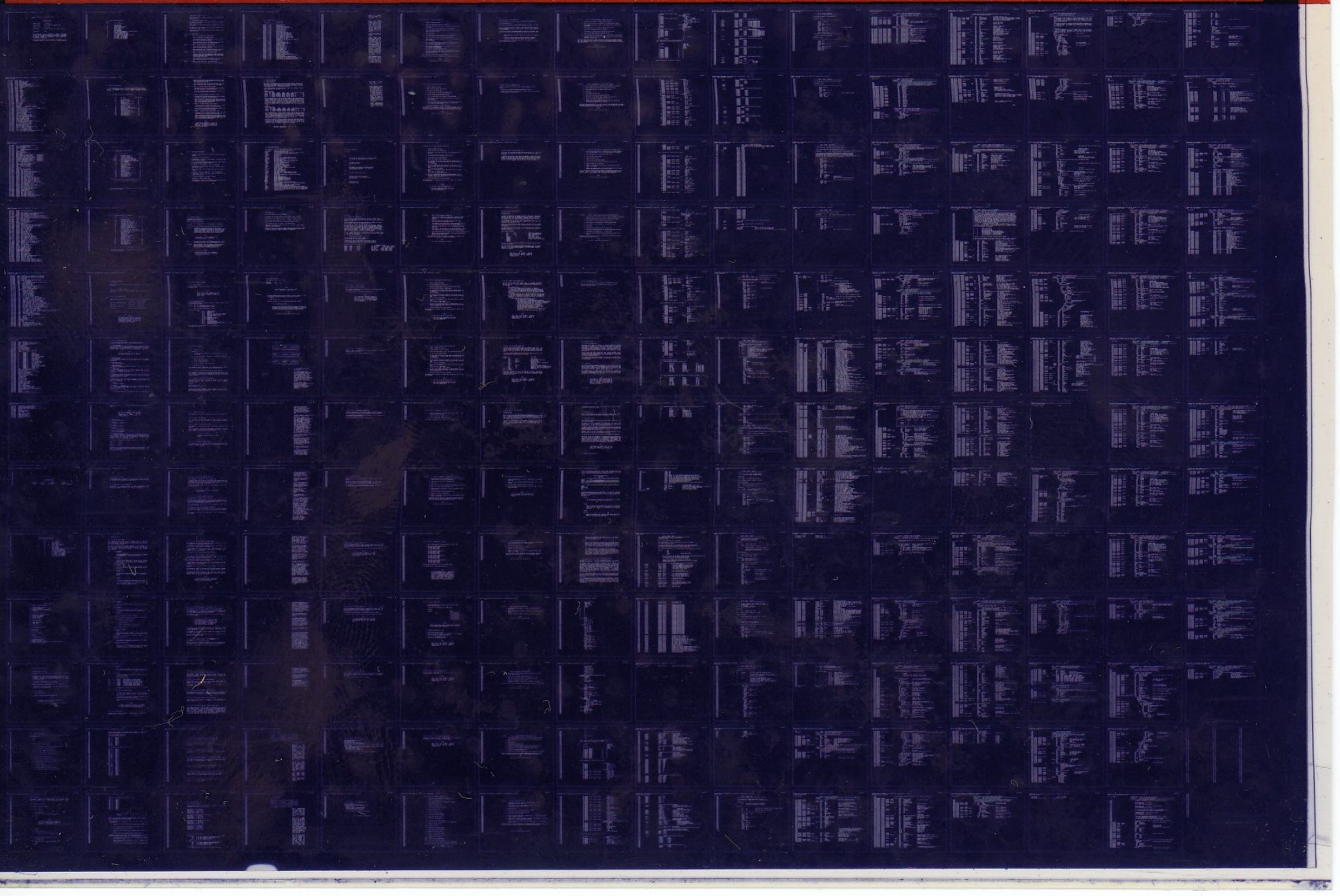
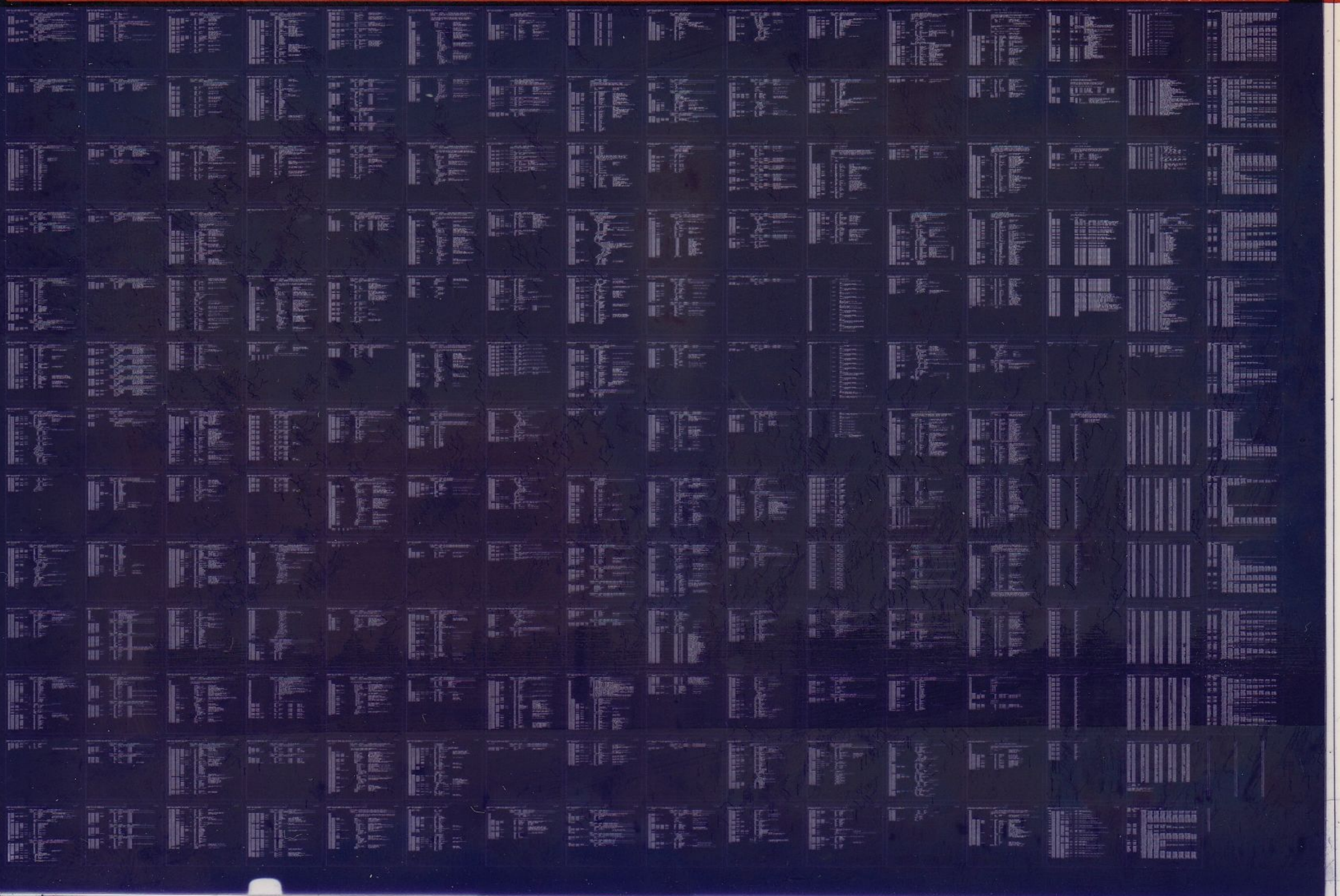
M8743,M7981 MS11-L/M/P MEM DIAG
M8722 CZMSPA0

AH-T157A-MC
FICHE 1 OF 3



M8743,M7981 MS11-L/M/P MEM DIAG AH-T157A-MC JUL 1982
M8722 • CZMSPA0 FICHE 2 OF 3
MADE IN USA



M8743,M7981 MS11-L/M/P MEM DIAG
M8722 CZMSPA0

AH-T157A-MC
FICHE 3 OF 3

```
                .TITLE  CZMSPA0 MS11-L/M/P MEMORY DIAG.
                .REM

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

                PRODUCT CODE:           AC-T156A-MC

                PRODUCT NAME:           CZMSPA0 MS11-L/M/P MEMORY DIAG

                PRODUCT DATE:           MAY  1982

                MAINTAINER:             STORAGE SYSTEMS S/W TEST APPLICATIONS
```



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

                THE SOFTWARE DESCRIBED IN THIS DOCUMENT IS FURNISHED TO  THE
                PURCHASER  UNDER  A  LICENSE  FOR  USE  ON A SINGLE COMPUTER
```

TABLE OF CONTENTS

110- 4559 DEFINE TRAPS
111- 4675 DEFINE BASIC PDP11 STUFF
111- 4758 DEFINE CACHE REGISTERS
111- 4765 DEFINE CPU REGISTERS
111- 4768 DEFINE MEMORY MANAGEMENT REGISTERS
113- 4898 DEFINE UNIBUS MAP REGISTERS
113- 4966 DEFINE SOFTWARE SWITCH & DISPLAY REGISTERS
113- 4970 DEFINE CONTROL STATUS REGISTERS
113- 4973 DEFINE PARAMETERS
115- 4980 MACRO   FATAL
115- 5000 MACRO   TYPE
117- 5018 MACRO   NEWTST
119- 5068 MACRO $$NEWTEST
119- 5089 MACRO   SUBTST
119- 5108 MACRO $SUBTST
121- 5123 MACRO   TYPOCT
123- 5163 MACRO   TYPOCS
125- 5219 MACRO   TYPDEC
126- 5261 MACRO   BMOV
128- 5327 MACRO   MAP
130- 5366 MACRO   SUPERVISOR
130- 5387 MACRO   USER
131- 5409 MACRO   TESTAREA
133- 5431 MACRO   SET4 & RES4
135- 5476 MACRO   DLEFT
137- 5500 TRAP CATCHER
137- 5508 ACT11 HOOKS
137- 5527 APT11 HOOKS
139- 5545 VARIABLES   INITIALIZED TO ZERO
141- 5717 VARIABLES   INITIALIZED TO NON ZERO
143- 5760 CONFIGURATION TABLE
144- 5787 ********** MAIN **********
144- 5788 INITIALIZE VARIABLES TO ZERO
144- 5800 CLEAR NON-PROGRAM SPACE
145- 5813 TYPE OF SYSTEM SIZER
147- 5863 INITIALIZE VARIABLES TO NON ZERO
147- 5873 INITIALIZE VECTORS
149- 5893 INITIALIZE PATTERNS
149- 5922 SUBR   PLUG IN NULL PATTERNS
151- 5933 CLEAR THE CONFIGURATION TABLE
151- 5945 SIZE FOR A HARDWARE SWITCH REGISTER
153- 5963 SETUP ACT, APT, & XXDP
154- 5988 PROTECT PROGRAM & LOADERS
154- 6000 CHECK SYSTEM FOR CACHE
155- 6032 SETUP USER & SUPERVISOR STACK
155- 6050 GET SOFTWARE SWITCH REGISTER IF NECESSARY
155- 6057 GET MEMORY MANAGEMENT READY
157- 6063 T1   BIT TEST OF ALL CSR'S
158- 6120 DETERMINE TYPE OF ECC MEMORY
159- 6140 PRINT CSR REGISTER MAP
160- 6181 READ AND WRITE ALL CSR BITS
164- 6419 CLEAR ALL MEMORY SPACE FROM BANK 2 ON
166- 6449 MATCH ALL CSR'S WITH MEMORY
167- 6694 T2   TEST BANK 0 ACCESSES
167- 6723 ENABLE ECC FOR CORRECT TRAPS
169- 6731 T3   TEST BANKS 1-200 (OCTAL) FOR ZEROS & ONES
170- 6856 FIND SHADOW INHIBIT MODE POINTERS

TABLE OF CONTENTS


```
   172- 6879    T4      ECC INHIBIT MODE POINTER TEST
   182- 7036    LEGAL CONFIGURATION CHECK
   184- 7150    PRINT CONFIGURATION DETAILS
   186- 7207    CHECK APT SIZING
   187- 7242    T5      DIAGNOSTIC MODE DISPATCH ROUTINE
   187- 7259    T6      UNIQUE BANK TEST
   187- 7273    FLUSH OUT DBE'S
   189- 7277    END OF PASS ROUTINE
   191- 7339    WRITE BACKGROUND PATTERNS
   193- 7353    MTEST MODES
   193- 7355    BANKS FORWARD,PATTERNS FORWARD       **RECURSIVE**
   195- 7385    BANKS FORWARD,PATTERNS REVERSE       **RECURSIVE**
   197- 7415    BANKS WORST FIRST,PATTERNS FORWARD   **RECURSIVE**
   199- 7452    BANKS WORST FIRST,PATTERNS REVERSE   **RECURSIVE**
   201- 7489    PATTERNS FORWARD,BANKS FORWARD       **RECURSIVE**
   203- 7527    PATTERNS FORWARD,BANKS WORST FIRST   **RECURSIVE**
   205- 7572    PATTERNS REVERSE,BANKS FORWARD       **RECURSIVE**
   207- 7610    PATTERNS REVERSE,BANKS WORST FIRST   **RECURSIVE**
   209- 7655    SUBR    SETUP MEMORY TEST
   211- 7675    SUBR    TEST ECC CSR LOGIC DISPATCH
   213- 7766    CHECK FOR SBE FREE LOCATIONS
   215- 7861    CSR PATTERN CASE STATEMENT
   217- 7907    SUBR    ECC TEST DISPATCH
   219- 7962    SUBR    PARITY TEST DISPATCH
   220- 8009    PATTERNS
   220- 8011    MEMORY TEST SETUP ROUTINES
   220- 8012    MT0000  SETUP DATA PATTERN TEST
   220- 8025    MT0001  SETUP ADDRESS TEST
   220- 8047    MT0002  SETUP COMPLEMENT ADDRESS TEST
   222- 8075    MT0003  SETUP 3 XOR 9 WORST CASE NOISE TEST
   222- 8111    MT0004  SETUP ROTATING ZEROS TEST
   222- 8129    MT0005  SETUP ROTATING ONES TEST
   224- 8151    MT0006  SETUP INITIAL DATA TEST
   224- 8158    MT0007  SETUP ADDRESS BIT TEST
   224- 8168    MT0010  SETUP BYTE ADDRESSING TEST
   226- 8177    MT0011  SETUP CREATE SINGLE BIT ERROR TEST
   226- 8186    MT0012  SETUP WRITE BYTE CLEARS SBE TEST
   226- 8201    MT0013  SETUP CREATE DOUBLE BIT ERROR TEST
   227- 8212    MT0014  SETUP BASIC DOUBLE BIT ERROR TEST
   229- 8226    MT0015  SETUP WRITE INHIBIT OF BYTE WITH DBE
   229- 8235    MT0016  SETUP WRITE INHIBIT OF WORD WITH DBE
   229- 8244    MT0017  SETUP HOLDING 1'S & 0'S
   231- 8251    MT0020  SETUP SYNDROMES TO CSR ON SINGLE BIT ERROR
   232- 8264    MT0021  SETUP MARCHING 0'S & 1'S TEST
   233- 8311    MT0022  SETUP REFRESH & SHIFTING DIAGONAL TEST
   233- 8319    MT0023  SHIFTING DIAGONAL TEST
   234- 8329    MT0024  SETUP FAST GALLOPING PATTERN TEST
   234- 8371    MT0025  SETUP INTERRUPT ENABLE TEST
   236- 8382    MT0026  SETUP RANDOM DATA TEST
   238- 8429    MT0027  UNIQUE BANK TEST
   240- 8500    MT0030  SETUP FLUSH OUT DBE'S TEST
   242- 8545    MT0031  SETUP SOB-A-LONG TEST
   244- 8574    MT0032  SETUP WRITE RECOVERY TEST
   246- 8638    MT0033  SETUP BRANCH GOBBLE TEST
   246- 8668    MT0034  SOFT ERROR - BACKGROUND PATTERN TEST
   246- 8698    MT0035  SETUP WORST CASE NOISE PARITY TEST
   247- 8719    MT0036  SETUP CORRECTION CODE TEST
```

TABLE OF CONTENTS


```
  248- 8731    MT0037  SETUP ECC DISABLE TEST
  248- 8743    MT0040
  249- 8746    MT0041  SETUP ADDRESS TO CSR ON DOUBLE BIT ERROR TEST
  250- 8755    MT0042  SETUP EXTENDED UNIBUS ADDRESS TO CSR TEST
  251- 8765    MT0043  SETUP WRITE BYTE CLEARS SBE TEST
  251- 8773    MT0044  SETUP SHIFTING 1/0'S THROUGH THE CHECK BITS TEST
  251- 8781    MT0045  SETUP SYNDROMES TO CSR ON DOUBLE BIT ERROR
  251- 8789    MT0046  SETUP CHECK SINGLE BIT ERRORS WITH ECC DISABLED TET
  251- 8797    MT0047  SETUP NO CSR UPDATE ON SBE WITH EXSISTING DBE TEST
  253- 8808    MT0999  SETUP NULL TEST
  253- 8813    CHECK FOR KAMIKAZE MODE
  255- 8821    SUBR    EXECUTE PATTERN IN SUPERVISOR
  259- 8892    MEMORY TEST PATTERN ROUTINES
  259- 8902    MTP000  BASIC DATA TEST
  259- 8913    MTP001  ADDRESS TEST
  259- 8925    MTP002  COMPLEMENT ADDRESS TEST (WRITE DOWN, READ UP)
  261- 8939    MTPA03  3 XOR 9 WORST CASE NOISE TEST (WRITE)
  261- 8962    MTPB03  3 XOR 9 WORST CASE NOISE TEST (READ)
  263- 8980    MTPC03  TEST DATA SUBPROGRAM
  263- 8988    MTPD03  TEST DATA SUBSUBPROGRAM
  265- 8998    MTPA04  ROTATING ZEROS TEST
  265- 9011    MTPB04  SUBR    ROTATING BIT
  265- 9020    MTP005  ROTATION ONES TEST
  267- 9034    MTP006  INITIAL DATA TEST
  269- 9077    MTP007  ADDRESS BIT TEST
  271- 9117    MTP010  BYTE ADDRESSING TEST
  273- 9153    MTP011  SINGLE BIT ERROR TEST
  275- 9294    MTP012  WRITE BYTE CLEARS SBE TEST
  277- 9376    MTP013  CREATE DOUBLE BIT ERROR TEST
  282- 9467    MTP014  BASIC DOUBLE BIT ERROR TEST
  283- 9515    MTP015  WRITE INHIBIT OF BYTE WITH DBE
  285- 9614    MTP016  WRITE INHIBIT OF WORD WITH DBE
  289- 9716    MTP017  HOLDING 1'S & 0'S TEST
  291- 9750    MTP020  SYNDROMES TO CSR ON SINGLE BIT ERROR TEST
  292- 9818    MTPA21  MARCHING 1'S & 0'S PATTERN TEST
  296- 9888    MTP022  REFRESH & SHIFTING DIAGONAL TEST
  297- 9960    SUBR    REFRESH DELAY
  299- 9982    MTPA24  FAST GALLOPING PATTERN TEST
 301-10026     MTPB24  FAST GALLOP PART B
 301-10034     MTPC24  FAST GALLOP PART C
 303-10044     MTP025  INTERRUPT ENABLE TEST
 307-10138     MTPA26  RANDOM DATA (WRITE)
 307-10145     MTPB26  RANDOM DATA (READ)
 307-10163     RANDOM NUMBER SUBPROGRAM
 307-10176     RANDOM NUMBER SUBSUBPROGRAM
 309-10184     MT0030  FLUSH OUT DBE'S
 309-10190     MTP031  SOB-A-LONG TEST
 311-10241     MTP032  WRITE RECOVERY TEST
 313-10261     MTP033  BRANCH GOBBLE TEST
 314-10307     MTP034  SOFT ERROR - BACKROUND PATTERN TEST
 315-10319     MTP035  WORST CASE NOISE PARITY TEST
 316-10351     MTP036  CORRECTION CODE TEST
 317-10404     MTP037  CHECK ECC DISABLE TEST
 319-10425     MTP041  ADDRESS TO CSR ON DOUBLE BIT ERROR TEST
 321-10466     MTP042  EXTENDED ADDRESS TO CSR ON ERROR TEST
 322-10519     MTP043  WRITE BYTE CLEARS SINGLE BIT ERROR TEST
 323-10560     MTP044  SHIFTING CHECK BITS THROUGH THE CSR TEST
```

TABLE OF CONTENTS

324-10640	MTP045	SYNDROMES TO CSR ON DOUBLE BIT ERROR TEST
325-10680	MTP046	CHECK SINGLE BIT ERRORS WITH ECC DISABLED
326-10756	MTP047	NO CSR UPDATE ON SBE WITH EXSISTING DBE
327-10800	MISC SUBROUTINES	
327-10802	SUBR	COPY R0 TO R4,R1 TO R3, & R2 TO R5
327-10808	FLIP WARNING CONSTANTS IN WORST CASE NOISE TESTS	
328-10835	SUBR	WRITE BACKGROUND
329-10854	SUBR	GET CSR INFORMATION FROM CONFIGURATION TABLE
331-10868	SUBR	PRINT CONFIGURATION MAP
333-10920	SUBR	TYPE CONFIGURATION
337-11050	TRAP	PARITY ERROR HANDLER
339-11082	TRAP	NON-EXISTANT MEMORY (HOLES) HANDLER
339-11102	TRAP	TIMEOUT (TRAP TO 4) HANDLER
339-11106	TRAP	MEMORY MANAGEMENT (TRAP TO 250) HANDLER
339-11109	TRAP	RESERVED INSTRUCTION HANDLER
339-11119	FIND BAD SP, PC, & PSW FROM STACK	
341-11127	TRAP	KERNEL TRAP HANDLER
341-11135	TRAP	ENERGIZE TRAP HANDLER
341-11139	TRAP	DEENERGIZE TRAP HANDLER
341-11143	TRAP	CACHON TRAP HANDLER
341-11150	TRAP	CACHOFF TRAP HANDLER
343-11158	TRAP	LOAD CSR TRAP HANDLER
343-11177	TRAP	READ CSR TRAP HANDLER
344-11185	TRAP	TEST (R1) & READ CSR CAREFULLY
346-11222	TRAP	ECC DISABLE ALL CSR'S TRAP HANDLER
346-11226	TRAP	ECC DISABLE OF 1 SELECTED CSR TRAP HANDLER
346-11230	TRAP	INITIALIZE ALL CSR'S TRAP HANDLER
346-11234	TRAP	INITIALIZE 1 SELECTED CSR TRAP HANDLER
346-11238	TRAP	ENABLE SBE PARITY TRAPS ON ALL CSR'S
346-11242	TRAP	ENABLE SBE PARITY TRAPS ON 1 SELECTED CSR
346-11246	TRAP	WRITE CHECKBITS THRU ALL CSR'S TRAP HANDLER
346-11251	TRAP	WRITE CHECKBITS THRU 1 SELECTED CSR TRAP HANDLER
348-11258	TRAP	WAS THERE A SBE ON ANY CSR TRAP HANDLER
348-11283	TRAP	WAS THERE A SBE IN 1 SELECTED CSR TRAP HANDLER
350-11293	TRAP	WAS THERE A DBE ON ANY CSR TRAP HANDLER
350-11318	TRAP	WAS THERE A DBE ON 1 SELECTED CSR TRAP HANDLER
352-11329	TRAP	CLEAR ALL ECC CSR'S TRAP HANDLER
352-11333	TRAP	CLEAR 1 SELECTED CSR TRAP HANDLER
352-11337	TRAP	ECC DISABLE, CHECK MODE, & WRITE CHECKBITS IN ALL CSR'S TRAP HANDLER
352-11342	TRAP	ECC DISABLE, CHECK MODE, & WRITE CHECKBITS IN 1 SELECTED CSR
354-11349	SUBR	WRITE IN ALL CSR'S
354-11364	TRAP	INVALIDATE BACKGROUND PATTERN
355-11373	TRAP	GENERATE AND TEST ERROR ADDRESS
355-11427	TRAP	ENABLE CHECK/SYNDROME BIT REGISTER
357-11434	SUBR	GENERATE CHECK BITS
361-11503	SUBR	MAPPER
361-11588	TRAP	MAP KERNEL (ALMOST 1 TO 1) TRAP HANDLER
363-11611	RELOCATE PROGRAM	
365-11716	UNRELOCATE PROGRAM	
365-11761	SETUP LOWER 16K OF UNIBUS MAP	
367-11774	MOVE BANKS	
369-11822	SUBR	MAP USER TO NEW BANK
369-11842	SUBR	SETUP KERNEL PAR'S FOR NEW BANK
369-11855	SUBR	MAP KERNAL PARS 4 AND 5 TO A BANK
369-11865	SUBR	SETUP KERNEL PAR'S FOR NEW LOADER BANK
369-11876	SUBR	UNMAP KERNAL PAR'S 4 AND 5
371-11883	SUBR	EXAMINE BANK

TABLE OF CONTENTS

373-11963 SUBR BANK OK?
373-11974 SUBR INCREMENT PATTERN TESTING
373-11982 SUBR SET HIGHEST PATTERN TESTING TYPE
373-11986 SUBR INCREMENT BANK & TEST
375-11993 BOOTSTRAP ROUTINE
377-12022 HALT PROGRAM
377-12031 SHUTDOWN DIAGNOSTIC
377-12058 APT SHUTDOWN SEQUENCE
379-12068 BLOCK MOVE SUBROUTINE
380-12095 FIELD SERVICE MODE
380-12097 SUBR FIELD SERVICE COMMAND MODE
382-12147 COMMAND 0 EXIT
382-12169 FS COMMAND 1 READ CSR
384-12184 FS COMMAND 2 LOAD CSR
386-12208 FS COMMAND 3 EXAMINE MEMORY
388-12250 FS COMMAND 4 MODIFY MEMORY
390-12302 FS COMMAND 5 SELECT BANK & PATTERN
391-12433 FS COMMAND 6 TYPE CONFIGURATION MAP
393-12439 FS COMMAND 7 SOB-A-LONG TEST
395-12480 FS COMMAND 8 ERROR SUMMARY
397-12510 FS COMMAND 9 REFRESH TEST
399-12551 FS COMMAND 10 SET FILL COUNT
399-12561 FS COMMAND 11 ENTER KAMIKAZE MODE
399-12566 FS COMMAND 12 EXIT KAMIKAZE MODE
399-12572 FS COMMAND 13 TURN CACHE OFF
399-12579 FS COMMAND 14 TURN CACHE ON
400-12598 FS COMMAND 15 TEST ONLY SELECTED BANKS
401-12618 FS COMMAND 16 RESUME TESTING ALL BANKS
402-12632 FS COMMAND 17 ENABLE TRACE
404-12638 FS COMMAND 18 DISABLE TRACE
406-12644 SUBR DETERMINE CORRECT CSR
421-13212 ERROR DATA (SUPERVISOR) SETUP STUFF
421-13226 DATA WAS 3 WORDS
423-13267 GET DATA FROM ABORTED AREA IF POSSIBLE
425-13283 POWER FAIL AUTO RESTART
425-13284 ROUTINE POWER DOWN AND UP
430-13472 POWER FAIL WHILE RELOCATED
432-13499 POWER UP FROM BANK 0 TO RELOCATION
434-13539 IO SUBROUTINES
434-13541 ROUTINE TYPE
449-14334 ERROR DATA SETUP
454-14583 DATA WAS A WORD
454-14595 DATA WAS A BYTE
456-14608 DATA WAS A 7 BIT BYTE
456-14623 DETERMINE XOR OF GOOD & BAD
458-14632 LOG ERROR ON BAD BANK
462-14721 ROUTINE SCOPE HANDLER
463-14791 SUBR DISPLAY
465-14808 ROUTINE ERROR HANDLER
468-14923 ROUTINE ERROR MESSAGE TYPEOUT
476-15150 SUBR DETAILED ERROR REPORT
481-15292 ROUTINE BINARY TO OCTAL (ASCII) AND TYPE
482-15370 ROUTINE CONVERT BINARY TO DECIMAL AND TYPE
483-15427 ROUTINE TTY INPUT
485-15522 CONTROL T
485-15547 CONTROL S & CONTROL Q
487-15667 ROUTINE READ AN OCTAL NUMBER FROM THE TTY

TABLE OF CONTENTS

487-15716 ROUTINE READ A DECIMAL NUMBER FROM THE TTY
488-15775 ROUTINE SAVE AND RESTORE R0-R5
489-15811 ROUTINE RANDOM NUMBER GENERATOR
491-15841 ROUTINE DOUBLE LENGTH BINARY TO OCTAL ASCII CONVERT
492-15883 TABLES
492-15885 APT MAILBOX-ETABLE
494-15967 ROUTINE TRAP DECODER
496-15994 TRAP TABLE
500-16097 TABLE   ERROR POINTER
509-16385 ERROR DATA TAGS (DT)
511-16412 ERROR DATA FORMATS (DF)
513-16429 ERROR MESSAGES (EM)
515-16475 ERROR DATA HEADERS (DH)
517-16502 MESSAGES

```
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
```

REVISION HISTORY

REVISION	DATE	AUTHOR	CHANGES
CZMSPA	1-JUN-82	IRA CHAVIS	NONE - NEW PROGRAM

```
                        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)
1.2 SYSTEM REQUIREMENTS
1.3 RELATED DOCUMENTS AND STANDARDS
1.4 DIAGNOSTIC HIERARCHY PREREQUISITES
1.5 ASSUMPTIONS

2.0 OPERATING INSTRUCTIONS

2.1 LOADING AND STARTING PROCEDURES
2.2 DEFAULT TEST SEQUENCE
2.3 SPECIAL ENVIRONMENTS
2.4 PROGRAM OPTIONS
2.5 EXECUTION TIMES

3.0 ERROR INFORMATION

3.1 ERROR REPORTING
3.2 ERROR ABBREVIATIONS
3.3 ERROR HALTS

4.0 PROGRESS REPORTS

5.0 CSR INFORMATION TABLES

5.1 MS11-P CSR
5.2 MS11-L CSR
5.3 MS11-M CSR

6.0 SUB-TEST SUMMARIES

6.1 TESTS
6.2 PATTERNS

7.0 PROGRAM FEATURES

7.1 FAST DATA ACCESS RATES
7.2 BANK ZERO TESTING
7.3 MEMORY CONFIGURATION MAP
7.4 EVERYTHING YOU'VE ALWAYS WANTED TO KNOW ABOUT SUPERMAC ...
7.5 MEMORY MANAGEMENT MAPPING

1.0 GENERAL PROGRAM INFORMATION

1.1 PROGRAM PURPOSE (ABSTRACT)

A. INTENDED FOR USE ON ALL PDP-11/24/44'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 MS11-P) 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-11-24/44 CPU WITH 22 BIT ADDRESSING AND AT LEAST 64K (16 BIT WORDS) OF MEMORY AND MEMORY MANAGEMENT.

NOTE

1. LIKE MEMORY TYPES MUST BE ON 16K WORD BOUNDARIES STARTING AT PHYSICAL ADDRESS 0.
2. PDP-11 SERIES 16/18 BIT PROCESSORS ARE NOT SUPPORTED.

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-11/04/24/34/44/70 PROCESSOR HANDBOOK (EB-19402)
2. PDP-11/44 USER'S GUIDE (EK-11044-UG)
3. MS11-M USER'S GUIDE (EK-MS11M-UG-001)
4. MS11-L USERS GUIDE (EK-MS11L-UG-001)
5. MS11-P TECHNICAL MANUAL (EK-MS11P-TM-001)

1.4 DIAGNOSTIC HIERARCHY PREREQUISITES

IF THE PROGRAM IN ANY WAY MISBEHAVES, THEN:

1. TRY IT AGAIN WITH CACHE OFF (REFEFENCE 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 MS11-L PARITY MEMORY -

	TEST	TEST 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 MS11-M ECC MEMORY -

	TEST	TEST 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
+@	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
*	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

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

2.2.3 TEST PROTOCOL FOR MS11-P ECC MEMEORY

	PATTERN	PATTERN NAME	TIME (SEC/16K)
	5	ROTATING 1'S TEST	1
	34	SOFT ERROR TEST	<1
	6	INITIAL DATA TEST	<1
	44	SHIFTING CHECK BITS THRU	1
	14	BASIC DBE TEST CSR TEST	<1
	45	SYNDROMES IN CSR ON DBE TEST	<1
	36	CORRECTION CODE TEST	1
	20	SYNDROMES IN CSR ON SBE TEST	1
	37	CHECK ECC DISABLE TEST	<1
	41	ADDRESS TO CSR ON DBE TEST	1
	42	EXTENDED ADDRESS TO CSR TEST	<1
	43	BYTE WRITE TEST	<1
	46	CHECK SBE WITH ECC DISABLE TEST	<1
	47	NO CSR UPDATE ON SBE WITH DBE 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
*	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

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
512K MS11-P	NOT ESTABLISHED AT RELEASE TIME	

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.

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.

604
605
606
607
608
609
610 2.3.3 NO SBE FREE BANKS -
611
612 IF THE PROGRAM CANNOT FIND ANY SBE (SINGLE BIT ERROR) FREE LOCATIONS
613 (IN NON-PROTECTED ECC MEMORY) IT WILL PRINT OUT AN ERROR MESSAGE AND
614 CONTINUE TESTING BY-PASSING THE ECC LOGIC TESTS.
615
616
617
618 2.3.4 MIXED PARITY ECC CONFIGURATIONS -
619
620 THE PROGRAM WILL FUNCTION NORMALLY IN MIXED ENVIRONMENTS. THE
621 SEQUENCE OF TESTING MAY SEEM STRANGE DUE TO THE RECURSIVE TEST MODE
622 ALGORITHM (REFERENCE SECTIONS 2.4.1.1, 2.4.1.2, 2.4.1.3).
623

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)

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

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

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

```
839
840                             ;RELOCATE  TEST PROGRAM SPACE (BANK 0 & 1)
841
842                             TEST  1.     BANK 0
843                             TEST  2.     BANK 0
844                             TEST  3.     BANK 0
845                             TEST  4.     BANK 0
846                             TEST  5.     BANK 0
847                             TEST 26.     BANK 0
848                             TEST  1.     BANK 1
849                             TEST  2.     BANK 1
850                             TEST  3.     BANK 1
851                             TEST  4.     BANK 1
852                             TEST  5.     BANK 1
853                             TEST 26.     BANK 1
854
855
856                                                  NOTE
857
858                                          THIS IS  AN  EXAMPLE    NOT  AN  ACTUAL
859                                          SEQUENCE.
860
861
862                             THE TEST SEQUENCE WAS FORWARD (THE SIMPLE PATTERNS  FIRST,  COMPLEX
863                             TESTS  LAST)  SEQUENCE OF PATTERNS (MS11-M = 17, 7, 1, 2, 4, 5, 21,
864                             20, 22, 26)(MS11-L = 1, 2, 3, 4, 5, 26).
865
866                             IF THE BANK SELECTION IS FORWARD THE  BANKS  WILL  BE  TESTED  IN  THE
867                             FOLLOWING ORDER:
868
869                                  1.  ECC BANKS THAT ARE NOT PROTECTED OR PROGRAM SPACE (FROM 0  TO
870                                      167).
871
872                                  2.  PARITY BANKS THAT ARE NOT PROGRAM SPACE (FROM 0 TO 167).
873
874                                  3.  THE PROGRAM NOW RELOCATES  TESTS:
875
876                                  4.  ECC BANKS THAT WERE PROTECTED OR PROGRAM  SPACE  (FROM  0  TO
877                                      167).
878
879                                  5.  PARITY BANKS THAT WERE PROGRAM SPACE (FROM 0 TO 167).
880
881                             IF BANK SELECTION IS WORST  FIRST  THE  CONFIGURATION  TABLE  WILL  BE
882                             CONSULTED AND BANKS WILL BE TESTED IN THE FOLLOWING ORDER.
883
884                                  1.  ECC BANKS THAT ARE KNOWN BAD AND ARE NOT PROTECTED OR PROGRAM
885                                      SPACE (FROM 0 TO 167).
886
887                                  2.  PARITY BANKS THAT ARE KNOWN BAD AND  ARE  NOT  PROGRAM  SPACE
888                                      (FROM 0 TO 167).
889
890                                  3.  ECC BANKS THAT ARE PRESUMED GOOD AND  ARE  NOT  PROTECTED  OR
```

```
892
893                                     PROGRAM SPACE (FROM 0 TO 167).
894
895                                 4.  PARITY BANKS THAT ARE PRESUMED GOOD AND ARE NOT PROGRAM SPACE
896                                     (FROM 0 TO 167).
897
898                                 5.  THE PROGRAM NOW RELOCATES  TESTS:
899
900                                 6.  ECC BANKS THAT ARE KNOWN BAD AND WERE  PROTECTED  OR  PROGRAM
901                                     SPACE (FROM 0 TO 167).
902
903                                 7.  PARITY BANKS THAT ARE KNOWN BAD AND WERE PROGRAM SPACE  (FROM
904                                     0 TO 167).
905
906                                 8.  ECC BANKS THAT  ARE  PRESUMED  GOOD  AND  WERE  PROTECTED  OR
907                                     PROGRAM SPACE (FROM 0 TO 167).
908
909                                 9.  PARITY BANKS THAT ARE PRESUMED GOOD AND  WERE  PROGRAM  SPACE
910                                     (FROM 0 TO 167).
911
912
913                             2.4.1.2  TEST MODE DETAILS -
914
915                             MODE 0     = 'BAFPAF'' BANKS FORWARD, PATTERNS FORWARD
916
917                                          THIS IS THE DEFAULT AND SIMPLEST MODE.
918
919                                          THIS MODE TESTS EACH BANK  COMPLETELY  FROM  0  TO  167
920                                          EXCEPT THOSE REQUIRING RELOCATION*.
921
922                                          WHILE TESTING EACH BANK THE PATTERNS ARE RUN  WITH  THE
923                                          SIMPLE ONES FIRST BUILDING TO THE MORE COMPLEX.
924
925                             MODE 1     = 'BAFPAR'' = BANKS FORWARD, PATTERNS REVERSE
926
927                                          THIS MODE TESTS EACH BANK  COMPLETELY  FROM  0  TO  167
928                                          EXCEPT THOSE REQUIRING RELOCATION*.
929
930                                          WHILE TESTING EACH BANK THE PATTERNS ARE RUN  WITH  THE
931                                          MOST COMPLEX ONES FIRST, WORKING TO THE SIMPLE ONES.
932
933                             MODE 2     = 'BAWPAF'' = BANKS WORST FIRST, PATTERNS FORWARD
934
935                                          THIS MODE FIRST TESTS EACH KNOWN  BAD  BANK  COMPLETELY
936                                          FROM  0 TO 167 EXCEPT THOSE REQUIRING RELOCATION*, THEN
937                                          PRESUMED GOOD BANKS ARE TESTED FROM  0  TO  167  EXCEPT
938                                          THOSE REQUIRING RELOCATION*.
939
940                                          WHILE TESTING EACH BANK THE PATTERNS ARE RUN  WITH  THE
941                                          SIMPLE ONES FIRST, BUILDING TO THE MORE COMPLEX.
942
943                             MODE 3     = 'BAWPAR'' = BANKS WORST FIRST, PATTERNS REVERSE
944
945                                          THIS MODE FIRST TESTS EACH KNOWN  BAD  BANK  COMPLETELY
```

```
                                        FROM  0 TO 167 EXCEPT THOSE REQUIRING RELOCATION*, THEN
                                        PRESUMED GOOD BANKS ARE TESTED FROM  0  TO  167  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
                                        167 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  167  EXCEPT THOSE REQUIRING RELOCATION* IS
                                        RUN, THEN PRESUMED GOOD BANKS ARE RUN  FROM  0  TO  167
                                        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
                                        167 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 167 EXCEPT THOSE THAT REQUIRE RELOCATION* IS
                                        RUN, THEN PRESUMED GOOD BANKS ARE RUN  FROM  0  TO  167
                                        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.

1042
1043
1044
1045
1046
1047
1048
1049 2.4.2 DISPLAY REGISTER -
1050
1051 A SOFTWARE DISPLAY REGISTER EXISTS IN LOCATION 174 IN ADDITION TO ANY
1052 HARDWARE DISPLAY EXISTENCE.
1053
1054 DISPLAY FIELDS ARE AS FOLLOWS:
1055
1056 15 | 14 13 12 11 10 9 8 | 7 6 5 | 4 3 2 1
1057 RELOCATED | BANK # | NOT USED | PATTERN #
1058 | | |
1059 ==
1060
1061 PATTERN # = THE NUMBER OF THE PATTERN PRESENTLY BEING RUN. ALL
1062 PATTERNS ARE DESCRIBED IN SECTION 6.2. ANY PATTERN CAN BE
1063 FOUND IN THE DIAGNOSTIC BY LOOKING UP THE SYMBOLIC TAGS
1064 'MTOONN'' AND 'MTPONN'' - WHERE 'NN'' IS THE TEST NUMBER.
1065 MTOONN REFERS TO THE ROUTINE THAT SETS UP FOR THE TEST
1066 PATTERN WHEREAS MTPONN IS THE ACTUAL PATTERN ITSELF.
1067
1068
1069 NOTE
1070
1071 THE PATTERN # IS NOT NECESSARILY AN
1072 INDICATION OF DEGREE OF DIFFICULTY.
1073
1074
1075
1076 BANK = THE NUMBER OF THE BANK (16K) OF MEMORY UNDER TEST (0-167).
1077 THESE BITS DIRECTLY MAP TO PHYSICAL ADDRESS BITS (21:15).
1078
1079 RELOCATED = THIS BIT INDICATES THAT THE PROGRAM IS RELOCATED AND NO
1080 LONGER IN BANK 0. IT WILL BE RELOCATED TO THE FIRST
1081 KNOWN GOOD NON-PROTECTED MEMORY BANK INDICATED ON
1082 THE CONFIGURATION MAP (REFERENCE SECTION 7.3).
1083
1084
1085
1086 NOTE
1087
1088 ANOTHER WAY TO OBTAIN THIS INFORMATION
1089 IS TO TYPE A CONTROL/T AT THE CONSOLE
1090 (REFERENCE SECTION 2.4.4.5).
1091
1092

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 "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.

2.4.4.3 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:

```
BANK = 17 TEST = 46
RELOCATED BANK= 0  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).

2.4.4.4 CONTROL "S" (STOP)

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

2.4.4.5 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.

2.4.4.6 CONTROL "F" (FIELD SERVICE MODE)

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

WHEN THE PROGRAM IS LOOKING FOR A SUB COMMAND ANY NUMBER THAT IS NOT A LEGAL COMMAND WILL CAUSE A MINI HELP MESSAGE TO BE TYPED. THEREFORE WHEN IN DOUBT TYPE 99 (CR) AND YOU WILL GET HELP.

NOTE

TYPING JUST CARRIAGE RETURN IS A DEFAULT COMMAND 0.

2.4.4.7.1 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.

2.4.4.7.2 FIELD SERVICE COMMAND 1 (READ CSR)

THIS COMMAND WILL TYPEOUT THE CONTENTS OF THE CSR.

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

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

NOTE

CSR REFERENCES ARE DONE IN ACCORDANCE WITH SECTION 5.0.

2.4.4.7.3 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.

2.4.4.7.4 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.

2.4.4.7.5 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.

2.4.4.7.6 FIELD SERVICE COMMAND 5 (SELECT BANK TEST)

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

THE PROGRAM WILL ASK YOU "BANK(0-167)" 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 "TEST (0-47)" 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 "TEST 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.

2.4.4.7.7 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.

2.4.4.7.8 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.

2.4.4.7.9 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.

2.4.4.7.10 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.

2.4.4.7.11 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.

2.4.4.7.12 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.

2.4.4.7.13 FIELD SERVICE COMMAND 12 (EXIT KAMIKAZE MODE)

RETURN TO THE DEFAULT MODE OF TESTING (UNDO COMMAND 12).

2.4.4.7.14 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).

2.4.4.7.16 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.

