#373-11723	#375-11775	#376-11887	#378-11893	#380-11934	#382-11964	#384-12005
	#384-12015	#384-12020	#384-12026	#384-12033	#385-12052	#385-12072	#387-12086	#389-12092	#391-12098	#406-12680
	#408-12721	#415-12926	#417-12953	#439-14037	#439-14049	#441-14062	#441-14077	#443-14086	#448-14245	#461-14604
	#470-14976	#470-15001								
SUPERV	#130-4809	#248-8694	#411-12786	#414-12883	#415-12930	#417-12971				
TESTAR	#131-4852	166-6201	166-6218	166-6235	166-6261	171-6381	210-7616	240-8490	242-8538	244-8584
	250-8727	329-10698	362-11533	371-11704	373-11744	373-11762	406-12669	406-12685	406-12695	408-12715
	408-12727									
THEN	#6-6900	#108-3976								
THRU	#11-4400	#108-3977								
TO	#10-2900	#108-3977								
TYPDEC	#125-4681	181-6995	181-6999	181-7003	181-7007	181-7011	181-7013	186-7122	380-11938	380-11940
	380-11955	457-14479								
TYPE	#115-4435	#154-5488	#154-5501	#154-5503	#154-5505	#154-5512	#154-5517	#154-5518	#160-5854	#179-6855
	#179-6938	#181-6992	#181-6996	#181-7000	#181-7004	#181-7008	#181-7012	#181-7014	#186-7117	#186-7119
	#208-7568	#208-7570	#316-10330	#316-10331	#316-10332	#316-10340	#316-10344	#316-10345	#316-10346	#316-10347
	#316-10348	#316-10349	#316-10358	#316-10359	#316-10360	#316-10361	#318-10386	#318-10389	#318-10391	#318-10400
	#318-10415	#320-10425	#320-10444	#320-10455	#320-10482	#320-10493	#320-10505	#320-10515	#320-10523	#320-10525
	#348-11109	#348-11139	#348-11147	#365-11572	#365-11575	#365-11585	#365-11590	#367-11621	#367-11651	#369-11662
	#369-11666	#369-11671	#369-11675	#371-11684	#371-11685	#371-11710	#371-11714	#373-11726	#373-11727	#373-11750
	#373-11754	#373-11757	#373-11759	#373-11766	#375-11778	#375-11779	#375-11788	#375-11792	#375-11797	#375-11807
	#375-11810	#378-11896	#378-11903	#378-11905	#378-11922	#380-11939	#380-11941	#380-11949	#380-11956	#382-11967
	#382-11974	#382-11976	#382-11992	#384-12007	#384-12016	#384-12021	#384-12027	#384-12034	#385-12053	#385-12057
	#385-12068	#385-12073	#387-12087	#389-12093	#391-12105	#414-12905	#419-13028	#450-14286	#450-14287	#451-14357
	#453-14385	#453-14409	#453-14410	#453-14412	#453-14419	#453-14421	#455-14448	#455-14458	#457-14489	#457-14493
	#458-14527	#458-14536	#458-14551	#458-14557	#458-14559	#463-14645	#463-14646	#463-14654	#463-14663	#463-14667
	#463-14669	#463-14679	#463-14682	#463-14684	#463-14688	#463-14694	#463-14697	#463-14699	#463-14703	#463-14705
	#466-14806	#467-14871	#468-14916	#468-14925	#468-14926	#468-14928	#468-14937	#468-14942	#468-14951	#468-14969
	#470-14978	#470-14989	#470-14991	#470-14993	#471-15059	#471-15072	#471-15078	#471-15083	#471-15087	#471-15092
	#471-15093	#471-15095	#471-15098	#471-15102	#472-15163	#472-15165	#472-15166	#472-15167	#472-15222	#472-15224
	#472-15225	#472-15226	#479-15444							
TYPOCS	#123-4628	179-6856	208-7569	371-11707	373-11758	373-11767	380-11952	457-14511	457-14517	458-14558
	470-14992	470-14994								