2.4.4.7.17 FIELD SERVICE COMMAND 16 (RESUME TESTING ALL BANKS)

RETURN TO THE DEFAULT MODE OF TESTING (UNDO COMMAND 15).

2.4.4.7.18 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.

2.4.4.7.19 FIELD SERVICE COMMAND 18 (RESUME TESTING ALL BANKS)

DISABLE "TRACE". (UNDO COMMAND 17).

```
1521
1522
1523
1524
1525
1526
1527                    2.5  EXECUTION TIMES
1528
1529                    2.5.1  TYPICAL (SYSTEM) -
1530
1531                    EXECUTION TIME DEPENDS ON  MANY  VARIABLES;   HOWEVER  HERE  ARE  SOME
1532                    MEASURED TIMES ON AN 11/44 WITH CACHE:
1533
1534
1535                    128K WORDS OF MS11-L MEMORY
1536                    NORMAL PASS      0 MIN  50 SEC
1537                    QUICK VERIFY     0 MIN  50 SEC
1538                    KAMIKAZE MODE   10 MIN   5 SEC
1539                    KAMIKAZE QV     10 MIN   5 SEC
1540
1541                    128K WORDS OF MS11-M MEMORY (NON-INTERLEAVED)
1542                    NORMAL PASS      2  MIN 25 SEC
1543                    QUICK VERIFY     1  MIN  0 SEC
1544                    KAMIKAZE MODE   11  MIN  0 SEC
1545                    KAMIKAZE QV     10  MIN 30 SEC
1546
1547                    128K WORDS OF MS11-M MEMORY (INTERLEAVED)
1548                    NORMAL PASS      3  MIN 55 SEC
1549                    QUICK VERIFY     1  MIN 50 SEC
1550                    KAMIKAZE MODE   22  MIN  0 SEC
1551                    KAMIKAZE QV     20  MIN  5 SEC
1552
1553                    512K WORDS OF MS11-P MEMORY
1554                    NORMAL PASS      3 MIN 55 SEC
1555                    QUICK VERIFY     3 MIN 25 SEC
1556                    KAMIKAZE MODE   42 MIN 30 SEC
1557                    KAMIKAZE QV     37 MIN  0 SEC
1558
1559
1560
1561
1562                    2.5.2  CALCULATIONS (SYSTEM)
1563
1564                    NORMAL PASS
1565                            ADD      18 SEC PER 16K BANK OF NON-INTEREAVED MS11-M
1566                            ADD      15 SEC PER 16K BANK OF INTERLEAVED MS11-M
1567                            ADD       6 SEC PER 16K BANK OF MS11-L
1568                            ADD      22 SEC PER 64K BANK OF MS11-P
1569
1570                    QUICK VERIFY PASS
1571                            ADD       8 SEC PER 16K BANK OF NON-INTERLEAVED MS11-M
1572                            ADD       7 SEC PER 16K BANK OF INTERLEAVED MS11-M
1573                            ADD       6 SEC PER 16K BANK OF MS11-L
1574                            ADD      20 SEC PER 64K BANK OF MS11-P
1575
1576                    KAMIKAZE MODE
1577                            ADD 10 MIN. PER 128K WORDS FOR APPROXIMATE PASS TIMES.
```

2.5.3 TYPICAL (TESTS)

TEST	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	BASIC DOUBLE BIT ERROR TEST
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	SYNDROMES TO CSR ON SINGLE BIT ERROR TEST
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 TES'
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
MT0036	: 1 SEC	CORRECTION CORE TEST
MT0037	:<1 SEC	CHECK ECC DISABLE TEST
MT0041	: 1 SEC	ADDRESS TO CSR ON DBC TEST
MT0042	:<1 SEC	EXTENDED ADDRESS TO CSR ON ERROR TEST
MT0043	:<1 SEC	WRITE BYTE TEST
MT0044	: 1 SEC	SHIFTING CHECKBITS THROUGH CSR TEST
MT0045	:<1 SEC	SYNDROME BITS TO THE CSR ON A DBE TEST
MT0046	: 1 SEC	CHECK SINGLE BIT ERRORS WITH ECC DISABLED TEST
MT0047	:<1 SEC	NO CSR UPDATE WITH EXISTING DBE 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		-	06
022132	37	060006	03700006	000000	000100	000100	0		-	06
022132	37	060006	03700006	000000	000100	000100	0		-	06
022132	37	060006	03700006	000000	000100	000100	0		-	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-P (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, OR MS11-P).
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.
AUT	ADDRESS UNDER TEST

```
1737
1738
1739                                    3.3  ERROR HALTS
1740
1741                                    THERE ARE SEVERAL HALTS IN THE PROGRAM.
1742
1743                                    ALL UNUSED TRAP VECTORS CONTAIN A TRAP CATCHER (.WORD .+2,HALT).
1744
1745                                    AN UNDEFINED TRAP INSTRUCTION HALTS AT SYMBOLIC LOCATION "$HALT2".
1746
1747                                    THE APT DOWN LOAD SEQUENCE WILL HALT AT SYMBOLIC LOCATION "APTHLT".
1748
1749                                    HALT ON ERROR OPTION (SW15 SET) AT SYMBOLIC LOCATION "$HALT".
1750
1751                                    HALT PROGRAM (SW8 SET) AT SYMBOLIC LOCATION "$EXHALT".
1752
1753                                    POWER FAIL WILL NORMALLY HALT AT THE END OF  THE  SHUT  DOWN  SEQUENCE
1754                                    (SYMBOLIC LOCATION "$DOWN").
1755
1756                                    POWER FAIL HAS A FATAL HALT AT SYMBOLIC LOCATION "$ILLUP" WHICH CAN BE
1757                                    CAUSED  BY  POWER UP OCCURRING BEFORE POWER DOWN SEQUENCE COMPLETED OR
1758                                    BY POWER DOWN BEFORE A POWER UP SEQUENCE IS COMPLETED.
1759
1760
```

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 ";" WILL PRINT OUT THE INFORMATION ENCODED IN THE DISPLAY REGISTER.

EXAMPLE:

```
BANK= 2 TEST= 34
```

REFERENCE SECTION 2.4.4.7.18 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

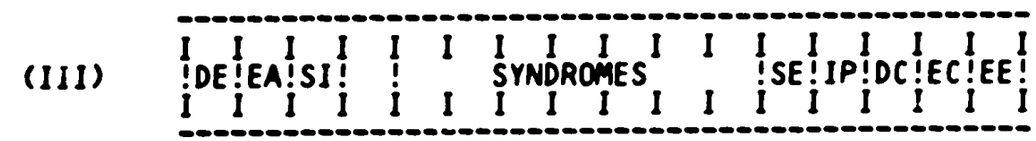
ALL UNUSED BITS IN EACH CSR ARE EQUAL TO ZERO.

5.1 MS11-P CSR

(I)

(II)

(III)



BIT ASSIGNMENTS ARE DEFINED
AS FOLLOWS:

BIT15 UNCORRECTABLE ERROR
ON A READ TO MEMORY (ECC DISABLE
BIT = 0), THIS BIT IS SET IF
A DOUBLE ERROR OCCURS. THE
ERROR ADDRESS IS STORED IN THE
CSR. SETTING THIS BIT ALSO
TURNS ON A RED LED AT THE REAR OF
THE CARD FOR A VISUAL INDICATION.
THIS BIT IS ALSO SET IN ECC
DISABLE MODE IF A SERR OR DERR
OCCURS.

BIT14 EUB ERROR ADDRESS
WITH BIT 14 = 1, A READ TO THE CSR
WILL FETCH ADDRESS A21
THROUGH A18. WHEN BIT 14 = 1,
DIAGNOSTIC DATA MAY NOT BE
LOADED INTO THE SYNDROME
REGISTER.

```
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
```

BIT13 SET INHIBIT MODE
WHEN THIS BIT IS SET TO
A "1", IT ENABLES THE INHIBIT
MODE POINTER TO INHIBIT
EITHER THE FIRST OR SECOND
16K FROM EVER GOING INTO THE
DIAG CHECK OR ECC DISABLE
MODE.

BITS05-10 CHECK BIT STORAGE (CSR II)
CHECK BIT STORAGE (DIAG CK
BIT 2 = 1)
WHEN IN THE DIAGNOSTIC CHECK
MODE THESE BITS ARE USED TO STORE
THE CHECK BITS TO BE WRITTEN
INTO MEMORY OR THE CHECK BITS
READ FROM MEMORY. IF A DOUBLE
ERROR OR SINGLE ERROR OCCURS
WHEN IN THE DIAGNOSTIC CHECK
MODE AND ECC DISABLE BIT 1 = 0,
THEN THE CHECK BITS ARE STORED
IN THE CSR TOGETHER WITH
THE DOUBLE OR SINGLE ERROR
BIT. THESE BITS ARE WRITEABLE
IN DIAGNOSTIC MODE. A "1"
IS STORED IN BIT 11 IF CSR
02, CSR 13, AND CSR 14 APE
SET TO INDICATE THAT THE
MEMORY UNDER TEST IS A MS11-P.

BITS05-11 UNIBUS ADDRESS STORAGE (CSR I)
(DIAG CK BITS 2 = 0, ECC
DISABLE BIT 1 = 0)

IF A DOUBLE OR SINGLE ERROR
OCCURS ON A READ CYCLE, THEN
ADDRESS BITS ALL THROUGH
A17 ARE STORED IN THESE BITS
THESE BITS ARE READ ONLY ON
THE CONDITION THAT SERR (CSR 4)
OR DERR (CSR 15) IS SET BUT
CSR 14 IS NOT SET.

EUB ADDRESS STORAGE (DIAG CK
BIT 2 = 0), ECC DISABLE BIT
1 = 0 OR 1).

IF A DOUBLE OR SINGLE ERROR
OCCURS ON A READ CYCLE,
ADDRESS BITS A17 THROUGH
A11 ARE STORED IN CSR BITS
11 THROUGH 5 AND ADDRESS
BITS A21 THROUGH A18 ARE

```
1925
1926                                                    STORED IN A BACKUP REGISTER.
1927                                                    THE EUB ERROR ADDRESS
1928                                                    RETRIEVAL BIT (CSR 14) IS
1929                                                    USED TO OBTAIN THE TOTAL
1930                                                    ERROR ADDRESS AS FOLLOWS:
1931
1932                                                    WITH CSR BIT 14 = 0 A READ TO
1933                                                    THE CSR WILL OBTAIN A17
1934                                                    THROUGH A11 FROM CSR BITS 11
1935                                                    THROUGH 5.
1936
1937                                                    CSR BIT 14 CAN THEN BE SET
1938                                                    TO A "1" AND A READ TO THE
1939                                                    CSR WILL THEN READ A21
1940                                                    THROUGH A18 FROM CSR BITS 8
1941                                                    THROUGH 5 AND 0'S FROM CSR
1942                                                    BITS 11 THROUGH 9.
1943
1944                                                    ADDRESS BITS A21 THROUGH
1945                                                    A11 ARE OBTAINED
1946                                                    TO LOCATE THE DOUBLE
1947                                                    ERROR TO A 1K SEGMENT OF
1948                                                    MEMORY.
1949
1950                                                    THE EUB ADDRESS A21
1951                                                    THROUGH A18 IS READ ONLY
1952                                                    WHENEVER CSR 14 = 1.
1953
1954                                                    BIT05-10 SYNDROME STORAGE (CSR III)
1955                                                    IF A DOUBLE OR SINGLE ERROR
1956                                                    OCCURS ON A READ OR WRITE
1957                                                    BYTE CYCLE, AND IF CSR BIT
1958                                                    2 IS SET TO A "0" SYNDROME
1959                                                    BITS X, 0, 1, 2, 4 AND 8
1960                                                    AND STORED IN CSR BITS 5
1961                                                    THROUGH 10.  TO READ THE
1962                                                    SYNDROME BITS FROM CSR, BIT
1963                                                    YOU MUST READ THE ERROR
1964                                                    ADDRESS, THEN SET 2 OF
1965                                                    THE CSR MUST BE ST TO
1966                                                    A "1" (DIAGNOSTIC MODE) AND
1967                                                    THE CSR READ AGAIN.  THIS OPERATION
1968                                                    WILL ALLOW SYNDROME BITS
1969                                                    FOR A SINGLE OR DOUBLE
1970                                                    FAILURE TO BE READ INSTEAD
1971                                                    OF THE ADDRESS BITS NORMALLY
1972                                                    READ WHEN CSR 02 IS SET TO "0".
1973
1974                                                    BIT04 SINGLE ERROR
1975                                                    IF ON A READ TO MEMORY A
1976                                                    SBE OCCURS, THE ERROR
1977                                                    ADDRESS A21-A11 AND
1978                                                    THE ERROR SYNDROMES WILL
```

```
1980                                    BE LOGGED IN CSR BITS 5-11
1981                                    UNLESS THE UNCORRECTABLE ERROR
1982                                    CSR 15 IS SET.  THE ERROR
1983                                    ADDRESS WILL BE LOGGED
1984                                    UNCONDITIONALLY IN THE ECC
1985                                    DISABLE MODE.  THIS BIT IS NOT
1986                                    SET IF INHIBIT MODE (BIT
1987                                    13 = 1) IS SET AND DIAGNOSTIC
1988                                    MODE (BIT 02 = 1) IS SET.
1989
1990                                    BIT03 INHIBIT MODE POINTER
1991                                    THIS BIT WORKS IN CONJUNCTION
1992                                    WITH THE SET INHIBIT MODE
1993                                    (BIT 13).  WHEN BIT 13 IS SET
1994                                    TO A 1, A 16K PORTION OF
1995                                    MEMORY IS INHIBITED FROM
1996                                    OPERATING IN THE ECC DISABLE
1997                                    MODE OR DIAGNOSTIC CHECK MODE.
1998
1999                                    THE INHIBIT MODE POINTER
2000                                    INDICATES WHICH 16K IS BEING
2001                                    INHIBITED, I.E., BIT 3 = 0
2002                                    THE FIRST 16K OF MEMORY IS
2003                                    INHIBITED, BIT 3 = 1, THE
2004                                    SECOND 16K OF MEMORY IS
2005                                    INHIBITED.
2006
2007                                    WITH BIT 13 SET TO A 0, BIT
2008                                    3 BECOMES INOPERATIVE.
2009
2010                                    BIT03, IN CONJUNCTION WITH
2011                                    BIT 13, THEREFORE ALLOWS A 16K
2012                                    CHUNK OF MEMORY TO ALWAYS HAVE
2013                                    ECC COVERAGE.  THE SYSTEMS
2014                                    DIAGNOSTIC CAN THEREFORE
2015                                    RESIDE IN THIS PROTECTED
2016                                    PORTION OF MEMORY AND CAN
2017                                    DISABLE ECC AND/OR RUN THE
2018                                    DIAGNOSTIC CHECK MODE IN THE
2019                                    REST OF MEMORY WITHOUT ITSELF
2020                                    BECOMING VULNERABLE TO SINGLE
2021                                    ERROR.  THIS BIT IS A READ/WRITE
2022                                    BIT RESET BY POWER UP AND BUS
2023                                    INIT.
2024
2025                                    BIT02 DIAGNOSTIC CHECK MODE
2026                                    THIS MODE ALLOWS A MEANS OF
2027                                    FORCING A SINGLE OR DOUBLE
2028                                    ERROR IN A DESIRED LOCATION.
2029                                    IT ALSO PROVIDES A MEANS OF
2030                                    EXAMINING THE CHECK BITS AND
2031                                    THE SYNDROME IN A GIVEN
2032                                    LOCATION.
```

```
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
```

THE CHECK BITS DESIRED FOR A
GIVEN DATA PATTERN ARE WRITTEN
INTO BITS 5 THROUGH 11 OF
THE CSR. A WORD OR WRITE BYTE
MEMORY WILL WRITE THE CHECK BITS
FROM THE CSR TO THE MOS ARRAY
(CSR 2 = 1) INSTEAD OF THE CHECK
BITS GENERATED ON THE DATA
TO BE WRITTEN. SINGLE ERRORS
ON THE READ PORTION OF THE DATOB
CYCLE ARE CORRECTED.

A READ TO THE MEMORY WILL
READ THE CHECK BITS STORED IN
MEMORY AND CLOCK THEM INTO
THE CSR.

IF A DOUBLE ERROR OR SINGLE
ERROR OCCURS THE DERR OR
SERR BIT IN THE CSR IS SET
AND THE ERROR SYNDROME BITS
READ FROM ECC ARE STORED IN
CSR BITS 10-5 AS WELL AS THE
ADDRESS BITS. IN DIAGNOSTIC
CHECK MODE THE ERROR SYNDROME
BITS WILL BE READ WHEN CSR
BITS 10-5 ARE READ.

THIS BIT IS A READ/WRITE BIT
AND IS RESET ON POWER UP AND
BUS INIT.

BIT01 DISABLE CORRECTION MODE
IF THIS BIT IS SET, NO SINGLE
ERRORS WILL BE CORRECTED. A
SINGLE ERROR WILL SET CSR 4
AND CSR 15 OR A DOUBLE ERROR
WILL SET CSR 15 AND ASSERT
BUS PBL IF CSR 00 IS ASSERTED.
THE 1K BLOCK OF ADDRESS WHERE
THE ERROR OCCURS WILL ALSO
RE STORED IN THE CSR. THE
PRIORITY OF A SERR AND DERR
WILL BE THE SAME, I.E., THE
LAST ERROR INFORMATION WILL
ALWAYS BE STORED UNLESS A DERR
PRECEDES A SERR. IF A DOUBLE
ERROR OCCURS DURING A WRITE BYTE
CYCLE, THE WRITE PORTION OF
THE CYCLE WILL NOT BE ABORTED.
THE CHECK BITS WRITTEN WILL

```
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
```

HAVE BEEN GENERATED ON THE
DATA WRITTEN. THIS MEANS THAT
IF A SINGLE OR DOUBLE ERROR
EXISTED IN THE LOCATION ACCESSED,
IT WOULD BE CLEARED (UNLESS THE
ERRORS WERE HARD).

THIS BIT IS A DIAGNOSTIC AID
TO ALLOW WRITING AND READING
DATA FROM MEMORY WITHOUT
INTERFERENCE FROM THE ERROR
CORRECTION LOGIC.

BIT00 UNCORRECTABLE ERROR
INDICATION ENABLE
IF A DOUBLE ERROR OCCURS WITH
ECC ENABLED OR A SINGLE ERROR
OR DOUBLE ERROR WITH ECC
DISABLED, ON A READ CYCLE TO THE
MEMORY AND THIS BIT IS SET,
THEN BUS PBL WILL BE ASSERTED.

5.2 MS11-L CSR

BIT ASSIGNMENTS ARE DEFINED AS FOLLOWS:

BIT15 PARITY ERROR

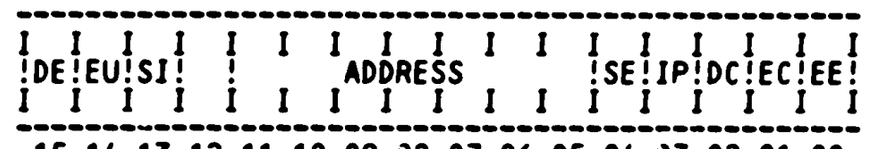
BIT14 EUB ERROR RETRIEVAL IF THE MEMORY IS ON AN EXTENDED UNIBUS, WHEN BIT14 IS ZERO, THE LOW ORDER FAILING ADDRESSES ARE AVAILABLE (BITS 11-17); WHEN BIT14 IS ONE, THE HIGH ORDER FAILING ADDRESSES ARE AVAILABLE (BITS 18-21 OF ADDRESS). IF THE MEMORY IS ON A UNIBUS, A JUMPER DISABLES THIS BIT SO THAT 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 WHEN SET.

5.3 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

```
2224
2225
2226
2227
2228
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2270
2271
2272
2273
2274
2275
2276
2277
```

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

```
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
```

ARE STORED IN BITS 11-5, NOT THE DBE ADDRESS BITS.

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

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

```
2311
2312
2313
2314
2315
2316                            6.0  SUB-TEST SUMMARIES
2317
2318                            6.1  TESTS
2319
2320
2321
2322
2323                            TEST     1
2324
2325                                     BIT TEST OF ALL CSR'S/MATCH ALL CSR'S WITH MEMORY
2326                                     (CSR ACCESS MAY CAUSE WRONG TYPE OF TRAPS)
2327
2328                            TEST     2
2329
2330                                     TEST BANK 0 ACCESSES
2331                                     FAILURES ARE FATAL.
2332
2333                            TEST     3
2334
2335                                     TEST BANKS 1-167 (OCTAL) FOR ZEROS AND ONES
2336                                     ERRORS ARE NOT TYPED HERE - ONLY LOGGED IN
2337                                     THE CONFIGURATION TABLE
2338
2339                            TEST     4
2340
2341                                     ECC INHIBIT MODE POINTER TEST
2342
2343                            TEST     5
2344
2345                                     DIAGNOSTIC MODE DISPATCH ROUTINE
2346                                     THIS TEST RUNS ALL THE PATTERNS IN THE
2347                                     MODE SELECTED.
2348
2349                            TEST     6
2350
2351                                     UNIQUE BANK TEST
2352                                     PATTERN 27 IS RUN
2353
2354
2355
```

6.2 TESTS

6.2.1 GENERAL TEST INFORMATION

ACTUAL TESTS 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 TEST.

SETUP PROCEDURES FOR EACH TEST ARE IDENTIFIED BY SYMBOLIC LOCATIONS 'MT00YY'' WHERE YY WILL BE THE NUMBER OF THE TEST.

TESTS RESIDE IN 4 SCRIPTS THAT ARE SCANNED FOR EXECUTION. SYMBOLIC LOCATION 'MKCSRT'' IS A TABLE OF TESTS THAT CAN RUN ONCE FOR EACH ECC BANK (TWICE FOR INTERLEAVED MS11-M'S). SYMBOLIC LOCATION 'MKPAT'' IS A TABLE OF TESTS THAT CAN RUN ON EACH BANK OF ECC MEMORY. SYMBOLIC LOCATION 'MJPAT'' IS A TABLE OF TESTS THAT CAN RUN ON EACH BANK OF PARITY MEMORY. SYMBOLIC LOCATION 'FSPAT'' IS A TABLE OF TESTS 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.

EXAMP'E E-TABLE SEGMENT:

```
;THE FOLLOWING LOCATI^NS SPECIFY WHICH TESTS
;ARE TO BE RUN FOR PA.FICULAR MEMORIES
;
;REFERENCE THE TABLE LISTED BELOW TO RELATE BITS TO TESTS.
;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 TESTS        103777 TABLE = MKPAT:
$DDW3:    .WORD       177777          ;ECC TESTS        177777 TABLE = MKPAT:
$DDW4:    .WORD       177777          ;PARITY TESTS     003777 TABLE = MJPAT:
$DDW5:    .WORD       177777          ;PARITY TESTS     177774 TABLE = MJPAT:
```

6.2.2 SPECIFIC TESTS

6.2.2.1 TEST 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 TEST.

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).

2435
2436
2437
2438
2439
2440
2441
2442
2443
2444 6.2.2.2 TEST 1 ADDRESS TEST
2445
2446 WRITES READS AN INCREMENTING PATTERN EQUIVALENT TO PHYSICAL
2447 ADDRESSED INTO A 16K BANK.
2448
2449 IT CAN EXECUTE OUT OF THE USER INSTRUCTION PAR'S.
2450
2451
2452

```
2454
2455
2456
2457
2458
2459                    6.2.2.3  TEST 2    COMPLEMENT ADDRESS TEST
2460
2461                    WRITES THE COMPLEMENT OF THE PHYSICAL ADDRESS FROM HIGH  ADDRESSES  TO
2462                    LOW (WRITE DOWN) AND READS FROM LOW ADDRESSES TO HIGH (READ UP).
2463
2464                    THIS PROVIDES THE COMPLEMENT OF THE COVERAGE OF TEST 1 IN BOTH DATA
2465                    PATTERN AND ADDRESSING SEQUENCE.
2466
2467                    IT CAN EXECUTE OUT OF THE USER INSTRUCTION PAR'S.
```

2469
2470
2471
2472
2473
2474
2475 6.2.2.4 TEST 3 3 XOR 9
2476
2477 WRITES READS A TEST THAT COMPLEMENTS AS ADDRESS BITS 3 AND 9
2478 CHANGE.
2479
2480 THIS TEST IS RUN 4 TIMES (1) WITH ZEROS ONES, (2) WITH ONES
2481 ZEROS, (3) WITH 401 ONES, AND (4) WITH ONES 401. THE TEST OF
2482 THE 401 IS TO FORCE A THE PARITY BITS TO BECOME INVOLVED.
2483
2484 IT CAN EXECUTE OUT OF THE USER DATA PDR'S, THE USER INSTRUCTION PAR'S,
2485 THE KERNEL DATA PAR'S AND THE SUPERVISOR DATA PAR'S.
2486

6.2.2.5 TEST 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 TEST 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 TEST 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.

```
2539
2540
2541
2542
2543
2544
2545                    6.2.2.7  TEST 6    INITIAL DATA TEST
2546
2547                    WRITES  READS A DOUBLE WORD FIRST WITH ALL BITS 0 EXCEPT 1 (FOR EVERY
2548                    BIT  POSITION),  SECOND  WITH  ALL  BITS  1  EXCEPT  1  (FOR EVERY BIT
2549                    POSITION).
2550
2551                    THIS IS A VERY QUICK CHECK OF THE DATA PATHS.
2552
```

```
2554
2555
2556
2557
2558
2559
2560
2561                    6.2.2.8  TEST 7    ADDRESS BIT TEST
2562
2563                    WRITES A BACKGROUND OF ALL ZEROS.
2564                    READ ADDRESS 1 FOR A 0 BYTE.
2565                    COMPLEMENT ADDRESS 1.
2566                    READ ADDRESS 1 FOR A NON 0 BYTE.
2567                    FOR EACH ADDRESS BIT POSITION FROM BIT 1:
2568                    VIRTUAL (2, 4, 10, 20, 40, 100, 200, 400,  1000,  2000,  4000,  10000,
2569                    60000, 20000)
2570                    PHYSICAL (60002, 60004, 60010,  60020,  60040,  60100,  60200,  60400,
2571                    61000, 62000, 64000, 70000, 140000, 100000)
2572                      READ ADDRESS FOR A 0 WORD.
2573                      COMPLEMENT ADDRESS CONTENTS.
2574                      READ ADDRESS FOR A NON-ZERO WORD.
2575
2576                    THIS IS A VERY QUICK CHECK OF THE ADDRESS BIT UNIQUENESS.
2577
2578
```

```
2580
2581
2582
2583
2584
2585
2586                    6.2.2.9  TEST 10   BYTE ADDRESSING TEST
2587
2588                    WITH ECC DISABLED.
2589                    WRITES ALL ONES TO A DOUBLE WORD.
2590                    FOR EACH OF THE 4 BYTES IN THE DOUBLE WORD.
2591                      CLEARS ONE BYTE.
2592                      READS ALL 4 BYTES FROM DOUBLE WORD.
2593                      CHECKS FOR ONLY PROPER BYTE CLEAR.
2594                      ALL OTHER BYTES SET TO ALL ONES.
2595
2596                    THIS IS ONLY DONE ON ONE DOUBLE WORD ADDRESS.
2597
2598
2599                                          NOTE
2600
2601                              THIS IS RUN FOR ECC MEMORY ONLY
2602
2603
```

6.2.2.10 TEST 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 MS11-M MEMORY ONLY

6.2.2.11 TEST 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.
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 MS11-M MEMORY ONLY.

6.2.2.12 TEST 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

1) THIS TEST IS RUN ON THE MS11-M ONLY.

2) 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 TEST 14 BASIC DOUBLE BIT ERROR TEST

1. WRITE THE CSR TO ENABLE DIAG MODE WITH A DOUBLE BIT ERROR CHECK BITS OF 110011 AND UNCORRECTABLE ERROR INDICATION ENABLED.

2. WRITE FIRST AUT IN A 16K BANK WITH DATA OF ALL ZERO'S. THIS WILL WRITE THE CHECK BITS IN (1)

3. READ ADDRESS, THIS SHOULD CAUSE A DOUBLE BIT ERROR. BUS PBL IS ASSERTED AND WE CHECK FOR A PARITY TRAP TO OCCUR.

4. READ THE CSR FOR CHECK BITS IN (1) AND UNCORRECTABLE ERROR INDICATOR.

5. WRITE ONES TO THE HIGH BYTE OF THE ADDRESS UNDER TEST. SINCE A DBE EXSISTS AT THIS ADDRESS THE WRITE SHOULD BE ABORTED.

6. READ ADDRESS AND CHECK FOR A PARITY TRAP TO OCCUR AS A RESULT OF (5)

7. REPEAT 5 AND 6 FOR DATA OF ONES IN THE LOW BYTE AND CHECK FOR WRITE ABORT AND PARITY TRAP.

THIS TEST CHECKS TO SEE IF A DOUBLE BIT ERROR WILL BE ABORTED AND A BYTE WRITE OF A DOUBLE BIT ERROR WILL BE ABORTED.

NOTE

THIS TEST IS ONLY RUN FOR THE MS11-P

6.2.2.14 TEST 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

1) THIS TEST IS RUN ON THE MS11-M ONLY.

2) 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 TEST 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

1) THIS TEST IS RUN ON THE MS11-M ONLY

2) 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 TEST 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 TEST.
3. DO (1-2) AGAIN FOR A COMPLEMENT TEST.

THIS CHECKS THE MEMORY FOR THE CAPABILITY OF HOLDING 0'S 1'S.

6.2.2.17 TEST 20 SYNDROME BITS TO THE CSR ON A SBE TEST

1. WRITE CSR WITH CHECK BITS TO CORRECT BIT 0 OF THE FIRST AUT 16K BANK FROM A 0 TO A 1 WITH DIAG MODE.
2. WRITE AUT WITH DATA OF 0'S CREATING A SBE.
3. CLEAR CSR.
4. READ THE AUT TO CLOCK THE ADDRESS AND SYNDROMES INTO THE CSR.
5. READ THE CSR FOR THE SBE INDICATOR, BIT 4.
6. WRITE THE CSR TO DIAG MODE TO CLOCK THE SYNDROME BITS INTO CSR BITS 5-11.
7. READ THE CSR FOR THE PROPER SYNDROME BITS.
8. REPEAT 1-7 FOR ALL 16 DATA BITS.
9. REPEAT 1-8 FOR DATA OF ONES SO THAT A CORRECTION WILL OCCUR FROM A 1 TO A ZERO.

THIS TEST CHECKS TO SEE THAT THE EDC CHIP CAN DETECT SINGLE BIT ERRORS FOR ALL 16 DATA BITS BY CHECKING FOR CSR BIT#4 AND THAT THE PROPER SYNDROME BITS ARE PLACED IN THE CSR.

NOTE

THIS TEST IS ONLY RUN FOR THE MS11-P

6.2.2.18 TEST 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 TEST 22 REFRESH TEST

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

   THIS TEST 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 TEST 23 SHIFTING DIAGONAL TEST

SIMILAR IN OVERALL OPERATION TO TEST 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 TEST 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 TEST 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 MS11-M MEMORY ONLY.

6.2.2.23 TEST 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 TEST (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 TEST 27 UNIQUE BANK TEST

THIS TEST USES TEST 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 TEST 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 TEST 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 TEST 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 TEST 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 TEST 34 SOFT ERROR TEST

RATIONALIZATION

MOS CHIPS HAVE A FAILURE MODE IN WHICH THEY CAN RANDOMLY PICK OR DROP BITS. THIS IS CAUSED BY ALPHA PARTICLES BOMBARDING THE CELL. IF THE CELL IS VERY SMALL (AND THEY ARE) THEN THE ELECTRONS DISPLACED BY THE ALPHA PARTICLE ARE SUFFICIENT TO CAUSE THE CELL TO CHANGE FROM A ONE TO A ZERO OR FROM A ZERO TO A ONE.

THIS TEST IS CONTROLLED BY THE MAIN PROGRAM SO THAT IT IS USED TO CREATE A TEST OF 125252 AND 52525 ON ALTERNATE PASSES OF THE PROGRAM. THE CONFIGURATION TABLE IS USED TO FLAG BANKS THAT HAVE THE TEST INVALIDATED BECAUSE ANOTHER TEST WAS WRITTEN OVER THIS BACKGROUND.

THIS TEST IS NOTHING MORE THAN A CLEVER USE OF TEST 0.

6.2.2.30 TEST 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 TEST 36 CORRECTION CODE TEST

1. WRITE CSR WITH CHECK BITS TO CORRECT BIT 0 OF THE FIRST ADDRESS IN A 16K BANK FROM A 0 TO A 1 WITH DIAG MODE.
2. WRITE AUT WITH DATA OF 0'S.
3. READ AUT FOR CORRECTION OF BIT 0 FROM A 0 TO A 1.
4. REPEAT 1-3 FOR ALL 16 DATA BITS.
5. REPEAT 1-4 FOR DATA OF ONES SO THAT A CORRECTION WILL OCCUR FROM A 1 TO A ZERO.

THIS TEST CHECKS TO SEE THAT THE EDC CHIP CAN CORRECT SINGLE BIT ERRORS FOR ALL 16K DATA BITS FROM A 1 TO A 0 AND VISA VERSA.

NOTE

THIS TEST IS ONLY RUN FOR THE MS11-P

6.2.2.32 TEST 37 CHECK ECC DISABLE TEST

1. WRITE CSR WITH ECC DISABLE, DIAG MODE, AND SBE CHECK BITS OF 000010.
2. WRITE AUT WITH DATA OF ZERO'S. THIS SHOULD WRITE CHECK BITS TO MEMORY.
3. READ AUT FOR DATA OF ZEROS INSURING NO CORRECTION WAS MADE.

NOTE

THIS TEST IS RUN ON THE MS11-P ONLY.

6.2.2.33 TEST 41 ADDRESS TO CSR ON DBE TEST

1. WRITE CSR WITH ECC DISABLE, DIAG MODE, AND DOUBLE BIT ERROR CHECK BITS OF 010011
2. WRITE AUT WITH DATA OF ZEROS CREATING A DBE.
3. READ AUT TO DETECT DBE AND TO CLOCK ADDRESS INTO CSR
4. READ CSR FOR CORRECT ADDRESS IN BITS 5-11.
5. INCREMENT ADDRESS BY 1K AND REPEAT 1-4 UNTIL 16K IS DONE.

THIS TEST INSURES THAT THE CORRECT ADDRESS APPEARS IN CSR BITS 5-11 ON A DBE

NOTE

THIS TEST IS RUN ON A MS11-P ONLY.

6.2.2.34 TEST 42 EXTENDED ADDRESS TO CSR ON ERROR TEST

1. WRITE CSR WITH SBE CHECK BITS OF 000010 WITH DIAGNOSTIC MODE.
2. WRITE LOW ADDRESS IN A 16K BANK WITH DATA OF ZEROS CREATING A SBE.
3. CLEAR THE CSR.
4. READ ADDRESS TO DETECT SBE.
5. READ CSR FOR CORRECT ADDRESS AND THE SBE INDICATOR BIT #4.
6. ENABLE CSR BIT 14 TO CHECK THE EXTENDED ADDRESS BITS.
7. READ CSR FOR CORRECT ADDRESS BITS
8. REPEAT 1-7 WITH A TEST ADDRESS THAT IS THE HIGHEST IN A 16K BANK.

THIS TEST CHECKS TO SEE THAT THE CORRECT ADDRESS BITS APPPEAR IN THE CSR.
THIS IS ALSO REPEATED FOR THE EXTENDED ADDRESS FUNCTION IN THE CSR.

NOTE

THIS TEST IS ONLY RUN FOR THE MS11-P

6.2.2.35 TEST 43 WRITE BYTE TEST

1. WRITE CSR TO DIAG MODE WITH CHECK BITS OF 001100. THESE CORRESPOND TO DATA OF ZEROS.
2. WRITE FIRST AUT WITH DATA OF ONE IN BIT ZERO. THE WRITE EFFECTIVELY CREATES A SBE IN BYTE 0.
3. CLEAR THE CSR
4. WRITE BYTE 1 OF THE AUT WITH DATA OF ALL ONES.
5. READ CSR TO CHECK FOR SBE INDICATION.
6. WRITE THE CSR TO DIAG MODE.
7. READ THE AUT TO CHECK FOR THE CORRECT DATA -- ALL ONES IN HIGH BYTE AND ALL ZEROS IN LOW BYTE.
8. READ THE CSR TO CHECK FOR CORRECT CHECK BITS CORRESONDING TO THE DATA READ IN (7). THESE CHECK BITS ARE 000110.
9. REPEAT (1)-(8) THIS TIME CREATING AN ERROR IN BYTE 1 (2) AND WRITING BYTE 0 IN (4).

THIS TEST CHECKS TO SEE THAT A SBE WILL BE CORRECTED DURING THE READ PORTION OF THE BYTE WRITE AND THAT CORRECT CHECKBITS WILL BE GENERATED ON THE WRITE.

NOTE

THIS TEST IS ONLY RUN FOR THE MS11-P

6.2.2.36 TEST 44 SHIFTING CHECKBITS THROUGH THE CSR TEST

1. WRITE CSR TO DIAG MODE TO ENABLE CHECKBIT REGISTER.
2. WRITE CSR WITH CHECK BITS OF 000001, ECC DISABLE AND DIAG MODE.
3. WRITE MEMORY WITH DATA OF ZEROS. THIS SHOULD WERITE THE CHECK BITS INTO MEMORY.
4. COMPLEMENT CHECK BITS PATTERN AND WRITE CSR AS IN (2).
5. READ CSR FOR COMPLMENT CHECK BIT PATTERN.
6. READ MEMORY TO READ CHECK BITS WRITTEN IN (2) INTO CSR.
7. READ CSR FOR CORRET CHECK BITS WRITTEN IN (2).
8. SHIFT CHECK BIT PATTERN AND REPEAT (1-7) TILL CSR BITS 5-10 ARE DONE.
9. COMPLEMENT CHECK BIT PATTERN IN (2) AND REPEAT (1-8) SHIFTING A ZERO THROUGH A FIELD OF ONES.

THIS TEST CHECKS THE ABILITY TO READ CHECK BITS FROM THE CSR TO MEMORY AND BACK. THE TEST IS DONE TWICE. ONCE SHIFTING A FIELD OF A ONE THROUGH A FIELD OF ZEROS AND A ZERO THROUGH A FIELD OF ONES. THIS TESTS THE CHECKBIT/SYNDROME BIT REGISTER AND CHECK BIT RAM'S

NOTE

THIS TEST IS ONLY RUN FOR THE MS11-P

6.2.2.37 TEST 45 SYNDROME BITS TO THE CSR ON A DBE TEST

1. WRITE CSR WITH DIAG MODE TO ENABLE CHECK/SYNDROME BIT REGISTER.
2. WRITE CSR WITH DBE CHECK BITS OF 110011 WITH DIAG MODE.
3. WRITE MEMORY WITH DATA OF ZEROS CREATING A DBE.
4. CLEAR CSR.
5. READ MEMORY TO DETECT DBE.
6. READ CSR FOR UNCORRECTABLE ERROR INDICATOR.
7. WRITE CSR TO DIAG MODE TO READ SYNDROME BITS INTO CSR.
8. READ CSR FOR CORRECT SYNDROME BITS OF 111111.
9. REPEAT (1-8) WITH MULIPLE BIT ERROR CHECK BITS OF 111100 AND CORRESPONDING SYNDROME BITS OF 110000.

THIS TEST CHECKS THE ABILITY OF THE CSR TO DETECT A DBE AND READ FOR THE PROPER SYNDROME BITS GENERATED BY THE EDC CHIP. THIS TEST IS THEN REPEATED WITH CHECK BITS CORRESPONDING TO A MULTIPLE BIT ERROR.

NOTE

THIS TEST IS ONLY RUN FOR THE MS11-P

6.2.2.38 TEST 46 CHECK SINGLE BIT ERRORS WITH ECC DISABLED TEST

1. WRITE CSR WITH CHECK BITS TO CORRECT BIT 0 OF THE FIRST ADDRESS IN A 16K BANK FROM A 0 TO A 1 WITH DIAG MODE AND ECC DISABLED.
2. WRITE AUT WITH DATA OF 0'S THUS CREATING A SBE.
3. WRITE THE CSR TO ECC DISABLE.
4. READ AUT TO DETECT SBE.
5. CHECK TO SEE THAT NO TRAP OCCURED.
6. READ CSR TO SEE THAT UNCORRECTABLE ERROR (CSR15) IS SET.
7. REPEAT 1-6 FOR ALL 16 DATA BITS.
8. REPEAT 1-7 FOR DATA OF ONES SO THAT A CORRECTION WILL OCCUR FROM A 1 TO A ZERO.
9. REPEAT 1-8 EXCEPT IN STEPS (3) THE CSR IS WRITTEN TO ECC DISABLE AND BUS PBL ENABLE AND (5) WE CHECK FOR TRAPS.

THIS TEST CHECKS TO SEE THAT SBE ARE TREATED A UNCORRECTABLE ERRORS WITH ECC DISABLE. THE TEST IS REPEATED 2 TIMES, ONCE WITH TRAPS DISABLED AND AGAIN WITH IT ENABLED. THIS IS DONE FOR ALL 16 POSSIBLE SBE CONDITIONS.

NOTE

THIS TEST IS ONLY RUN FOR THE MS11-P

6.2.2.39 TEST 47 NO CSR UPDATE ON SBE WITH EXSISTING DBE TEST

1. WRITE THE CSR TO DIAG MODE TO ENABLE CHECKBIT/SYNDROME BIT REGISTER.
2. WRITE THE CSR WITH DBE CHECK BITS OF 110011 AND DIAG MODE.
3. WRITE MEMORY WITH DATA OF ZEROS CREATING A DBE.
4. WRITE CSR WITH SBE CHECK BITS OF 000010 AND DIAG MODE.
5. WRITE MEMORY 4K ABOVE ADDRESS IN (3) CREATING A SBE.
6. CLEAR CSR.
7. READ MEMORY WITH ADDRESS IN (3) TO DETECT DBE.
8. READ CSR FOR CORRECT ADDRESS AND UNCORRECTABLE ERROR INDICATOR
9. READ MEMORY WITH ADDRESS IN (5) TO DETECT SBE.
10. READ CSR FOR SBE INDICATOR AND NO CHANGE IN DBE STATUS IN CSR IN (8)

THIS TEST CHECKS TO SEE THAT NO UPDATE WILL OCCUR IN THE CSR WITH A SBE IN MEMORY WHEN A DBE ALREADY EXISTS.

NOTE

THIS TEST IS ONLY RUN FOR THE MS11-P

```
3754
3755
3756
3757
3758
3759
3760                    6.2.2.40  TEST 999 NULL TEST
3761
3762                    THIS IS AN INSTANT RETURN ADDED TO PRESERVE THE SOFTWARE STRUCTURE.
3763
3764                    THIS TEST REPLACES ANY REAL TESTS WHEN THE APT E-TABLE DOES  NOT
3765                    SPECIFY A TEST TO BE RUN.
3766
```

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

```
3878                    THIS MAP IS PRINTED OUT IMMEDIATELY AFTER SIZING THE MEMORY UNLESS SW6
3879                    IS SET (REFERENCE SECTION 2.4.1).  IT CAN ALSO BE PRINTED AT ANY LATER
3880                    TIME IN FIELD SERVICE MODE (REFERENCE SECTION 2.4.4.8.7)
3881
3882                    EXAMPLE:
3883                                          MEMORY CONFIGURATION MAP
3884                                                16K BANKS
3885                                 1         2         3         4         5         6         7
3886                          01234567012345670123456701234567012345670123456701234567012345670123
3887                    ERRORS              XX
3888                    INTRLV  ----------------333333333333333----------------------------------
3889                    MEMTYPE LLLLLLLLMMMMMMMMMMMMMMMMMMMMMMMMPPPPPPPPPPPPPPPPPPPPPPPPPPPPPP
3890                    CSR     00000000111111112222222222222222444444444444444444444444444444
3891                    PROTECT PP      I       I       P       I
3892                                 0         1         2         3         4         5         6
3893                          45670123456701234567012345670123456701234567012345670123456701234567
3894                    ERRORS
3895                    INTRLV  ----
3896                    MEMTYPE PPPP
3897                    CSR     4444
3898                    PROTECT
3899
3900
3901                    DISPLAYED ARE BANKS 0-73 OCTAL (2 MEG WORDS).  IF THE FAT TERMINAL
3902                    SWITCH WAS SET (REFERENCE SECTION 2.4.1) THEN ALL BANKS (0-167) WOULD
3903                    BE SHOWN.  IF THIS WAS AN 18-BIT PDP-11 (EG - 11/34), ONLY BANKS 0-7
3904                    WOULD BE PRINTED.
3905                    THE FIELDS:
3906
3907                         ERRORS:
3908
3909                              THE SIZING ROUTINE COULD NOT WRITE ZEROS AND ONES IN BANKS
3910                              10  11, HENCE THEY ARE MARKED AS BAD WITH X'S.
3911
3912
3913
3914                         INTRLV:
3915
3916                              THERE IS INTERLEAVING ON BANKS 20-37, WITH CSR 2 (172104)
3917                              CONTROLLING THE ADDRESS BIT 1 NON-ASSERTED ADDRESSES, AND
3918                              CSR 3 (172106) CONTROLLING THE ADDRESS BIT 1 ASSERTED
3919                              ADDRESSES.
3920
3921                         ERRORS:
3922
3923                         MEMTYPE:
3924
3925                              BANKS 0-7 ARE MEMORY TYPE L (MS-11L), AND BANKS 10-37 ARE
3926                              MEMORY TYPE M (MS11-M) AND BANKS 40-77 ARE MEMORY TYPE
3927                              P(MS11-P).  BANKS 100-167 DO NOT EXIST.
3928
3929                         CSR:
3930
```

BANKS 0-7 ARE ASSIGNED TO CSR 172100, 10-17 TO CSR 172102, AND 20-37 TO INTERLEAVED CSR'S 172104 AND 172106 AND BANKS 40-77 ARE ASSIGNED CSR 17210.

PROTECT:

BANKS 0 AND 1 ARE PROTECTED BECAUSE THEY ARE PROGRAM SPACE. BANK 0 AND 1 CAN ALSO BE PROTECTED BECAUSE THEY ARE IN THE BOTTOM 16K OF AN MS11-M CSR. THE PROTECTION IS HIERARCHICAL AND PROGRAM SPACE OVERSHADOWS MS11-M PROTECTION. BANKS 0 AND 1 WILL NOT BE TESTED UNTIL THE PROGRAM RELOCATES. IF

ANY BANK IS PROTECTED BY MS11-M 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.

```
3985
3986                                    7.4.1  SAMPLE SOURCE FILE -
3987                                           .ENABL ABS
3988                                           .ENABL AMA
3989                                           .MCALL .SUPER
3990                                           .SUPER
3991                                           ;LST$$=0
3992                                           BIT5=40
3993                                    A:     0
3994                                    B:     0
3995                                    C:     0
3996                                    D:     0
3997                                    E:     0
3998                                    F:     0
3999                                    G:     0
4000                                    H:     0
4001                                    I:     0
4002                                    J:     0
4003                                           .PAGE
4004                                    ;LET EXAMPLES
4005                                           LET R0 := A
4006                                           LET B := C + D
4007                                           LET E := F +  1
4008                                           LET G := H +  2
4009                                           LET J := J +  01
4010                                           LET A :B= B
4011                                    ;IF EXAMPLES
4012                                           IF A IS TRUE
4013                                             MOV    23,D
4014                                           END ;OF IF A
4015                                           IF B IS FALSE
4016                                             MOV    34,E
4017                                           END ;OF IF B
4018                                           IF A EQ B THEN LET C := D
4019                                           IF A LT B
4020                                             MOV   C,D
4021                                           ELSE
4022                                             MOV   E,D
4023                                           END ;OF IF A
4024                                           IF A EQ B AND C NE D
4025                                             MOV   F,G
4026                                           END ;OF IF A
4027                                           IF A EQ B OR C NE D
4028                                             MOV   F,G
4029                                           END ;OF IF A
4030                                           IFB A EQ B AND C EQ  1
4031                                             MOV   H,J
4032                                           ELSE
4033                                             MOV   E,J
4034                                           END ;OF IFB A
4035                                           IFB A EQ B ANDB C EQ  1
4036                                             MOV   H,J
4037                                           ELSE
4038                                             MOV   E,J
```

```
                                        END ;OF IFB A
                                        IF RESULT IS EQ
                                          MOV   A,B
                                        END ;OF IF RESULT
                                        IF  BIT5 SET.IN A
                                          MOV   B,C
                                        END ;OF IF  BIT5
                                        IF  BIT5 OFF.IN A
                                          MOV   C,D
                                        END ;OF IF  BIT5
                                ;ON.ERROR IS LIKE AN IF STATEMENT ON THE C-BIT
                                ;ON.ERROR EXAMPLES
                                        ON.ERROR
                                          MOV   A,B
                                        ELSE
                                          MOV   C,B
                                        END ;OF ON.ERROR
                                        ON.NOERROR


                                          MOV   C,B
                                        ELSE
                                          MOV   A,B
                                        END ;OF ON.NOERROR
                                        ON.ERROR THEN LET A :B= B
                                ;FOR EXAMPLES
                                        FOR I :=  -5 TO  23
                                          INC   A
                                        END ;OF FOR I
                                        FOR 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


                                7.4.2  SAMPLE LISTING FILE (WITH NO EXPANDED MACROS) - -
                                .MAIN.  MACRO M1111  01-APR-79 16:41  PAGE 2

                                     1 000000                                   .ENABL ABS
                                     2                                          .ENABL AMA
                                     3                                          .MCALL .SUPER
                                     4 000000                                   .SUPER
                                     5                                          ;LST$$=0
                                     6          000040                          BIT5=40
                                     7 000000   000000                  A:      0
                                     8 000002   000000                  B:      0
                                     9 000004   000000                  C:      0
                                    10 000006   000000                  D:      0
                                    11 000010   000000                  E:      0
                                    12 000012   000000                  F:      0
                                    13 000014   000000                  G:      0
                                    14 000016   000000                  H:      0
                                    15 000020   000000                  I:      0
                                    16 000022   000000                  J:      0


                                .MAIN.  MACRO M1111  01-APR-79 16:41  PAGE 3


                                    18                                  ;LET EXAMPLES
                                    19 000024                                   LET R0 := A
                                    20 000030                                   LET B := C + D
                                    21 000044                                   LET E := F +  1
                                    22 000056                                   LET G := H +  2
                                    23 000072                                   LET J := J +  01
                                    24 000100                                   LET A :B= B
                                    25                                  ;IF EXAMPLES
                                    26 000106                                   IF A IS TRUE
                                    27 000114   012737   000023   000006          MOV    23,D
                                    28 000122                                   END ;OF IF A
                                    29 000122                                   IF B IS FALSE
                                    30 000130   012737   000034   000010          MOV    34,E
                                    31 000136                                   END ;OF IF B
                                    32 000136                                   IF A EQ B THEN LET C := D
                                    33 000154                                   IF A LT B
                                    34 000164   013737   000004   000006          MOV   C,D
                                    35 000172                                   ELSE
```

```
4148                                    36 000174  013737  000010  000006          MOV   E,D
4149                                    37 000202                               END ;OF IF A
4150                                    38 000202                               IF A EQ B AND C NE D
4151                                    39 000222  013737  000012  000014          MOV   F,G
4152                                    40 000230                               END ;OF IF A
4153                                    41 000230                               IF A EQ B OR C NE D
4154                                    42 000250  013737  000012  000014          MOV   F,G
4155                                    43 000256                               END ;OF IF A
4156                                    44 000256                               IFB A EQ B AND C EQ  1
4157                                    45 000276  013737  000016  000022          MOV   H,J
4158                                    46 000304                               ELSE
4159                                    47 000306  013737  000010  000022          MOV   E,J
4160                                    48 000314                               END ;OF IFB A
4161                                    49 000314                               IFB A EQ B ANDB C EQ  1
4162                                    50 000334  013737  000016  000022          MOV   H,J
4163                                    51 000342                               ELSE
4164                                    52 000344  013737  000010  000022          MOV   E,J
4165                                    53 000352                               END ;OF IFB A
4166                                    54 000352                               IF RESULT IS EQ
4167                                    55 000354  013737  000000  000002          MOV   A,B
4168                                    56 000362                               END ;OF IF RESULT
4169                                    57 000362                               IF  BIT5 SET.IN A
4170                                    58 000372  013737  000002  000004          MOV   B,C
4171                                    59 000400                               END ;OF IF  BIT5
4172                                    60 000400                               IF  BIT5 OFF.IN A
4173                                    61 000410  013737  000004  000006          MOV   C,D
4174                                    62 000416                               END ;OF IF  BIT5
4175                                    63                              ;ON.ERROR IS LIKE AN IF STATEMENT ON THE C-BIT
4176                                    64                              ;ON.ERROR EXAMPLES
4177                                    65 000416                               ON.ERROR
4178                                    66 000420  013737  000000  000002          MOV   A,B
4179                                    67 000426                               ELSE
4180                                    68 000430  013737  000004  000002          MOV   C,B
4181                                    69 000436                               END ;OF ON.ERROR
4182                                    70 000436                               ON.NOERROR
4183                                    71 000440  013737  000004  000002          MOV   C,B
4184                                    72 000446                               ELSE
4185                                    73 000450  013737  000000  000002          MOV   A,B
4186                                    74 000456                               END ;OF ON.NOERROR
4187
4188
4189                            .MAIN.  MACRO M1111  01-APR-79 16:41  PAGE 3-1
4190
4191
4192                                    75 000456                               ON.ERROR THEN LET A :B= B
4193                                    76                              ;FOR EXAMPLES
4194                                    77 000466                               FOR I :=  -5 TO  23
4195                                    78 000474  005237  000000               INC   A
4196                                    79 000500                               END ;OF FOR I
4197                                    80 000514                               FOR R0 :=  0 TO  140 BY  4
4198                                    81 000516  005360  000000               DEC   A(R0)
4199                                    82 000522                               END ;OF FOR R0
4200                                    83 000534                               FOR I :=  133 DOWNTO  3 BY  2
```

```
                                     84 000542  063737  000000  000002          ADD   A,B
                                     85 000550                                END ;OF FOR I
                                     86                             ;BEGIN EXAMPLES
                                     87 000566                                BEGIN ALPHA
                                     88 000566                                  FOR R0 :=  0 TO  167
                                     89 000570  116037  000000  000002              MOVB        A(R0),B
                                     90 000576                                    IF B LT  0 THEN LEAVE ALPHA
                                     91 000604                                  END ;OF FOR R0
                                     92 000614                                  FOR R0 :=  400 TO  567
                                     93 000620                                    IF B GE  0 THEN LEAVE ALPHA
                                     94 000626                                  END ;OF FOR R0
                                     95 000636                                END ALPHA
                                     96                             ;$RETURN EXAMPLES
                                     97 000636                                $RETURN
                                     98 000640                                $RETURN ERROR
                                     99 000644                                $RETURN NOERROR
                                    100                             ;CASE EXAMPLES
                                    101 000650  013700  000000                MOV     A,R0
                                    102 000654                                CASE R0
                                    103 000664  000000                          A
                                    104 000666  000002                          B
                                    105 000670  000004                          C
                                    106 000672  000006                          D
                                    107 000674  000010                          E
                                    108 000676  000012                          F
                                    109 000700                                END ;OF CASE R0
                                    110
                                    111         000001                        .END
```

7.4.3 SAMPLE LISTING FILE (WITH EXPANDED MACROS) - -

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

```
                                      1 000000                                .ENABL ABS
                                      2                                       .ENABL AMA
                                      3                                       .MCALL .SUPER
                                      4 000000                                .SUPER
                                      5         000000                        LST$$=0
                                      6         000040                        BIT5=40
                                      7 000000  000000                  A:    0
                                      8 000002  000000                  B:    0
                                      9 000004  000000                  C:    0
                                     10 000006  000000                  D:    0
                                     11 000010  000000                  E:    0
                                     12 000012  000000                  F:    0
                                     13 000014  000000                  G:    0
                                     14 000016  000000                  H:    0
                                     15 000020  000000                  I:    0
                                     16 000022  000000                  J:    0
```

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

```
4255                                    18
4256                                    19 000024
4257                                       000024  013700  000000
4258                                    20 000030
4259                                       000030  013737  000004  000002
4260                                       000036  063737  000006  000002
4261                                    21 000044
4262                                       000044  013737  000012  000010
4263                                       000052  005237  000010
4264                                    22 000056
4265                                       000056  013737  000016  000014
4266                                       000064  062737  000002  000014
4267                                    23 000072
4268                                       000072  062737  000001  000022
4269                                    24 000100
4270                                       000100  113737  000002  000000
4271                                    25
4272                                    26 000106
4273                                       000106  005737  000000
4274                                       000112  001403
4275                                    27 000114  012737  000023  000006
4276                                    28 000122
4277                                       000122
4278                                    29 000122
4279                                       000122  005737  000002
4280                                       000126  001003
4281                                    30 000130  012737  000034  000010
4282                                    31 000136
4283                                       000136
4284                                    32 000136
4285                                       000136  023737  000000  000002
4286                                       000144  001003
4287                                       000146  013737  000006  000004
4288                                       000154
4289                                    33 000154
4290                                       000154  023737  000000  000002
4291                                       000162  002004
4292                                    34 000164  013737  000004  000006
4293                                    35 000172
4294                                       000172  000403
4295                                       000174
4296                                    36 000174  013737  000010  000006
4297                                    37 000202
4298                                       000202
4299                                    38 000202
4300                                       000202  023737  000000  000002
4301                                       000210  001007
4302                                       000212  023737  000004  000006
4303                                       000220  001403
4304                                    39 000222  013737  000012  000014
4305                                    40 000230

                                        ;LET EXAMPLES
                                                LET R0 := A
                                                MOV A,R0
                                                LET B := C + D
                                                MOV C,B
                                                ADD D,B
                                                LET E := F +  1
                                                MOV F,E
                                                INC E
                                                LET G := H +  2
                                                MOV H,G
                                                ADD  2,G
                                                LET J := J +  01
                                                ADD  01,J
                                                LET A :B= B
                                                MOVB B,A
                                        ;IF EXAMPLES
                                                IF A IS TRUE
                                                TST A
                                                BEQ L0
                                                  MOV    23,D
                                                END ;OF IF A
                                        L0:
                                                IF B IS FALSE
                                                TST B
                                                BNE L1
                                                  MOV    34,E
                                                END ;OF IF B
                                        L1:
                                                IF A EQ B THEN LET C := D
                                                CMP A,B
                                                BNE L2
                                                MOV D,C
                                        L2:
                                                IF A LT B
                                                CMP A,B
                                                BGE L3
                                                  MOV   C,D
                                                ELSE
                                                BR L4
                                        L3:
                                                  MOV   E,D
                                                END ;OF IF A
                                        L4:
                                                IF A EQ B AND C NE D
                                                CMP A,B
                                                BNE L5
                                                CMP C,D
                                                BEQ L5
                                                  MOV   F,G
                                                END ;OF IF A
```

```
4307                                    000230                                  L5:
4308                                 41 000230
4309                                    000230  023737  000000  000002          CMP A,B
4310                                    000236  001404                          BEQ L6
4311                                    000240  023737  000004  000006          CMP C,D
4312                                    000246  001403                          BEQ L7
4313
4314
4315                            .MAIN.  MACRO M1111  01-APR-79 16:10  PAGE 3-1
4316
4317
4318                                    000250                                  L6:
4319                                 42 000250  013737  000012  000014            MOV   F,G
4320                                 43 000256                                  END ;OF IF A
4321                                    000256                                  L7:
4322                                 44 000256                                  IFB A EQ B AND C EQ  1
4323                                    000256  123737  000000  000002          CMPB A,B
4324                                    000264  001010                          BNE L10
4325                                    000266  023727  000004  000001          CMP C, 1
4326                                    000274  001004                          BNE L10
4327                                 45 000276  013737  000016  000022            MOV   H,J
4328                                 46 000304                                  ELSE
4329                                    000304  000403                          BR L11
4330                                    000306                                  L10:
4331                                 47 000306  013737  000010  000022            MOV   E,J
4332                                 48 000314                                  END ;OF IFB A
4333                                    000314                                  L11:
4334                                 49 000314                                  IFB A EQ B ANDB C EQ  1
4335                                    000314  123737  000000  000002          CMPB A,B
4336                                    000322  001010                          BNE L12
4337                                    000324  123727  000004  000001          CMPB C, 1
4338                                    000332  001004                          BNE L12
4339                                 50 000334  013737  000016  000022            MOV   H,J
4340                                 51 000342                                  ELSE
4341                                    000342  000403                          BR L13
4342                                    000344                                  L12:
4343                                 52 000344  013737  000010  000022            MOV   E,J
4344                                 53 000352                                  END ;OF IFB A
4345                                    000352                                  L13:
4346                                 54 000352                                  IF RESULT IS EQ
4347                                    000352  001003                          BNE L14
4348                                 55 000354  013737  000000  000002            MOV   A,B
4349                                 56 000362                                  END ;OF IF RESULT
4350                                    000362                                  L14:
4351                                 57 000362                                  IF  BIT5 SET.IN A
4352                                    000362  032737  000040  000000          BIT  BIT5,A
4353                                    000370  001403                          BEQ L15
4354                                 58 000372  013737  000002  000004            MOV   B,C
4355                                 59 000400                                  END ;OF IF  BIT5
4356                                    000400                                  L15:
4357                                 60 000400                                  IF  BIT5 OFF.IN A
4358
4359                                    000400  032737  000040  000000          BIT  BIT5,A
```

```
4361                                    61 000410  013737  000004  000006            MOV   C,D
4362
4363                                    62 000416                                   END ;OF IF  BIT5
4364                                       000416                           L16:
4365                                    63                                  ;ON.ERROR IS LIKE AN IF STATEMENT ON THE C-BIT
4366                                    64                                  ;ON.ERROR EXAMPLES
4367                                    65 000416                                   ON.ERROR
4368                                       000416  103004                           BCC L17
4369                                    66 000420  013737  000000  000002            MOV   A,B
4370                                    67 000426                                   ELSE
4371                                       000426  000403                           BR L20
4372                                       000430                           L17:
4373
4374                                    68 000430  013737  000004  000002            MOV   C,B
4375
4376                                    69 000436                                   END ;OF ON.ERROR
4377                                       000436                           L20:
4378                                    70 000436                                   ON.NOERROR
4379
4380
4381                                .MAIN.  MACRO M1111  01-APR-79 16:10  PAGE 3-2
4382
4383
4384                                       000436  103404                           BCS L21
4385                                    71 000440  013737  000004  000002            MOV   C,B
4386                                    72 000446                                   ELSE
4387                                       000446  000403                           BR L22
4388                                       000450                           L21:
4389                                    73 000450  013737  000000  000002            MOV   A,B
4390                                    74 000456                                   END ;OF ON.NOERROR
4391                                       000456                           L22:
4392                                    75 000456                                   ON.ERROR THEN LET A :B= B
4393                                       000456  103003                           BCC L23
4394                                       000460  113737  000002  000000           MOVB B,A
4395                                       000466                           L23:
4396                                    76                                  ;FOR EXAMPLES
4397                                    77 000466                                   FOR I :=  -5 TO  23
4398                                       000466  012737  177773  000020           MOV  -5,I
4399                                       000474                           B0:
4400                                    78 000474  005237  000000                    INC   A
4401                                    79 000500                                   END ;OF FOR I
4402                                       000500  005237  000020                   INC I
4403                                       000504  023727  000020  000023           CMP I, 23
4404                                       000512  003770                           BLE B0
4405                                       000514                           E0:
4406                                    80 000514                                   FOR R0 :=  0 TO  140 BY  4
4407                                       000514  005000                           CLR R0
4408                                       000516                           B1:
4409                                    81 000516  005360  000000                    DEC   A(R0)
4410                                    82 000522                                   END ;OF FOR R0
4411                                       000522  062700  000004                   ADD  4,R0
4412                                       000526  020027  000140                   CMP R0, 140
```

```
4414                                    000532  003771          BLE B1
4415                                    000534                  E1:
4416                                 83 000534                      FOR I :=  133 DOWNTO  3 BY  2
4417                                    000534  012737  000133  000020  MOV  133,I
4418                                    000542                  B2:
4419                                 84 000542  063737  000000  000002    ADD   A,B
4420                                 85 000550                      END ;OF FOR I
4421                                    000550  162737  000002  000020  SUB  2,I
4422                                    000556  023727  000020  000003  CMP I, 3
4423                                    000564  002366          BGE B2
4424                                    000566                  E2:
4425                                 86                         ;BEGIN EXAMPLES
4426                                 87 000566                      BEGIN ALPHA
4427                                    000566                  B3:
4428                                 88 000566                      FOR R0 :=  0 TO  167
4429                                    000566  005000          CLR R0
4430                                    000570                  B4:
4431                                 89 000570  116037  000000  000002    MOVB        A(R0),B
4432                                 90 000576                      IF B LT  0 THEN LEAVE ALPHA
4433                                    000576  005737  000002  TST B
4434                                    000602  002415          BLT E3
4435                                 91 000604                      END ;OF FOR R0
4436                                    000604  005200          INC R0
4437                                    000606  020027  000167  CMP R0, 167
4438                                    000612  003766          BLE B4
4439                                    000614                  E4:
4440                                 92 000614                      FOR R0 :=  400 TO  567
4441                                    000614  012700  000400  MOV  400,R0
4442
4443
4444  .MAIN.  MACRO M1111  01-APR-79 16:10  PAGE 3-3
4445
4446
4447                                    000620                  B5:
4448                                 93 000620                      IF B GE  0 THEN LEAVE ALPHA
4449                                    000620  005737  000002  TST B
4450                                    000624  002004          BGE E3
4451                                 94 000626                      END ;OF FOR R0
4452                                    000626  005200          INC R0
4453                                    000630  020027  000567  CMP R0, 567
4454                                    000634  003771          BLE B5
4455                                    000636                  E5:
4456                                 95 000636                      END ALPHA
4457                                    000636                  E3:
4458                                 96                         ;$RETURN EXAMPLES
4459                                 97 000636                      $RETURN
4460                                    000636  000207          RTS PC
4461                                 98 000640                      $RETURN ERROR
4462                                    000640  000261          SEC
4463                                    000642  000207          RTS PC
4464                                 99 000644                      $RETURN NOERROR
4465                                    000644  000241          CLC
4466                                    000646  000207          RTS PC
```

```
                        100
                        101 000650  013700  000000          ;CASE EXAMPLES
                        102 000654                                MOV     A,R0
                            000654  010046                        CASE R0
                            000656  006316                        MOV R0,-(SP)
                            000660  004737  000700                ASL @SP
                        103 000664  000000                        JSR PC,L24
                        104 000666  000002                          A
                        105 000670  000004                          B
                        106 000672  000006                          C
                        107 000674  000010                          D
                        108 000676  000012                          E
                        109 000700                                  F
                            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	TES AREA	PROGRAM	DST BK
5	TES 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~

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

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

```
                                .SBTTL  DEFINE  TRAPS
                        ;ALL ENTRIES HERE MUST HAVE A CORRESPONDING ENTRY IN THE
                        ;TRAP TABLE "$TRPAD" (NEAR END OF PROGRAM).
                        ;*TRAP DEFINITIONS
                        ;
                        ;HERE IS HOW TRAPS WORK IN THIS PROGRAM
                        ;
                        ;ALL TRAPS EXECUTE A "TRAP" INSTRUCTION WHICH TAKES THE PROGRAM
                        ;TO SYMBOLIC LOCATION "$TRAP"
                        ;
                        ;AT $TRAP THE PROGRAM PICKS UP THE RIGHT BYTE OF THE TRAP INSTRUCTION
                        ;AND INDEXES INTO A TABLE AT LOCATION "$TRPAD" WHICH SENDS THE PROGRAM TO
                        ;THE SPECIFIC ROUTINE TO HANDLE THAT SPECIFIC TRAPS TASK.
                        ;
                        ;THE ULTIMATE DESTINATION OF A TRAP INSTRUCTION CAN BE GUESSED AT AS FOLLOWS
                        ;
                        ;EXAMPLE:       NOP
                        ;               NOP
                        ;               NOP
                        ;               KERNEL                  ;ENTER KERNEL MODE
                        ;               NOP
                        ;
                        ;       ADD A DOLLAR SIGN TO THE SYMBOLIC NAME AND CHECK THE CRF FOR SOMETHING CLOSE
                        ;       IN THIS CASE THE CRF HAS $KERNE LISTED AS 032546
                        ;       AT LOCATION 32546 YOU FIND THE ROUTINE $KERNEL
                        ;
                        ;       NOTE THAT CRF SYMBOLS ARE TRUCNATED TO 6 CHARACTERS
                        ;       SYMBOLIC NAMES GREATER THAT 6 CHARACTERS ARE USED SO I CAN
                        ;       REMEMBER WHAT THEY MEAN!
                        ;
                        ;       ALL GOOD TRAP ROUTINES RETURN VIA AN 'RTI" INSTRUCTION
104401                  TYPEIT= 104401          ;;TTY TYPEOUT ROUTINE
104402                  TYPOC=  104402          ;;TYPE OCTAL NUMBER (WITH LEADING ZEROS)
104403                  TYPO=   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          ;ENTFR KERNEL MODE

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

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

DEFINE TRAPS

```
4616
4617        104425          LOADCSR=104425          ;LOAD CORRECT CSR
4618        104426          READCSR=104426          ;READ CORRECT CSR
4619
4620        104427          PERR01= 104427          ;PROGRAM DETECTED ERROR
4621        104430          PERR02= 104430          ;PROGRAM DETECTED ERROR
4622        104431          PERR03= 104431          ;PROGRAM DETECTED ERROR
4623        104432          PERR04= 104432          ;PROGRAM DETECTED ERROR
4624        104433          PERR07= 104433          ;PROGRAM DETECTED ERROR
4625        104434          PERR10= 104434          ;PROGRAM DETECTED ERROR
4626        104435          PERR11= 104435          ;PROGRAM DETECTED ERROR
4627        104436          PERR12= 104436          ;PROGRAM DETECTED ERROR
4628        104437          PERR13= 104437          ;PROGRAM DETECTED ERROR
4629        104440          PERR14= 104440          ;PROGRAM DETECTED ERROR
4630        104441          PERR15= 104441          ;PROGRAM DETECTED ERROR
4631        104442          PERR16= 104442          ;PROGRAM DETECTED ERROR
4632        104443          PERR17= 104443          ;PROGRAM DETECTED ERROR
4633        104444          PERR20= 104444          ;PROGRAM DETECTED ERROR
4634        104445          PERR21= 104445          ;PROGRAM DETECTED ERROR
4635        104446          PERR22= 104446          ;PROGRAM DETECTED ERROR
4636        104447          PERR23= 104447          ;PROGRAM DETECTED ERROR
4637        104450          PERR24= 104450          ;PROGRAM DETECTED ERROR
4638        104451          PERR25= 104451          ;PROGRAM DETECTED ERROR
4639        104452          PERR26= 104452          ;PROGRAM DETECTED ERROR
4640        104453          PERR27= 104453          ;PROGRAM DETECTED ERROR
4641        104454          PERR30= 104454          ;PROGRAM DETECTED ERROR
4642        104455          PERR31= 104455          ;PROGRAM DETECTED ERROR
4643        104456          PERR32= 104456          ;PROGRAM DETECTED ERROR
4644        104457          PERR33= 104457          ;PROGRAM DETECTED ERROR
4645        104460          PEPR34= 104460          ;PROGRAM DETECTED ERROR
4646        104461          PERR35= 104461          ;PROGRAM DETECTED ERROR
4647        104462          PERR36= 104462          ;PROGRAM DETECTED ERROR
4648        104463          PERR37= 104463          ;PROGRAM DETECTED ERROR
4649        104464          PERR40= 104464          ;PROGRAM DETECTED ERROR
4650        104465          PERR41= 104465          ;PROGRAM DETECTED ERROR
4651        104466          PERR42= 104466          ;PROGRAM DETECTED ERROR
4652        104467          PERR43= 104467          ;PROGRAM DETECTED ERROR
4653
4654        104470          ECCDIS= 104470          ;DISABLE ECC ON ALL CSR'S
4655        104471          ECC1DIS=104471          ;DISABLE ECC ON 1 SELECTED CSR
4656        104472          ECCINIT=104472          ;INITIALIZE ALL ECC CSR'S
4657        104473          ECC1INIT=104473         ;INITIALIZE 1 SELECTED ECC CSR
4658        104474          CBCSR=  104474          ;WRITE GENERATED CHECKBITS IN ALL CSR'S
4659        104475          CB1CSR= 104475          ;WRITE GENERATED CHECKBITS IN 1 SELECTED CSR
4660        104476          WASSBE= 104476          ;WAS THERE A SBE ON ANY CSR?
4661        104477          WAS1SBE=104477          ;WAS THERE A SBE ON 1 SELECTED CSR?
4662        104500          WASDBE= 104500          ;WAS THERE A DBE ON ANY CSR?
4663        104501          WAS1DBE=104501          ;WAS THERE A DBE ON 1 SELECTED CSR?
4664        104502          CLRCSR= 104502          ;CLEAR ALL CSR'S
4665        104503          CLR1CSR=104503          ;CLEAR 1 SELECTED CSR
4666        104504          CHKDIS= 104504          ;DISABLE ECC & WRITE CHECKBITS FROM ALL CSR'S
4667        104505          CHK1DIS=104505          ;DISABLE ECC & WRITE CHECKBITS FROM 1 SELECTED CSR
4668        104506          ENASBE= 104506          ;ENABLE TRAPS ON SBE'S FROM ALL CSR'S
4669        104507          ENA1SBE=104507          ;ENABLE TRAPS ON SBE'S FROM 1 SELECTED CSR
4670        104510          TSTREAD=104510          ;TEST LOC (R1) & TST FOR SBE (WITHOUT FETCHES)
4671        104511          INVALID=104511          ;INVALIDATE BACKGROUND PATTERN ON 'BANK''
4672        104512          ERRGEN =104512          ;CHECK ERROR ADDRESS
```

DEFINE TRAPS

```
   4673         104513                  CBREG  =104513          ;ENABLES CHECK/SYNDROME BIT REGISTER
```

```
                                .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
```

```
4732          000200                BIT7=   200
4733          000100                BIT6=   100
4734          000040                BIT5=   40
4735          000020                BIT4=   20
4736          000010                BIT3=   10
4737          000004                BIT2=   4
4738          000002                BIT1=   2
4739          000001                BIT0=   1
4740
4741                                 *BASIC ''CPU'' TRAP VECTOR ADDRESSES
4742          000004                ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
4743          000010                RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
4744                                ;TBITVEC=14             ;;''T'' BIT
4745                                ;TRTVEC=        14              ;;TRACE TRAP
4746                                ;BPTVEC=        14              ;;BREAKPOINT TRAP (BPT)
4747          000020                IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
4748          000024                PWRVEC= 24              ;;POWER FAIL
4749          000030                EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
4750          000034                TRAPVEC=34              ;;''TRAP'' TRAP
4751          000060                TKVEC=  60              ;;TTY KEYBOARD VECTOR
4752                                ;TPVEC= 64              ;;TTY PRINTER VECTOR
4753                                ;LKVEC= 100             ;;LINE CLOCK (KW11-L) VECTER
4754          000114                CACHVEC=114             ;;CACHE ERROR INTERRUPT VECTOR
4755          000114                PARVEC=CACHVEC
4756                                ;PIRQVEC=240            ;;PROGRAM INTERRUPT REQUEST VECTOR
4757          000250                MMVEC=  250             ;;MEMORY MANAGEMENT VECTOR
4758                                        .SBTTL  DEFINE  CACHE REGISTERS
4759                                ;MEMERR = 177744                ;;CACHE ERROR REGISTER
4760          177746                CONTRL = 177746                 ;;MEMORY CONTROL REGISTER
4761          177750                MAINT  = 177750                 ;;MEMORY MAINTENENCE REGISTER
4762                                ;HITMIS = 177752                ;;HIT MISS REGISTER ''1'' IMPLIES HIT IN CACHE
4763          177754                DATARG  = 177754                ;;DATA REGISTER
4764
4765                                        .SBTTL  DEFINE  CPU REGISTERS
4766          177766                CPUERR = 177766         ;;CPU ERROR REGISTER HOLDS CONDITION THAT CAUSED
4767
4768                                        .SBTTL  DEFINE  MEMORY MANAGEMENT REGISTERS
4769                                        ;*MEMORY MANAGEMENT STATUS REGISTER ADDRESSES
4770          177572                MMR0=   177572
4771          177574                MMR1=   177574
4772          177576                MMR2=   177576
4773          172516                MMR3=   172516
4774
4775                                        ;*USER ''I'' PAGE DESCRIPTOR REGISTERS
4776          177600                UIPDR0= 177600
4777                                ;UIPDR1=        177602
4778                                ;UIPDR2=        177604
4779                                ;UIPDR3=        177606
4780                                ;UIPDR4=        177610
4781                                ;UIPDR5=        177612
4782                                ;UIPDR6=        177614
4783                                ;UIPDR7=        177616
4784
4785                                        ;*USER ''D'' PAGE DESCRIPTOR REGISTORS
4786                                ;UDPDR0=        177620
4787                                ;UDPDR1=        177622
4788                                ;UDPDR2=        177624
```

DEFINE MEMORY MANAGEMENT REGISTERS

```
                                ;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
```

DEFINE MEMORY MANAGEMENT REGISTERS

```
4846                                    ;*SUPERVISOR 'D" PAGE ADDRESS REGISTERS
4847            172260          SDPAR0= 172260
4848                            ;SDPAR1=        172262
4849                            ;SDPAR2=        172264
4850                            ;SDPAR3=        172266
4851                            ;SDPAR4=        172270
4852            172272          SDPAR5= 172272
4853            172274          SDPAR6= 172274
4854            172276          SDPAR7= 172276
4855
4856                                    ;*KERNEL "I" PAGE DESCRIPTOR REGISTERS
4857            172300          KIPDR0= 172300
4858                            ;KIPDR1=        172302
4859                            ;KIPDR2=        172304
4860                            ;KIPDR3=        172306
4861                            ;KIPDR4=        172310
4862                            ;KIPDR5=        172312
4863                            ;KIPDR6=        172314
4864                            ;KIPDR7=        172316
4865
4866                                    ;*KERNEL 'D" PAGE DESCRIPTOR REGISTERS
4867                            ;KDPDR0=        172320
4868                            ;KDPDR1=        172322
4869                            ;KDPDR2=        172324
4870                            ;KDPDR3=        172326
4871                            ;KDPDR4=        172330
4872                            ;KDPDR5=        172332
4873                            ;KDPDR6=        172334
4874                            ;KDPDR7=        172336
4875
4876                                    ;*KERNEL "I" PAGE ADDRESS REGISTERS
4877            172340          KIPAR0= 172340
4878                            ;KIPAR1=        172342
4879                            ;KIPAR2=        172344
4880                            ;KIPAR3=        172346
4881            172350          KIPAR4= 172350
4882            172352          KIPAR5= 172352
4883            172354          KIPAR6= 172354
4884                            ;KIPAR7=        172356
4885
4886                                    ;*KERNEL 'D" PAGE ADDRESS REGISTERS
4887            172360          KDPAR0= 172360
4888                            ;KDPAR1=        172362
4889                            ;KDPAR2=        172364
4890                            ;KDPAR3=        172366
4891                            ;KDPAR4=        172370
4892                            ;KDPAR5=        172372
4893            172374          KDPAR6= 172374
4894            172376          KDPAR7= 172376
4895
```

DEFINE UNIBUS MAP REGISTERS

```
                                        .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
```

DEFINE UNIBUS MAP REGISTERS

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

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

                                        .SBTTL  DEFINE  CONTROL STATUS REGISTERS
172100                          CSRADD=172100

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

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

                                        .SBTTL  MACRO   TYPE
                                        .MACRO  TYPE    ARG
                                        .NLIST
                                        .DSABL  CRF
                                        .IIF DF LST$$ .LIST  ME
                                        .ENABL  CRF
                                        .LIST
                                        .IF B ARG
                                        TYPEIT
                                        .IFF
                                        TYPEIT  ,ARG
                                        .ENDC
                                        .DSABL  CRF
                                        .IIF DF LST$$ .NLIST  ME
                                        .ENABL  CRF
                                        .ENDM   TYPE
```

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

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

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

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

                        .SBTTL  MACRO   $SUBTST
                        .MACRO  $SUBTST ASC
                        .IRP    ASCI,<ASC>
                        .SBTTL  ASCI
                        .NLIST
                        .LIST   ME
                        .LIST
                ;*******************************************************************************************
                ;*SUBTEST        ASCI
                        .ENDM
                ;*******************************************************************************************
                        .NLIST  ME
                        .ENDM   $SUBTST
```

```
                        .SBTTL  MACRO   TYPOCT
                ;********************* TYPOCT *********************
                ;
                ;TYPOCT IS USED TO CHANGE A BINARY NUMBER
                ;       TO A 6 DIGIT OCTAL NUMBER AND TYPE IT
                ;
                ;ARGUMENTS:
                ;
                ;1)     NUM     THE NUMBER TO BE TYPED
                ;
                ;2)     REMARK  ALLOWS A COMMENT TO BE MADE
                ;
                ;ROUTINES REQUIRED
                ;
                ;1)     CONVERT BINARY TO OCTAL AND TYPE (.$TYPOCT)
                ;
                ;2)     TYPE AN ASCIZ STRING (.$TYPE)
                ;
                ;EXAMPLES:
                ;
                ;1)     TYPOCT  HILMT,<TYPES THE CONTENTS OF HILMT>
                ;2)     TYPOCT  #5,<TYPES ' 000005'>
                ;
                ;*****************************************************************

                        .MACRO  TYPOCT  NUM,REMARK
                        .NLIST
                        .DSABL  CRF
                        .IIF DF LST$$ .LIST  ME
                        .ENABL  CRF
                        .LIST
                        MOV     NUM,-(SP)       ;;SAVE NUM FOR TYPEOUT
                        .IIF NB <REMARK>,                       ;;REMARK
                        TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
                        .DSABL  CRF
                        .IIF DF LST$$ .NLIST  ME
                        .ENABL  CRF
                        .ENDM   TYPOCT
```

```
                        .SBTTL  MACRO   TYPOCS
                ;******************** TYPOCS ********************
                ;
                ;TYPOCS IS USED TO CHANGE A BINARY NUMBER TO AN OCTAL
                ;       NUMBER AND TYPE 1 TO 6 DIGITS
                ;       WITH OR WITHOUT LEADING ZEROS.
                ;
                ;ARGUMENTS:
                ;
                ;1)     NUM     NUMBER TO BE TYPED
                ;
                ;2)     REMARK  ALLOWS A COMMENT TO BE MADE
                ;
                ;3)     N       NUMBER OF DIGITS (1 TO 6) TO BE TYPED
                ;
                ;4)     Z       BLANK=SUPPRESS LEADING ZEROS (TYPES SPACES)
                ;               NON-BLANK=TYPE LEADING ZEROS
                ;
                ;ROUTINES REQUIRED
                ;
                ;1)     CONVERT BINARY TO OCTAL AND TYPE (.$TYPOCT)
                ;
                ;2)     TYPE AN ASCIZ STRING (.$TYPE)
                ;
                ;EXAMPLES:
                ;1)     TYPOCS  #12345,<TYPES '5'>,1
                ;2)     TYPOCS  #004,<TYPES '04'>,2,X
                ;3)     TYPOCS  #004,<TYPES ' 4'>,2
                ;
                ;*************************************************************

                .MACRO  TYPOCS  NUM,REMARK,N,Z
                .NLIST
                .DSABL  CRF
                .IIF DF LST$$ .LIST  ME
                .ENABL  CRF
                .LIST
                MOV     NUM,-(SP)       ;;SAVE NUM FOR TYPEOUT
                .IIF NB <REMARK>,                               ;;REMARK
                TYPOS                   ;;GO TYPE--OCTAL ASCII
                .IF NB N
                .BYTE   N               ;;TYPE N DIGIT(S)
                .IFF
                .BYTE   6               ;;TYPE 6 DIGITS
                .ENDC
                .IF NB Z
                .BYTE   1               ;;TYPE LEADING ZEROS
                .IFF
                .BYTE   0               ;;SUPPRESS LEADING ZEROS
                .ENDC
                .DSABL  CRF
                .IIF DF LST$$ .NLIST  ME
                .ENABL  CRF
                .ENDM   TYPOCS
```

```
                        .SBTTL  MACRO   TYPDEC
                ;******************** TYPDEC ********************
                ;
                ;TYPDEC IS USE TO CHANGE A BINARY NUMBER TO A SIGNED
                ;       DECIMAL NUMBER AND TYPE IT REPLACING LEADING ZERO
                ;       WITH SPACES.
                ;NOTE:  IF THE NUMBER IS NEGATIVE A
                ;       MINUS SIGN WILL BE TYPED.
                ;
                ;ARGUMENTS:
                ;
                ;1)     NUM     NUMBER TO BE TYPED
                ;
                ;2)     REMARK ALLOWS A COMMENT TO BE MADE
                ;
                ;ROUTINES REQUIRED
                ;
                ;1)     CONVERT BINARY TO DECIMAL AND TYPE (.$TYPDEC)
                ;
                ;2)     TYPE AN ASCIZ STRING (.$TYPE)
                ;
                ;EXAMPLES
                ;
                ;1)     TYPDEC  SIZE,<TYPE THE CONTENTS OF SIZE>
                ;2)     TYPDEC  #-10.,<TYPE A MINUS TEN>
                ;
                ;*****************************************************************

                .MACRO  TYPDEC  NUM,REMARK
                .NLIST
                .DSABL  CRF
                .IIF DF LST$$ .LIST  ME
                .ENABL  CRF
                .LIST
                MOV     NUM,-(SP)       ;;SAVE NUM FOR TYPEOUT
                .IIF NB <REMARK>,                               ;;REMARK
                TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
                .DSABL  CRF
                .IIF DF LST$$ .NLIST  ME
                .ENABL  CRF
                .ENDM   TYPDEC
```

```
                                        .SBTTL  MACRO   BMOV
                                ;****************** BMOV ****************************
                                ;
                                ; THIS MACRO MOVES A BLOCK OF DATA.
                                ;
                                ;ARGUEMENTS:
                                ;
                                ;1)      FROMHERE        THE FIRST ADDRESS OF THE SOURCE BLOCK.
                                ;
                                ;2)      TOHERE          THE FIRST ADDRESS OF THE DESTINATION BLOCK.
                                ;                        IF BLANK THE 1ST ADDRESS OF THE USER INSTRUCTION
                                ;                        PAR'S IS USED (FASTCITY).
                                ;
                                ;3)      SIZE            THE SIZE OF THE SOURCE BLOCK.
                                ;                        IF BLANK A 16 WORD TRANSFER IS ASSUMED.
                                ;                        "WHY DEFAULT TO 16 WORDS?" YOU ASK!
                                ;                        "BECAUSE THAT'S HOW MANY WORDS TO THE USER PAR
                                ;                        REGISTERS & THAT'S WHERE I INTEND TO MOVE LOTS
                                ;                        OF STUFF." I REPLY!
                                ;
                                ;*****************************************************

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

MACRO   BMOV

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

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

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

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

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

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

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

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

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

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

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

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

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

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