TYPOCT	#121-4582	#453-14390	#457-14473	#463-14653	#463-14668	#463-14670	#463-14683	#463-14685	#463-14698	#463-14700
	#468-14927									
UNTIL	#11-2900	#108-3977								
UNTILB	#18-1600	#108-3979								
USER	#130-4830	348-11158	350-11217	352-11276	352-11292	411-12780	414-12887			

MACRO CROSS REFERENCE

MACRO NAME	REFERENCES									
WHILE	#8-100	#108-3976	292-9822	293-9843	293-9880					
WHILEB	#18-2100	#108-3979								
$CALL	#15-100	#108-3980								
$RETUR	#15-3800	#108-3980	210-7671	210-7679	246-8665	#246-8667	348-11111	348-11149	348-11191	
$SUBTS	#119-4543	144-5263	144-5281	145-5301	147-5351	147-5365	149-5389	149-5418	151-5429	151-5441
	153-5459	154-5484	154-5491	155-5566	155-5584	155-5597	161-5857	163-5885	164-6153	167-6286
	179-6793	181-6943	183-7020	184-7097	188-7163	190-7179	192-7209	194-7239	196-7276	198-7313
	200-7351	202-7396	204-7434	206-7479	208-7499	210-7588	212-7683	214-7717	216-7775	217-7828
	217-7841	217-7863	219-7891	219-7927	219-7945	221-7967	221-7974	221-7984	223-7993	223-8001
	223-8015	223-8024	225-8035	225-8043	225-8051	227-8058	230-8203	231-8250	231-8258	232-8268
	232-8310	234-8320	236-8367	238-8438	240-8483	242-8512	244-8576	244-8606	244-8636	246-8658
	246-8663	248-8671	252-8752	252-8763	252-8775	254-8789	254-8812	256-8830	256-8838	258-8848
	258-8861	258-8870	260-8884	262-8927	264-8967	266-9003	268-9144	270-9226	274-9316	276-9424
	278-9523	282-9625	284-9658	288-9732	292-9802	293-9874	296-9897	298-9941	298-9949	300-9959
	304-10053	304-10060	304-10078	304-10091	306-10099	306-10105	308-10156	310-10176	311-10222	312-10234
	313-10268	313-10274	314-10301	316-10321	318-10373	324-10602	339-10832	339-10847	340-10856	342-10911
	346-10980	348-11088	350-11194	350-11239	352-11252	354-11300	354-11320	354-11333	356-11345	358-11436
	358-11447	358-11455	358-11459	360-11466	362-11495	362-11504	362-11531	364-11541	365-11570	367-11620
	367-11642	369-11657	371-11681	373-11723	375-11775	376-11887	378-11893	380-11934	382-11964	384-12005
	384-12015	384-12020	384-12026	384-12033	385-12052	385-12072	387-12086	389-12092	391-12098	406-12680
	408-12721	415-12926	417-12953	439-14037	439-14049	441-14062	441-14077	443-14086	448-14245	461-14604
	470-14976	470-15001								
$$END	#12-7400	#108-3978	503-16177							
$$NEWT	#119-4503	157-5609	164-6124	166-6161	169-6309	184-7066	184-7083			
.ARITH	#17-7700	#108-3979								
.EMIT	#1-4600	#108-3973								
.EMITL	#1-6800	#108-3973								
.EMITN	#1-5800	#108-3973								
.EMITR	#1-7300	#108-3973								
.GENBR	#2-5100	#108-3974								
.GOTO	#5-4500	#108-3976								
.IFARI	#5-1500	#108-3976								
.IFOPR	#2-100	#108-3974								
.IS	#2-3300	#108-3974								
.LEAVE	#5-3100	#108-3976								
.OPADD	#2-7800	#108-3974								
.OPSUB	#2-9300	#108-3974								
.OR	#5-100	#108-3976								
.SIMPL	#17-100	#108-3979								
.SUPER	#19-2500									
..BRAN	#1-3500	#108-3973								
..POP	#1-2200	#108-3973								
..PUSH	#1-1500	#108-3973								
..TAG	#1-3200	#108-3973								