```
                                                .SBTTL  TRAP CATCHER
                        000000                  .=0
000000  000000  000000                          .WORD   0,0
                        000177                  .REPT   177             ;.WORD  .+2,HALT
                                                .SBTTL  ACT11 HOOKS
                                        ;*THE HOOKS REQUIRED BY ACT11 ARE DEFINED AND SETUP BELOW:
                                        ;*      DEFINITIONS:
                                        ;*                      1)LOC.46        "END-OF-PASS" HOOK
                                        ;*                              =ADDRESS OF END OF PASS ROUTINE
                                        ;*                               MODIFIED BY ACT11.
                                        ;*                      2)LOC.52        PROGRAM NEEDS HOOK
                                        ;*                              BIT 15=1 PROGRAM SHOULD BE POWER
                                        ;*                                       FAILED WHILE RUNNING
                                        ;*                                     =0 NO POWER FAIL
                                        ;*                              BIT 14=1 PROGRAM MEMORY SIZE DEPENDENT
                                        ;*                                     =0 NOT MEMORY SIZE DEPENDENT
                                        ;*                              BIT 13=1 PROGRAM REQUIRES MANUAL INTERVENTION
                                        ;*                                     =0  MANUAL INTERVENTION NOT REQUIRED
                                        ;*                              BITS 12-0  MUST BE ZERO'S
                        000046                  .=46
000046  015232                          $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  065740                          $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  002612          START1: CLR     RESTART
000304  000137  003654                  JMP     START
000310                          START2: SET     RESTART
000316  000137  003654                  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                  PMEMFLG:.WORD   0               ;"MEMORY UNDER TEST IS A MS11-P" FLAG
002142  000000                  ABORTFLAG:.WORD 0               ;"ABORT OCCURED" FLAG
002144  000000                  CTLKVEC:.WORD   0               ;HOLDS OLD KERNAL STACK POINTER IN CASE OF CNTL/K
```

```
VARIABLES       INITIALIZED TO ZERO

   5602 002146  000000                    CSR:     .WORD   0               ;DATA TO OR FROM CSR
   5603 002150  000000                    CSRNO:   0                       ;CSR ADDRESS NUMBER (4 LSB'S)
   5604 002152  000000                    SAVCSR: .WORD    0               ;LOCATION TO SAVE CSRNO DURING FS COMMAND
   5605 002154  000000                    OLDCSR: .WORD    0               ;OLD CSR NUMBER(USED IN INH PTR TEST)
   5606                                                    ;THESE LOCATIONS STORE GPR'S DURING SUPERVISOR TESTS
   5607 002156  000000                    SUPDR0: 0
   5608 002160  000000                    SUPDR1: 0
   5609 002162  000000                    SUPDR2: 0
   5610 002164  000000                    SUPDR3: 0
   5611 002166  000000                    SUPDR4: 0
   5612 002170  000000                    SUPDR5: 0
   5613 002172  000000                    SUPDR6: 0
   5614 002174  000000                    DUMMY:   0                       ;DUMMY LOCATION FOR ADDRESS PASSING
   5615                                    ;THESE LOCATIONS STORE GPR'S & PSW DURING DETAILED ERROR PRINTOUTS
   5616 002176  000000                    DETR0:   0
   5617 002200  000000                    DETR1:   0
   5618 002202  000000                    DETR2:   0
   5619 002204  000000                    DETR3:   0
   5620 002206  000000                    DETR4:   0
   5621 002210  000000                    DETR5:   0
   5622 002212  000000                    DETSP:   0
   5623 002214  000000                    DETPSW:  0
   5624 002216  000000                    DETFLAG:0                        ;DETAILED REPORT FLAG
   5625 002220  000000                    CONTFLAG:0                       ;CSR'S HAVE BEEN TESTED FLAG
   5626 002222  000000                    TOTCSRS:.WORD    0               ;1 BIT PER EXISTING CSR, EG-
   5627                                                                    ;CSR 0 REPRESENTED BY BIT 15, ETC.
   5628 002224  000000                    CSRFIRST:.WORD   0               ;FIRST ADDRESS UNDER CONTROL OF THIS CSR
   5629 002226  000000                    CSRLAST: .WORD   0               ;
   5630 002230  000000                    CSRFBANK:.WORD   0               ;
   5631 002232  000000                    CSRLBANK:.WORD   0               ;
   5632 002234  000000                    CSRINT:   .WORD  0               ;
   5633 002236  000000                    SPLTCSR: .WORD   0               ;
   5634 002240  000000  000000            DATBUF: .WORD    0,0             ;TWO WORD DATA BUFFER
   5635 002244  000000  000000            TSTDAT: .WORD    0,0             ;TWO WORD TEST DATA
   5636 002250  000000  000000            SBEMSK: .WORD    0,0             ;TWO WORD SINGLE BIT ERROR MASK
   5637 002254  000000  000000            DBEMSK: .WORD    0,0             ;TWO WORD DOUBLE BIT ERROR MASK
   5638 002260  000000                    SUPDOADD:.WORD   0               ;ADDRESS OF SUBROUTINE TO EXECUTE IN SUPERVISOR MODE
   5639 002262     000                    PASFLG: .BYTE    0               ;LOCAL LOOP PASS CONTROL
   5640 002263     000                    UPPFLG: .BYTE    0               ;LOCAL LOOP PASS CONTROL
   5641 002264  000000                    PASSNO: .WORD    0               ;LOCAL LOOP PASS CONTROL
   5642 002266  000000                    SAVPAR: .WORD    0               ;USED TO SAVE KERNAL PAR 5
   5643 002270  000000                    SAVMON: .WORD    0               ;XXDP MONITOR RETURN ADDRESS
   5644 002272  000000                    MONFLG: .WORD    0               ;RETURN TO MONITOR FLAG
   5645 002274  000000                    REALPAT:.WORD    0               ;REAL PATTERN UNDER TEST
   5646 002276  000000                    OLDCACHE:.WORD   0               ;BACKED UP VALUE OF CACHE CONTROL REGISTER
   5647 002300  000000                    PARTHERE:.WORD   0               ;PARITY TRAPS SOMETIMES GO TO ADDRESS STORED HERE
   5648 002302  000000                    FSSTACK:.WORD    0               ;STACK SAVED HERE IF IN FIELD SERVICE MODE
   5649 002304  000C00                    NEWBANK:.WORD    0               ;USED FOR RELOCATION TO A NEW BANK
   5650 002306  000000                    SOURCE: .WORD    0               ;SOURCE OF DATA WORDS FOR CHECKBIT GENERATION SUBROUTINE
   5651 002310  000000                    CHECK:   .WORD   0               ;CHECKBITS TO BE LOADED INTO CSR
   5652 002312  000000                    CBITS:   .WORD   0               ;CHECK BITS TO BE WRITTEN
   5653 002314  000000                    MASK:    .WORD   0               ;BIT MASK FOR CSR
   5654 002316  000000                    CSR1S:   .WORD   0               ;CSR ALL 1'S PATTERN
   5655 002320  000000                    BITNO:   .WORD   0               ;BIT POINTER
   5656 002322  000000                    PCBUMP: .WORD    0               ;VALUE TO BUMP THE PC BY TO RECOVER AFTER A PARITY TRAP
   5657 002324  000000                    CSRINC: .WORD    0               ;VALUE TO INCREMENT ADDRESS BY TO REMAIN IN THE SAME CSR
   5658 002326  000000                    CSRLOOP:.WORD    0               ;LOOP CONTROL FOR CSR TESTING
```

```
VARIABLES       INITIALIZED TO ZERO

 5659 002330  000000                  SUCCESS:.WORD   0               ;FLAG SET BY SUCCESSFULL TASK OR SUBROUTINE
 5660 002332  000000                  ZEROS:  .WORD   0               ;FOR AID IN 'MOV" INSTRUCTIONS
 5661 002334  000000                  TIME:   .WORD   0               ;SECONDS THAT BATTERIES SHOULD LAST
 5662 002336  000000                  SKIPMK: .WORD   0               ;FLAG TO SKIP MKCONTROL SUBROUTINE
 5663 002340  000000                  NULLFLAG:.WORD  0               ;SET WHEN RUNNING NULL PATTERNS
 5664 002342  000000                  QVFLAG: 0                       ;FLAGS QUICK VERIFY PASS UNDER APT, ACT, OR XXDP CHAIN MODE
 5665 002344  000000                  ACTFLAG:0                       ;FLACS ACT AUTOMATIC MODE PROGRAMMING RULES
 5666 002346  000000                  APTFLAG:0                       ;FLAG. APT AUTOMATIC MODE PROGRAMMING RULES
 5667 002350  000000                  XXDPCHAIN:0                     ;FLAGS XXDP CHAIN MODE PROGRAMMING RULES
 5668                                  ;NOTE: THESE TWO BYTES MUST STAY TOGETHER
 5669 002352     000                  $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER FOR FILLS
 5670 002353     000                  $FILLS: .BYTE   0               ;CONTAINS # OF FILL CHARACTERS
 5671 002354     000                  $TPFLG: .BYTE   0               ;;"TERMINAL NOT AVAILABLE" FLAG
 5672                                  .EVEN
 5673 002356  000000                  $ESCAPE:0                       ;;ESCAPE ON ERROR ADDRESS
 5674 002360  000000                  EVEN:   0                       ;USED FOR ALTERNATE DATA PATTERNS
 5675 002362  000000                  STRIPES:0                       ;COUNTS DIAGONAL STRIPES
 5676 002364  000000                  COUNT:  0                       ;BACKED UP COPY OF STRIPES
 5677 002366  000000                  NOTAB:  0                       ;NO TABLE BEING PRINTED - NOW
 5678 002370  000000                  BSIZE:  0                       ;SIZE OF 11/45 MOS MEMORY IN K WORDS
 5679 002372  000000                  KSIZE:  0                       ;SIZE OF MF11S-K MEMORY IN K WORDS
 5680 002374  000000                  LSIZE:  0                       ;SIZE OF MS11-L MEMORY IN K WORDS
 5681 002376  000000                  MSIZE:  0                       ;SIZE OF MS11-M MEMORY IN K WORDS
 5682 002400  000000                  PSIZE:  0                       ;SIZE OF UNIBUS PARITY MEMORY IN K WORDS
 5683 002402  000000                  TOOMANY:0                       ;FLAGS WHEN TOO MANY ERRORS HAVE BEEN PRINTED FOR A BANK
 5684 002404  000000                  READONLY:0                      ;FLAG TO PATTERNS TO READ ONLY
 5685 002406  000000  000000          TESTADD:0,0                     ;THE ADDRESS TO RUN CSR TESTS ON
 5686 002412  000000                  UNITOP: 0                       ;HIGHEST ACCESSABLE BANK OF MEMORY THRU UNIBUS MAP
 5687 002414  000000                  STOPOK: 0                       ;FLAG TO ALLOW STOPPING WITH SWITCH REGISTER
 5688 002416  000000                  APTPAR: .WORD   0               ;AMOUNT OF PARITY MEMORY ACCORDING TO APT
 5689 002420  000000                  APTECC: .WORD   0               ;AMOUNT OF ECC MEMORY ACCORDING TO APT
 5690 002422  000000                  NOFSMODE:0                      ;FLAG TO DISABLE FIELD SERVICE MODE
 5691 002424  000000                  NOERROR:0                       ;"THIS IS NOT AN ERROR" FLAG
 5692 002426  000000                  LOADBANK:0                      ;BANK LOADERS ARE RELOCATED TO
 5693 002430  000000                  TEMP:   0                       ;USED FOR JUNK
 5694 002432  000000                  QUICK:  0                       ;QUICK STOP FLAG FOR APT POWER FAIL
 5695 002434  000000                  NOSCOPE:0                       ;"NO SCOPE LOOP ALLOWED" FLAG
 5696 002436  000000                  FSINFLAG:0                      ;"FIELD SERVICE - NO INTERNAL INTERLEAVE" FLAG
 5697 002440  000000                  APTSIZE:0                       ;APT SIZING INFO FLAG
 5698 002442  000000                  FS7FLAG:0                       ;TRUE WHEN IN FIELD SERVICE COMMAND 7
 5699 002444  000000                  CONFGERROR:0                    ;CONFIGURATION ERROR FLAG
 5700 002446  000000                  I:      0                       ;USED FOR GENERAL PURPOSE INDEXING
 5701 002450  000000                  NO22BIT:0                       ;NO 22-BIT MODE FLAG
 5702 002452  000000                  NOSUPER:0                       ;NO SUPERVISOR MODE FLAG
 5703 002454  000000                  ERRADD: .WORD   0               ;HOLDS THE CSR'S ERROR ADDRESS
 5704 002456  000000  000000  000000  CSRINFO:0,0,0,0,0,0,0,0         ;USED TO STORE INFORMATION ABOUT THE 16
      002464  000000  000000  000000
      002472  000000  000000
 5705 002476  000000  000000  000000          0,0,0,0,0,0,0,0         ;POSSIBLE CSR'S
      002504  000000  000000  000000
      002512  000000  000000
 5706 002516  000000                  LINK1:  0                       ;USED TO HOLD LINKS TO PATTERNS WHICH
 5707 002520  000000                  LINK2:  0                       ;CAN EXECUTE IN THE PAR/PDR'S OR NOT
 5708 002522  000000                  CSRHOLD:0                       ;USED TO STORE CSR VALUES FOR CSR TESTS
 5709 002524  000000                  KFLAG:  0                       ;USED TO FLAG MF11S-K MEMORY TO TESTS
 5710 002526  000000  000000          PGMCSR: .WORD   0,0             ;POINTS TO PROGRAM CSR
 5711 002532  000000                  INHECC: .WORD   0               ;FLAGS INHIBIT ECC TESTS ON RELOCATION
```

VARIABLES INITIALIZED TO ZERO

```
   5712 002534  000000                INHBANK:.WORD   0               ;
   5713 002536  000000                FULLREL:.WORD   0               ;
   5714 002540                        $CMTGE: ;*END OF COMMON TAGS
```

VARIABLES INITIALIZED TO NON ZERO

```
                                        .SBTTL  VARIABLES       INITIALIZED TO NON ZERO
002540  000001  000000          CACHKN: 1,0                     ;CACHE CONSTANT (MOVED TO CONTRL TO TURN ON CACHE)
002544  001415                  CACHKF: 1415                    ;CACHE CONSTANT (MOVED TO CONTRL TO TURN OFF CACHE)
002546  040000                  TESTMODE:40000                  ;USED TO SELECT THE PROPER TEST MODE FOR A PATTERN RUN
002550  000012                  ERRMAX: 10.                     ;MAX # OF ERRORS PER BANK WITH SW11
002552  000167                  LASTBANK:167                    ;HIGHEST BANK OF MEMORY
002554  170000                  LASTBLOCK:170000                ;HIGHEST BANK OF MEMORY+1 (IN PAR FORMAT)
002556  000031                  SOBK:   25.                     ;SOB CONSTANT
002560  002000                  KSTACK: STACK                   ;STACK BEGINNING
002562  000001                  LOADHOME:1                      ;HOME BANK OF LOADERS
002564  177777                  WORST:  177777                  ;SET IF TESTING BANKS IN WORST FIRST MODE(1ST PASS)
002566  176543                  SEEDHI: 176543                  ;WORKING SEED HI (USED FOR RANDOM NUMBER GENERATOR)
002570  123456                  SEEDLO: 123456                  ;WORKING SEED LO (USED FOR RANDOM NUMBER GENERATOR)
002572  176543                  MSEEDH: 176543                  ;MASTER SEED HI (USED FOR RANDOM NUMBER GENERATOR)
002574  123456                  MSEEDL: 123456                  ;MASTER SEED LO (USED FOR RANDOM NUMBER GENERATOR)
002576  177777                  HEADER: 177777                  ;USED TO PRINT HEADINGS ONLY ONCE
002600  177777                  ONES:   177777                  ;FOR AID IN "MOV" INSTRUCTIONS
002602  000003                  FLIPLOC:3                       ;COUNTER FOR FLIPING DATA ON WORST CASE NOISE TEST
002604  052525                  SOFTPAT:52525                   ;PATTERN FOR SOFT ERROR BACKGROUND TESTS
002606  000000                  $LPADR: .WORD   0               ;;CONTAINS SCOPE LOOP ADDRESS
002610  000000                  $LPERR: .WORD   0               ;;CONTAINS SCOPE RETURN FOR ERRORS
002612  000000                  RESTART:0                       ;RESTART (START ADD 202) FLAG
002614  000000                  $ERTTL: .WORD   0               ;;CONTAINS TOTAL ERRORS

                                ;********************** NOTE THESE TWO LOCATIONS MUST STAY TOGETHER ************
002616  000377                  BAKPAT: .WORD   377             ;BACKGROUND PATTERN                            *
002620  177400                  SWAPAT: .WORD   177400          ;SWAPPED BAKPAT                                *
                                ;*******************************************************************************

002622  177570                  SWR:    .WORD   DSWR            ;;ADDRESS OF SWITCH REGISTER
002624  177570                  DISPLAY: .WORD  DDISP           ;;ADDRESS OF DISPLAY REGISTER
002626  177560                  $TKS:   177560                  ;;TTY KBD STATUS
002630  177562                  $TKB:   177562                  ;;TTY KBD BUFFER
002632  177564                  $TPS:   177564                  ;;TTY PRINTER STATUS REG. ADDRESS
002634  177566                  $TPB:   177566                  ;;TTY PRINTER BUFFER REG. ADDRESS
002636     012                  $FILLC: .BYTE   12              ;;INSERT FILL CHARS. AFTER A "LINE FEED"
002637     207  377  377         $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
002642     000
002643     077                  $QUES:  .ASCII  /?/             ;;QUESTION MARK
002644     015                  $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
002645     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
 5782 002650  000201            CONFIG: .REPT   201
 5785 003654                    CONFIEND:
```

```
5787                                            .SBTTL  ********** MAIN **********
5788 003654                                     START:  SUBTST  <<INITIALIZE VARIABLES TO ZERO>>
                                                ;*************************************************************************
                                                ;*SUBTEST        INITIALIZE VARIABLES TO ZERO
                                                ;*************************************************************************
5789 003654  105737  065660                             TSTB    $ENV
5790 003660  001001                                     BNE     NORES
5791 003662  000005                                     RESET
5792 003664                                     NORES:  CLEAR   MONFLG          ;;CLEAR RETURN TO MONITOR FLAG
5793 003670  010637  002270                             MOV     SP,SAVMON       ;;SAVE XXDP MONITOR RESTART ADDRESS
5794 003674  013706  002560                             MOV     KSTACK,SP       ;;SETUP THE STACK POINTER
5795 003700  012700  002000                             MOV     #$CMTAG,R0      ;;FIRST LOCATION TO BE CLEARED
5796 003704  005020                             1$:     CLR     (R0)+           ;;CLEAR MEMORY LOCATION
5797 003706  022700  002540                             CMP     #$CMTGE,R0      ;;DONE?
5798 003712  001374                                     BNE     1$              ;LOOP BACK IF NO
5799 003714  012737  000167  002552                     MOV     #167,LASTBANK   ;RESTORE LASTBANK (THIS MUST BE DONE PRIOR TO SYSTEM SIZING)
5800 003722                                             SUBTST  <<CLEAR NON-PROGRAM SPACE>>
                                                ;*************************************************************************
                                                ;*SUBTEST        CLEAR NON-PROGRAM SPACE
                                                ;*************************************************************************
5801                                                    ;THIS ATTEMPS TO GET RID OF ANY PARITY ERRORS BY WRITING INTO
5802                                                    ;EVERY LOCATION THAT IS NOT LOADED INTO BY THE PROGRAM OR ALLOCATED
5803                                                    ;TO THE XXDP LOADERS
5804 003722  012737  000001  002074                     MOV     #1,NOPAR                ;PARITY ACTION = COUNT & IGNORE
5805 003730  005000                                     CLR     R0
5806 003732  000241                             2$:     CLC
5807 003734  005520                                     ADC     (R0)+
5808 003736  020027  160000                             CMP     R0,#160000
5809 003742  103773                                     BLO     2$
5810 003744  005037  002074                             CLR     NOPAR                   ;RESTORE DEFAULT PARITY ACTION
5811
```

```
5813 003750                                    SUBTST  <<TYPE OF SYSTEM SIZER>>
                                       ;*****************************************************************************
                                       ;*SUBTEST       TYPE OF SYSTEM SIZER
                                       ;*****************************************************************************
5814 003750  000401                            BR      SYSSIZ                  ;SKIP OVER VARIABLE LOCATION
5815 003752  000000                    PROTYP: .WORD   0
5816 003754                            SYSSIZ: SET4    #4$
5817 003762  005737  177746                    TST     CONTRL                  ;SEE IF CACHE REGISTER RESPONDS
5818 003766                                    SET4    #9$                     ;YES - DO WE HAVE 11/44 TYPE CACHE
5819 003774  005737  177750                    TST     MAINT                   ;OR 11/60 TYPE CACHE?
5820 004000  000411                            BR      5$                      ;BRANCH IF 11/44 TYPE CACHE
5821 004002  012737  000014  002544    9$:     MOV     #14,CACHKF              ;TURN OFF CONSTANT FOR 11/60 CACHE
5822 004010  000405                            BR      5$                      ;
5823 004012  005037  002540            4$:     CLR     CACHKN                  ;NO CACHE ON SYSTEM
5824 004016  012737  002332  067330            MOV     #ZEROS,DT14             ;DO NOT PRINT CONTRL ERROR MESSAGES
5825 004024                            5$:     SET4    #6$
5826 004032  005737  172516                    TST     MMR3                    ;DO WE HAVE AN MMR3?
5827 004036  005037  172516                    CLR     MMR3                    ;YES WE DO
5828 004042  052737  000020  172516            BIS     #BIT4,MMR3              ;SEE IF THERE IS 22-BIT MODE
5829 004050  032737  000020  172516            BIT     #BIT4,MMR3
5830 004056  001026                            BNE     10$                     ;BRANCH IF 22-BIT RELOCATION
5831 004060  000413                            BR      7$                      ;BRANCH IF MMR3 BUT NO 22-BIT RELOC.
5832                                   ;* 11/34 TYPE MACHINES ENTER HERE
5833 004062  012737  140000  002546    6$:     MOV     #140000,TESTMODE        ;MAKE TESTMODE USER
5834 004070  005237  002452                    INC     NOSUPER                 ;
5835 004074  005037  067200                    CLR     DT5+10                  ;
5836 004100  005037  067340                    CLR     DT14+10                 ;
5837 004104  005237  002450                    INC     NO22BIT                 ;
5838                                   ;* 11/45 TYPE MACHINES ENTER HERE
5839 004110  005237  002450            7$:     INC     NO22BIT                 ;
5840 004114  012737  000007  002552            MOV     #7,LASTBANK             ;
5841 004122  005037  067202                    CLR     DT5+12                  ;
5842 004126  005037  067342                    CLR     DT14+12                 ;
5843 004132  000417                            BR      8$                      ;
5844 004134                            10$:    SET4    #8$
5845 004142  000007                            MFPT                            ;TYPE OF PROCESSOR TEST: THIS INSTRUCTION
5846                                                                           ; (AVAILABLE ON NEWER PROCESSORS ONLY) PLACES
5847                                                                           ; A CODE IN THE LOWER BYTE OF R0 THAT
5848                                                                           ; INDICATES THE  PROCESSOR TYPE. 1=11/44
5849                                                                           ; 3=11/24
5850 004144  110037  003752                    MOVB    R0,PROTYP               ;MOV THE CODE TO PROTYP
5851 004150  022737  000003  003752            CMP     #3,PROTYP               ;IS THIS AN 11/24?
5852 004156  001005                            BNE     8$                      ;BRANCH IF NOT - WE HAVE AN 11/44
5853 004160  005237  002452                    INC     NOSUPER                 ;NO SUPERVISOR MODE
5854 004164  012737  140000  002546            MOV     #140000,TESTMODE        ;MAKE TEST MODE USER
5855
5856 004172                            8$:     SET4    #11$                    ;TRAPS GO TO 11$                    ;R-C
5857 004200  005037  061322                    CLR     CPERRF                  ;CLEAR THE FLAG                     ;R-C
5858 004204  005737  177766                    TST     @#177766                ;IS THERE A CPU ERROR REGISTER?     ;R-C
5859 004210  012737  177777  061322            MOV     #-1,CPERRF              ;YES-TRAPPED
5860 004216                            11$:    RES4                                                                ;R-C
```

```
                                                        SUBTST  <<INITIALIZE VARIABLES TO NON ZERO>>
                                                ;*****************************************************************************
                                                ;*SUBTEST        INITIALIZE VARIABLES TO NON ZERO
                                                ;*****************************************************************************
                                                        SET     WORST
                                                        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
```

```
5893 004470                                     SUBTST  <<INITIALIZE PATTERNS>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       INITIALIZE PATTERNS
                                        ;*****************************************************************************
                                        ;
5894                                            ;THE APT E-TABLE DETERMINES WHICH PATTERNS ARE GOING TO BE RUN.
5895                                            ;EACH BIT SET REPRESENTS A PATTERN TABLE ENTRY THAT IS TO BE LEFT
5896                                            ;ALONE (TO BE RUN).  EACH BIT CLEARED REPRESENTS A PATTERN TABLE ENTRY
5897                                            ;THAT IS TO BE OVERLAYED WITH THE ADDRESS OF A NULL PATTERN.
5898 004470  012700  065724                    MOV     #$DDW0,R0
5899 004474  012001                            MOV     (R0)+,R1
5900 004476  012703  020204                    MOV     #MKCSRT,R3
5901 004502  012702  000020                    MOV     #16.,R2
5902 004506  004737  004606                    CALL    PATPLUG
5903 004512  012001                            MOV     (R0)+,R1
5904 004514  012702  000010                    MOV     #8.,R2
5905 004520  004737  004606                    CALL    PATPLUG
5906 004524  012001                            MOV     (R0)+,R1
5907 004526  012703  020434                    MOV     #MKPAT,R3
5908 004532  012702  000020                    MOV     #16.,R2
5909 004536  004737  004606                    CALL    PATPLUG
5910 004542  012001                            MOV     (R0)+,R1
5911 004544  012702  000010                    MOV     #8.,R2
5912 004550  004737  004606                    CALL    PATPLUG
5913 004554  012001                            MOV     (R0)+,R1
5914 004556  012703  020620                    MOV     #MJPAT,R3
5915 004562  012702  000020                    MOV     #16.,R2
5916 004566  004737  004606                    CALL    PATPLUG
5917 004572  012001                            MOV     (R0)+,R1
5918 004574  012702  000010                    MOV     #8.,R2
5919 004600  004737  004606                    CALL    PATPLUG
5920 004604  000417                            BR      SUBAAA
5921
5922 004606                             PATPLUG:SUBTST  <<SUBR  PLUG IN NULL PATTERNS>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       SUBR    PLUG IN NULL PATTERNS
                                        ;*****************************************************************************
                                        ;
5923 004606                                     FOR I := #1 TO R2
5924 004614  006001                                 ROR   R1
5925 004616                                         ON.NOERROR              ;IF CARRY CLEAR
5926 004620  012713  026760                            MOV #MT0999,(R3)
5927 004624                                         END ;OF ON.ERROR
5928 004624  062703  000002                        ADD #2,R3
5929 004630                                     END ;OF FOR
5930 004642  000207                             RETURN
```

```
                                SUBAAA: SUBTST  <<CLEAR THE CONFIGURATION TABLE>>
                                ;*****************************************************************************
                                ;*SUBTEST        CLEAR THE CONFIGURATION TABLE
                                ;*****************************************************************************
                                        ;THIS ZEROS (UNLESS WE STARTED AT ADDRESS 202) THE CONFIG TABLE
                                        ;WHICH IS FULLY DISCRIBED AT LOCATION "CONFIG".
                                        .ENABLE LSB
                                        IF RESTART IS FALSE
                                          MOV   #CONFIG,R0
                                1$:       CLR   (R0)+
                                          CMP   #CONFIEND,R0
                                          BNE   1$
                                        END ;OF IF RESTART
                                        .DSABL  LSB
004644                                  MOV     #BIT1,CPUBIT            ;SET ID BIT
004666                                  SUBTST  <<SIZE FOR A HARDWARE SWITCH REGISTER>>
004674                          ;*****************************************************************************
                                ;*SUBTEST        SIZE FOR A HARDWARE SWITCH REGISTER
                                ;*****************************************************************************
                                        ;;IF NOT FOUND OR IT IS
                                        ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
                                        .ENABL  LSB
004674                                  SET4    #3$             ;TRAPS TO 4 GOTO 3$
004702                                  MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWITCH REGISTER
004710                                  MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
004716                                  IF #-1 EQ @SWR          ;IF NO TRAP FROM REFERENCE TO @SWR AND @SWR = #-1
004726                                    BR    2$              ;;BRANCH IF NO TIMEOUT
004730                          3$:       MOV   #2$,(SP)        ;;SET UP FOR TRAP RETURN
004734                                    RTI
004736                          2$:       RES4                  ;RESET TRAPS TO 4 TO DEFAULT
004760                                    MOV   #SWREG,SWR      ;;POINT TO SOFTWARE SWR
004766                                    MOV   #DISPREG,DISPLAY
004774                                  END ;OF IF #-1
                                        .DSABL  LSB
```

SIZE FOR A HARDWARE SWITCH REGISTER

```
 5963 004774                            SUBAAB: SUBTST  <<SETUP ACT, APT, & XXDP>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       SETUP ACT, APT, & XXDP
                                        ;*****************************************************************************
 5964                                           ;THIS SETS UP A BUNCH OF FLAGS TO TELL THE PROGRAM EVERYTHING
 5965                                           ;IT CARES TO KNOW ABOUT APT, ACT, & XXDP.
 5966 004774  005037  065646                    CLR     $PASS           ;CLEAR PASS COUNT
 5967 005000                                    IFB #BIT5 SET.IN $ENVM
 5968 005010                                      SET   $TPFLG          ;INDICATE NO TERMINAL
 5969 005016                                    END ;OF IFB #BIT5
 5970 005016                                    IFB #BIT7 SET.IN $ENVM
 5971 005026                                      SET   APTSIZE
 5972 005034                                    END ;OF IFB #BIT7
 5973 005034                                    IFB $ENV EQ #1
 5974 005044                                      SET   APTFLAG,QVFLAG,$AUTO,QUICK
 5975 005074  012737  047746  000024              MOV   #APTDOWN,PWRVEC
 5976 005102  012737  065662  002622              MOV   #$SWREG,SWR     ;USE APT SWR
 5977 005110                                    ELSE
 5978 005112                                      IF 42 NE #STACK AND 42 NE #0
 5979 005130                                        SET QVFLAG,$AUTO
 5980 005144                                        IF 42 EQ #$ENDAD
 5981 005154                                          SET       ACTFLAG
 5982 005162                                        ELSE
 5983 005164                                          SET       XXDPCHAIN
 5984 005172                                        END ;OF IF 42
 5985 005172                                      END ;OF IF 42
 5986 005172                                    END ;OF IFB $ENV
```

```
 5988 005172                                    SUBTST  <<PROTECT PROGRAM & LOADERS>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       PROTECT PROGRAM & LOADERS
                                        ;*****************************************************************************
 5989 005172  052737  000200  002650            BIS     #BIT7,CONFIG            ;PROTECT PROGRAM SPACE (BANK 0)
 5990 005200  052737  000200  002654            BIS     #BIT7,CONFIG+4          ;PROTECT LOADER SPACE (BANK 1)
 5991 005206                                    IF #$ENDAD NE 42                ;NOT ACT-11?
 5992 005216                                      IF NO22BIT NE #0              ;
 5993 005224                                        SET     MONFLG              ;RETURN TO XXDP MONITOR
 5994 005232  104064                                ERROR +64                   ;ILLEGAL PROCESSOR
 5995 005234                                      ELSE                          ;
 5996 005236                                        TYPE    MSG000              ;TYPE PROGRAM TITLE
 5997 005242                                      END                           ;
 5998 005242                                    END ;OF IF #$ENDAD
 5999
 6000 005242                                    SUBTST  <<CHECK SYSTEM FOR CACHE>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       CHECK SYSTEM FOR CACHE
                                        ;*****************************************************************************
 6001                                           ;* THIS FIGURES OUT IF THERE IS A CACHE ON THE SYSTEM,
 6002                                           ;* WHAT TYPE OF SYSTEM IT IS, AND WHETHER IT IS ENABLED
 6003                                           ;* OR DISABLED.
 6004 005242                                    SET4    #3$
 6005 005250  005737  177746                    TST     CONTRL                  ;IS THERE A CONTROL REGISTER?
 6006 005254                                    SET4    #2$
 6007 005262  005737  177750                    TST     MAINT                   ;IS THERE A MAINTENANCE REGISTER?
 6008 005266                                    SET4    #1$
 6009 005274  005737  177754                    TST     DATARG                  ;IS THERE A DATA REGISTER?
 6010 005300                                    TYPE    MSG117                  ; 11/44
 6011 005304  000410                            BR      4$
 6012 005306                            1$:     SET     MONFLG                  ; 11/34
 6013 005314  104064                            ERROR   +64                     ;
 6014                                                                           ;PROCESSOR NOT SUPPORTED BY THIS DIAGNOSTIC
 6015 005316                            2$:     SET     MONFLG                  ; 11/60
 6016 005324  104064                            ERROR   +64                     ;
 6017                                                                           ;PROCESSOR NOT SUPPORTED BY THIS DIAGNOSTIC
 6018 005326  052737  000014  177746    4$:     BIS     #BIT2!BIT3,CONTRL       ;SET CACHE DISABLE BITS
 6019 005334  042737  000014  177746            BIC     #BIT2!BIT3,CONTRL       ;CLEAR CACHE DISABLE BITS
 6020 005342  032737  000004  177746            BIT     #BIT2,CONTRL            ;IS THE BIT SET?
 6021 005350  001004                            BNE     7$                      ;BRANCH IF THE BIT IS SET
 6022 005352  032737  000010  177746            BIT     #BIT3,CONTRL            ;IS THE BIT SET?
 6023 005360  001413                            BEQ     6$                      ;BRANCH IF THE BIT IS SET
 6024 005362                            7$:     TYPE    MSG121                  ; CACHE BYPASSED
 6025 005366  104424                            CACHOFF
 6026 005370  013737  002540  002542            MOV     CACHKN,CACHKN+2         ;SAVE INFO ABOUT CACHE
 6027 005376  005037  002540                    CLR     CACHKN                  ;CACHE CANNOT BE USED - IT'S BYPASSED
 6028 005402  000404                            BR      8$                      ;
 6029 005404                            3$:     TYPE    MSG119                  ;       NO
 6030 005410                            6$:     TYPE    MSG120                  ;CACHE AVAILABLE
```

CHECK SYSTEM FOR CACHE

```
                                        SUBTST  <<SETUP USER & SUPERVISOR STACK>>
                                ;*******************************************************************************
                                ;*SUBTEST        SETUP USER & SUPERVISOR STACK
                                ;*******************************************************************************
 6033 005414  104421            8$:     DEENERGIZE              ;TURN OFF MEMORY MANAGEMENT
 6034 005416  005737  002452            TST     NOSUPER         ;IS THERE A SUPERVISOR MODE?
 6035 005422  001011                    BNE     5$              ;NO-SKIP SUPERVISOR SETUP.
 6036
 6037                                   ;SET PREVIOUS MODE TO SUPERVISOR
 6038 005424  042737  030000  177776    BIC     #BIT13!BIT12,PSW
 6039 005432  052737  010000  177776    BIS     #BIT12,PSW
 6040
 6041 005440                            PUSH    #SUPSTK
 6042 005444  006606                    MTPI    SSP
 6043
 6044                                   ;SET PREVIOUS MODE TO USER
 6045 005446  052737  030000  177776    5$:     BIS     #BIT13!BIT12,PSW
 6046
 6047 005454                            PUSH    #USESTK
 6048 005460  006606                    MTPI    USP
 6049
 6050 005462                            SUBTST  <<GET SOFTWARE SWITCH REGISTER IF NECESSARY>>
                                ;*******************************************************************************
                                ;*SUBTEST        GET SOFTWARE SWITCH REGISTER IF NECESSARY
                                ;*******************************************************************************
 6051 005462                            IF $AUTO IS FALSE                       ;IF NOT(APT OR ACT)
 6052 005470                              IF SWR EQ #SWREG               ;&F SOFTWARE SWITCH REG SELECTED
 6053 005500  104407                        GTSWR                               ;;GET SOFT-SWR SETTINGS
 6054 005502                              END ;OF IF SWR
 6055 005502                            END ;OF IF $AUTO
 6056
 6057 005502                            SUBTST  <<GET MEMORY MANAGEMENT READY>>
                                ;*******************************************************************************
                                ;*SUBTEST        GET MEMORY MANAGEMENT READY
                                ;*******************************************************************************
 6058 005502  104422                    KMAP                    ;MAP KERNEL SPACE 1 TO 1
 6059 005504                            MAP                     ;MAP SUPERVISOR SPACE (TEST AREA) 1 TO 1
 6060 005520  104420                    ENERGIZE                ;TURN ON MEMORY MANAGEMENT
```

```
6063 005522                                             NEWTST  <<BIT TEST OF ALL CSR'S>>
                                                ;*****************************************************************************
                                                ;*TEST 1        BIT TEST OF ALL CSR'S
                                                ;*****************************************************************************
     005522  000004                             TST1:   SCOPE
6064                                            ;* THE FIRST PART OF THE CONFIGURATION ANALYSIS DOES THE FOLLOWING:
6065                                            ;*      1) FINDS WHICH CSR'S RESPOND, AND PUTS THEM INTO THE CSR INFORMATION
6066                                            ;*         TABLE, AND STORES ANOTHER BIT FOR "TOTCSRS".
6067                                            ;*      2) TESTS THE CSR BITS COMMON TO ALL CSR'S.
6068                                            ;*      3) FIGURES OUT IF THE MODULE IS A ECC OR PARITY MEMORY
6069                                            ;*      4) TESTS THE BITS PARTICULAR TO THAT TYPE OF CSR.
6070                                            ;*      5) IF ANY BITS TEST BAD IN THE CSR UNDER TEST, THE CSR OK BIT IN THE
6071                                            ;*              CSR INFORMATION TABLE IS CLEARED.
6072                                            ;* THE INFORMATION BITS ONE THROUGH THREE FORM A CODE WHICH GIVES THE TYPE
6073                                            ;* OF CSR:
6074                                            ;*                      NOT USED     ECC TYPE      ECC
6075                                            ;*       TYPE             BIT2         BIT1         BIT0        CODE TOTALS
6076                                            ;*
6077                                            ;*      MS11-L            0            0            0            0
6078                                            ;*      MS11-M            0            0            1            1
6079                                            ;*      MS11-P            0            1            1            3
6080                                            ;*
6081                                            ;* THIS MEMORY CODE WILL BE USED IN THE SECOND PART OF THIS ANALYSIS
6082                                            ;*
6083 005524  005005                                     CLR     R5              ;R5 IS THE TOTAL CSR NUMBER
6084 005526  005000                                     CLR     R0              ;R0 IS A TABLE INDEX
6085 005530  012703  172100                             MOV     #CSRADD,R3      ;R3 HAS THE CSR ADDRESS
6086 005534  012737  000001  002074                     MOV     #1,NOPAR        ;IGNORE PARITY ERRORS
6087 005542                                             SET4    #NXTCSR
6088 005550                                             REPEAT
6089 005550  005713                                        TST (R3)             ;DOES THIS CSR RESPOND???
6090 005552  052705  000001                                BIS  #1,R5           ;MARK IT IN CSR MAP
6091 005556  005004                                        CLR  R4              ;CLEAR THE LAST CSR INDICATOR
6092 005560  042760  000004  002456                        BIC  #4,CSRINFO(R0)  ;CLEAR UNUSED BITS
6093 005566  052760  000030  002456                        BIS  #BIT4!BIT3,CSRINFO(R0) ;YES-MARK IT IN CSR INFORMATION TABLE
6094 005574  005013                                        CLR  (R3)            ;CLEAR THE CSR UNDER TEST
6095 005576                                                LET (R3) := #BIT13   ;IS THIS AN ECC MEMORY???
6096 005602                                                IF #BIT13 SET.IN (R3) ;IS BIT 13 SET FOR ECC MEMORY???
6097 005610  004737  005714                                   CALL ECCTYPE      ;FIGURE OUT WHAT KIND OF ECC MEMORY WE HAVE
6098 005614                                                END                  ;
6099 005614  005013                                        CLR (R3)             ;CLEAR CSR UNDER TEST
6100 005616  004737  006206                                CALL RWCSR           ;BIT TEST OF ALL BITS IN CSR'S
6101 005622                                                IF CSRINFO(R0) MI #30 ;DO WE HAVE A LEGAL CONFIGURATION?
6102 005632  016037  002456  002050                           MOV CSRINFO(R0),BAD ;MOVE IN BAD DATA
6103 005640  104021                                           ERROR +21         ;
6104 005642                                                END                  ;
6105 005642  062700  000002          NXTCSR:              ADD #2,R0            ;GO TO NEXT CSR
6106 005646  062703  000002                               ADD #2,R3            ;GO TO NEXT CSR
6107 005652  006305                                       ASL  R5              ;SHIFT CSR MAP
6108 005654                                               ON.ERROR             ;IS THERE A CSR 0
6109 005656  005204                                         INC R4             ;YES-SET CSR PRESENT FLAG
6110 005660                                               END                  ;
6111 005660                                             UNTIL R0 EQ #40        ;UNTIL ALL CSR'S ARE DONE
6112 005666  006005                                     ROR     R5              ;RESYNC R5
6113 005670  005704                                     TST     R4              ;WAS THERE A CSR 0?
6114 005672                                             BNE     22$             ;BRANCH IF NOT EQUAL
6115 005674  052705  100000                             BIS     #BIT15,R5       ;YES SET IT IN CSR TABLE
```

T1   BIT TEST OF ALL CSR'S

```
   6116 005700                              22$:    LET TOTCSRS := R5       ;STORE CSR MAP IN TOTCSRS
   6117 005704  004737  005774                     CALL CSRMAP             ;PRINT CSR MAP
   6118 005710                                      JUMPTO  CTEST           ;
```

```
                                        SUBTST  <<DETERM.NE TYPE OF ECC MEMORY>>
                                ;*******************************************************************************
                                ;*SUBTEST       DETERMINE TYPE OF ECC MEMORY
                                ;*******************************************************************************
                                ;
                                ;       THIS ROUTINE WILL DETERMINE IF THE ECC MEMORY UNDER TEST IS
                                ;       A MS11-M OR A MS11-P
                                ;
005714                          ECCTYPE:
005714  052760  000001  002456          BIS #BIT0,CSRINFO(R0)    ;MARK IT IN THE TABLE AS BEING A ECC MEMORY
005722                                  LET (R3) := #60004       ;IS THIS A MS11-P???
005726                                  IF #BIT11 SET.IN (R3)    ;IS BIT 11 SET???
005734  005013                             CLR (R3)              ;CLEAR CSR
005736                                     LET (R3) := #20004    ;ENABLE CHECK/SYNDROME BIT REGISTER
005742                                     LET (R3) := #27744    ;IT IS BUT MAKE SURE AGAIN
005746                                     IF (R3) EQ #23744     ;DO WE HAVE 6 OR 7 CHECK BITS IN CSR
005754  052760  000002  002456                BIS #BIT1,CSRINFO(R0) ;6 CHECK BITS-MARK IT A MS11-P
005762                                     END                   ;
005762                                  ELSE                     ;IT IS A MS11-M
005764  042760  000002  002456             BIC #BIT1,CSRINFO(R0) ;MARK IT IN THE TABLE
005772                                  END                      ;
005772  000207                          RETURN
```

```
6140 005774                                CSRMAP: SUBTST  <<PRINT CSR REGISTER MAP>>
                                           ;*****************************************************************************
                                           ;*SUBTEST       PRINT CSR REGISTER MAP
                                           ;*****************************************************************************
6141 005774  005000                                CLR     R0                  ;CLEAR CSR INFO POINTER
6142 005776                                        TYPE    MSG008              ;PRINT TITLE
6143 006002                                        TYPE    MSG016              ;PRINT CSR NUMBERS
6144 006006  005001                                CLR     R1                  ;
6145 006010                                        REPEAT                      ;
6146 006010  010102                                   MOV  R1,R2               ;
6147 006012  022702  000011                           CMP  #9.,R2              ;
6148 006016  100002                                   BPL  1$                  ;JUMP AROUND NEXT INSTRUCTION
6149 006020  062702  000007                           ADD  #7,R2               ;
6150 006024  062702  000060                1$:        ADD  #o0,R2              ;MAKE IT ASCII
6151 006030  110237  074704                           MOVB R2,MSG015           ;
6152 006034                                           TYPE MSG015              ;
6153 006040                                           TYPE MSG014              ;TYPE SINGLE SPACE
6154 006044  005201                                   INC  R1                  ;
6155 006046                                        UNTIL R1 EQ #16.            ;
6156 006054                                        TYPE  MSG009                ;TYPE MEMTYPE
6157 006060                                        REPEAT                      ;
6158 006060                                           IF CSRINFO(R0) NE #0     ;IS CSR NONEXSISTANT????
6159 006066                                              IF #BIT0 SET.IN CSRINFO(R0)
6160 006076                                                 IF #BIT1 SET.IN CSRINFO(R0)
6161 006106  112737  000120  074704                           MOVB  #'P,MSG015 ;IT IS A MS11-P
6162 006114                                                ELSE
6163 006116  112737  000115  074704                           MOVB  #'M,MSG015 ;IT IS A MS11-M
6164 006124                                                END
6165 006124                                             END
6166 006124                                             IF #BIT1!BIT0 OFF.IN CSRINFO(R0)
6167 006134  112737  000114  074704                        MOVB  #'L,MSG015 ;IT IS A MS11-L
6168 006142                                             END
6169 006142                                          ELSE
6170 006144  112737  000040  074704                     MOVB #' ,MSG015
6171 006152                                          END
6172 006152                                          TYPE MSG015                ;TYPE MEMORY TYPE
6173 006156                                          TYPE MSG014                ;TYPE SPACE
6174 006162  000240                                  NOP                        ;
6175 006164  062700  000002                          ADD  #2,R0                 ;POINT TO NEXT ENTRY
6176 006170                                       UNTIL R0 EQ #40               ;
6177 006176                                       TYPE   MSG129                 ;
6178 006202  000207                               RETURN                        ;
6179 006204  000000                     TRACE:    .WORD   0
```

```
   6181 006206                                  SUBTST  <<READ AND WRITE ALL CSR BITS>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       READ AND WRITE ALL CSR BITS
                                        ;*****************************************************************************
                                        ;
   6182                                 ; THIS ROUTINE "RWCSR" CHECK TO SEE THAT THE CSR CAN BEWRITTEN ON CORRECTLY
   6183                                 ; BY WRITING AND CHECKING FOR THE FOLLOWING PATTERNS:
   6184                                 ;
   6185                                 ;               1-ZEROS
   6186                                 ;               2-ONES
   6187                                 ;               3-SHIFTING A ONE THROUGH A FIELD OF ZEROS
   6188                                 ;               4-SHIFTING A ZEROS THROUGH A FIELD OF ONES
   6189                                 ;
   6190                                 ;
   6191 006206                          RWCSR:
   6192 006206                                  PUSH R4,R5,UIPAR0       ;SAVE R4,R5, AND UIPAR0 ON STACK
   6193 006216                                  LET R5 := R0            ;GET CSR NUMBER FOR POSSIBLE ERROR
   6194 006220  006205                          ASR R5                  ;
   6195 006222                                  LET CSRNO := R5         ;
   6196 006226                                  LET ADDRESS := R3       ;GET ADDRESS FOR POSSIBLE ERROR
   6197 006232                                  IF #BIT0 SET.IN CSRINFO(R0) ;WHAT KIND OF MEMORY IS THIS???     ;GET BIT MASKS FOR D
   6198 006242                                     LET R5 := #17740     ;MASK FOR MS11-M/P
   6199 006246                                  ELSE                    ;IT IS A MS11-L
   6200 006250                                     LET R5 := #70032     ;MASK FOR MS11-L
   6201 006254                                  END                     ;
   6202 006254                                  LET CSR1S := #177777    ;SET CSR1S TO ALL ONES
   6203 006262  040537  002316                  BIC R5,CSR1S            ;CLEAR BITS FOR GOOD DATA
   6204 006266                                  LET (R3) := #0          ;0----->CSR
   6205 006270                                  LET R4 := (R3)          ;MASK OUT UNWANTED BITS
   6206 006272  040504                          BIC R5,R4               ;
   6207 006274                                  IF R4 NE #0             ;DO WE HAVE A CORRECT READ
   6208 006300                                    LET GOOD := #0        ;GOOD DATA=0'S
   6209 006304                                    LET CSR := R4         ;BAD DATA=CSR
   6210 006310  104035                            ERROR +35             ;BIT SET ERROR
   6211 006312  042760  000010  002456            BIC #BIT3,CSRINFO(R0) ;CLEAR CSR OK BIT
   6212 006320                                  END
   6213 006320                                  LET (R3) := CSR1S       ;ONES--->(R3)
   6214 006324                                  LET R4 := (R3)          ;MASK OUT CORRECT FIELD
   6215 006326  005013                          CLR     (R3)            ;CLEAR OUT CSR
   6216 006330  040504                          BIC R5,R4               ;
   6217 006332                                  IF R4 NE CSR1S          ;WAS PATTERN WRITTEN CORRECTLY?
   6218 006340                                      LET GOOD := CSR1S   ;GOOD DATA = ALL LEGAL BITS SET IN CSR
   6219 006346                                      LET CSR := R4       ;BAD DATA=CSR
   6220 006352  104010                              ERROR +10           ;BIT CLEAR ERROR
   6221 006354  042760  000010  002456              BIC #BIT3,CSRINFO(R0) ;CLEAR CSR OK BIT
   6222 006362                                  END                     ;
   6223 006362                                  LET PASFLG := #0        ;SET UP LOOP COUNTER
   6224 006366                                  REPEAT                  ;REPEAT WITH A FIELD OF 1'S THROUGH 0'S
   6225                                                                 ;                       0'S THROUGH 1'S
   6226 006366  005237  002262                     INC PASFLG           ;INCREMENT LOOP COUNTER
   6227 006372                                     LET UIPAR0 := #-1    ;USE USER PAR FOR BIT COUNTER
   6228 006400                                     IF PASFLG EQ #1      ;PASS 1
   6229 006410                                       LET R2 := #1       ;1----->FIELD OF ZEROS
   6230 006414                                     ELSE                 ;PASS 2
   6231 006416                                       LET R2 := #177776  ;0----->FIELD OF ONES
   6232 006422                                     END                  ;
   6233 006422                                     REPEAT               ;DO BITS 0-4 AND 13-15
   6234 006422  005237  177640                       INC UIPAR0         ;INCREMENT BIT POINTER
```

```
6235 006426                                     IF PASFLG EQ #2 AND #BIT0 OFF.IN CSRINFO(R0) ;
6236 006446  042702  040004                        BIC #BIT14!BIT2,R2  ;IF THIS IS PASS 2 ON A MS11-L, CLEAR EUB BIT AND WRITE
6237 006452                                     END                    ;
6238 006452                                     LET (R3) := R2         ;WRITE DATA
6239 006454                                     LET R1 := R2           ;GET GOOD DATA AND MASK IT OUT
6240 006456  040501                             BIC R5,R1              ;GET GOOD DATA
6241 006460                                     LET R4 := (R3)         ;GET DATA THAT IS READ
6242 006462  040504                             BIC R5,R4              ;MASK OUT CSR BITS
6243 006464                                     IF R1 NE R4            ;IS DATA CORRECT???
6244 006470                                        LET BAD := R4       ;BAD DATA = CSR CONTENTS
6245 006474                                        LET CSR := R1       ;GET GOOD DATA
6246 006500                                        IF PASFLG EQ #1     ;SELECT ERROR DEPENDING ON PASS
6247 006510  104035                                  ERROR +35         ;BIT SET ERROR
6248 006512                                        ELSE                ;PASS 2
6249 006514  104010                                  ERROR +10         ;BIT CLEAR ERROR
6250 006516                                        END
6251 006516  042760  000010  002456                BIC #BIT3,CSRINFO(R0) ;CLEAR CSR OK BIT
6252 006524                                     END                    ;
6253 006524                                     IF PASFLG EQ #1        ;GET DATA FOR NEXT LOOP
6254 006534  006302                                ASL  R2             ;SHIFT 1 ACROSS 0'S
6255 006536                                     ELSE                   ;
6256 006540  000261                                SEC                 ;SET CARRY
6257 006542  006102                                ROL R2              ;ROTATE A 0 ACROSS A FIELD OF ONES
6258 006544                                     END                    ;
6259 006544                                  UNTIL UIPAR0 EQ #15.      ;UNTIL ALL BITS ARE DONE
6260 006554                               UNTIL PASFLG EQ #2           ;DONE WITH 2 PASSES
6261 006564                               IF #BIT0 SET.IN CSRINFO(R0) THEN JUMPTO DONE ;IF MS11-L DO ONE LAST WRITE
6262 006600                               LET (R3) := #140005          ;WRITE ONES TO CSR WITH EUB BIT ENABLED
6263 006604                               LET R2 := (R3)               ;READ CSR FOR CORRECT BITS
6264 006606  042702  037772               BIC #37772,R2               ;CLEAR UNWANTED BITS
6265 006612                               IF R2 NE #140005             ;WAS WRITE CORRECT
6266 006620                                  LET GOOD := #140005       ;GOOD DATA
6267 006626                                  LET CSR := R2             ;BAD DATA
6268 006632  104010                          ERROR +10                 ;BIT CLEAR ERROR
6269 006634  042760  000010  002456          BIC #BIT3,CSRINFO(R0)     ;CLEAR CSR OK BIT!
6270 006642                               END                          ;
6271 006642                        DONE:  LET (R3) := #0               ;CLEAR OUT CSR
6272 006644                               POP UIPAR0,R5,R4             ;RESTORE UIPAR0,R4, AND R5
6273 006654  000207                       RETURN
6274
```

```
                                        ;THE FOLLOWING ROUTINE DETERMINES WHICH CSR CONTROLS PROGRAM SPACE
                                        ;
006656  104424                  CTEST:  CACHOFF
006660  012737  177777  002526          MOV     #177777,PGMCSR
006666  012737  002000  172350          MOV     #2000,KIPAR4            ;SET UP MAP REGISTER
006674  012701  002406                  MOV     #TESTADD,R1
006700  012737  100000  002406          MOV     #100000,TESTADD
006706  012737  100002  002410          MOV     #100002,TESTADD+2
006714  005000                          CLR     R0                      ;CLEAR CSR COUNTER
006716  005037  002150                  CLR     CSRNO
006722  013703  002222                  MOV     TOTCSRS,R3              ;OBTAIN CSR MAP
006726  000240                          NOP                             ;DEBUG AID
006730  006303                  4$:     ASL     R3                      ;PUT HIGH ORDER BIT INTO C BIT
006732  103407                          BCS     2$                      ;BRANCH IF CSR EXISTS
006734  062700  000002          1$:     ADD     #2,R0                   ;UPDATE CSR COUNTER
006740  010037  002150                  MOV     R0,CSRNO
006744  005703                          TST     R3                      ;IS MAP EMPTY?
006746  001474                          BEQ     3$                      ;BRANCH IF SO
006750  000767                          BR      4$
006752  000240                  2$:     NOP                             ;DEBUG AID
006754  000241                          CLC                             ;CLEAR CARRY
006756  032760  000003  002456          BIT     #BIT1!BIT0,CSRINFO(R0)  ;IS THIS PARITY MEMORY?
006764  001014                          BNE     5$                      ;BRACH IF NOT
006766  052760  000004  172100          BIS     #BIT2,CSRADD(R0)        ;SET WRITE WRONG PARITY
006774  012771  123456  000000          MOV     #123456,@(R1)           ;WRITE DATA
007002  012771  123456  000002          MOV     #123456,@2(R1)
007010  005060  172100                  CLR     CSRADD(R0)              ;RESTORE CSR
007014  000414                          BR      6$
007016  012760  000000  172100  5$:     MOV     #0,CSRADD(R0)           ;CLEAR THE CSR UNDER TEST
007024  012771  123456  000000          MOV     #123456,@(R1)           ;WRITE DATA
007032  012771  123456  000002          MOV     #123456,@2(R1)
007040  012760  020006  172100          MOV     #20006,CSRADD(R0)       ;SET DIAG CHECK MODE
007046  005771  000000          6$:     TST     @(R1)                   ;WRITE CHECKBITS TO CSR
007052  016004  172100                  MOV     CSRADD(R0),R4           ;WRITE CSR TO R4
007056  032760  000003  002456          BIT     #BIT1!BIT0,CSRINFO(R0)  ;PARITY MEMORY?
007064  001003                          BNE     7$                      ;BRANCH IF NOT
007066  005704                          TST     R4                      ;PARITY ERROR?
007070  100421                          BMI     8$                      ;BRACH IF SO
007072  000720                          BR      1$                      ;TRY NEXT CSR
007074  000240                  7$:     NOP                             ;DEBUG AID
007076  072427  177773                  ASH     #-5,R4
007102  042704  177600                  BIC     #^C177,R4
007106  032760  000002  002456          BIT     #BIT1,CSRINFO(R0)       ;WHAT KIND OF ECC MEMORY IS THIS
007114  001003                          BNE     10$                     ;BRANCH IF MS11-P
007116  012702  000157                  MOV     #157,R2                 ;LOAD IN CORRECT CHECK BITS FOR MS11-M
007122  000402                          BR      11$
007124  012702  000040          10$:    MOV     #40,R2                  ;CORRECT CHECK BITS FOR MS11-P
007130  020204                  11$:    CMP     R2,R4                   ;CORRECT CHECKBITS?
007132  001300                          BNE     1$                      ;BRANCH IF NOT
007134  010037  002526          8$:     MOV     R0,PGMCSR
007140  000240                  3$:     NOP                             ;DEBUG AID
007142  104502                          CLRCSR                          ;CLEAR ALL CSR'S
007144  012771  000000  000000          MOV     #0,@(R1)                ;RESTORE TEST LOCATIONS
007152  012771  000000  000002          MOV     #0,@2(R1)
007160  023727  002526  177777          CMP     PGMCSR,#177777
007166  001402                          BEQ     FINT                    ;IF PROGRAM CSR NOT FOUND GO TO FINT
007170  000137  007642                  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
                                                ;
007174                                  FINT:   SET4    #2$                     ;NE MEMORY TRAPS GO TO 2$
007202  012771  123456  000000          1$:     MOV     #123456,@(R1)           ;WRITE DATA AT FIRST LOCATION OF BANK 2 IN BOARD
007210  012771  123456  000002                  MOV     #123456,@2(R1)          ;WRITE DATA AT SECOND LOCATION OF BANK 2 IN BOARD
007216  062737  010000  172350                  ADD     #10000,KIPAR4           ;UPDATE PAR4 TO POINT TO UPPER BOARDS
007224  000766                                  BR      1$                      ;KEEP GOING TILL NO MORE MEMORY
007226  012700  177776                  2$:     MOV     #-2,R0
007232  013703  002222                          MOV     TOTCSRS,R3              ;PUT CSR MAP IN R3
007236  062700  000002                  3$:     ADD     #2,R0                   ;UPDATE CSR COUNTER
007242  010037  002150                          MOV     R0,CSRNO                ;UPDATE CSRNO
007246  006303                                  ASL     R3
007250  103403                                  BCS     4$                      ;BRANCH IF CSR EXISTS
007252  005703                                  TST     R3                      ;ANY CSR'S LEFT?
007254  001405                                  BEQ     5$                      ;BRANCH IF NOT
007256  000767                                  BR      3$                      ;LOOK FOR NEXT CSR
007260  012760  020006  172100          4$:     MOV     #20006,CSRADD(R0)       ;SET DIAGNOSTIC CHECK MODE IN CSR
007266  000763                                  BR      3$                      ;LOOK FOR NEXT CSR
007270                                  5$:     SET4    #6$                     ;NE MEMORY TRAPS NOW GO TO 6$
007276  012700  177776                          MOV     #-2,R0                  ;RESET CSR POINTER
007302  012737  002000  172350                  MOV     #2000,KIPAR4            ;REMAP PAR4 TO POINT TO BANK 2
007310  005771  000000                          TST     @(R1)                   ;TEST NONASSERTED LOCATIONS
007314  062700  000002                  6$:     ADD     #2,R0                   ;UPDTAE CSR POINTER
007320  010037  002150                          MOV     R0,CSRNO
007324  022700  000040                          CMP     #40,R0                  ;NOT FOUND?
007330  001535                                  BEQ     10$                     ;BRANCH IF NOT
007332  032760  000002  002456                  BIT     #BIT1,CSRINFO(R0)       ;GET TYPE OF ECC MEMORY
007340  001003                                  BNE     55$                     ;BRANCH IF MS11-P
007342  012702  000157                          MOV     #157,R2                 ;MS11-M CHECK BITS
007346  000402                                  BR      56$
007350  012702  000040                  55$:    MOV     #40,R2                  ;MS11-P CHECK BITS
007354  016004  172100                  56$:    MOV     CSRADD(R0),R4           ;GET CSR CONTENTS
007360  072427  177773                          ASH     #-5,R4
007364  042704  177600                          BIC     #^C177,R4               ;CLEAR ALL BUT CHECKBITS
007370  020204                                  CMP     R2,R4                   ;PROPER CHECKBITS?
007372  001401                                  BEQ     7$                      ;BRANCH IF SO
007374  000747                                  BR      6$                      ;TRY NEXT CSR IF NOT
007376  110037  002526                  7$:     MOVB    R0,PGMCSR               ;WRITE NON-ASSERTED CSR # IN PGMCSR
007402                                          SET4    #8$                     ;NE TRAPS GO TO 8$
007410  012700  177776                          MOV     #-2,R0
007414  013703  002222                          MOV     TOTCSRS,R3              ;PUT CSR MAP IN R3
007420  062700  000002                  23$:    ADD     #2,R0                   ;UPDATE CSR COUNTER
007424  010037  002150                          MOV     R0,CSRNO                ;UPDATE CSRNO
007430  006303                                  ASL     R3
007432  103403                                  BCS     24$                     ;BRANCH IF CSR EXISTS
007434  005703                                  TST     R3                      ;ANY CSR'S LEFT?
007436  001405                                  BEQ     25$                     ;BRANCH IF NOT
007440  000767                                  BR      23$                     ;LOOK FOR NEXT CSR
007442  012760  020006  172100          24$:    MOV     #20006,CSRADD(R0)       ;SET DIAGNOSTIC CHECK MODE IN CSR
007450  000763                                  BR      23$                     ;LOOK FOR NEXT CSR
007452  012700  177776                  25$:    MOV     #-2,R0
007456  005771  000002                          TST     @2(R1)                  ;TEST ASSERTED LOCATIONS
007462  062700  000002                  8$:     ADD     #2,R0
007466  010037  002150                          MOV     R0,CSRNO
007472  022700  000040                          CMP     #40,R0
```

READ AND WRITE ALL CSR BITS

```
6392 007476  001452                        BEQ     10$
6393 007500  032760  000002  002456        BIT     #BIT1,CSRINFO(R0)       ;CHECK FOR TYPE OF ECC MEMORY
6394 007506  001003                        BNE     76$                     ;BRANCH IF MS11-P
6395 007510  012702  000157                MOV     #157,R2                 ;CHECK BITS FOR MS11-M
6396 007514  000402                        BR      77$
6397 007516  012702  000040        76$:    MOV     #40,R2                  ;CHECK BITS FOR MS11-P
6398 007522  016004  172100        77$:    MOV     CSRADD(R0),R4
6399 007526  072427  177773                ASH     #-5,R4
6400 007532  042704  177600                BIC     #^C177,R4
6401 007536  020204                        CMP     R2,R4                   ;PROPER CHECKBITS?
6402 007540  001401                        BEQ     9$                      ;BRANCH IF SO
6403 007542  000747                        BR      8$                      ;TRY NEXT CSR IF NOT
6404 007544  110037  002527        9$:     MOVB    R0,PGMCSR+1             ;WRITE ASSERTED CSR # IN PGMCSR
6405 007550  052737  100000  002526        BIS     #BIT15,PGMCSR           ;SET INTERLEAVED INDICATOR IN PGMCSR
6406 007556  104502                        CLRCSR
6407 007560  012737  002000  172350        MOV     #2000,KIPAR4
6408 007566                                SET4    #12$                    ;NE MEMORY TRAPS GO TO 12$
6409 007574  012771  000000  000000  11$:  MOV     #0,@(R1)                ;WRITE DATA AT FIRST LOCATION OF BANK 2 IN BOARD
6410 007602  012771  000000  000002        MOV     #0,@2(R1)               ;WRITE DATA AT SECOND LOCATION OF BANK 2 IN BOARD
6411 007610  062737  010000  172350        ADD     #10000,KIPAR4           ;UPDATE PAR4 TO POINT TO UPPER BOARDS
6412 007616  000766                        BR      11$
6413 007620  104423                12$:    CACHON
6414 007622  000407                        BR      CLRMEM
6415 007624  012737  001000  172350  10$:  MOV     #1000,KIPAR4            ;
6416 007632                                TYPE    MSG126                  ;ERROR - PROGRAM CSR NOT FOUND!
6417 007636  005037  002526                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.
                                ;
007642                  CLRMEM: SET4    #CLREX          ;NONEM TRAPS GO TO CLREX
007650  005037  006204          CLR     TRACE
007654  012737  000001  002074          MOV     #1,NOPAR        ;IGNORE PARITY ERRORS
007662  012737  002000  172350          MOV     #2000,KIPAR4    ;SET UP MAP TO START AT BANK 2
007670  012701  100000              MOV     #100000,R1      ;R1 MAPS TO KIPAR4
007674  020127  117776      1$:     CMP     R1,#117776      ;WHOLE 16K BANK DONE?
007700  001003                  BNE     2$              ;KEEP GOING IF NOT
007702  012737  177777  006204          MOV     #-1,TRACE       ;USE TRACE FLAG TO FLAG END OF BANK
007710  005021              2$:     CLR     (R1)+           ;CLEAR CONTENTS & INCREMENT
007712  005737  006204              TST     TRACE           ;EOB FLAG SET?
007716  001001                  BNE     3$              ;GO TO NEXT BANK IF SO
007720  000765                  BR      1$
007722  062737  000200  172350  3$:     ADD     #200,KIPAR4     ;SET MAP FOR NEXT BANK
007730  022737  170000  172350          CMP     #170000,KIPAR4  ;ARE WE AT THE PERPHERIAL PAGE
007736  001405                  BEQ     CLREX           ;YES-GO ON
007740  005037  006204              CLR     TRACE           ;RESET FLAG
007744  012701  100000              MOV     #100000,R1      ;RESET R1
007750  000751                  BR      1$              ;CLEAR NEXT BANK
007752  000240              CLREX:  NOP
007754  005037  006204              CLR     TRACE
007760                          RES4
```

```
   6449 010002                          ANA2:   SUBTST  <<MATCH ALL CSR'S WITH MEMORY>>
                                        ;*********************************************************************************
                                        ;*SUBTEST       MATCH ALL CSR'S WITH MEMORY
                                        ;*********************************************************************************
   6450                                 ;* THE SECOND PART OF THE ANALYSIS MATCHES UP THE CSR'S WITH THE MEMORY, AND
   6451                                 ;* ".S;ALLS ALL THE INFORMATION FOUND IN THE CONFIGURATION TABLE.  FOR ECC,
   6452                                 ;* THIS IS DONE BY TAKING EACH CSR FOUND IN THE PREVIOUS SECTION SEQUENTIALLY
   6453                                 ;* AND CHECKING THROUGH ALL OF MEMORY, ONE BANK AT AT TIME, TO SEE WHICH BANKS
   6454                                 ;* RESPOND TO THE CSR IN QUESTION. THE FIRST DOUBLE WORD PAIR IN EACH BANK ARE
   6455                                 ;* WRITTEN WITH DATA AND DIAGNOSTIC CHECK MODE SET IN THE CSR IN ORDER TO AC-
   6456                                 ;* COMPLISH THIS. ALL POSSIBLE CONFIGURATIONS OF DOUBLE WORD PAIRS (NON-INTER-
   6457                                 ;* LEAVED, 64K INTERLEAVED, OR 128K INTERLEAVED) ARE CHECKED FOR EACH BANK
   6458                                 ;* THROUGH USE OF TESTADD AND KERNEL INSTRUCTION PAGE ADDRESS REGISTERS 4 AND
   6459                                 ;* 5. IF WE GET THE PROPER CHECKBITS BACK, WE HAVE A MATCH. IF NOT, THE ROUT-
   6460                                 ;* INE CHECKS FOR SINGLE OR DOUBLE BIT ERRORS.
   6461                                 ;*                 IF ONE OR THE OTHER IS FOUND, THE ERROR ADDRESS IS CHECKED
   6462                                 ;* TO SEE IF IT IS THAT BANK.  IF IT IS, WE HAVE A MATCH.  AT THE END OF EACH
   6463                                 ;* BANK PASS, FOR EACH CSR PASS, THE PROGRAM COMES UP WITH A NUMBER, STORED IN
   6464                                 ;* "I", WHICH DENOTES THE FOLLOWING:
   6465                                 ;*
   6466                                 ;*       I       MEMORY DESCRIP:ION
   6467                                 ;*       -       -----------------
   6468                                 ;*       0       NON-EXISTANT MEMORY
   6469                                 ;*       1       NON-INTERLEAVED MEMORY MS11-L,MS11-P
   6470                                 ;*       2       64K INTERLEAVED, A1 NOT ASSERTED MEMORY
   6471                                 ;*       3       128K INTERLEAVED, A1 NOT ASSERTED MEMORY
   6472                                 ;*       4       64K INTERLEAVED, A1 ASSERTED MEMORY
   6473                                 ;*       5       128K INTERLEAVED, A1 ASSERTED MEMORY
   6474                                 ;*
   6475                                 ;*  NOTE - I=2 THROUGH I=5 CAN ONLY OCCUR WITH MS11-M MEMORY.
   6476                                 ;*
   6477                                 ;* NOTE THAT PARITY MEMORY WRITES WRONG PARITY TO THE DOUBLE WORDS, THEN LOOKS
   6478                                 ;* FOR THE PARITY ERROR BIT TO BE SET.  IF THE BIT IS SET, WE HAVE A MATCH.
   6479                                 ;*
   6480 010002                                  SET4    #100$                   ;NE MEMORY TRAPS GO TO 100$
   6481 010010  005037  002310                  CLR     CHECK                   ;CLEAR CHECK
   6482 010014  012701  002406                  MOV     #TESTADD,R1             ;SET UP THE VIRTUAL ADDR. POINTER
   6483 010020  013703  002222                  MOV     TOTCSRS,R3              ;MOVE CSR MAP INTO R3
   6484 010024  005000                          CLR     R0                      ;CLEAR THE CSR POINTER
   6485 010026  005005                          CLR     R5                      ;CLEAR THE PROGRAM CSR STATUS POINTER
   6486 010030  005737  002450                  TST     NO22BIT                 ;IS THIS AN 11/44 OR 11/24?
   6487 010034  001403                          BEQ     7$                      ;BRANCH IF IT IS
   6488 010036  005037  002554                  CLR     LASTBLOCK               ;ADJUST LASTBLOCK INDICATOR FOR 124K MACHINE
   6489 010042  000413                          BR      1$                      ;BRANCH OVER NEXT PIECE OF CODE
   6490 010044  022737  000167  002552  7$:     CMP     #167,LASTBANK           ;IS THERE UNIBUS MEMORY ABOVE 17000000?
   6491 010052  001407                          BEQ     1$                      ;BRANCH IF NOT
   6492 010054  013702  002552                  MOV     LASTBANK,R2             ;SET UP A NEW LAST BLOCK INDICATOR
   6493 010060  005202                          INC     R2
   6494 010062  072227  000011                  ASH     #9.,R2
   6495 010066  010237  002554                  MOV     R2,LASTBLOCK
   6496 010072  012702  000004          1$:     MOV     #4,R2                   ;R2 IS INDEX FOR CONFIG TABLE
   6497 010076  012737  001000  172350          MOV     #1000,KIPAR4            ;SET KIPAR4 FOR BANK 1
   6498 010104  012737  001000  172352          MOV     #1000,KIPAR5            ;SET KIPAR5 FOR BANK 1
   6499 010112  006303                  2$:     ASL     R3                      ;DOES THIS CSR EXIST?
   6500 010114  103420                          BCS     3$                      ;BRANCH IF IT DOES EXIST
   6501 010116  062700  000002                  ADD     #2,R0                   ;INCREMENT THE CSR POINTER
   6502 010122  010037  002150                  MOV     R0,CSRNO                ;STORE IT IN CSRNO ALSO
```

```
6503 010126  005703                          TST     R3                      ;ARE THERE ANY MORE CSR'S TO DO?
6504 010130  001370                          BNE     2$                      ;BRANCH IF ALL CSRS NOT DONE
6505 010132  012737  001000  172350          MOV     #1000,KIPAR4            ;RESTORE KIPAR4
6506 010140  012737  001200  172352          MOV     #1200,KIPAR5            ;RESTORE KIPAR5
6507 010146  013706  002560                  MOV     KSTACK,SP               ;RESTORE STACK
6508 010152  000137  011400                  JMP     SUBAAS                  ;JUMP TO SUBAAS IF ALL CSR'S ARE DONE
6509 010156  010037  002150          3$:     MOV     R0,CSRNO                ;MAKE SURE CSRNO IS UPDATED
6510 010162  104424                  13$:    CACHOFF                         ;TURN THE CACHE OFF
6511 010164  000240                          NOP
6512 010166  012737  100000  002406  45$:    MOV     #100000,TESTADD         ;SET UP VIRTUAL ADDRESS TO KIPAR4
6513 010174  012737  120002  002410          MOV     #120002,TESTADD+2       ;SET UP VIRTUAL ADDRESS TO KIPAR5
6514 010202  032762  000040  002650          BIT     #BIT5,CONFIG(R2)        ;IS THIS A BANK TO SKIP?
6515 010210  001402                          BEQ     43$                     ;NO - BRANCH AROUND NEXT INSTRUCTION
6516 010212  000137  011314                  JMP     6$                      ;YES - GO TO END OF BANK
6517 010216  005037  002446          43$:    CLR     I                       ;CLEAR THE MEMORY CONFIGURATION COUNTER
6518 010222  005771  000000          4$:     TST     @(R1)                   ;TEST TO SEE THAT THERE IS MEMORY PRESENT
6519 010226  005237  002446                  INC     I
6520 010232                                  PUSH    @(R1),@2(R1)            ;SAVE THE LOCATIONS UNDER TEST
6521 010242  032760  000003  002456          BIT     #BIT1!BIT0,CSRINFO(R0)  ;IS THIS PARITY MEMORY?
6522 010250  001014                          BNE     34$                     ;NO - BRANCH
6523 010252  052760  000004  172100          BIS     #BIT2,CSRADD(R0)        ;SET WRITE WRONG PARITY
6524 010260  012771  123456  000000          MOV     #123456,@(R1)           ;SET THE FIRST LOCATION UNDER TEST
6525 010266  012771  123456  000002          MOV     #123456,@2(R1)          ;SET THE SECOND LUT
6526 010274  005060  172100                  CLR     CSRADD(R0)              ;CLEAR THE CSR
6527 010300  000411                          BR      41$                     ;TEST LOCATIONS
6528 010302  012771  123456  000000  34$:    MOV     #123456,@(R1)           ;SET THE FIRST LOCATION UNDER TEST
6529 010310  012771  123456  000002          MOV     #123456,@2(R1)          ;SET THE SECOND LUT
6530 010316  104503                          CLR1CSR                         ;RESET CSR
6531 010320  104475                          CB1CSR                          ;SET DIAG. CHECK MODE IN CSR UNDER TEST
6532 010322  000240                          NOP                             ;DEBUG AID
6533 010324  005771  000000          41$:    TST     @(R1)                   ;READ THE FIRST LUT TO WRITE CKBITS. INTO CSR
6534 010330  104426                          READCSR                         ;READ THE CSR UNDER TEST
6535 010332  000240                          NOP                             ;DEBUG AID
6536 010334  013704  002146                  MOV     CSR,R4                  ;GET THE CHECKBITS FROM THE CSR
6537 010340  000240                          NOP                             ;DEBUG AID
6538 010342  010437  002430                  MOV     R4,TEMP                 ;SAVE IN TEMP FOR LATER
6539 010346  104503                          CLR1CSR                         ;RESET CSR
6540 010350                                  POP     @2(R1),@(R1)            ;RESTORE LOCATIONS UNDER TEST
6541 010360  032760  000003  002456          BIT     #BIT1!BIT0,CSRINFO(R0)  ;IS THIS PARITY MEMORY?
6542 010366  001004                          BNE     42$                     ;NO - BRANCH
6543 010370  005704                          TST     R4                      ;DID WE GET A PARITY ERROR?
6544 010372  100431                          BMI     25$                     ;YES - FILL IN CONFIG TABLE
6545 010374  000137  011314                  JMP     6$                      ;NO - JUMP TO END OF BANK
6546 010400  072427  177773          42$:    ASH     #-5,R4                  ;MANIPULATE THE CSR BITS
6547 010404  042704  177600                  BIC     #^C177,R4               ;INTO A USABLE FORM.
6548 010410  032760  000002  002456          BIT     #BIT1,CSRINFO(R0)       ;WHAT KIND OF ECC MEMORY IS THIS??
6549 010416  001005                          BNE     76$                     ;BRANCH IF MS11-P
6550 010420  012737  000157  002312          MOV     #157,CBITS              ;MS11-M CHECK BITS
6551 010426  000240                          NOP                             ;DEBBUG AIDE
6552 010430  000404                          BR      77$                     ;
6553 010432  012737  000040  002312  76$:    MOV     #40,CBITS               ;MS11-P CHECK BITS
6554 010440  000240                          NOP                             ;DEBBUGGING AIDE
6555 010442  023704  002312          77$:    CMP     CBITS,R4                ;DO THE CHECKBITS COMPARE TO WHAT WAS WRITTEN?
6556 010446  000240                          NOP                             ;DEBBUG AIDE
6557 010450  001402                          BEQ     25$                     ;BRANCH IF THERE IS A MATCH
6558 010452  000137  011002                  JMP     22$                     ;ELSE BRANCH IF NOT THE SAME
6559                                         ;*
```

```
                                                ;* WE COME HERE IF THERE IS A MATCH
                                                ;*
010456  010004                        25$:    MOV     R0,R4                   ;GET THE CSR NUMBER
010460  000240                                NOP                             ;
010462  006204                                ASR     R4                      ;SET IT UP FOR USE IN THE
010464  000304                                SWAB    R4                      ;CONFIGURATION TABLE.
010466  042704  170377                        BIC     #170377,R4              ;CLEAR OFF EXTRANEOUS BITS
010472  032737  000004  002446                BIT     #BIT2,I                 ;INTERLEAVED A1 ASSERTED MEMORY FOUND?
010500  001402                                BEQ     15$                     ;BRANCH IF NOT
010502  072427  000004                        ASH     #4,R4                   ;PUT CSR NUMBER IN INTERLEAVED CSR SLOT
010506  050462  002650                15$:    BIS     R4,CONFIG(R2)           ;PUT CSR NUMBER IN CONFIG. TABLE
010512  016004  002456                        MOV     CSRINFO(R0),R4          ;GET MEMORY TYPE
010516  042704  177770                        BIC     #^C7,R4                 ;CLEAR OFF THE EXTRANEOUS BITS
010522  000304                                SWAB    R4                      ;MOVE INTO PROPER POSITION
010524  050462  002652                        BIS     R4,CONFIG+2(R2)         ;SET IT INTO THE CONFIG TABLE
010530  022737  000001  002446                CMP     #1,I                    ;WAS THIS NON-INTERLEAVED MEMORY?
010536  001431                                BEQ     24$                     ;BRANCH IF IT WAS
010540  052762  010000  002652                BIS     #BIT12,CONFIG+2(R2)     ;SET THE INTERLEAVED BIT
010546  010204                                MOV     R2,R4                   ;SAVE THE CURRENT BANK INDEX
010550  032737  000001  002446                BIT     #BIT0,I                 ;WAS THIS 128K INTERLEAVED?
010556  001006                                BNE     5$                      ;BRANCH IF TRUE
010560  052762  004000  002652                BIS     #BIT11,CONFIG+2(R2)     ;SET 64K INTERLEAVED FLAG IN CONFIG
010566  062704  000020                        ADD     #20,R4                  ;SET NEW BANK POINTER TO 4 BANKS AHEAD
010572  000402                                BR      16$                     ;JUMP OVER NEXT INSTRUCTION
010574  062704  000040                5$:     ADD     #40,R4                  ;SET NEW BANK POINTER 8 BANKS AHEAD
010600  052764  000040  002650        16$:    BIS     #BIT5,CONFIG(R4)        ;SET SKIP ECC LOGIC TESTS FLAG
010606  056264  002650  002650                BIS     CONFIG(R2),CONFIG(R4)   ;SET OTHER INFO INTO THAT BANK
010614  056264  002652  002652                BIS     CONFIG+2(R2),CONFIG+2(R4)
                                                ;*
                                                ;* THIS SECTION IS EXECUTED ONLY WHEN THE BANK=1
                                                ;*
010622  022737  001000  172350  24$:    CMP     #1000,KIPAR4            ;IS THIS BANK 1 ?
010630  001402                                BEQ     30$                     ;BRANCH IF TRUE
010632  000137  011154                        JMP     33$                     ;ELSE JUMP TO END OF THIS BANK
010636  032737  100020  002430  30$:    BIT     #BIT15!BIT4,TEMP        ;WAS THERE A SBE OR DBE?
010644  001417                                BEQ     10$                     ;BRANCH IF NOT
010646  013704  002430                        MOV     TEMP,R4                 ;GET CSR CONTENTS
010652  072427  177767                        ASH     #-9.,R4                 ;MAKE ERROR ADDRESS INTO BANK #
010656  022704  000001                        CMP     #1,R4                   ;ERROR IN BANKS 0 OR 1?
010662  003010                                BGT     10$                     ;BRANCH IF NOT
010664  052762  000001  002650                BIS     #BIT0,CONFIG(R2)        ;SET ERROR FLAG IN CONFIG TABLE
010672  105262  002652                        INCB    CONFIG+2(R2)            ;ADD ONE TO BANK ERROR COUNT
010676                                        SET     CONFGERROR              ;PRINT CONFIG TABLE
010704  053737  002654  002650  10$:    BIS     CONFIG+4,CONFIG         ;SET UP INFORMATION IN BANK ZERO
010712  053737  002656  002652                BIS     CONFIG+6,CONFIG+2
010720  000240                                NOP                             ;DEBUG AID
010722  022737  000001  002446                CMP     #1,I                    ;WAS THIS NON-INTERLEAVED MEMORY
010730  001002                                BNE     46$                     ;NO - BRANCH OVER NEXT STMT.
010732  000137  011314                        JMP     6$                      ;YES - JUMP TO END OF THIS BANK
010736  012704  000020                46$:    MOV     #20,R4                  ;SET UP COUNTER FOR 64K INTERLEAVED
010742  032737  000001  002446                BIT     #BIT0,I                 ;WAS IT 128K INTERLEAVED?
010750  001402                                BEQ     26$                     ;BRANCH IF NOT
010752  062704  000020                        ADD     #20,R4                  ;SET UP COUNTER FOR 128K INTERLEAVED
010756  053764  002650  002650  26$:    BIS     CONFIG,CONFIG(R4)       ;SET OTHER BANK WITH SAME INFORMATION
010764  053764  002652  002652                BIS     CONFIG+2,CONFIG+2(R4)   ;AS IN BANK 0
010772  052764  000040  002650                BIS     #BIT5,CONFIG(R4)        ;SET SKIP ECC LOGIC TESTS FLAG
011000  000465                                BR      33$                     ;BRANCH
```

```
6617                                                    ;*
6618                                                    ;* IF CHECKBITS DID NOT MATCH, WE COME HERE
6619                                                    ;*
6620 011002  032737  100020  002146  22$:    BIT     #BIT15!BIT4,CSR         ;SBE OR DBE FLAGS SET?
6621 011010  001001                         BNE     8$                      ;BRANCH IF TRUE
6622 011012  000460                         BR      33$                     ;CHECK TO SEE IF IT IS MS11-M
6623 011014  013704  002150                 8$:     MOV     CSRNO,R4        ;GET CSRNO
6624 011020  042764  000006  172100                 BIC     #6,CSRADD(R4)   ;TURN OFF DIAG CHECK & ECC DISABLE
6625 011026                                 PUSH    R0,R1                   ;SAVE R0 & R1
6626 011032  016401  172100                 MOV     CSRADD(R4),R1           ;GET CSR INFORMATION
6627 011036  072127  177773                 ASH     #-5,R1                  ;SET UP ERROR ADDRESS
6628 011042  042701  177600                 BIC     #^C177,R1
6629 011046  052764  040000  172100         BIS     #BIT14,CSRADD(R4)       ;GET EXTENDED ERROR ADDRESS BITS
6630 011054  016400  172100                 MOV     CSRADD(R4),R0           ;READ FROM CSR
6631 011060  042764  040000  172100         BIC     #BIT14,CSRADD(R4)       ;TURN OFF EUB BIT
6632 011066  042700  177037                 BIC     #^C740,R0               ;SET UP EXTENDED BITS
6633 011072  006300                         ASL     R0
6634 011074  006300                         ASL     R0
6635 011076  060001                         ADD     R0,R1                   ;SET UP TOTAL ERROR ADDRESS
6636 011100  010104                 27$:    MOV     R1,R4                   ;SAVE IN R4
6637 011102                                 POP     R1,R0                   ;RESTORE R0 & R1
6638 011106  072427  000005                 ASH     #5,R4                   ;SET ERROR ADDRESS UP IN PAR NOTATION
6639 011112  020437  172350                 CMP     R4,KIPAR4               ;DOES IT EQUAL KIPAR4?
6640 011116  001001                         BNE     28$                     ;BRANCH IF FALSE
6641 011120  000403                         BR      35$                     ;YES - MARK INFO IN CONFIG TABLE
6642 011122  020437  172352                 28$:    CMP     R4,KIPAR5       ;DOES IT EQUAL KIPAR5?
6643 011126  001012                         BNE     33$                     ;BRANCH IF FALSE
6644 011130  052762  000001  002650  35$:    BIS     #BIT0,CONFIG(R2)        ;SET BANK ERROR FLAG
6645 011136  105262  002652                 INCB    CONFIG+2(R2)            ;INCREMENT BANK ERROR COUNTER
6646 011142                                 SET     CONFGERROR              ;PRINT CONFIG TABLE
6647 011150  000137  010456                 JMP     25$                     ;YES - MARK INFO IN CONFIG TABLE
6648                                                    ;*
6649                                                    ;* THIS SECTION SETS UP ALL THE POSSIBLE CONFIGURATIONS OF
6650                                                    ;* MS11-M MEMORY.
6651                                                    ;*
6652 011154  032760  000001  002456  33$:    BIT     #BIT0,CSRINFO(R0)       ;IS THIS MS11-M MEMORY?
6653 011162  001454                         BEQ     6$                      ;NO - GO TO END OF BANK
6654 011164  032760  000002  002456         BIT     #BIT1,CSRINFO(R0)
6655 011172  001050                         BNE     6$
6656 011174  022737  000001  002446         CMP     #1,I                    ;IS THIS 1ST TIME THROUGH?
6657 011202  103410                         BLO     18$                     ;BRANCH IF NOT
6658 011204  162737  000002  002410         SUB     #2,TESTADD+2            ;TRY AS 64K INTERLEAVED
6659 011212  062737  004000  172352         ADD     #4000,KIPAR5            ;A1 NON-ASSERTED MEMORY
6660 011220  000137  010222                 JMP     4$                      ;TRY TO MATCH AGAIN
6661 011224  022737  000004  002446  18$:    CMP     #4,I                    ;4TH TIME THROUGH?
6662 011232  001404                         BEQ     20$                     ;YES - BRANCH
6663 011234  022737  000002  002446         CMP     #2,I                    ;2ND TIME THROUGH
6664 011242  103405                         BLO     12$                     ;NO - BRANCH
6665 011244  062737  004000  172352  20$:    ADD     #4000,KIPAR5            ;TRY AS 128K INTERLEAVED
6666 011252  000137  010222                 JMP     4$                      ;TRY TO MATCH AGAIN
6667 011256  022737  000003  002446  12$:    CMP     #3,I                    ;THIRD TIME THROUGH?
6668 011264  103413                         BLO     6$                      ;NO - BRANCH
6669 011266  062737  000002  002406         ADD     #2,TESTADD              ;TRY TESTING THE BANK
6670 011274  062737  000002  002410         ADD     #2,TESTADD+2            ;AS A1 ASSERTED
6671 011302  162737  004000  172352         SUB     #4000,KIPAR5            ;64K INTERLEAVED MEMORY
6672 011310  000137  010222                 JMP     4$                      ;TRY TO MATCH AGAIN
6673                                                    ;*
```

```
6674                                                    ;*END OF BANK ROUTINE
6675                                                    ;*
6676 011_ 4  104503                          6$:        CLR1CSR                 ;CLEAR THE CSR UNDER TEST
6677 0113    062702  000004                            ADD     #4,R2           ;UPDATE CONFIGURATION POINTER
6678 011322  062737  001000  172350                    ADD     #1000,KIPAR4    ;UPDATE KIPAR4 TO NEXT BANK
6679 011330  013737  172350  172352                    MOV     KIPAR4,KIPAR5   ;AND UPDATE KIPAR5
6680 011336  000240                                    NOP                     ;DEBUG AID
6681 011340  02373_  002554  172350                    CMP     LASTBLOCK,KIPAR4 ;HAVE WE DONE THE WHOLE MEMORY SPACE?
6682 011346  10140 ·                                   BLOS    19$             ;BRANCH IF DONE                 ;R-C
6683 011350  000137  010166                            JMP     45$             ;JUMP IF NOT DONE
6684 011354  062700  000002                 19$:       ADD     #2,R0           ;INCREMENT CSR POINTER
6685 011360  000240                                    NOP                     ;DEBUG AID
6686 011362  104423                                    CACHON                  ;TURN ON THE CACHE
6687 011364  000137  010072                            JMP     1$              ;JUMP TO TRY NEXT CSR
6688                                          ;
6689 011370  062706  000004                 100$:      ADD     #4,SP           ;RESTORE STACK                  ;R-C
6690 011374  000137  011314                            JMP     6$              ;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
                                ;*****************************************************************************
011404  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
011406  005037  002070                  CLR     PARCNT                  ;CLEAR PARITY ERROR COUNTER
011412  012737  000001  002074          MOV     #1,NOPAR                ;SET THE NO PARITY ERROR FLAG
011420  005037  002066                  CLR     NEMCNT                  ;CLEAR NON-EXISTANT MEMORY ERROR COUNTER
011424  012737  000001  002076          MOV     #1,NONEM                ;SET THE NON-EXISTANT MEMORY ERROR MODE TO COUNT
011432                                  SET4    #NONEXIST               ;TRAPS TO 4 GOTO NONEXIST
011440  005000                          CLR     R0
011442  012701  040000                  MOV     #SIZE,R1
011446  104424                          CACHOFF                         ;TURN CACHE OFF
011450  005720                  1$:     TST     (R0)+                   ;SEE IF I CAN DO A READ ACCESS WITHOUT A PARITY TRAP
011452  077102                          SOB     R1,1$
011454  104423                          CACHON                          ;TURN CACHE ON
                                        ;SEE IF ANY FAILURES
011456  005737  002070                  TST     PARCNT                  ;ANY PARITY ERRORS?
011462  001403                          BEQ     2$                      ;NO - SKIP
011464                                  FATAL   3
011472  005737  002066          2$:     TST     NEMCNT                  ;ANY NON-EXISTANT MEMORY (HOLES)?
011476  001406                          BEQ     3$                      ;SKIP IF EQUAL
011500  162737  000002  002032          SUB     #2,ADDRESS              ;UPDATE 1ST ADDRESS FAILURE FROM AUTO INCREMENT #
011506                                  FATAL   4
011514  053737  002104  002650  3$:     BIS     CPUBIT,CONFIG           ;SET CORRECT ACCESSED BIT ON BANK 0
011522                                  RES4                            ;RESET TRAPS TO 4 TO DEFAULT

011544                                  SUBTST  <<ENABLE ECC FOR CORRECT TRAPS>>
                                ;*****************************************************************************
                                ;*SUBTEST        ENABLE ECC FOR CORRECT TRAPS
                                ;*****************************************************************************
011544                                  IF #SW0 SET.IN @SWR OR ACTFLAG IS TRUE
104506                                    ENASBE                        ;TRAP ON SINGLE BIT ERRORS
011564                                  ELSE
104472                                    ECCINIT                       ;TRAP ON DOUBLE BIT ERRORS (NORMAL)
011570                                  END ;OF IF #SW0
```

ENABLE ECC FOR CORRECT TRAPS

```
                                        NEWTST  <<TEST BANKS 1-200 (OCTAL) FOR ZEROS & ONES>>
                                ;*******************************************************************************
                                ;*TEST 3         TEST BANKS 1-200 (OCTAL) FOR ZEROS & ONES
                                ;*******************************************************************************
011570  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!
011572  005037  002100                  CLR     BANK
011576  012737  000001  002074          MOV     #1,NOPAR                ;SET NO PARITY ERROR FLAG
011604  012737  000002  002076          MOV     #2,NONEM                ;SET NON-EXISTANT MEMORY MODE TO EXIT TEST LOOP
011612                                  SET4    #NONEXIST               ;TRAPS TO 4 GOTO NONEXIST
011620  022737  000001  003752          CMP     #1,PROTYP               ;IS THIS AN 11/44?
011626  001407                          BEQ     1$                      ;BRANCH IF TRUE
011630  012737  012426  002516          MOV     #MTST3+4,LINK1          ;SET UP LINKS
011636  012737  012430  002520          MOV     #MTST3+6,LINK2
011644  000411                          BR      TAG9$
011646                          1$:     BMOV    MTST3                   ;PUT IN FAST MEMORY
011654  012737  177644  002516          MOV     #UIPAR2,LINK1           ;SET UP LINKS
011662  012737  177646  002520          MOV     #UIPAR3,LINK2
011670  005237  002100          TAG9$:  INC     BANK
011674  023737  002552  002100          CMP     LASTBANK,BANK           ;DONE?
011702  103457                          BLO     TAG2$                   ;YES - SKIP TO NEXT TEST
011704  013701  002100                  MOV     BANK,R1
011710  006301                          ASL     R1
011712  006301                          ASL     R1                      ;BANK * 4
011714  010137  002102                  MOV     R1,BANKINDEX
011720  005037  002072                  CLR     PATERR                  ;CLEAR PATTERN ERROR COUNTER
011724  005037  002070                  CLR     PARCNT                  ;CLEAR PARITY ERROR COUNTER
011730  005037  002066                  CLR     NEMCNT                  ;CLEAR NON-EXISTANT MEMORY COUNTER (HOLES)
011734                                  MAP     BANK                    ;MAP SUPERVISOR SPACE (TEST AREA) TO BANK
011750  105761  002650                  TSTB    CONFIG(R1)              ;IS THIS BANK PROTECTED?
011754  100555                          BMI     TSTBANK                 ;YES - GO TEST BANK SPECIAL
011756  012777  000207  170532  WARN1:  MOV     #207,@LINK1             ;PUT "RETURN" INSTRUCTION AFTER WRITE ROUTINE
011764  012700  060000                  MOV     #FIRST,R0
011770  010004                          MOV     R0,R4
011772  012701  040000                  MOV     #SIZE,R1
011776  010103                          MOV     R1,R3
012000  005002                          CLR     R2                      ;DATA IS ZEROS
012002  104424                          CACHOFF                         ;TURN CACHE OFF
012004                                  TESTAREA                        ;ENTER SUPERVISOR MODE
012012  022737  000001  003752          CMP     #1,PROTYP               ;IS THIS AN 11/44?
012020  001403                          BEQ     1$                      ;BRANCH IF TRUE
012022  004737  012422                  CALL    MTST3
012026  000402                          BR      2$
01203C  004737  177640          1$:     CALL    FASTCITY                ;CALL TO THE USER INSTRUCTION PAR'S
012034  104417                  2$:     KERNEL                          ;ENTER KERNEL MODE
012036  104423                          CACHON                          ;TURN CACHE ON
012040  000416                          BR      TAG3$                   ;SKIP NEXT INSTRUCTION
012042  005037  002100          TAG2$:  CLR     BANK
012046                                  RES4                            ;RESET TRAPS TO 4 TO DEFAULT
012070  005037  002074                  CLR     NOPAR                   ;INDICATE DEFAULT PARITY ACTION
012074  000564                          BR      SUBAAI
```

```
6784 012076  005737  002066          TAG3$:  TST     NEMCNT                  ;ANY TRAPS?
6785 012102  001401                          BEQ     1$                      ;NO - SKIP
6786 012104  000671                          BR      TAG9$                   ;NOW - TRY NEXT BANK
6787 012106  104424                  1$:     CACHOFF                         ;TURN CACHE OFF
6788 012110                                  TESTAREA                        ;ENTER SUPERVISOR MODE
6789 012116  004777  170376                  CALL    @LINK2                  ;FINISH PATTERN
6790 012122  104417                          KERNEL                          ;ENTER KERNEL MODE
6791 012124  104423                          CACHON                          ;TURN CACHE ON
6792 012126  005737  002072                  TST     PATERR                  ;ANY PATTERN ERRORS
6793 012132  001040                          BNE     2$                      ;YES - SKIP
6794 012134  005737  002070                  TST     PARCNT                  ;ANY PARITY ERRORS
6795 012140  001035                          BNE     2$                      ;YES - SKIP
6796 012142  005737  002066                  TST     NEMCNT                  ;ANY NON EXISTANT MEMORY
6797 012146  001032                          BNE     2$                      ;YES - SKIP
6798 012150  012700  060000                  MOV     #FIRST,R0
6799 012154  010004                          MOV     R0,R4
6800 012156  012701  040000                  MOV     #SIZE,R1
6801 012162  010103                          MOV     R1,R3
6802 012164  013702  002600                  MOV     ONES,R2                 ;DATA IS ONES
6803 012170  012777  000240  170320          MOV     #000240,@LINK1          ;PUT 'NOP' INSTRUCTION BACK IN SUBROUTINE
6804 012176  104424                          CACHOFF                         ;TURN CACHE OFF
6805 012200                                  TESTAREA                        ;ENTER TEST MODE
6806 012206  022737  000001  003752          CMP     #1,PROTYP               ;IS THIS AN 11/44?
6807 012214  001403                          BEQ     5$                      ;BRANCH IF IT IS
6808 012216  004737  012422                  CALL    MTST3                   ;DO IN MEMORY IF NOT
6809 012222  000402                          BR      6$                      ;JUMP OVER NEXT INSTRUCTION
6810 012224  004737  177640          5$:     CALL    FASTCITY                ;CALL TO THE USER INSTRUCTION PAR'S
6811 012230  104417                  6$:     KERNEL                          ;ENTER KERNEL MODE
6812 012232  104423                          CACHON                          ;TURN CACHE ON
6813 012234  013700  002102          2$:     MOV     BANKINDEX,R0
6814 012240  005737  002072                  TST     PATERR                  ;ANY PATTERN ERRORS?
6815 012244  001006                          BNE     3$                      ;YES - SKIP
6816 012246  005737  002070                  TST     PARCNT                  ;ANY PARITY ERRORS?
6817 012252  001003                          BNE     3$                      ;YES - SKIP
6818 012254  005737  002066                  TST     NEMCNT                  ;ANY HOLES?
6819 012260  001406                          BEQ     4$                      ;NONE - SKIP
6820 012262  052760  000001  002650  3$:     BIS     #BIT0,CONFIG(R0)        ;SET ERROR BIT IN THIS BANK
6821 012270                                  SET     CONFGERROR              ;FORCE PRINTING OF CONFIGURATION TABLE
6822 012276  053760  002104  002650  4$:     BIS     CPUBIT,CONFIG(R0)       ;SET ACCESSED BIT
6823 012304  000137  011670                  JMP     TAG9$
6824
6825                                         ;TEST A PROTECTED BANK
6826 012310                          TSTBANK:PUSH    R1
6827 012312  012737  000001  002076          MOV     #1,NONEM                ;SET NON-EXISTANT MEMORY TO COUNT
6828 012320  012700  060000                  MOV     #FIRST,R0
6829 012324  012701  020000                  MOV     #20000,R1
6830 012330  104424                          CACHOFF                         ;TURN CACHE OFF
6831 012332                                  TESTAREA                        ;ENTER TEST MODE
6832 012340  005720                  4$:     TST     (R0)+
6833 012342  077102                          SOB     R1,4$
6834 012344  104417                          KERNEL                          ;ENTER KERNEL MODE
6835 012346  104423                          CACHON                          ;TURN CACHE ON
6836 012350  012737  000002  002076          MOV     #2,NONEM                ;RESET NON-EXISTANT MEMORY TO EXIT TEST LOOP
6837 012356                                  POP     R1
6838 012360                                  IF PARCNT NE #0
6839 012366  052761  000001  002650            BIS   #BIT0,CONFIG(R1)        ;ERROR BANK
6840 012374                                    SET   CONFGERROR
```

```
                                                        END ;OF IF PARCNT
                                                        IF NEMCNT EQ #0
 6843 012410  053761  002104  002650                      BIS   CPUBIT,CONFIG(R1)       ;ACCESSED BANK
                                                        END ;OF IF NEMCNT
 6845 012416  000137  011670                            JMP   TAG9$
 6846 012422  010220                          MTST3:    MOV   R2,(R0)+                ;V177640
 6847 012424  077102                                    SOB   R1,MTST3                ;V177642
 6848 012426  000240                                    NOP                           ;V177644
 6849 012430  012401                          2$:       MOV   (R4)+,R1                ;V177646
 6850 012432  020102                                    CMP   R1,R2                   ;V177650
 6851 012434  001402                                    BEQ   3$                      ;V177652
 6852 012436  005237  002072                            INC   PATERR                  ;V177654
 6853 012442  077306                          3$:       SOB   R3,2$                   ;V177660
 6854 012444  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.
012446  005037  002100                CLR     BANK                    ;RESET BANK TO ZERO
012452  004737  047020        SHADL1: CALL    EXBANK                  ;SET BANK PARAMETERS
012456  013700  002102                MOV     BANKINDEX,R0
012462                                IF ACFLAG IS TRUE AND INTFLAG IS TRUE
012476                                  IF INT64K IS TRUE
012504  062700  000020                    ADD #20,R0                  ;POINT TO BANKINDEX + 4
012510  062737  000010  002100             ADD #10,BANK                ;POINT TO BANK + 8
012516                                  ELSE
012520  062702  000040                    ADD #40,R2                  ;POINT TO BANKINDEX + 8
012524  062737  000020  002100             ADD #20,BANK                ;POINT TO BANK + 16
012532                                  END; OF IF INT64K
012532  052760  000200  002650          BIS   #BIT7,CONFIG(R0)        ;MAKE NEW BANK PROGRAM SPACE
012540                                ELSE
012542  005237  002100                  INC   BANK                    ;GO TO NEXT BANK
012546                                END; OF IF ACFLAG
012546  023737  002552  002100        CMP     LASTBANK,BANK           ;HAVE WE DONE ALL THE BANKS?
012554  002336                        BGE     SHADL1                  ;BRANCH IF NOT
```

```
                                        NEWTST  <<ECC INHIBIT MODE POINTER TEST>>
                                ;*****************************************************************************
                                ;*TEST 4         ECC INHIBIT MODE POINTER TEST
                                ;*****************************************************************************
     012556  000004             TST4:   SCOPE
                                        ;THE MS11-M OR MF11S-K INHIBIT ECC DISABLE AND DIAGNOSTIC CHECK MODE
                                        ;ON THE BOTTOM  FIRST OR SECOND 16K WORDS CONTROLLED BY A CSR.  THIS
                                        ;IS CONSIDERED  TO BE A  PROTECTED BANK  BY THE  PROGRAM.  IT MAY BE
                                        ;QUITE  COMPLEX TO DETERMINE ON A  GIVEN SYSTEM  CONFIGURATION WHICH
                                        ;BANKS CAN BE PROTECTED;
                                        ;SO
                                        ;THIS ROUTINE ATTEMPS TO  CREATE A DOUBLE BIT ERROR IN ADDRESS 0 & 2
                                        ;OF  EVERY ECC BANK.  ECC HARDWARE WILL PREVENT THIS  FROM HAPPENING
                                        ;IN  PROTECTED BANKS  WHICH SHOULD ALWAYS  INCLUDE BANK ZERO - WHERE
                                        ;THE PROGRAM IS.
                                        ;
                                        ;
                                        ;WARNING:!!!!!!!!!!
                                        ;  IN CASE OF HARDWARE FAILURE IT IS COMMON  THAT A DOUBLE BIT ERROR
                                        ;  WILL BE  CREATED ON  THE KERNEL  STACK & "CRASH"  THE  DIAGNOSTIC
                                        ;DURING  THIS ROUTINE.  YOUR ONLY CLUE IS THAT YOU CAN GET AS FAR AS
                                        ;THIS ROUTINE BUT NOT PAST IT!
 012560  104424                        CACHOFF                                 ;TURN CACHE OFF
 012562  012737  177777  002154        MOV     #-1,OLDCSR
 012570                                FOR BANK := #0 TO LASTBANK
 012574  012701  060000                  MOV   #FIRST,R1                       ;SET UP VIRT ADDR POINTER
 012600  004737  047020                  CALL  EXBANK
 012604  013700  002102                  MOV   BANKINDEX,R0
 012610                                  IF ACFLAG IS TRUE
 012616                                    IF MKFLAG IS TRUE
 012624                                    IF SKIPMK IS FALSE
 012632                                      IF INTFLAG IS TRUE
 012640  012703  040000                        MOV     #40000,R3               ;SET INDEX COUNTER
 012644  012737  000001  002236                MOV     #1,SPLTCSR              ;MAP AS INTERLEAVED BANK
 012652                                      ELSE
 012654  012703  000002                        MOV     #2,R3                   ;SET INDEX COUNTER
 012660                                      END; OF IF INTFLAG
 012660  116002  002651                      MOVB    CONFIG+1(R0),R2
 012664  00630?                             ASL     R2
 012666  042702  177741                     BIC     #^C36,R2
 012672  010237  002150                     MOV     R2,CSRNO
 012676                                     IF CSRNO NE OLDCSR
 012706  013737  002150  002154               MOV     CSRNO,OLDCSR
 012714                                       IF PFLAG IS FALSE
 012722  052760  000100  002650                 BIS  #BIT6,CONFIG(R0)
 012730                                       END; OF IF PFLAG
 012730  004737  013064                       CALL    IMPTEST
```

```
 6923 012734                                                    IF INTFLAG IS TRUE
 6924 012742  116002  002651                                      MOVB  CONFIG+1(R0),R2
 6925 012746  072227  177775                                      ASH   #-3,R2
 6926 012752  042702  177741                                      BIC   #^C36,R2
 6927 012756  010237  002150                                      MOV   R2,CSRNO
 6928 012762  062701  000002                                      ADD   #2,R1            ;FIX POINTER FOR A1 ASSERTED HALF
 6929 012766  004737  013064                                      CALL  IMPTEST
 6930 012772  005037  002236                                      CLR   SPLTCSR
 6931 012776                                                    END; OF IF INTFLAG
 6932 012776                                                  END; OF IF CSRNO
 6933 012776                                                END; OF IF SKIPMK
 6934 012776                                                END; OF IF MKFLAG
 6935 012776                                              END; OF IF ACFLAG
 6936 012776                                            END; OF FOR BANK
 6937 013012                                            MAP                             ;MAP TEST SPACE TO BANK 0
 6938 013026  005037  002100                            CLR     BANK
 6939 013032                                            IF #SW0 SET.IN @SWR OR ACTFLAG IS TRUE
 694C 013050  104506                                      ENASBE                        ;TRAP ON SINGLE BIT ERRORS
 6941 013052                                            ELSE
 6942 013054  104472                                      ECCINIT                       ;TRAP ON DOUBLE BIT ERRORS (NORMAL)
 6943 013056                                            END; OF IF #SW0
 6944 013056  104423                                    CACHON                          ;TURN THE CACHE BACK ON
 6945 013060  000137  013372                            JMP     SUBAAR                  ;JUMP OVER THE SUBROUTINE
```

```
6947 013064  005004                    IMPTEST:CLR     R4
6948 013066                                    MAP BANK                        ;MAP SUPERVISOR SPACE (TEST AREA) TO BANK
6949 013102  005005                            CLR     R5
6950 013104  012737  020000  002146            MOV     #BIT13,CSR
6951 013112                                    TESTAREA                        ;ENTER TEST MODE
6952 013120                                    PUSH    (R1)                    ;SAVE TEST LOCATION
6953 013122  060301                            ADD     R3,R1                 ;INDEX TO NEXT LOCATION
6954 013124                                    PUSH    (R1)                  ;SAVE TEST LOCATION
6955 013126  104505                            CHK1DIS                       ;DISABLE ECC & WRITE CHECKBITS FOR 1 CSR
6956 013130  010411                            MOV     R4,(R1)               ;WRITE CHECKBITS (ALL ZEROS)
6957 013132  160301                            SUB     R3,R1
6958 013134  010411                            MOV     R4,(R1)
6959 013136  104503                            CLR1CSR                         ;CLEAR CSR
6960 013140  005711                            TST     (R1)                    ;READ CHECKBITS INTO REAL CSR
6961 013142  104501                            WAS1DBE                         ;WAS THERE A DOUBLE BIT ERROR
6962
6963                                           ;THIS MAKES SURE THAT SBE'S DON'T LOOK LIKE PROTECTED AREAS
6964
6965 013144                                    ON.NOERROR ;1
6966 013146  012737  020000  002146              MOV   #BIT13,CSR
6967 013154  104505                              CHK1DIS                       ;DISABLE ECC & WRITE CHECKBITS FOR 1 CSR
6968 013156  013711  002600                      MOV   ONES,(R1)
6969 013162  060301                              ADD   R3,R1
6970 013164  013711  002600                      MOV   ONES,(R1)
6971 013170  160301                              SUB   R3,R1
6972 013172  104503                              CLR1CSR                       ;CLEAR CSR
6973 013174  005711                              TST   (R1)
6974 013176  104501                              WAS1DBE                       ;WAS THERE A DOUBLE BIT ERROR
6975 013200                                      ON.NOERROR ;2
6976 013202                                        IF #BIT9 SET.IN CONFIG+2(R0) ;IS THIS A MS11-P
6977 013212  104513                                   CBREG                   ;ENABLE CHECK/SYNDROME BIT REGISTER
6978 013214  012737  023140  002146                  MOV #23140,CSR           ;WRITE DBE'S IN CSR
6979 013222                                        ELSE                        ;OR A MS11-M
6980 013224  012737  027400  002146                  MOV #27400,CSR           ;WRITE DBE'S IN CSR
6981 013232                                        END                         ;
6982 013232  104505                                CHK1DIS                     ;DISABLE ECC & WRITE CHECKBITS FOR 1 CSR
6983 013234  010411                                MOV R4,(R1)
6984 013236  060301                                ADD R3,R1                   ;ADD INDEX TO GET TO SECOND WORD
6985 013240  010411                                MOV R4,(R1)
6986 013242  160301                                SUB R3,R1                   ;SUBTRACT INDEX TO FIRST WORD
6987 013244  104503                                CLR1CSR                       ;CLEAR CSR
6988 013246  005711                                TST (R1)
6989 013250  104501                                WAS1DBE                       ;WAS THERE A DOUBLE BIT ERROR
6990 013252                                        ON.NOERROR ;3
6991 013254                                          IF #BIT9 SET.IN CONFIG+2(R0)     ;IS THIS A MS11-P
6992 013264  104513                                     CBREG                 ;ENABLE CHECK/SYNDROME BIT REGISTER
6993 013266  012737  023604  002146                    MOV #23604,CSR        ;WRITE DBE'S IN CSR
6994 013274                                          ELSE                      ;IS IT A MS11-P
6995 013276  012737  074000  002146                    MOV #74000,CSR        ;WRITE DBE'S IN CSR
6996 013304                                          END                       ;
6997 013304  104505                                  CHK1DIS                   ;DISABLE ECC & WRITE CHECKBITS FOR 1 CSR
6998 013306  010411                                  MOV       R4,(R1)
6999 013310  060301                                  ADD       R3,R1           ;INDEX TO SECOND WORD
7000 013312  010411                                  MOV       R4,(R1)
7001 013314  104503                                  CLR1CSR                     ;CLEAR CSR
7002 013316  160301                                  SUB       R3,R1           ;GO BACK TO FIRST WORD
7003 013320  005711                                  TST       (R1)
```

```
7004 013322  104501                          WAS1DBE                          ;WAS THERE A DOUBLE BIT ERROR
7005 013324                                 END ;OF ON.NOERROR ;3
7006 013324                               END ;OF ON.NOERROR ;2
7007 013324                             END ;OF ON.NOERROR ;1
7008 013324                             ON.ERROR
7009 013326  005205                       INC   R5                           ;IDENTIFY AS BAD BANK
7010 013330                             END ;OF ON.ERROR
7011 013330  104471                     ECC1DIS                            ;DISABLE ERROR CORRECTION
7012 013332  010411                     MOV   R4,(R1)                ;CLEAR OUT DOUBLE BIT ERROR!
7013 013334  060301                     ADD   R3,R1                  ;INDEX TO SECOND WORD
7014 013336  010411                     MOV   R4,(R1)                ;CLEA  OUT DOUBLE BIT ERROR!
7015 013340  104503                     CLR1CSR
7016 013342  005705                     TST   R5
7017 013344  001405                     BEQ   1$
7018 013346  050560  002650             BIS   R5,CONFIG(R0)
7019 013352  105260  002652             INCB  CONFIG+2(R0)
7020 013356  104036                     ERROR +36
7021 013360                      1$:    POP   (R1)                   ;RESTORE TEST LOCATION (2ND WORD)
7022 013362  160301                     SUB   R3,R1                  ;GO BACK TO FIRST WORD
7023 013364                             POP   (R1)                   ;RESTORE TEST LOCATION (1ST WORD)
7024 013366  104417                     KERNEL
7025 013370  000207                     RETURN
7026
7027 013372                      SUBAAR: SET   STOPOK          ;PROGRAM CAN NOW BE HALTED
```

```
7036 013400                                             SUBTST  <<LEGAL CONFIGURATION CHECK>>
                                                ;*****************************************************************************
                                                ;*SUBTEST       LEGAL CONFIGURATION CHECK
                                                ;*****************************************************************************
7037 013400  012700  000020                             MOV     #16.,R0
7038 013404  012701  002456                             MOV     #CSRINFO,R1
7039 013410  005021                             1$:     CLR     (R1)+
7040 013412  077002                                     SOB     R0,1$
7041 013414                                               FOR BANK := #0 TO LASTBANK
7042 013420  004737  047020                                 CALL        EXBANK
7043 013424  013700  002102                                 MOV BANKINDEX,R0
7044
7045 013430                                                 IF ACFLAG IS TRUE
7046 013436  116003  002651                                  MOVB       CONFIG+1(R0),R3
7047 013442  042703  177760                                  BIC        #^C17,R3
7048 013446  006303                                          ASL        R3
7049 013450  005263  002456                                  INC        CSRINFO(R3)
7050 013454                                                  IF MKFLAG IS TRUE
7051                                                         ;MAKE SURE THAT EACH BANK HAS NO MORE THAN 2 CSRS
7052 013462                                                   BEGIN   LEGALCSR
7053 013462                                                     IF INTFLAG IS TRUE
7054 013470  116003  002651                                      MOVB       CONFIG+1(R0),R3
7055 013474  010304                                              MOV R3,R4
7056 013476  042703  177760                                      BIC #^C17,R3
7057 013502  072427  177774                                      ASH #-4,R4
7058 013506  042704  177760                                      BIC #^C17,R4
7059 013512                                                      IF R3 EQ R4
7060 013516  042760  014000  002652                                BIC        #BIT11!BIT12,CONFIG+2(R0)
7061 013524  042760  170000  002650                                BIC        #170000,CONFIG(R0)
7062 013532                                                        LEAVE LEGALCSR
7063 013534                                                      END; OF IF R3
7064 013534                                                    ELSE
7065 013536                                                      LEAVE       LEGALCSR
7066 013540                                                    END; OF IF INTFLAG
7067 013540                                                  SET      CONFGERROR
7068 013546                                                  END      LEGALCSR
7069 013546                                                END ;OF IF MKFLAG
7070 013546                                              END ;OF IF ACFLAG
7071 013546                                            END; OF FOR BANK
7072 013562                                     PUSH    R5,R0                           ;SAVE CONTENTS OF R5, R0
7073 013566  005000                             CLR     R0                              ;CLEAR REGISTERS
7074 013570  005001                             CLR     R1
7075 013572  005005                             CLR     R5
7076 013574  005037  014002                     CLR     MBERR                           ;CLEAR ERROR INDICATOR
7077 013600  022761  000040  002456  2$:        CMP     #40,CSRINFO(R1)                 ;IS CURRENT CSR <= 40
7078 013606  002043                             BGE     5$                              ;BRANCH IF SO
7079 013610  022761  000010  002456             CMP     #10,CSRINFO(R1)                 ;IS CURRENT CSR < 10
7080 013616  002003                             BGE     3$                              ;BRANCH IF SO
7081 013620  004737  014124                     CALL    :LLCSR                          ;CALL ERROR ROUTINE
7082 013624  000434                             BR      5$                              ;TRY NEXT CSR
7083 013626  016005  002650             3$:     MOV     CONFIG(R0),R5                   ;MOVE LOW WORD TO R5
7084 013632  032705  000002                     BIT     #BIT1,R5                        ;DOES MEMORY EXIST HERE?
7085 013636  001415                             BEQ     4$                              ;BRANCH IF NOT
7086 013640  042705  170377                     BIC     #^C7400,R5                      ;ISOLATE CSR NUMBER IN
7087 013644  072527  177771                     ASH     #-7,R5                          ;REGISTER 5
7088 013650  020501                             CMP     R5,R1                           ;IS IT THE CURRENT CSR?
7089 013652  001007                             BNE     4$                              ;TRY NEXT WORD OF CONFIG IF NOT
```

LEGAL CONFIGURATION CHECK

```
7090 013654  032760  010000  002652          BIT     #BIT12,CONFIG+2(R0)             ;IS IT INTERLEAVED?
7091 013662  001003                          BNE     4$                              ;BRANCH IF SO
7092 013664  012737  000001  014002          MOV     #1,MBERR                        ;SET ERROR INDICATOR
7093 013672  062700  000004            4$:   ADD     #4,R0                           ;UPDATE CONFIG COUNTER
7094 013676  022700  000340                  CMP     #340,R0                         ;CONFIG TABLE ALL DONE?
7095 013702  001351                          BNE     3$                              ;BRANCH IF NOT
7096 013704  005737  014002                  TST     MBERR                           ;ERRORS FOUND?
7097 013710  001402                          BEQ     5$                              ;TRY NEXT CSR IF NOT
7098 013712  004737  014124                  CALL    ILLCSR                          ;CALL ERROR ROUTINE
7099 013716  005000                    5$:   CLR     R0                              ;REINITIALIZE CONFIG COUNTER
7100 013720  005037  014002                  CLR     MBERR                           ;CLEAR ERROR INDICATOR
7101 013724  062701  000002                  ADD     #2,R1                           ;UPDATE CSR COUNTER
7102 013730  022701  000040                  CMP     #40,R1                          ;ALL CSR'S DONE?
7103 013734  001321                          BNE     2$                              ;BRANCH IF NOT
7104 013736                                  POP     R0,R5                           ;RESTORE REGISTERS
7105 013742  005037  014002                  CLR     MBERR                           ;RESET ERROR INDICATOR
7106 013746  012700  000734                  MOV     #734,R0                 ;INDEX TO TOP OF CONFIG TABLE   ;R-C
7107 013752  032760  000002  002650    6$:   BIT     #BIT1,CONFIG(R0)        ;MEMORY PRESENT?                ;R-C
7108 013760  001003                          BNE     7$                      ;BRANCH IF SO                   ;R-C
7109 013762  162700  000004                  SUB     #4,R0                   ;TRY NEXT LOWER ENTRY IN CONFIG TABLE   ;R-C
7110 013766  000771                          BR      6$                                                      ;R-C
7111 013770  006200                    7$:   ASR     R0                                                      ;R-C
7112 013772  006200                          ASR     R0                      ;DIVIDE INDEX BY 4 TO GET BANK #;R-C
7113 013774  010037  002552                  MOV     R0,LASTBANK             ;STORE IN LASTBANK              ;R-C
7114 014000  000402                          BR      SKUJ
7115 014002  000000              MBERR:      .WORD   0                       ;SAVE SPACE FOR ERROR INDICATOR
7116 014004  000000              PHEBE:      .WORD   0                       ;SAVE SPACE FOR ODD BOUNDARY INTERLEAVED INDICATOR
7117 014006  005000              SKUJ:       CLR     R0                      ;CLEAR CONFIG COUNTER
7118 014010  005037  014004                  CLR     PHEBE                   ;CLEAR COUNTER
7119 014014  032760  000002  002650    1$:   BIT     #BIT1,CONFIG(R0)        ;IS THERE MEMORY PRESENT?
7120 014022  001431                          BEQ     3$                      ;BRANCH IF NOT
7121 014024  032760  010000  002652          BIT     #BIT12,CONFIG+2(R0)     ;IS IT INTERLEAVED?
7122 014032  001005                          BNE     2$                      ;BRANCH IF SO
7123 014034  005237  014004                  INC     PHEBE                   ;INCREMENT COUNTER
7124 014040  062700  000004                  ADD     #4,R0                   ;INCREMENT CONFIG COUNTER
7125 014044  000763                          BR      1$                      ;TRY NEXT BANK
7126 014046  023727  014004  000010    2$:   CMP     PHEBE,#10               ;IS THE COUNTER EQUAL TO...
7127 014054  001417                          BEQ     4$                      ;ONE OF THE SPECIAL VALUES.
7128 014056  023727  014004  000030          CMP     PHEBE,#30               ;IF IT IS...
7129 014064  001413                          BEQ     4$                      ;BRANCH TO 4$
7130 014066  023727  014004  000050          CMP     PHEBE,#50
7131 014074  001407                          BEQ     4$
7132 014076  023727  014004  000070          CMP     PHEBE,#70
7133 014104  001403                          BEQ     4$
7134 014106  005037  014004            3$:   CLR     PHEBE                           ;CLEAR INDICATOR
7135 014112  000403                          BR      5$
7136 014114  012737  000001  014004    4$:   MOV     #1,PHEBE                        ;SET INDICATOR
7137 014122  000421                    5$:   BR      SUBAAP                  ;BRANCH TO NEXT SUBTEST
7138 014124  010102              ILLCSR:     MOV     R1,R2                   ;R2 HAS CSR NUMBER
7139 014126  006202                          ASR     R2                      ;MAKE ACCEPTABLE FOR PRINTING
7140 014130  022702  000012                  CMP     #10.,R2
7141 014134  100002                          BPL     1$
7142 014136  062702  000007                  ADD     #7,R2
7143 014142  062702  000060            1$:   ADD     #60,R2
7144 014146  110237  100014                  MOVB    R2,MSGA122              ;PUT NUMBER INTO ERROR MESSAGE
7145 014152                                  TYPE    MSG122
7146 014156                                  SET     CONFGERROR
```

```
   7147 014164  000207                          RETURN
```

```
                                SUBAAP: SUBTST  <<PRINT CONFIGURATION DETAILS>>
                                ;*****************************************************************************
                                ;*SUBTEST       PRINT CONFIGURATION DETAILS
                                ;*****************************************************************************
                                        CLEAR   LSIZE,MSIZE,PSIZE
                                        MOV     LASTBANK,R2
                                        ASL     R2
                                        ASL     R2
                                        FOR R1 := #0 TO R2 BY #4
                                         IF CPUBIT SET.IN CONFIG(R1)
                                           IF #BIT8 SET.IN CONFIG+2(R1)
                                             IF #BIT9 SET.IN CONFIG+2(R1)
                                                 LET PSIZE := PSIZE + #1
                                             ELSE
                                                 LET MSIZE := MSIZE + #1
                                             END;IF BIT9
                                           ELSE
                                             LET LSIZE := LSIZE + #1
                                           END;IF BIT8
                                         END; OF IF CPUBIT
                                        END ;OF FOR ALL BANKS IN TABLE

                                        CLR     I
                                        FOR R1 := #0 TO #10 BY #2
                                          ASL   BSIZE(R1)
                                          ASL   BSIZE(R1)
                                          ASL   BSIZE(R1)
                                          ASL   BSIZE(R1)               ;BSIZE(R1) := BSIZE(R1) * 16.
                                          ADD   BSIZE(R1),I             ;I <- I + BSIZE(R1)
                                        END; FOR R1
                                        FOR R1 := #0 TO #200 BY #4
                                           IF CPUBIT SET.IN CONFIG(R1)
                                             LET UNITOP := UNITOP + #1
                                           END; OF IF CPUBIT
                                        END; OF FOR R1
                                        ASL     UNITOP
                                        ASL     UNITOP
                                        ASL     UNITOP
                                        ASL     UNITOP          ;UNITOP := UNITOP * 16.
                                        IF I LT UNITOP THFN LET I := UNITOP
                                        TYPE    $CRLF
                                2$:     TST     LSIZE
                                        BEQ     3$
                                        TYPDEC  LSIZE
                                        TYPE    MSG112
                                3$:     TST     MSIZE
                                        BEQ     4$
                                        TYPDEC  MSIZE
                                        TYPE    MSG113
                                4$:     TST     PSIZE
                                        BEQ     5$
                                        TYPDEC  PSIZE
                                        TYPE    MSG114
                                5$:     TYPDEC  I
                                        TYPE    MSG070
                                        IF #SW6 OFF.IN @SWR
                                          CALL  PCONFIG
```

```
   7204 014542                                  END; OF IF #SW6
```

```
7207 014542                                     SUBTST  <<CHECK APT SIZING>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       CHECK APT SIZING
                                        ;*****************************************************************************
7208 014542                                     IF APTFLAG IS TRUE AND APTSIZE IS TRUE
7209 014556  005037  002430                       CLR    TEMP
7210 014562  012700  065670                       MOV    #$MAMS1,R0
7211 014566                                       FOR R2 := #0 TO #4
7212 014570                                         IFB 1(R0) NE #0
7213 014576  111001                                   MOVB       (R0),R1
7214 014600  042701  177400                           BIC        #177400,R1
7215 014604                                           IF 2(R0) LT #0
7216 014612  000261                                     SEC
7217 014614                                           ELSE
7218 014616  000241                                     CLC
7219 014620                                           END ;OF IF 2(R0)
7220 014620  006101                                   ROL        R1
7221 014622  005201                                   INC        R1                ;TO COMPENSATE FOR 4 BANKS BEING (0-3)
7222 014624  006301                                   ASL        R1
7223 014626  006301                                   ASL        R1
7224 014630  006301                                   ASL        R1
7225 014632  006301                                   ASL        R1
7226 014634  163701  002430                           SUB        TEMP,R1
7227 014640  010137  002430                           MOV        R1,TEMP
7228 014644                                           IFB 1(R0) EQ #3
7229 014654  060137  002416                             ADD      R1,APTPAR
7230 014660                                           END ;OF IFB 1(R0)
7231 014660                                           IFB 1(R0) EQ #4
7232 014670  060137  002420                             ADD      R1,APTECC
7233 014674                                           END ;OF IFB 1(R0)
7234 014674  062700  000004                           ADD        #4,R0
7235 014700                                         END ;OF IFB 1(R0)
7236 014700                                       END ;OF FOR R2
7237 014710                                       IF APTPAR NE LSIZE OR APTECC NE MSIZE OR APTECC NE PSIZE
7238 014740  104046                                 ERROR        +46
7239 014742                                       END ;OF IF APTPAR
7240 014742                                     END ;OF IF APTFLAG
```

```
   7242 014742                          LOOP:   NEWTST  <<DIAGNOSTIC MODE DISPATCH ROUTINE>>
                                        ;*****************************************************************************
                                        ;*TEST 5         DIAGNOSTIC MODE DISPATCH ROUTINE
                                        ;*****************************************************************************
        014742  000004                  TST5:   SCOPE
   7243 014744  005037  002220                  CLR     CONTFLAG
   7244 014750  017700  165646                  MOV     @SWR,R0         ;GET SWITCHES
   7245 014754  042700  177761                  BIC     #^C16,R0        ;MASK TO ONLY MODE BITS
   7246 014760  004770  014770                  CALL    @DISPTBL(R0)    ;DISPATCH TO ROUTINE THROUGH NEXT TABLE
   7247 014764  000137  015010                  JMP     MEMDONE         ;GO TO NEXT TEST
   7248 014770  015452                  DISPTBL:BAFPAF  ;MODE 0;BANKS FORWARD, PATTERNS FORWARD
   7249 014772  015560                          BAFPAR  ;MODE 1;BANKS FORWARD, PATTERNS REVERSE
   7250 014774  015666                          BAWPAF  ;MODE 2;BANKS WORST FIRST, PATTERNS FORWARD
   7251 014776  016016                          BAWPAR  ;MODE 3;BANKS WORST FIRST, PATTERNS REVERSE
   7252 015000  016146                          PAFBAF  ;MODE 4;PATTERNS FORWARD, BANKS FORWARD
   7253 015002  016276                          PAFBAW  ;MODE 5;PATTERNS FORWARD, BANKS WORST FIRST
   7254 015004  016450                          PARBAF  ;MODE 6;PATTERNS REVERSE, BANKS FORWARD
   7255 015006  016600                          PARBAW  ;MODE 7;PATTERNS REVERSE, BANKS WORST FIRST
   7256
   7257 015010  004737  015352          MEMDONE:CALL    DOBACK                          ;CHECK BACKGROUND PATTERN
   7258
   7259 015014                                  NEWTST<<UNIQUE BANK TEST>>
                                        ;*****************************************************************************
                                        ;*TEST 6         UNIQUE BANK TEST
                                        ;*****************************************************************************
        015014  000004                  TST6:   SCOPE
   7260                                         ;MAKE SURE THAT EACH BANK CAN HAVE UNIQUE DATA
   7261                                         ;WRITE AND READ THE BANK NUMBER IN EACH BANK (EXCEPT WHERE THE PROGRAM IS)
   7262 015016                                  IF SELONLY IS FALSE
   7263 015024                                    SET   HEADER,MUT
   7264 015040  004737  024102                    CALL  MT0027
   7265 015044                                    SET   HEADER
   7266 015052  005037  002106                    CLR   MUT
   7267 015056                                  END ;OF IF SELONLY
   7268 015056  004737  015352                  CALL    DOBACK          ;RESTORE BACKROUND PATTERN
   7272
   7273 015062                          FLUSH:  SUBTST  <<FLUSH OUT DBE'S>>
                                        ;*****************************************************************************
                                        ;*SUBTEST        FLUSH OUT DBE'S
                                        ;*****************************************************************************
   7274 015062  004737  024566                  CALL    MT0030
```

END OF PASS ROUTINE

```
                                                .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
015066  005037  002436          $EOP:   CLR     FSINFLAG
015072  012700  002652                  MOV     #CONFIG+2,R0    ;MOVE 2ND WORD OF CONFIG TO R0
015076  042710  020000          1$:     BIC     #BIT13,(R0)     ;CLEAR BACKGROUND VALID BIT
015102  062700  000004                  ADD     #4,R0           ;INCREMENT TO NEXT BANK
015106  020027  003620                  CMP     R0,#3620        ;DONE?
015112  003771                          BLE     1$              ;NO - BRANCH
015114  013737  002614  002014          MOV     $ERTTL,LASTERROR
015122  005237  065646                  INC     $PASS           ;;INCREMENT THE PASS NUMBER
015126  042737  100000  065646          BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
015134                                  TYPE    MSG077          ;;TYPE 'END PASS #''
015140                                  IF #SW11 SET.IN @SWR OR QVFLAG IS TRUE OR $PASS EQ #1
015166                                    TYPE  MSG035          ;QV
015172  005037  002342                    CLR   QVFLAG
015176                                  END ;OF IF SW11
015176                                  TYPDEC  $PASS
015204  013700  000042                  MOV     42,R0           ;;GET MONITOR ADDRESS
015210  001456                          BEQ     $DOAGAIN        ;;BRANCH IF NO MONITOR
015212  022700  002000          $ZAP42: CMP     #STACK,R0       ;ARE WE UNDER RT11
015216  001453                          BEQ     $DOAGAIN        ;YES - BRANCH
                                        ;WE ARE UNDER (HEAVEN HELP US) XXDP!
015220                                  PUSH    R0
015222  004737  047664                  CALL    SHUTUP
015226                                  POP     R0
015230  000005                          RESET                   ;;CLEAR THE WORLD
015232  004710                  $ENDAD: CALL    (R0)            ;;GO TO MONITOR
015234  000240                          NOP                     ;;SAVE ROOM
015236  000240                          NOP                     ;;FOR
015240  000240                          NOP                     ;;ACT11
015242                          $DOAGN: ;UNDO SHUTUP STUFF
                                        ;       RESTORE STACK
                                        ;       ENERGIZE UNIBUS MAP & 22 BIT ADDRESSING
                                        ;       ENERGIZE MEMORY MANAGEMENT
                                        ;       PUT LOADERS BACK HOME
015242  013706  002560                  MOV     KSTACK,SP
015246  005737  002450                  TST     NO22BIT                 ;IS THIS AN 11/44 OR 11/24?
015252  001003                          BNE     1$
015254  052737  000060  172516          BIS     #BIT5!BIT4,MMR3
015262  104420                  1$:     ENERGIZE                        ;TURN ON MEMORY MANAGEMENT
015264  013700  002562                  MOV     LOADHOME,R0             ;DESTINATION BANK
015270  012701  000001                  MOV     #1,R1                   ;SOURCE BANK
015274  004737  046410                  CALL    BANKMOV
015300                                  IF APTFLAG IS TRUE
015306                                    IF $USWR EQ $PASS
015316  012701  000050          APTHANG:    MOV #50,R1
015322  077001                  2$:         SOB R0,2$
015324  062737  000001  065650            ADD #1,$DEVCT
015332  005537  065652                    ADC $UNIT
015336  077107                            SOB R1,2$
015340  005237  065646                    INC $PASS
015344  000764                            BR  APTHANG
```

END OF PASS ROUTINE

```
 7334 015346                                        END ;OF IF $USWR
 7335 015346                                      END ;OF IF APTFLAG
 7336 015346  000137  014742          $DOAGAIN: JMP   LOOP              ;RETURN
```

END OF PASS ROUTINE

```
 7339 015352                        DOBACK: SUBTST  <<WRITE BACKGROUND PATTERNS>>
                                    ;*****************************************************************************
                                    ;*SUBTEST       WRITE BACKGROUND PATTERNS
                                    ;*****************************************************************************
 7340 015352  005037  002110                CLR     PATTERN
 7341 015356                                FOR BANK := #0 TO LASTBANK
 7342 015362  004737  047020                  CALL  EXBANK
 7343 015366                                  IF ACFLAG IS TRUE AND RRFLAG IS FALSE
 7344 015402                                    SET         HEADER,MUT
 7345 015416  004737  020274                    CALL        MKTEST          ;CALL   MJTEST WOULD ALSO WORK
 7346 015422  005037  002106                    CLR         MUT
 7347 015426                                    SET         HEADER
 7348 015434                                  END ;OF IF ACFLAG
 7349 015434                                END ;OF FOR BANK
 7350 015450  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
```

```
7385 015560                             BAFPAR: SUBTST  <<BANKS FORWARD,PATTERNS REVERSE      **RECURSIVE**>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       BANKS FORWARD,PATTERNS REVERSE      **RECURSIVE**
                                        ;*****************************************************************************
7386 015560  005037  002100                        CLR     BANK            ;SET BANK TO 0
7387                                                ;START OF BANK LOOP
7388 015564  004737  047020             1$:        CALL    EXBANK          ;EXAMINE BANK
7389 015570  005737  002114                        TST     ACFLAG          ;CAN WE ACCESS THIS BANK?
7390 015574  001412                                BEQ     4$              ;NO - GO TO BANK LOOP TERMINATION
7391 015576  005737  002122                        TST     RRFLAG          ;RELOCATION REQUIRED?
7392 015602  001007                                BNE     4$              ;YES - GO TO BANK LOOP TERMINATION
7393 015604  004737  047500                        CALL    SETPAT          ;SET HIGH PATTERN FOR CORRECT MEMORY
7394                                                ;START OF PATTERN LOOP
7395 015610  004737  016752             2$:        CALL    MTEST           ;GO TEST CORRECT MEMORY
7396                                                ;TERMINATION OF PATTERN LOOP
7397 015614  005337  002110                        DEC     PATTERN         ;IS THIS THE LAST PATTERN?
7398 015620  100373                                BPL     2$              ;NO - LOOP ON THIS PATTERN
7399                                                ;TERMINATION OF BANK LOOP
7400 015622  005037  002220             4$:        CLR     CONTFLAG
7401 015626  004737  047510                        CALL    INCBNK          ;NEXT HIGHER BANK
7402 015632  002354                                BGE     1$              ;IF NOT DONE - LOOP ON THIS BANK
7403                                                ;END OF LOOPS
7404 015634  005737  002124                        TST     RLFLAG          ;HAVE WE BEEN RELOCATED?
7405 015640  001401                                BEQ     5$              ;NO - SKIP
7406 015642  000207                                RETURN                  ;YES - RETURN
7407 015644  004737  045172             5$:        CALL    RELOCATE        ;MOVE & MAP PROGRAM
7408 015650                                        ON.ERROR THEN $RETURN
7409                                                ;**NOTE** RECURSIVE CALL
7410 015654  004737  015560                        CALL    BAFPAR          ;CALL SELF
7411 015660  004737  046056                        CALL    UNRELOCATE      ;UNMOVE & UNMAP PROGRAM
7412 015664  000207                                RETURN
```

```
7415 015666                                BAWPAF: SUBLIST  <<BANKS WORST FIRST,PATTERNS FORWARD  **RECURSIVE**>>
                                           ;*****************************************************************************
                                           ;*SUBTEST       BANKS WORST FIRST,PATTERNS FORWARD  **RECURSIVE**
                                           ;*****************************************************************************
7416 015666  005037  002100                        CLR     BANK            ;SET BANK TO 0
7417                                               ;START OF BANK LOOP
7418 015672  004737  047020                1$:     CALL    EXBANK          ;EXAMINE BANK
7419 015676  005737  002114                        TST     ACFLAG          ;CAN WE ACCESS THIS BANK?
7420 015702  001415                               BEQ     4$              ;NO - GO TO BANK LOOP TERMINATION
7421 015704  005737  002126                        TST     BMFLAG          ;IS THIS BAD MEMORY (WORST FIRST)?
7422 015710  001412                               BEQ     4$              ;NO - GO TO BANK LOOP TERMINATION
7423 015712  005737  002122                        TST     RRFLAG          ;RELOCATION REQUIRED?
7424 015716  001007                               BNE     4$              ;YES - GO TO BANK LOOP TERMINATION
7425 015720  005037  002110                        CLR     PATTERN         ;SET PATTERN TO 0
7426                                               ;START OF PATTERN LOOP
7427 015724  004737  016752                2$:     CALL    MTEST           ;GO TEST CORRECT MEMORY
7428                                               ;TERMINATION OF PATTERN LOOP
7429 015730  004737  047464                        CALL    INCPAT          ;GO SEE IF THIS IS THE LAST PATTERN
7430 015734  001373                               BNE     2$              ;NO - LOOP ON THIS PATTERN
7431                                               ;TERMINATION OF BANK LOOP
7432 015736  005037  002220                4$:     CLR     CONTFLAG
7433 015742  004737  047510                        CALL    INCBNK          ;NEXT HIGHER BANK
7434 015746  002351                               BGE     1$              ;IF NOT DONE - LOOP ON THIS BANK
7435                                               ;END OF LOOPS
7436 015750  005137  002564                        COM     WORST           ;IS THIS AN EVEN NUMBERED PASS?
7437 015754  001003                               BNE     5$              ;YES - SKIP
7438                                               ;**NOTE** RECURSIVE CALL
7439 015756  004737  015666                        CALL    BAWPAF          ;CALL SELF
7440 015762  000207                               RETURN
7441 015764  005737  002124                5$:     TST     RLFLAG          ;HAVE WE BEEN RELOCATED?
7442 015770  001401                               BEQ     6$              ;NO - SKIP
7443 015772  000207                               RETURN                  ;YES - RETURN
7444 015774  004737  045172                6$:     CALL    RELOCATE        ;MOVE & MAP PROGRAM
7445 016000                                        ON.ERROR THEN $RETURN
7446                                               ;**NOTE** RECURSIVE CALL
7447 016004  004737  015666                        CALL    BAWPAF          ;CALL SELF
7448 016010  004737  046056                        CALL    UNRELOCATE      ;UNMOVE & UNMAP PROGRAM
7449 016014  000207                               RETURN
```

```
 7452 016016                            BAWPAR: SUBTST  <<BANKS WORST FIRST,PATTERNS REVERSE  **RECURSIVE**>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       BANKS WORST FIRST,PATTERNS REVERSE  **RECURSIVE**
                                        ;*****************************************************************************
 7453 016016  005037  002100                    CLR     BANK            ;SET BANK TO 0
 7454                                           ;START OF BANK LOOP
 7455 016022  004737  047020            1$:     CALL    EXBANK          ;EXAMINE BANK
 7456 016026  005737  002114                    TST     ACFLAG          ;CAN WE ACCESS THIS BANK?
 7457 016032  001415                            BEQ     4$              ;NO - GO TO BANK LOOP TERMINATION
 7458 016034  005737  002126                    TST     BMFLAG          ;IS THIS BAD MEMORY (WORST FIRST)
 7459 016040  001412                            BEQ     4$              ;NO - GO TO BANK LOOP TERMINATION
 7460 016042  005737  002122                    TST     RRFLAG          ;RELOCATION REQUIRED?
 7461 016046  001007                            BNE     4$              ;YES - GO TO BANK LOOP TERMINATION
 7462 016050  004737  047500                    CALL    SETPAT          ;SET HIGH PATTERN FOR CORRECT MEMORY
 7463                                           ;START OF PATTERN LOOP
 7464 016054  004737  016752            2$:     CALL    MTEST           ;GO TEST CORRECT MEMORY
 7465                                           ;TERMINATION OF PATTERN LOOP
 7466 016060  005337  002110                    DEC     PATTERN         ;IS THIS THE LAST PATTERN?
 7467 016064  100373                            BPL     2$              ;NO - LOOP ON THIS PATTERN
 7468                                           ;TERMINATION OF BANK LOOP
 7469 016066  005037  002220            4$:     CLR     CONTFLAG
 7470 016072  004737  047510                    CALL    INCBNK          ;NEXT HIGHER BANK
 7471 016076  002351                            BGE     1$              ;IF NOT DONE - LOOP ON THIS BANK
 7472                                           ;END OF LOOPS
 7473 016100  005137  002564                    COM     WORST           ;IS THIS AN EVEN NUMBERED PASS?
 7474 016104  001003                            BNE     5$              ;YES - SKIP
 7475                                           ;**NOTE** RECURSIVE CALL
 7476 016106  004737  016016                    CALL    BAWPAR          ;CALL SELF
 7477 016112  000207                            RETURN
 7478 016114  005737  002124            5$:     TST     RLFLAG          ;HAVE WE BEEN RELOCATED?
 7479 016120  001401                            BEQ     6$              ;NO - SKIP
 7480 016122  000207                            RETURN                  ;YES - RETURN
 7481 016124  004737  045172            6$:     CALL    RELOCATE        ;MOVE & MAP PROGRAM
 7482 016130                                    ON.ERROR THEN $RETURN
 7483                                           ;**NOTE** RECURSIVE CALL
 7484 016134  004737  016016                    CALL    BAWPAR          ;CALL SELF
 7485 016140  004737  046056                    CALL    UNRELOCATE      ;UNMOVE & UNMAP PROGRAM
 7486 016144  000207                            RETURN
```

```
 7489 016146                            PAFBAF: SUBTST  <<PATTERNS FORWARD,BANKS FORWARD      **RECURSIVE**>>
                                        ;*************************************************************************
                                        ;*SUBTEST       PATTERNS FORWARD,BANKS FORWARD      **RECURSIVE**
                                        ;*************************************************************************
                                        ;
 7490 016146  005037  002110                    CLR     PATTERN          ;SET PATTERN TO 0
 7491                                           ;START OF PATTERN LOOP
 7492 016152  005037  002100            1$:     CLR     BANK             ;SET BANK TO 0
 7493                                           ;START OF BANK LOOP
 7494 016156  004737  047020            2$:     CALL    EXBANK           ;EXAMINE BANK
 7495 016162  004737  047446                    CALL    BANKOK           ;CORRECT MEMORY FOR THIS BANK?
 7496 016166  001010                            BNE     4$               ;NO - GO TO BANK LOOP TERMINATOR
 7497 016170  005737  002114                    TST     ACFLAG           ;CAN WE ACCESS THIS BANK?
 7498 016174  001405                            BEQ     4$               ;NO - GO TO BANK LOOP TERMINATION
 7499 016176  005737  002122                    TST     RRFLAG           ;RELOCATION REQUIRED?
 7500 016202  001002                            BNE     4$               ;YES - GO TO BANK LOOP TERMINATION
 7501 016204  004737  016752                    CALL    MTEST            ;GO TEST CORRECT MEMORY
 7502                                           ;TERMINATION OF BANK LOOP
 7503 016210  005037  002220            4$:     CLR     CONTFLAG
 7504 016214  004737  047510                    CALL    INCBNK           ;NEXT HIGHER BANK
 7505 016220  002356                            BGE     2$               ;IF NOT DONE - LOOP ON THIS BANK
 7506                                           ;TERMINATION OF PATTERN LOOP
 7507 016222  004737  047464                    CALL    INCRPT           ;NEXT HIGHER PATTERN
 7508 016226  001351                            BNE     1$               ;OK - LOOP; ELSE CONTINUE
 7509                                           ;END OF LOOPS
 7510 016230  005137  002132                    COM     TMFLAG           ;COMPLEMENT TYPE OF MEMORY
 7511                                                                    ;IS THIS AN EVEN NUMBER PASS?
 7512 016234  001403                            BEQ     5$               ;YES - SKIP
 7513                                           ;**NOTE** RECURSIVE CALL
 7514 016236  004737  016146                    CALL    PAFBAF           ;CALL SELF
 7515 016242  000207                            RETURN
 7516 016244  005737  002124            5$:     TST     RLFLAG           ;HAVE WE BEEN RELOCATED?
 7517 016250  001401                            BEQ     6$               ;NO - SKIP
 7518 016252  000207                            RETURN                   ;YES - RETURN
 7519 016254  004737  045172            6$:     CALL    RELOCATE         ;MOVE & MAP PROGRAM
 7520 016260                                    ON.ERROR THEN $RETURN
 7521                                           ;**NOTE** RECURSIVE CALL
 7522 016264  004737  016146                    CALL    PAFBAF           ;CALL SELF
 7523 016270  004737  046056                    CALL    UNRELOCATE       ;UNMOVE & UNMAP PROGRAM
 7524 016274  000207                            RETURN
```

```
 7527 016276                            PAFBAW: SUBTST  <<PATTERNS FORWARD,BANKS WORST FIRST  **RECURSIVE**>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       PATTERNS FORWARD,BANKS WORST FIRST  **RECURSIVE**
                                        ;*****************************************************************************
 7528 016276  005037  002110                    CLR     PATTERN         ;SET PATTERN TO 0
 7529                                           ;START OF PATTERN LOOP
 7530 016302  005037  002100            1$:     CLR     BANK            ;SET BANK TO 0
 7531                                           ;START OF BANK LOOP
 7532 016306  004737  047020            2$:     CALL    EXBANK          ;EXAMINE BANK
 7533 016312  004737  047446                    CALL    BANKOK          ;CORRECT MEMORY FOR THIS BANK?
 7534 016316  001013                            BNE     4$              ;NO - GO TO BANK LOOP TERMINATOR
 7535 016320  005737  002114                    TST     ACFLAG          ;CAN WE ACCESS THIS BANK?
 7536 016324  001410                            BEQ     4$              ;NO - GO TO BANK LOOP TERMINATION
 7537 016326  005737  002126                    TST     BMFLAG          ;IS THIS BAD MEMORY (WORST FIRST)
 7538 016332  001405                            BEQ     4$              ;NO - GO TO BANK LOOP TERMINATION
 7539 016334  005737  002122                    TST     RRFLAG          ;RELOCATION REQUIRED?
 7540 016340  001002                            BNE     4$              ;YES - GO TO BANK LOOP TERMINATION
 7541 016342  004737  016752                    CALL    MTEST           ;GO TEST CORRECT MEMORY
 7542                                           ;TERMINATION OF BANK LOOP
 7543 016346  005037  002220            4$:     CLR     CONTFLAG
 7544 016352  004737  047510                    CALL    INCBNK          ;NEXT HIGHER BANK
 7545 016356  002353                            BGE     2$              ;IF NOT DONE - LOOP ON THIS BANK
 7546                                           ;TERMINATION OF PATTERN LOOP
 7547 016360  004737  047464                    CALL    INCRPT          ;NEXT HIGHER PATTERN
 7548 016364  001346                            BNE     1$              ;OK - LOOP; ELSE CONTINUE
 7549                                           ;END OF LOOPS
 7550 016366  005137  002132                    COM     TMFLAG          ;COMPLEMENT TYPE OF MEMORY
 7551                                                                   ;IS THIS AN EVEN NUMBER PASS?
 7552 016372  001403                            BEQ     5$              ;YES - SKIP
 7553                                           ;**NOTE** RECURSIVE CALL
 7554 016374  004737  016276                    CALL    PAFBAW          ;CALL SELF
 7555 016400  000207                            RETURN
 7556 016402  005137  002564            5$:     COM     WORST           ;4TH PASS?
 7557 016406  001003                            BNE     6$              ;YES - SKIP
 7558                                           ;**NOTE** RECURSIVE CALL
 7559 016410  004737  016276                    CALL    PAFBAW          ;CALL SELF
 7560 016414  000207                            RETURN
 7561 016416  005737  002124            6$:     TST     RLFLAG          ;HAVE WE BEEN RELOCATED?
 7562 016422  001401                            BEQ     7$              ;NO - SKIP
 7563 016424  000207                            RETURN                  ;YES - RETURN
 7564 016426  004737  045172            7$:     CALL    RELOCATE        ;MOVE & MAP PROGRAM
 7565 016432                                    ON.ERROR THEN $RETURN
 7566                                           ;**NOTE** RECURSIVE CALL
 7567 016436  004737  016276                    CALL    PAFBAW          ;CALL SELF
 7568 016442  004737  046056                    CALL    UNRELOCATE      ;UNMOVE & UNMAP PROGRAM
 7569 016446  000207                            RETURN
```

```
7572 016450                             PARBAF: SUBTST  <<PATTERNS REVERSE,BANKS FORWARD      **RECURSIVE**>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       PATTERNS REVERSE,BANKS FORWARD      **RECURSIVE**
                                        ;*****************************************************************************
7573 016450  004737  047500                        CALL    HIPAT           ;SET HIGHTEST PATTERNS
7574                                                ;START OF PATTERN LOOP
7575 016454  005037  002100            1$:         CLR     BANK            ;SET BANK TO 0
7576                                                ;START OF BANK LOOP
7577 016460  004737  047020            2$:         CALL    EXBANK          ;EXAMINE BANK
7578 016464  004737  047446                        CALL    BANKOK          ;CORRECT MEMORY FOR THIS BANK?
7579 016470  001010                                BNE     4$              ;NO - GO TO BANK LOOP TERMINATOR
7580 016472  005737  002114                        TST     ACFLAG          ;CAN WE ACCESS THIS BANK?
7581 016476  001405                                BEQ     4$              ;NO - GO TO BANK LOOP TERMINATION
7582 016500  005737  002122                        TST     RRFLAG          ;RELOCATION REQUIRED?
7583 016504  001002                                BNE     4$              ;YES - GO TO BANK LOOP TERMINATION
7584 016506  004737  016752                        CALL    MTEST           ;GO TEST CORRECT MEMORY
7585                                                ;TERMINATION OF BANK LOOP
7586 016512  005037  002220            4$:         CLR     CONTFLAG
7587 016516  004737  047510                        CALL    INCBNK          ;NEXT HIGHER BANK
7588 016522  002356                                BGE     2$              ;IF NOT DONE - LOOP ON THIS BANK
7589                                                ;TERMINATION OF PATTERN LOOP
7590 016524  005337  002110                        DEC     PATTERN         ;NEXT LOWER PATTERN
7591 016530  100351                                BPL     1$              ;OK - LOOP; ELSE CONTINUE
7592                                                ;END OF LOOPS
7593 016532  005137  002132                        COM     TMFLAG          ;COMPLEMENT TYPE OF MEMORY
7594                                                                        ;IS THIS AN EVEN NUMBER PASS?
7595 016536  001403                                BEQ     5$              ;YES - SKIP
7596                                                ;**NOTE** RECURSIVE CALL
7597 016540  004737  016450                        CALL    PARBAF          ;CALL SELF
7598 016544  000207                                RETURN
7599 016546  005737  002124            5$:         TST     RLFLAG          ;HAVE WE BEEN RELOCATED?
7600 016552  001401                                BEQ     6$              ;NO - SKIP
7601 016554  000207                                RETURN                  ;YES - RETURN
7602 016556  004737  045172            6$:         CALL    RELOCATE        ;MOVE & MAP PROGRAM
7603 016562                                        ON.ERROR THEN $RETURN
7604                                                ;**NOTE** RECURSIVE CALL
7605 016566  004737  016450                        CALL    PARBAF          ;CALL SELF
7606 016572  004737  046056                        CALL    UNRELOCATE      ;UNMOVE & UNMAP PROGRAM
7607 016576  000207                                RETURN
```

```
7610 016600                             PARBAW: SUBTST  <<PATTERNS REVERSE,BANKS WORST FIRST  **RECURSIVE**>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       PATTERNS REVERSE,BANKS WORST FIRST  **RECURSIVE**
                                        ;*****************************************************************************
7611 016600  004737  047500                        CALL    HIPAT           ;SET HIGHTEST PATTERN
7612                                               ;START OF PATTERN LOOP
7613 016604  005037  002100            1$:         CLR     BANK            ;SET BANK TO 0
7614                                               ;START OF BANK LOOP
7615 016610  004737  047020            2$:         CALL    EXBANK          ;EXAMINE BANK
7616 016614  004737  047446                        CALL    BANKOK          ;CORRECT MEMORY FOR THIS BANK?
7617 016620  001013                               BNE     4$              ;NO - GO TO BANK LOOP TERMINATOR
7618 016622  005737  002114                        TST     ACFLAG          ;CAN WE ACCESS THIS BANK?
7619 016626  001410                               BEQ     4$              ;NO - GO TO BANK LOOP TERMINATION
7620 016630  005737  002126                        TST     BMFLAG          ;IS THIS BAD MEMORY (WORST FIRST)
7621 016634  001405                               BEQ     4$              ;NO - GO TO BANK LOOP TERMINATION
7622 016636  005737  002122                        TST     RRFLAG          ;RELOCATION REQUIRED?
7623 016642  001002                               BNE     4$              ;YES - GO TO BANK LOOP TERMINATION
7624 016644  004737  016752                        CALL    MTEST           ;GO TEST CORRECT MEMORY
7625                                               ;TERMINATION OF BANK LOOP
7626 016650  005037  002220            4$:         CLR CONTFLAG
7627 016654  004737  047510                        CALL    INCBNK          ;NEXT HIGHER BANK
7628 016660  002353                               BGE     2$              ;IF NOT DONE - LOOP ON THIS BANK
7629                                               ;TERMINATION OF PATTERN LOOP
7630 016662  005337  002110                        DEC     PATTERN         ;NEXT LOWER PATTERN
7631 016666  100346                               BPL     1$              ;OK - LOOP; ELSE CONTINUE
7632                                               ;END OF LOOPS
7633 016670  005137  002132                        COM     TMFLAG          ;COMPLEMENT TYPE OF MEMORY
7634                                                                       ;IS THIS AN EVEN NUMBER PASS?
7635 016674  001403                               BEQ     5$              ;YES - SKIP
7636                                               ;**NOTE** RECURSIVE CALL
7637 016676  004737  016600                        CALL    PARBAW          ;CALL SELF
7638 016702  000207                               RETURN
7639 016704  005137  002564            5$:         COM     WORST           ;4TH PASS?
7640 016710  001003                               BNE     6$              ;YES - SKIP
7641                                               ;**NOTE** RECURSIVE CALL
7642 016712  004737  016600                        CALL    PARBAW          ;CALL SELF
7643 016716  000207                               RETURN
7644 016720  005737  002124            6$:         TST     RLFLAG          ;HAVE WE BEEN RELOCATED?
7645 016724  001401                               BEQ     7$              ;NO - SKIP
7646 016726  000207                               RETURN                  ;YES - RETURN
7647 016730  004737  045172            7$:         CALL    RELOCATE        ;MOVE & MAP PROGRAM
7648 016734                                        ON.ERROR THEN $RETURN
7649                                               ;**NOTE** RECURSIVE CALL
7650 016740  004737  016600                        CALL    PARBAW          ;CALL SELF
7651 016744  004737  046056                        CALL    UNRELOCATE      ;UNMOVE & UNMAP PROGRAM
7652 016750  000207                               RETURN
```

```
7655 016752                                 MTEST:  SUBTST  <<SUBR  SETUP MEMORY TEST>>
                                            ;*****************************************************************************
                                            ;*SUBTEST       SUBR    SETUP MEMORY TEST
                                            ;*****************************************************************************
7656 016752                                         SET     HEADER                  ;INITIALIZE HEADER MESSAGE TYPEOUT
7657 016760                                         SET     MUT                     ;INDICATE THERE IS A MEMORY UNDER TEST
7658 016766  005037  002262                         CLR     PASFLG
7659 016772  005737  002116                         TST     MKFLAG                  ;ECC?
7660 016776  001413                                 BEQ     MT1                     ;NO - SKIP
7661 017000                                         BEGIN   HOLDLOOP
7662 017000                                           IF CONTFLAG IS TRUE THEN LEAVE HOLDLOOP
7663 017006                                           IF SKIPMK IS FALSE
7664 017014  004737  017046                             CALL        MKCONTROL
7665 017020                                           END; OF IF SKIPMK
7666 017020                                         END     HOLDLOOP
7667 017020  004737  020274                         CALL    MKTEST                  ;YES - DO ECC TESTS
7668 017024  000402                                 BR      MT2
7669 017026  004737  020514            MT1:         CALL    MJTEST                  ;DO PARITY TESTS
7670 017032  005037  002106            MT2:         CLR     MUT                     ;NOW - NO MEMORY UNDER TEST
7671 017036                                         SET     HEADER                  ;ALLOW HEADERS NORMAL
7672 017044  000207                                 RETURN
```

```
 7675 017046                                MKCONTROL:SUBTST        <<SUBR  TEST ECC CSR LOGIC DISPATCH>>
                                            ;*****************************************************************************
                                            ;*SUBTEST       SUBR    TEST ECC CSR LOGIC DISPATCH
                                            ;*****************************************************************************
 7676                                               ;THE NEXT TWO MODULES SOLVE THE PROBLEM OF
 7677                                               ;HOW TO RUN THE CSR TESTS ON EACH ECC MEMORY
 7678                                               ;
 7679 017046                                        IF SELONLY IS TRUE THEN $RETURN
 7680 017056                                        IF INHECC IS TRUE THEN $RETURN
 7681 017066                                        PUSH    BANK,R0,R1,R2,R3
 7682 017102  012737  060000  002230                MOV     #FIRST,CSRFBANK         ;SET FIRST TEST ADDRESS TO FIRST ADDR.
 7683 017110  012737  157776  002232                MOV     #LAST,CSRLBANK
 7684 017116  005037  002234                        CLR     CSRINT
 7685 017122  005037  002236                        CLR     SPLTCSR
 7686 017126  005037  002326                        CLR     CSRLOOP                 ; AND ZERO THE LOOP COUNTER
 7687 017132  013700  002102                        MOV     BANKINDEX,R0            ;GET THE BANK INDEX
 7688 017136  016001  002650                        MOV     CONFIG(R0),R1           ;GET CSR NUMBER
 7689 017142  000301                                SWAB    R1
 7690 017144  042701  177760                        BIC     #^C17,R1
 7691 017150  006301                                ASL     R1
 7692 017152  010137  002522                        MOV     R1,CSRHOLD              ;STORE IN THE LOW BYTE
 7693 017156  005737  002134                        TST     INTFLAG                 ;IS THIS BANK INTERLEAVED?
 7694 017162  001421                                BEQ     1$                      ;BRANCH IF NOT INTERLEAVED
 7695 017164  005237  002236                        INC     SPLTCSR
 7696 017170  012737  120000  002232                MOV     #120000,CSRLBANK
 7697 017176  005237  002326                        INC     CSRLOOP                 ;WE MUST LOOP TWICE FOR AN INTERLEAVED BANK
 7698 017202  005237  002234                        INC     CSRINT
 7699 017206  016001  002650                        MOV     CONFIG(R0),R1           ;GET THE INTERLEAVE CSR NUMBER
 7700 017212  072127  177775                        ASH     #-3,R1
 7701 017216  042701  160777                        BIC     #^C17000,R1
 7702 017222  050137  002522                        BIS     R1,CSRHOLD              ;STORE IT IN CSRHOLD'S UPPER BYTE
 7703 017226  005003                        1$:     CLR     R3
 7704 017230  116337  002522  002150    MKLOOP:     MOVB    CSRHOLD(R3),CSRNO
 7705 017236  042737  177741  002150                BIC     #^C36,CSRNO             :CLEAR ANY UNNECESSARY BITS
 7706 017244                                          FOR MKCNT := #0 TO CSRINT
 7707 017250                                            FOR CSRFIRST := CSRFBANK TO CSRLBANK BY #4000
 7708 017256                                            MAP BANK                    ;MAP TEST SPACE TO BANK
 7709 017272  104511                                    INVALIDATE                  ;INVALIDATE BACKROUND PATTERN
 7710 017274                                              BEGIN CSRSTUFF
 7711 017274  005037  002330                              CLR       SUCCESS
 7712 017300                                                IF ACFLAG IS TRUE AND RRFLAG IS FALSE
 7713 017314  013737  002224  002226                          MOV   CSRFIRST,CSRLAST
 7714 017322  062737  004000  002226                          ADD   #4000,CSRLAST
 7715 017330                                                    FOR TESTADD := CSRFIRST TO CSRLAST BY #4
 7716 017336  013737  002406  002410                              MOV       TESTADD,TESTADD+2
 7717 017344  005737  002236                                    TST       SPLTCSR
 7718 017350  001404                                            BEQ       1$
 7719 017352  062737  040000  002410                          ADD       #40000,TESTADD+2
 7720 017360  000403                                          BR        2$
 7721 017362  062737  000002  002410                    1$:   ADD       #2,TESTADD+2
 7722 017370  004737  017670                              2$:   CALL      SBETEST
 7723 017374                                                  ON.NOERROR
 7724 017376  104424                                           CACHOFF                  ;TURN CACHE OFF
 7725 017400  005037  002074                                   CLR     NOPAR            ;INDICATE PARITY ACTION
 7726 017404                                                    FOR I := #0 TO #27
 7727 017410                                                      SET   HEADER
 7728 017416  005037  002262                                      CLR   PASFLG
```

```
7729 017422                                                     LET R0 := I
7730 017426                                                     PUSH  R3                ;SAVE LOOP COUNTER
7731 017430  010637  002144                                     MOV   SP,CTLKVEC        ;SAVE VECTOR IN CSR OF ^K
7732 017434   62737  000002  002144                            SUB   #2,CTLKVEC
7733 017442  004737  020174                                     CALL CSRCASE
7734 017446                                                     POP   R3                ;RESTORE LOOP COUNTER
7735 017450                                                   END ;OF FOR I
7736 017464  104423                                           CACHON                  ;TURN CACHE ON
7737 017466                                                   SET     SUCCESS
7738 017474                                                   LEAVE CSRSTUFF
7739 017476                                                 END ;OF ON.NOERROR
7740 017476                                               END ;OF FOR TESTADD
7741 017514                                             END ;OF IF
7742 017514                                           END CSRSTUFF
7743 017514                                           IF SUCCESS IS FALSE
7744 017522                                             TYPE    MSGA34
7745 017526                                             TYPOCS  BANK,<TYPES BANK NUMBER>,3
7746 017536                                             TYPE    MSGB34
7747 017542  004737  057476                               CALL    PERBNK
7748 017546                                           END ;OF IF SUCCESS
7749 017546                                         END; OF FOR CSRFIRST
7750 017564  005237  002236                         INC  SPLTCSR
7751 017570                                       END; OF FOR MKCNT
7752 017604  062737  000002  002230           ADD     #2,CSRFBANK
7753 017612  012737  000001  002236           MOV     #1,SPLTCSR
7754 017620  005203                           INC     R3
7755 017622  020337  002326                   CMP     R3,CSRLOOP
7756 017626  003002                           BGT     1$
7757 017630  000137  017230                   JMP     MKLOOP
7758 017634  104472                 1$:      ECCINIT         ;TRAP ON DOUBLE BIT ERRORS (NORMAL)
7759 017636                                   SET     CONTFLAG
7760 017644  005037  002236                   CLR     SPLTCSR
7761 017650                                   POP     R3,R2,R1,R0,BANK
7762 017664  000207                          RETURN
7763 017666  000000                 MKCNT:   .WORD   0                       ;COUNTER FOR MKLOOP
```

```
                                SBETEST:SUBTST  <<CHECK FOR SBE FREE LOCATIONS>>
                                ;*****************************************************************************
                                ;*SUBTEST       CHECK FOR SBE FREE LOCATIONS
                                ;*****************************************************************************
                                        ;IN ORDER TO DETERMINE IF A LOCATION IS SBE FREE I DO THIS
                                        ;
                                        ;WRITE ZEROS WITH ECC DISABLE
                                        ;READ ZEROS BACK
                                        ;IF NOT ZEROS THEN RETURN ERROR
                                        ;
                                        ;WRITE ZEROS WITH ECC ENABLED BUT TRAPS DISABLED
                                        ;READ ZEROS BACK
                                        ;IF NOT ZEROS THEN RETURN ERROR
                                        ;
                                        ;TEST THE LOCATION FROM THE PAR'S (WITH NO PROGRAM FETCHES)
                                        ;IF THERE WERE ANY SBE'S OR DBE'S THEN RETURN ERROR
                                        ;
                                        ;COMPLIMENT ZEROS TO ONES WITH ECC DISABLE
                                        ;READ ONES BACK
                                        ;IF NOT ONES THEN RETURN ERROR
                                        ;
                                        ;WRITE 100,,100000,00000 (CHECKBITS COMPLIMENT OF BEFORE)
                                        ;  WITH ECC ENABLED BUT TRAPS DISABLED
                                        ;TEST THE LOCATION FROM THE PAR'S (WITH NO PROGRAM FETCHES)
                                        ;IF THERE WERE ANY SBE'S OR DBE'S THEN RETURN ERROR
                                        ;
                                        ;IF NONE OF THE ABOVE FORCES A RETURN ERROR THEN RETURN NO.ERROR
                                        .ENABL  LSB
017670                                  PUSH    R0,R1,R4                ;PUSH R0,R1,R4 ONTO STACK
017676  013701  002406                  MOV     TESTADD,R1
017702  013704  002410                  MOV     TESTADD+2,R4
017706                                  TESTAREA                        ;ENTER TEST MODE
017714  104424                          CACHOFF                         ;TURN CACHE OFF
017716  104471                          ECC1DIS                         ;DISABLE ECC ON 1 SELECTED CSR
017720                                  CLEAR   (R1),(R4)
017724  005711                          TST     (R1)
017726  001107                          BNE     SBENT
017730  005714                          TST     (R4)
017732  001105                          BNE     SBENT
                                        
017734  104503                          CLR1CSR                         ;CLEAR 1 SELECTED CSR
017736                                  CLEAR   (R1),(R4)
017742  005711                          TST     (R1)
017744  001100                          BNE     SBENT
017746  005714                          TST     (R4)
017750  001076                          BNE     SBENT
                                        
017752  104510                          TSTREAD                 ;TEST LOC (R1) & TST FOR SBE (WITHOUT FETCHES)
017754                                  IF #BIT15!BIT4 SET.IN CSR
017764                                    SET SKPERR                    ;DISABLE ERRGEN'S ERROR PRINTOUT
017772  104512                            ERRGEN
017774  013700  002454                      OV    ERRADD,R0
020000  072027  177774                    ASH   #-4,R0
020004  042700  177600                    BIC   #^C177,R0
020010                                    IF BANK EQ R0 THEN GOTO SBENT
020016                                  END; OF IF #BIT15
020016  104471                          ECC1DIS                         ;DISABLE ECC ON 1 SELECTED CSR
```

CHECK FOR SBE FREE LOCATIONS

```
7820 020020  005111                          COM     (R1)
7821 020022  005114                          COM     (R4)
7822 020024  023711  002600                  CMP     ONES,(R1)
7823 020030  001046                          BNE     SBENT
7824 020032  023714  002600                  CMP     ONES,(R4)
7825 020036  001043                          BNE     SBENT
7826
7827 020040  104503                          CLR1CSR                         ;CLEAR 1 SELECTED CSR
7828 020042  005011                          CLR     (R1)
7829 020044  012714  100000                  MOV     #BIT15,(R4)
7830 020050  005711                          TST     (R1)
7831 020052  001035                          BNE     SBENT
7832 020054  022714  100000                  CMP     #BIT15,(R4)
7833 020060  001032                          BNE     SBENT
7834
7835 020062  104510                          TSTREAD                 ;TEST LOC (R1) & TST FOR SBE (WITHOUT FETCHES)
7836 020064                                  IF #BIT15!BIT4 SET.IN CSR
7837 020074                                    SET SKPERR                    ;DISABLE ERRGEN'S ERROR PRINTOUT
7838 020102  104512                            ERRGEN
7839 020104  013700  002454                    MOV   ERRADD,R0
7840 020110  072027  177774                    ASH   #-4,R0
7841 020114  042700  177600                    BIC   #^C177,R0
7842 020120                                    IF BANK EQ R0 THEN GOTO SBENT
7843 020126                                  END; OF IF #BIT15
7844
7845 020126  104417                          KERNEL                          ;ENTER KERNEL MODE
7846 020130  104473                          ECC1INIT                        ;INITIALIZE 1 SELECTED CSR
7847 020132  104423                          CACHON                          ;TURN CACHE ON
7848 020134                                  POP     R4,R1,R0                ;POP R0,R1 & R4 FROM STACK
7849 020142                                  $RETURN NOERROR
7850
7851 020146  104503          SBENT:          CLR1CSR                         ;CLEAR 1 SELECTED CSR
7852 020150                                  CLEAR   (R1),(R4)
7853 020154  104417                          KERNEL                          ;ENTER KERNEL MODE
7854 020156  104473                          ECC1INIT                        ;INITIALIZE 1 SELECTED CSR
7855 020160  104423                          CACHON                          ;TURN CACHE ON
7856 020162                                  POP     R4,R1,R0                ;POP R0,R1 & R4 FROM STACK
7857 020170                                  $RETURN ERROR
7858                                         .DSABL  LSB
```

CHECK FOR SBE FREE LOCATIONS

```
                                CSRCASE:SUBTST  <<CSR PATTERN CASE STATEMENT>>
                                ;*****************************************************************************
                                ;*SUBTEST       CSR PATTERN CASE STATEMENT
                                ;*****************************************************************************
                                        CASE R0
                                          ;WARNING IF YOU CHANGE THIS TABLE ALSO
                                          ;CHANGE "$DDW0" - "$DDW5" (THE PATTERN BIT MAP)

                                          ;PAT          TIME            DESCRIPTION
7867 020204  021750             MKCSRT:   MT0006        ;<1 SEC         INITIAL DATA TEST

                                          ;
                                          ;             MS11-P ECC TESTS
                                          ;
                                          MT0044        ; 1 SEC         SHIFTING 1/0'S THROUGH CHECK BITS
7873 020210  022350                       MT0014        ; 1 SEC         BASIC DOUBLE ERROR TEST
7874 020212  026614                       MT0045        ;<1 SEC         SYNDROMES TO CSR ON DOUBLE BIT ERROR TEST
7875 020214  026264                       MT0036        ; 1 SEC         CORRECTION CODE TEST
7876 020216  022616                       MT0020        ; 1 SEC         SYNDROMES TO CSR ON SINGLE BIT ERROR TEST
7877 020220  026336                       MT0037        ;<1 SEC         ECC DISABLE TEST
7878 020222  026406                       MT0041        ; 1 SEC         ADDRESS TO CSR ON DOUBLE BIT ERROR
7879 020224  026450                       MT0042        ;<1 SEC         EXTENDED ADDRESS TO CSR ON ERROR TEST
7880 020226  026514                       MT0043        ;<1 SEC         WRITE BYTE CLEARS SBE TEST
7881 020230  026654                       MT0046        ; 1 SEC         CHECK SINGLE BIT ERRORS WITH ECC DISABLED TEST
7882 020232  026714                       MT0047        ;<1 SEC         NO CSR UPDATE ON SBE WITH EXSISTING DBE

7884 020234  022046                       MT0010        ;<1 SEC         BYTE ADDRESSING TEST

                                          ;
                                          ;             MS11-M ECC TESTS
                                          ;
7889 020236  023522                       MT0025        ;<1 SEC         INTERRUPT ENABLE TEST
7890 020240  022102                       MT0011        ;<2 SEC         CREATE SINGLE BIT ERROR TEST
7891 020242  022160                       MT0012        ;<1 SEC         WRITE BYTE CLEARS SBE TEST
7892 020244  022264                       MT0013        ; 1 SEC         CREATE DOUBLE BIT ERROR TEST
7893 020246  022440                       MT0015        ; 1 SEC         WRITE INHIBIT OF BYTE WITH DBE
7894 020250  022516                       MT0016        ;<1 SEC         WRITE INHIBIT OF WORD WITH DBE


7898 020252  026760                       MT0999        ; 0 SEC         NULL TEST
7899 020254  026760                       MT0999        ; 0 SEC         NULL TEST
7900 020256  026760                       MT0999        ; 0 SEC         NULL TEST
7901 020260  026760                       MT0999        ; 0 SEC         NULL TEST
7902 020262  026760                       MT0999        ; 0 SEC         NULL TEST
7903 020264                     END ;OF CASE R0
7904 020272  000207             RETURN
```

```
7907 020274                                MKTEST: SUBTST  <<SUBR  ECC TEST DISPATCH>>
                                           ;*****************************************************************************
                                           ;*SUBTEST       SUBR    ECC TEST DISPATCH
                                           ;*****************************************************************************
7908 020274                                        IF #SW0 SET.IN @SWR OR ACTFLAG IS TRUE
7909 020312  104470                                  ECCDIS                          ;DISABLE ERROR CORRECTION
7910 020314                                        ELSE
7911 020316  104502                                  CLRCSR                          ;CLEAR ALL CSR'S
7912 020320                                        END ;OF IF
7913 020320  012737  000002  002074                MOV     #2,NOPAR                ;INDICATE PARITY ACTION
7914 020326  012737  000002  002322                MOV     #2,PCBUMP               ;TRAPS ADD 2 TO PC
7915 020334  013700  002110                        MOV     PATTERN,R0              ;GET PATTERN NUMBER
7916 020340  006300                                ASL     R0                      ;MAKE IT A WORD ADDRESS
7917 020342                                        IF MKPAT(R0) NE #MT0034 AND MKPAT(R0) NE #MT0999
7918 020362  104511                                  INVALIDATE                    ;INVALIDATE BACKGROUND PATTERN ON 'BANK"
7919 020364                                        END ;OF IF MKPAT(R0)
7920 020364  010637  002144                        MOV     SP,CTLKVEC              ;SAVE VECTOR IN CASE OF ^K
7921 020370  162737  000002  002144                SUB     #2,CTLKVEC
7922 020376  004770  020434                        CALL    @MKPAT(R0)              ;INDEX OFF TABLE
7923 020402                                        IF #SW0 SET.IN @SWR OR ACTFLAG IS TRUE
7924 020420  104506                                  ENASBE                        ;TRAP ON SINGLE BIT ERRORS
7925 020422                                        ELSE
7926 020424  104472                                  ECCINIT                       ;TRAP ON DOUBLE BIT ERRORS (NORMAL)
7927 020426                                        END ;OF IF #SW0
7928 020426  005037  002074                        CLR     NOPAR                   ;INDICATE PARITY ACTION
7929 020432  000207                                RETURN
7930
7931                                               ;WARNING IF YOU CHANGE THIS TABLE ALSO
7932                                               ;CHANGE '$DDW0" - "$DDW5" (THE PATTERN BIT MAP)
7933                                               ;PAT    TIME              DISCRIPTION
7934 020434                                MKPAT:  ;NOTE MT0034 MUST BE FIRST & LAST
7935 020434  026000                               MT0034  ;<1 SEC           ;SOFT ERROR - BACKGROUND PATTERN TEST
7936 020436  022574                               MT0017  ;<1 SEC           ;HOLDING 1'S & 0'S TEST
7937 020440  022004                               MT0007  ;<1 SEC           ;ADDRESS BIT TEST
7938 020442  020760                               MT0001  ;<1 SEC           ;ADDRESS TEST
7939 020444  021100                               MT0002  ;<1 SEC           ;COMPLEMENT ADDRESS TEST
7940 020446  021472                               MT0004  ; 1 SEC           ;ROTATING ZEROS TEST
7941 020450  021614                               MT0005  ; 1 SEC           ;ROTATING ONES TEST
7942 020452  022706                               MT0021  ; 1 SEC           ;MARCHING 0'S & 1'S TEST
7943 020454  023160                               MT0022  ;10 SEC           ;REFRESH & SHIFTING DIAGONAL  TEST
7944 020456  023600                               MT0026  ;<1 SEC           ;RANDOM DATA TEST
7945 020460  023256                               MT0024  ;20 SEC           ;FAST GALLOPING PATTERN TEST
7946 020462  025070                               MT0031  ; 3 SEC           ;SOB-A-LONG TEST
7947 020464  025260                               MT0032  ;<1 SEC           ;WRITE RECOVERY TEST
7948 020466  025612                               MT0033  ;35 SEC           ;BRANCH GOBBLE TEST
7949 020470  026000                               MT0034  ;<1 SEC           ;SOFT ERROR - BACKGROUND PATTERN TEST
7950                                               ;NOTE MT0034 MUST BE FIRST & LAST
7951 020472  026760                               MT0999  ; 0 SEC           ;NULL TEST
7952 020474  026760                               MT0999  ; 0 SEC           ;NULL TEST
7953 020476  026760                               MT0999  ; 0 SEC           ;NULL TEST
7954 020500  026760                               MT0999  ; 0 SEC           ;NULL TEST
7955 020502  026760                               MT0999  ; 0 SEC           ;NULL TEST
7956 020504  026760                               MT0999  ; 0 SEC           ;NULL TEST
7957 020506  026760                               MT0999  ; 0 SEC           ;NULL TEST
7958 020510  026760                               MT0999  ; 0 SEC           ;NULL TEST
7959 020512  026760                               MT0999  ; 0 SEC           ;NULL TEST
```

```
7962 020514                                MJTEST: SUBTST  <<SUBR  PARITY TEST DISPATCH>>
                                           ;*****************************************************************************
                                           ;*SUBTEST       SUBR    PARITY TEST DISPATCH
                                           ;*****************************************************************************
7963 020514  012737  000002  002074                MOV     #2,NOPAR                ;INDICATE PARITY ACTION
7964 020522  012737  000002  002322                MOV     #2,PCBUMP               ;TRAPS ADD 2 TO PC
7965 020530  012737  060000  002406                MOV     #FIRST,TESTADD
7966 020536  012737  060002  002410                MOV     #FIRST+2,TESTADD+2
7967 020544  013700  002110                        MOV     PATTERN,R0              ;GET PATTERN NUMBER
7968 020550  006300                                ASL     R0                      ;MAKE IT A WORD ADDRESS
7969 020552                                        IF MJPAT(R0) NE #MT0034 AND MJPAT(R0) NE #MT0999
7970 020572  104511                                  INVALIDATE                    ;INVALIDATE BACKGROUND PATTERN ON 'BANK"
7971 020574                                        END ;OF IF MJPAT(R0)
7972 020574  010637  002144                        MOV     SP,CTLKVEC              ;SAVE VECTOR IN CASE OF ^K
7973 020600  162737  000002  002144                SUB     #2,CTLKVEC
7974 020606  004770  020620                        CALL    @MJPAT(R0)              ;INDEX OFF TABLE
7975 020612  005037  002074                        CLR     NOPAR                   ;INDICATE PARITY ACTION
7976 020616  000207                                RETURN
7977
7978                                               ;WARNING IF YOU CHANGE THIS TABLE ALSO
7979                                               ;CHANGE "$DDW0" - "$DDW5" (THE PATTERN BIT MAP)
7980
7981                                               ;PAT    TIME            DISCRIPTION
7982 020620                                MJPAT:  ;NOTE MT0034 MUST BE FIRST & LAST
7983 020620  026000                                MT0034  ;<1 SEC         ;SOFT ERROR - BACKGROUND PATTERN TEST
7984 020622  021750                                MT0006  ;<1 SEC         ;INITIAL DATA TEST
7985 020624  022574                                MT0017  ;<1 SEC         ;HOLDING 1'S & 0'S TEST
7986 020626  022004                                MT0007  ;<1 SEC         ;ADDRESS BIT TEST
7987 020630  020760                                MT0001  ;<1 SEC         ;ADDRESS TEST
7988 020632  021100                                MT0002  ;<1 SEC         ;COMPLEMENT ADDRESS TEST
7989 020634  021240                                MT0003  ; 1 SEC         ;3 XOR 9 WORST CASE NOISE TEST
7990 020636  021472                                MT0004  ; 1 SEC         ;ROTATING ZEROS TEST
7991 020640  021614                                MT0005  ; 1 SEC         ;ROTATING ONES TEST
7992 020642  022706                                MT0021  ; 1 SEC         ;MARCHING 0'S & 1'S TEST
7993 020644  026152                                MT0035  ;<1 SEC         ;WORSE CASE NOISE PARITY TEST
7994 020646  023160                                MT0022  ;10 SEC         ;REFRESH TEST
7995 020650  023212                                MT0023  ;10 SEC         ;SHIFTING DIAGONAL  TEST
7996 020652  023600                                MT0026  ;<1 SEC         ;RANDOM DATA TEST
7997 020654  023256                                MT0024  ;20 SEC         ;FAST GALLOPING PATTERN TEST
7998 020656  025070                                MT0031  ; 3 SEC         ;SOB-A-LONG TEST
7999 020660  025260                                MT0032  ;<1 SEC         ;WRITE RECOVERY TEST
8000 020662  025612                                MT0033  ;35 SEC         ;BRANCH GOBBLE TEST
8001 020664  026000                                MT0034  ;<1 SEC         ;SOFT ERROR - BACKGROUND PATTERN TEST
8002                                               ;NOTE MT0034 MUST BE FIRST & LAST
8003 020666  026760                                MT0999  ; 0 SEC         ;NULL TEST
8004 020670  026760                                MT0999  ; 0 SEC         ;NULL TEST
8005 020672  026760                                MT0999  ; 0 SEC         ;NULL TEST
8006 020674  026760                                MT0999  ; 0 SEC         ;NULL TEST
8007 020676  026760                                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
                                                ;*******************************************************************************
 8013 020700  005037  002274                            CLR     REALPAT                 ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
 8014 020704  012700  060000                            MOV     #FIRST,R0
 8015 020710  012701  040000                            MOV     #SIZE,R1
 8016 020714  004737  041064                            CALL    REGCOPY
 8017 020720  022737  000001  003752                    CMP     #1,PROTYP               ;ARE WE ON AN 11/44?
 8018 020726  001406                                    BEQ     1$                      ;BRANCH IF YES
 8019 020730  012737  027400  002260                    MOV     #MTP000,SUPDOADD        ;ELSE DO PATTERN IN MAIN MEMORY
 8020 020736  004737  027206                            CALL    SUPD03
 8021 020742  000207                                    RETURN
 8022 020744                                    1$:     BMOV    MTP000
 8023 020752  004737  027030                            CALL    SUPD01                  ;DO IT IN SUPERVISOR MODE
 8024 020756  000207                                    RETURN
 8025 020760                                    MT0001: SUBTST  <<MT0001        SETUP ADDRESS TEST>>
                                                ;*******************************************************************************
                                                ;*SUBTEST        MT0001  SETUP ADDRESS TEST
                                                ;*******************************************************************************
 8026 020760  012737  000001  002274                    MOV     #1,REALPAT              ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
 8027 020766  012700  060000                            MOV     #FIRST,R0
 8028 020772  012701  040000                            MOV     #SIZE,R1
 8029 020776  005737  002452                            TST     NOSUPER
 8030 021002  001005                                    BNE     2$
 8031 021004  023737  172252  172254                    CMP     SIPAR5,SIPAR6
 8032 021012  001007                                    BNE     4$
 8033 021014  000404                                    BR      3$
 8034 021016  023737  177652  177654            2$:     CMP     UIPAR5,UIPAR6
 8035 021024  001002                                    BNE     4$
 8036 021026  012701  030000                    3$:     MOV     #30000,R1
 8037 021032  005002                            4$:     CLR     R2
 8038 021034  004737  041064                            CALL    REGCOPY
 8039 021040  022737  000001  003752                    CMP     #1,PROTYP               ;IS THIS AN 11/44?
 8040 021046  001406                                    BEQ     1$                      ;BRANCH IF IT IS
 8041 021050  012737  027424  002260                    MOV     #MTP001,SUPDOADD        ;SET UP CALLING ADDRESS
 8042 021056  004737  027206                            CALL    SUPD03
 8043 021062  000207                                    RETURN
 8044 021064                                    1$:     BMOV    MTP001
 8045 021072  004737  027030                            CALL    SUPD01                  ;DO IT IN SUPERVISOR MODE
 8046 021076  000207                                    RETURN
 8047 021100                                    MT0002: SUBTST  <<MT0002        SETUP COMPLEMENT ADDRESS TEST>>
                                                ;*******************************************************************************
                                                ;*SUBTEST        MT0002  SETUP COMPLEMENT ADDRESS TEST
                                                ;*******************************************************************************
 8048 021100  012737  000002  002274                    MOV     #2,REALPAT              ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
 8049 021106  012700  160000                            MOV     #LAST+2,R0
 8050 021112  012701  040000                            MOV     #SIZE,R1
 8051 021116  012704  060000                            MOV     #FIRST,R4
 8052 021122  012705  100001                            MOV     #100001,R5
 8053 021126  005737  002452                            TST     NOSUPER
 8054 021132  001005                                    BNE     2$
 8055 021134  023737  172252  172254                    CMP     SIPAR5,SIPAR6
 8056 021142  001013                                    BNE     4$
```

```
8057 021144  000404                          BR      3$
8058 021146  023737  177652  177654  2$:     CMP     UIPAR5,UIPAR6
8059 021154  001006                          BNE     4$
8060 021156  012701  030000          3$:     MOV     #30000,R1
8061 021162  012700  140000                  MOV     #140000,R0
8062 021166  012705  120001                  MOV     #120001,R5
8063 021172  012702  000001          4$:     MOV     #1,R2
8064 021176  010103                          MOV     R1,R3
8065 021200  022737  000001  003752          CMP     #1,PROTYP               ;IS THIS AN 11/44?
8066 021206  001406                          BEQ     1$                      ;BRANCH IF TRUE
8067 021210  012737  027456  002260          MOV     #MTP002,SUPDOADD        ;SET UP CALLING ADDRESS
8068 021216  004737  027206                  CALL    SUPD03
8069 021222  000207                          RETURN
8070 021224                          1$:     BMOV    MTP002
8071 021232  004737  027030                  CALL    SUPD01
8072 021236  000207                          RETURN
```

```
 8075 021240                                    MT0003: SUBTST  <<MT0003        SETUP 3 XOR 9 WORST CASE NOISE TEST>>
                                                ;*****************************************************************************
                                                ;*SUBTEST       MT0003  SETUP 3 XOR 9 WORST CASE NOISE TEST
                                                ;*****************************************************************************
 8076 021240                                            IF EUFLAG IS TRUE THEN $RETURN
 8077 021250  012737  000003  002274                    MOV     #3,REALPAT              ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
 8078 021256  005037  002322                    CLR     PCBUMP                  ;TRAPS DO NOT ADD TO PC
 8079 021262  004737  041074            1$:     CALL    FLIPWARN                ;SETUP WARNING CONSTANTS & R2
 8080 021266  012701  060000            2$:     MOV     #FIRST,R1               ;R1 <-- STARTING ADDRESS
 8081 021272  012703  020000                    MOV     #20000,R3
 8082 021276  072327  177770                    ASH     #-8.,R3                 ;R3 <-- R3 / 256.
 8083 021302  012702  000004                    MOV     #4,R2                   ;SMALL LOOP SIZE
 8084 021306  012705  000100                    MOV     #64.,R5                 ;MEDIUM LOOP SIZE
 8085 021312  022737  000001  003752            CMP     #1,PROTYP               ;IS THIS AN 11/44?
 8086 021320  001415                            BEQ     3$                      ;BRANCH IF IT IS
 8087 021322  104415                            SAVREG
 8088 021324  012737  027510  002260            MOV     #MTPA03,SUPDOADD
 8089 021332  004737  027206                    CALL    SUPD03                  ;DO IT IN MAIN MEMORY
 8090 021336  104416                            RESREG
 8091 021340  012737  027550  002260            MOV     #MTPB03,SUPDOADD
 8092 021346  004737  027222                    CALL    SUPD04
 8093 021352  000442                            BR      4$
 8094 021354                            3$:     BMOV    MTPA03
 8095 021362  104415                            SAVREG
 8096 021364  004737  027030                    CALL    SUPD01
 8097 021370                                    BMOV    MTPB03
 8098 021376                                    BMOV    MTPC03,KDPAR0,8.
 8099 021410                                    BMOV    MTPD03,SDPAR0,8.
 8100 021422  012737  172360  177642            MOV     #KDPAR0,UIPAR1          ;SET UP PAR LINKS
 8101 021430  012737  172260  172374            MOV     #SDPAR0,KDPAR6
 8102 021436  012737  177644  172276            MOV     #UIPAR2,SDPAR7
 8103 021444  012737  001032  172272            MOV     #1032,SDPAR5            ;CHANGE INST TO BR .+66 (BR TO KDPAR1)
 8104 021452  104416                            RESREG
 8105 021454  004737  027044                    CALL    SUPD02
 8106 021460  022737  000003  002602    4$:     CMP     #3,FLIPLOC              ;DONE WITH 4 PATTERNS
 8107                                                                           ;[(0,177777);(177777,0);(401,177777);(177777,401)]?
 8108 021466  001275                            BNE     1$                      ;NO - LOOP
 8109 021470  000207                            RETURN
 8110
 8111 021472                                    MT0004: SUBTST  <<MT0004        SETUP ROTATING ZEROS TEST>>
                                                ;*****************************************************************************
                                                ;*SUBTEST       MT0004  SETUP ROTATING ZEROS TEST
                                                ;*****************************************************************************
 8112 021472  012737  000004  002274                    MOV     #4,REALPAT              ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
 8113 021500  012737  000004  002322                    MOV     #4,PCBUMP               ;TRAPS ADD 4 TO PC
 8114 021506  013702  002600                            MOV     ONES,R2
 8115 021512  004737  041224                            CALL    BACKGND                 ;WRITE BACKGROUND OF ONES
 8116 021516  012700  060000                            MOV     #FIRST,R0
 8117 021522  012701  040000                            MOV     #SIZE,R1
 8118 021526  022737  000001  003752                    CMP     #1,PROTYP               ;IS THIS AN 11/44?
 8119 021534  001406                                    BEQ     1$                      ;BRANCH IF IT IS
 8120 021536  012737  027646  002260                    MOV     #MTPA04,SUPDOADD        ;SET UP LINKS
 8121 021544  004737  027222                            CALL    SUPD04
 8122 021550  000207                                    RETURN
 8123 021552                                    1$:     BMOV    MTPA04
 8124 021560                                            BMOV    MTPB04,KDPAR0,8.
 8125 021572  012737  172360  177652                    MOV     #KDPAR0,UIPAR5
```

```
 8126 021600  012737  177654  172376          MOV     #UIPAR6,KDPAR7
 8127 021606  004737  027044                  CALL    SUPD02
 8128 021612  000207                          RETURN
 8129 021614                          MT0005: SUBTST  <<MT0005        SETUP ROTATING ONES TEST>>
                                      ;*****************************************************************************
                                      ;*SUBTEST        MT0005  SETUP ROTATING ONES TEST
                                      ;*****************************************************************************
 8130 021614  012737  000005  002274          MOV     #5,REALPAT              ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
 8131 021622  012737  000004  002322          MOV     #4,PCBUMP               ;TRAPS ADD 4 TO PC
 8132 021630  005002                          CLR     R2
 8133 021632  004737  041224                  CALL    BACKGND                 ;WRITE BACKGROUND OF ZEROS
 8134 021636  012700  060000                  MOV     #FIRST,R0
 8135 021642  012701  040000                  MOV     #SIZE,R1
 8136 021646  022737  000001  003752          CMP     #1,PROTYP               ;IS THIS AN 11/44?
 8137 021654  001414                          BEQ     1$                      ;BRANCH IF IT IS
 8138 021656  012737  027722  002260          MOV     #MTP005,SUPD0ADD        ;SET UP LINKS
 8139 021664  012737  027736  027720          MOV     #MTP005+14,MTPB04+16
 8140 021672  004737  027222                  CALL    SUPD04
 8141 021676  012737  027662  027720          MOV     #MTPA04+14,MTPB04+16    ;RESET TEST'S ORIGINAL VALUE
 8142 021704  000207                          RETURN
 8143 021706                          1$:     BMOV    MTP005
 8144 021714                                  BMOV    MTPB04,KDPAR0,8.
 8145 021726  012737  172360  177652          MOV     #KDPAR0,UIPAR5
 8146 021734  012737  177654  172376          MOV     #UIPAR6,KDPAR7
 8147 021742  004737  027044                  CALL    SUPD02
 8148 021746  000207                          RETURN
```

```
 8151 021750                                MT0006: SUBTST  <<MT0006        SETUP INITIAL DATA TEST>>
                                            ;*****************************************************************************
                                            ;*SUBTEST       MT0006  SETUP INITIAL DATA TEST
                                            ;*****************************************************************************
 8152 021750  012737  000006  002274               MOV     #6,REALPAT              ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
 8153 021756  012737  000004  002322               MOV     #4,PCBUMP               ;TRAPS ADD 4 TO PC
 8154 021764  012701  002406                       MOV     #TESTADD,R1
 8155 021770  012737  027756  002260               MOV     #MTP006,SUPDOADD
 8156 021776  004737  027206                       CALL    SUPD03                  ;DO IT IN SUPERVISOR MODE
 8157 022002  000207                               RETURN
 8158 022004                                MTC007: SUBTST  <<MT0007        SETUP ADDRESS BIT TEST>>
                                            ;*****************************************************************************
                                            ;*SUBTEST       MT0007  SETUP ADDRESS BIT TEST
                                            ;*****************************************************************************
 8159 022004  012737  000007  002274               MOV     #7,REALPAT              ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
 8160 022012  005002                               CLR     R2
 8161 022014  004737  041224                       CALL    BACKGND                 ;OF ZEROS
 8162 022020  012701  060000                       MOV     #FIRST,R1
 8163 022024  012702  000001                       MOV     #1,R2
 8164 022030  050201                               BIS     R2,R1
 8165 022032  012737  030156  002260               MOV     #MTP007,SUPDOADD
 8166 022040  004737  027206                       CALL    SUPD03                  ;DO IT IN SUPERVISOR MODE -
 8167 022044  000207                               RETURN
 8168 022046                                MT0010: SUBTST  <<MT0010        SETUP BYTE ADDRESSING TEST>>
                                            ;*****************************************************************************
                                            ;*SUBTEST       MT0010  SETUP BYTE ADDRESSING TEST
                                            ;*****************************************************************************
 8169 022046  012737  000010  002274               MOV     #10,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
 8170 022054  012737  000004  002322               MOV     #4,PCBUMP               ;TRAPS ADD 4 TO PC
 8171 022062  013704  002406                       MOV     TESTADD,R4
 8172 022066  012737  030256  002260               MOV     #MTP010,SUPDOADD
 8173 022074  004737  027206                       CALL    SUPD03                  ;DO IT IN SUPERVISOR MODE
 8174 022100  000207                               RETURN
```

```
                                MT0011: SUBTST  <<MT0011        SETUP CREATE SINGLE BIT ERROR TEST>>
                                ;*****************************************************************************
                                ;*SUBTEST       MT0011  SETUP CREATE SINGLE BIT ERROR TEST
                                ;*****************************************************************************
                                        IF ACTFLAG IS TRUE OR APTFLAG IS TRUE
                                          IF $PASS NE #0 THEN $RETURN
                                        END; OF IF ACTFLAG
                                        IF PMEMFLG IS TRUE THEN $RETURN ;EXIT IF NOT MS11-M
8182 022136  012737  000011  002274       MOV     #11,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
8183 022144  012737  030364  002260       MOV     #MTP011,SUPDOADD
8184 022152  004737  027206               CALL    SUPD03                  ;DO IT IN SUPERVISOR MODE
8185 022156  000207                       RETURN
                                MT0012: SUBTST  <<MT0012        SETUP WRITE BYTE CLEARS SBE TEST>>
                                ;*****************************************************************************
                                ;*SUBTEST       MT0012  SETUP WRITE BYTE CLEARS SBE TEST
                                ;*****************************************************************************
                                        IF ACTFLAG IS TRUE OR APTFLAG IS TRUE
                                          IF $PASS NE #0 THEN $RETURN
                                        END; OF IF ACTFLAG
                                        IF PMEMFLG IS TRUE THEN $RETURN ;IS THIS A MS11-M?
8191 022214  012737  000012  002274       MOV     #12,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
8192 022222  013700  002102               MOV     BANKINDEX,R0
8193 022226                               IF #BIT12 SET.IN CONFIG+2(R0)
8194 022236  012705  040000                 MOV   #40000,R5
8195 022242                               ELSE
8196 022244  012705  000002                 MOV   #2,R5
8197 022250                               END; OF IF #BIT12
8198 022250  012737  031162  002260       MOV     #MTP012,SUPDOADD
8199 022256  004737  027206               CALL    SUPD03                  ;DO IT IN SUPERVISOR MODE
8200 022262  000207                       RETURN
                                MT0013: SUBTST  <<MT0013        SETUP CREATE DOUBLE BIT ERROR TEST>>
                                ;*****************************************************************************
                                ;*SUBTEST       MT0013  SETUP CREATE DOUBLE BIT ERROR TEST
                                ;*****************************************************************************
                                        IF ACTFLAG IS TRUE OR APTFLAG IS TRUE
                                          IF $PASS NE #0 THEN $RETURN
                                        END; OF IF ACTFLAG
                                        IF PMEMFLG IS TRUE THEN $RETURN ;EXIT IF NOT MS11-M
8206 022320  012737  000013  002274       MOV     #13,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
8207 022326  012737  031550  002260       MOV     #MTP013,SUPDOADD
8208 022334  012737  000003  002074       MOV     #3,NOPAR                ;INDICATE PARITY ACRION
8209 022342  004737  027206               CALL    SUPD03                  ;DO IT IN SUPERVISOR MODE
8210 022346  000207                       RETURN
```

```
 8212 022350                                MT0014: SUBTST  <<MT0014        SETUP BASIC DOUBLE BIT ERROR TEST>>
                                            ;*****************************************************************************
                                            ;*SUBTEST       MT0014  SETUP BASIC DOUBLE BIT ERROR TEST
                                            ;*****************************************************************************
 8213 022350                                        IF ACTFLAG IS TRUE OR APTFLAG IS TRUE
 8214 022364                                          IF $PASS NE #0 THEN $RETURN
 8215 022374                                        END; OF IF ACTFLAG
 8216 022374                                        IF PMEMFLG IS FALSE THEN $RETURN ;EXIT IF NOT MS11-P
 8217 022404  012737  000014  002274               MOV     #14,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
 8218 022412  004737  046706                       CALL  MAPKERNAL                 ;MAP KERNAL SPACE
 8219 022416                                        LET R1 := #100000               ;SETUP TEST ADDRESS
 8220 022422  004737  041324                       CALL GETCSR                     ;GET CSR INFO FROM CONFIGURATION TABLE
 8221 022426  004737  032264                       CALL MTP014                     ;DO BASIC DOUBLE BIT ERROR TEST
 8222 022432  004737  046774                       CALL  UNMAP                     ;UNMAP KERNAL SPACE
 8223 022436  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
                                                IF PMEMFLG IS TRUE THEN $RETURN ;EXIT IF NOT MS11-M
8231 022474  012737  000015  002274            MOV     #15,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
8232 022502  012737  032510  002260            MOV     #MTP015,SUPDOADD
8233 022510  004737  027206                    CALL    SUPD03                  ;DO IT IN SUPERVISOR MODE
8234 022514  000207                            RETURN
8235 022516                             MT0016: SUBTST  <<MT0016        SETUP WRITE INHIBIT OF WORD WITH DBE>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       MT0016  SETUP WRITE INHIBIT OF WORD WITH DBE
                                        ;*****************************************************************************
                                        ;
                                                IF PMEMFLG IS TRUE THEN $RETURN ;EXIT IF NOT MS11-M
                                                IF ACTFLAG IS TRUE OR APTFLAG IS TRUE
8236 022516                                       IF $PASS NE #0 THEN $RETURN
8237 022526                                     END ;OF IF ACTFLAG
8238 022542
8239 022552
8240 022552  012737  000016  002274            MOV     #16,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
8241 022560  012737  033254  002260            MOV     #MTP016,SUPDOADD
8242 022566  004737  027206                    CALL    SUPD03                  ;DO IT IN SUPERVISOR MODE
8243 022572  000207                            RETURN
8244 022574                             MT0017: SUBTST  <<MT0017        SETUP HOLDING 1'S & 0'S>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       MT0017  SETUP HOLDING 1'S & 0'S
                                        ;*****************************************************************************
                                        ;
8245 022574  012737  000017  002274            MOV     #17,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
8246 022602  012737  034036  002260            MOV     #MTP017,SUPDOADD
8247 022610  004737  027206                    CALL    SUPD03                  ;DO IT IN SUPERVISOR MODE
8248 022614  000207                            RETURN
```

```
                                        MT0020: SUBTST  <<MT0020        SETUP SYNDROMES TO CSR ON SINGLE BIT ERROR>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       MT0020  SETUP SYNDROMES TO CSR ON SINGLE BIT ERROR
                                        ;*****************************************************************************
                                                IF ACTFLAG IS TRUE OR APTFLAG IS TRUE
                                                  IF $PASS NE #0 THEN $RETURN
                                                END: OF IF ACTFLAG
8255 022642  012737  000020  002274            MOV     #20,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
8256 022650                                     IF PMEMFLG IS FALSE THEN $RETURN ;EXIT IF NOT MS11-P
8257 022660  004737  046706                     CALL  MAPKERNAL                 ;MAP KERNAL SPACE
8258 022664                                     LET R1 := #100000               ;SETUP TEST ADDRESS
8259 022670  004737  041324                     CALL GETCSR                     ;GET CSR INFO FROM CONFIGURATION TABLE
8260 022674  004737  034114                     CALL MTP020                     ;DO SYNDROMES TO CSR ON SINGLE ERROR TEST
8261 022700  004737  046774                     CALL  UNMAP                     ;UNMAP KERNAL SPACE
8262 022704  000207                           RETURN
```

```
 8264 022706                                MT0021: SUBTST  <<MT0021        SETUP MARCHING 0'S & 1'S TEST>>
                                            ;*****************************************************************************
                                            ;*SUBTEST       MT0021  SETUP MARCHING 0'S & 1'S TEST
                                            ;*****************************************************************************
 8265 022706                                        SET NOSCOPE
 8266 022714  012737  000021  002274               MOV     #21,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
 8267 022722  013702  002616                       MOV     BAKPAT,R2
 8268 022726  004737  041224                       CALL    BACKGND
 8269 022732  010203                               MOV     R2,R3
 8270 022734  000303                               SWAB    R3
 8271 022736  012701  160000                       MOV     #LAST+2,R1
 8272 022742  010105                               MOV     R1,R5
 8273 022744  012704  060000                       MOV     #FIRST,R4
 8274 022750  022737  000001  003752               CMP     #1,PROTYP       ;IS THIS AN 11/44?
 8275 022756  001441                               BEQ     1$              ;BRANCH IF IT IS
 8276 022760  022737  000003  003752               CMP     #3,PROTYP       ;IS THIS AN 11/24?
 8277 022766  001407                               BEQ     3$              ;BRANCH IF SO
 8278 022770  022737  000007  002100               CMP     #7,BANK
 8279 022776  001003                               BNE     3$
 8280 023000  012701  140000                       MOV     #140000,R1
 8281 023004  010105                               MOV     R1,R5
 8282 023006  012737  034430  002260  3$:         MOV     #MTPA21,SUPDOADD
 8283 023014  004737  027206                       CALL    SUPD03
 8284 023020  012737  034460  002260               MOV     #MTPB21,SUPDOADD
 8285 023026  004737  027222                       CALL    SUPD04
 8286 023032  010401                               MOV     R4,R1
 8287 023034  012737  034514  002260               MOV     #MTPC21,SUPDOADD
 8288 023042  004737  027222                       CALL    SUPD04
 8289 023046  012737  034550  002260               MOV     #MTPD21,SUPDOADD
 8290 023054  004737  027222                       CALL    SUPD04
 8291 023060  000434                               BR      2$
 8292 023062  022737  000177  002100  1$:         CMP     #177,BANK
 8293 023070  001003                               BNE     4$
 8294 023072  012701  140000                       MOV     #140000,R1
 8295 023076  010105                               MOV     R1,R5
 8296 023100                             4$:       BMOV    MTPA21
 8297 023106  004737  027030                       CALL    SUPD01
 8298
 8299 023112                                       BMOV    MTPB21
 8300 023120  004737  027044                       CALL    SUPD02
 8301
 8302 023124  010401                               MOV     R4,R1
 8303 023126                                       BMOV    MTPC21
 8304 023134  004737  027044                       CALL    SUPD02
 8305
 8306 023140                                       BMOV    MTPD21
 8307 023146  004737  027044                       CALL    SUPD02
 8308 023152  005037  002434             2$:       CLR     NOSCOPE
 8309 023156  000207                               RETURN
```

```
                                        MT0022: SUBTST  <<MT0022         SETUP REFRESH & SHIFTING DIAGONAL TEST>>
                                        ;*****************************************************************************
                                        ;*SUBTEST        MT0022  SETUP REFRESH & SHIFTING DIAGONAL TEST
                                        ;*****************************************************************************
8312 023160  004737  026774                     CALL    KAMITEST                ;CHECK FOR KAMIKAZE MODE
8313 023164                                     ON.ERROR THEN $RETURN           ;IF NOT IN KAMIKAZE MODE RETURN
8314 023170  012737  000022  002274             MOV     #22,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
8315 023176  012737  034600  002260             MOV     #MTP022,SUPDOADD
8316 023204  004737  027206                     CALL    SUPD03                  ;DO IT IN SUPERVISOR MODE
8317 023210  000207                            RETURN
8318
8319 023212                             MT0023: SUBTST  <<MT0023         SHIFTING DIAGONAL TEST>>
                                        ;*****************************************************************************
                                        ;*SUBTEST        MT0023  SHIFTING DIAGONAL TEST
                                        ;*****************************************************************************
8320 023212  004737  026774                     CALL    KAMITEST                ;CHECK FOR KAMIKAZE MODE
8321 023216                                     ON.ERROR THEN $RETURN           ;IF NOT IN KAMIKAZE MODE RETURN
8322 023222  012737  000023  002274             MOV     #23,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
8323 023230  012737  034600  002260             MOV     #MTP022,SUPDOADD
8324 023236                                     SET     DIAGFLAG                ;IDENTIFY DIAGONAL TEST TO MTP022
8325 023244  004737  027206                     CALL    SUPD03                  ;DO IT IN SUPERVISOR MODE
8326 023250  005037  002002                     CLR     DIAGFLAG
8327 023254  000207                            RETURN
```

```
                                        MT0024: SUBTST  <<MT0024        SETUP FAST GALLOPING PATTERN TEST>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       MT0024  SETUP FAST GALLOPING PATTERN TEST
                                        ;*****************************************************************************
                                        ;
 8330 023256  004737  026774                    CALL    KAMITEST                ;CHECK FOR KAMIKAZE MODE
 8331 023262                                    ON.ERROR THEN $RETURN           ;IF NOT IN KAMIKAZE MODE RETURN
 8332 023266                                    SET     NOSCOPE
 8333 023274  012737  000024  002274            MOV     #24,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
 8334 023302  013702  002616                    MOV     BAKPAT,R2
 8335 023306  004737  041224                    CALL    BACKGND
 8336 023312  010203                            MOV     R2,R3
 8337 023314  010304                            MOV     R3,R4
 8338 023316  000304                            SWAB    R4
 8339 023320  012701  060000                    MOV     #FIRST,R1
 8340 023324  012705  157776                    MOV     #LAST,R5
 8341 023330  022737  000001  003752            CMP     #1,PROTYP
 8342 023336  001417                            BEQ     1$
 8343 023340  022737  000003  003752            CMP     #3,PROTYP
 8344 023346  001406                            BEQ     3$
 8345 023350  022737  000007  002100            CMP     #7,BANK
 8346 023356  001002                            BNE     3$
 8347 023360  012705  137776                    MOV     #137776,R5
 8348 023364  104415                    3$:     SAVREG
 8349 023366  012737  035314  002260            MOV     #MTPB24,SUPDOADD
 8350 023374  000440                            BR      2$
 8351 023376  022737  000177  002552    1$:     CMP     #177,LASTBANK
 8352 023404  001002                            BNE     4$
 8353 023406  012705  137776                    MOV     #137776,R5
 8354 023412  104415                    4$:     SAVREG
 8355 023414                                    BMOV    MTPA24
 8356 023422                                    BMOV    MTPB24,SDPAR0,8.
 8357 023434                                    BMOV    MTPC24,KDPAR0,8.
 8358 023446  012737  172260  002260            MOV     #SDPAR0,SUPDOADD
 8359 023454  012737  172260  177676            MOV     #SDPAR0,UDPAR7          ;SET UP PAR LINKS
 8360 023462  012737  172360  172272            MOV     #KDPAR0,SDPAR5
 8361 023470  012737  177660  172374            MOV     #UDPAR0,KDPAR6
 8362 023476  004737  027222            2$:     CALL    SUPD04
 8363
 8364                                           ;DO IT AGAIN FOR COMPLEMENT DATA
 8365 023502  104416                            RESREG
 8366 023504  000302                            SWAB    R2
 8367 023506  000303                            SWAB    R3
 8368 023510  004737  027222                    CALL    SUPD04
 8369 023514  005037  002434                    CLR     NOSCOPE
 8370 023520  000207                            RETURN
 8371 023522                            MT0025: SUBTST  <<MT0025        SETUP INTERRUPT ENABLE TEST>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       MT0025  SETUP INTERRUPT ENABLE TEST
                                        ;*****************************************************************************
                                        ;
 8372 023522                                    IF ACTFLAG IS TRUE OR APTFLAG IS TRUE
 8373 023536                                      IF $PASS NE #0 THEN $RETURN
 8374 023546                                    END ;OF IF ACTFLAG
 8375 023546                                    IF PMEMFLG IS TRUE THEN $RETURN ;EXIT IF NOT MS11-P
 8376 023556  012737  000025  002274            MOV     #25,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
 8377 023564  012737  035346  002260            MOV     #MTP025,SUPDOADD
 8378 023572  004737  027206                    CALL    SUPD03                  ;DO IT IN SUPERVISOR MODE
 8379 023576  000207                            RETURN
```

```
 8382 023600                                MT0026: SUBTST  <<MT0026        SETUP RANDOM DATA TEST>>
                                            ;*****************************************************************************
                                            ;*SUBTEST       MT0026  SETUP RANDOM DATA TEST
                                            ;*****************************************************************************
 8383 023600  012737  000026  002274               MOV     #26,REALPAT
 8384 023606  005037  002322                       CLR     PCBUMP                  ;TRAPS DO NOT ADD TO THE PC
 8385 023612  013703  002570                       MOV     SEEDLO,R3               ;INITIALIZE RANDOM NUMBERS
 8386 023616  013702  002566                       MOV     SEEDHI,R2
 8387 023622  010305                               MOV     R3,R5
 8388 023624  010204                               MOV     R2,R4
 8389 023626  012701  060000                       MOV     #FIRST,R1
 8390 023632  012700  020000                       MOV     #SIZE/2,R0
 8391 023636  022737  000001  003752               CMP     #1,PROTYP               ;DO WE HAVE AN 11/44?
 8392 023644  001437                               BEQ     1$                      ;BRANCH IF WE DO
 8393 023646  022737  000003  003752               CMP     #3,PROTYP               ;11/24?
 8394 023654  001406                               BEQ     3$                      ;BRANCH IF SO
 8395 023656  022737  000007  002100               CMP     #7,BANK
 8396 023664  001002                               BNE     3$
 8397 023666  012700  014000                       MOV     #14000,R0
 8398 023672  104415                      3$:      SAVREG
 8399 023674  012737  036020  036120               MOV     #MTPA26+4,MTPD26+14
 8400 023702  012737  036014  002260               MOV     #MTPA26,SUPDOADD
 8401 023710  004737  027206                       CALL    SUPDO3
 8402 023714  005037  036044                       CLR     RANODD                  ;FOR ERROR REPORTING
 8403 023720  012737  036034  036120               MOV     #MTPB26+4,MTPD26+14     ;SET UP NEXT LINK
 8404 023726  012737  036030  002260               MOV     #MTPB26,SUPDOADD
 8405 023734  104416                               RESREG
 8406 023736  004737  027206                       CALL    SUPDO3
 8407 023742  000452                               BR      2$
 8408 023744  022737  000177  002100  1$:         CMP     #177,BANK
 8409 023752  001002                               BNE     4$
 8410 023754  012700  014000                       MOV     #14000,R0
 8411 023760  104415                      4$:      SAVREG
 8412 023762                                       BMOV    MTPA26                  ;WRITE ROUTINE TO FAST MEMORY
 8413 023770                                       BMOV    MTPC26,KDPAR0,8.        ;RANDOM SUBPROGRAM TO FAST MEMORY
 8414 024002  012737  000730  172376               MOV     #730,KDPAR7             ;WRITES 'BR .-116'' IN (BR SDPAR0)
 8415 024010                                       BMOV    MTPD26,SDPAR0,8.        ;RANDOM SUBSUBPROGRAM TO FAST MEMORY
 8416 024022  012737  172360  177642               MOV     #KDPAR0,UIPAR1
 8417 024030  012737  177644  172274               MOV     #UIPAR2,SDPAR6
 8418 024036  004737  027030                       CALL    SUPDO1                  ;WRITE RANDOM DATA
 8419 024042  005037  036044                       CLR     RANODD                  ;FOR ERROR REPORTING
 8420 024046                                       BMOV    MTPB26                  ;READ ROUTINE TO FAST MEMORY
 8421 024054  012737  172360  177642               MOV     #KDPAR0,UIPAR1          ;SET UP PAR LINK
 8422 024062  104416                               RESREG
 8423 024064  004737  027030                       CALL    SUPDO1                  ;READ RANDOM DATA
 8424 024070  010337  002570          2$:          MOV     R3,SEEDLO               ;UPDATE FOR NEW RANDOM NUMBERS
 8425 024074  010237  002566                       MOV     R2,SEEDHI
 8426 024100  000207                               RETURN
```

```
8429 024102                                MT0027: SUBTST  <<MT0027        UNIQUE BANK TEST>>
                                           ;*****************************************************************************
                                           ;*SUBTEST       MT0027  UNIQUE BANK TEST
                                           ;*****************************************************************************
                                           ;
8430                                               ;MAKE SURE THAT EACH BANK CAN HAVE UNIQUE DATA
8431                                               ;WRITE AND READ THE BANK NUMBER IN EACH BANK (EXCEPT WHERE THE PROGRAM IS)
8432 024102  012737  000027  002274          MOV     #27,REALPAT                ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
8433 024110  104502                        CLRCSR                      ;CLEAR CSRS
8434 024112  022737  000001  003752          CMP     #1,PROTYP                  ;IS THIS AN 11/44?
8435 024120  001404                          BEQ     1$                         ;BRANCH IF TRUE
8436 024122  012737  027206  002516          MOV     #SUPDO3,LINK1              ;SET UP LINK
8437 024130  000414                          BR      STAR27                     ;BRANCH TO RUN
8438 024132                                1$:       BMOV    MTP034
8439 024140  012737  177646  002260    WARN7:  MOV     #UIPAR3,SUPDOADD
8440 024146  012737  027030  002516          MOV     #SUPDO1,LINK1              ;SET UP LINK
8441 024154                                      SET NOFSMODE
8442 024162                                STAR27: FOR I := #1 TO #2
8443 024170                                          FOR BANK := #0 TO LASTBANK
8444 024174  004737  047020                             CALL EXBANK
8445 024200                                             IF ACFLAG IS TRUE AND RRFLAG IS FALSE
8446 024214  104511                                        INVALIDATE              ;INVALIDATE BACKGROUND PATTERN ON 'BANK'
8447 024216                                                LET R2 := BANK
8448 024222  012700  060000                               MOV       #FIRST,R0
8449 024226  010004                                       MOV       R0,R4
8450 024230  012701  040000                               MOV       #SIZE,R1
8451 024234  010103                                       MOV       R1,R3
8452 024236                                               IF I EQ #1
8453 024246  022737  000001  003752                          CMP     #1,PROTYP
8454 024254  001403                                          BEQ     2$
8455 024256  012737  036340  002260                          MOV     #MTP034,SUPDOADD
8456 024264  004777  156226                             2$: CALL    @LINK1
8457 024270                                               END ;OF IF
8458 024270                                               IF I EQ #2
8459 024300  022737  000001  003752                          CMP     #1,PROTYP
8460 024306  001403                                          BEQ     3$
8461 024310  012737  036346  002260                          MOV     #MTP034+6,SUPDOADD
8462 024316  004737  027206                             3$: CALL    SUPDO3
8463 024322                                               END ;OF IF
8464 024322                                             END ;OF IF
8465 024322                                          END ;OF FOR BANK
8466 024336                                     END ;OF FOR I
8467 024352                                     IF FS7FLAG IS TRUE
8468 024360  005037  002422                       CLR   NOFSMODE
8469 024364  000207                               RETURN
8470 024366                                     END ;OF IF FS7FLAG
8471 024366                                     FOR I := #1 TO #2
8472 024374                                       FOR BANK := LASTBANK DOWNTO #0
8473 024402  004737  047020                         CALL EXBANK
8474 024406                                         IF ACFLAG IS TRUE AND RRFLAG IS FALSE
8475 024422                                           LET R2 := BANK
8476 024426  005102                                   COM       R2
8477 024430  012700  060000                           MOV       #FIRST,R0
8478 024434  010004                                   MOV       R0,R4
8479 024436  012701  040000                           MOV       #SIZE,R1
8480 024442  010103                                   MOV       R1,R3
8481 024444                                           IF I EQ #1
8482 024454  022737  000001  003752                      CMP     #1,PROTYP
```

```
8483 024462  001403                                    BEQ     4$
8484 024464  012737  036340  002260                    MOV     #MTP034,SUPDOADD
8485 024472  004777  156020                          4$: CALL    @LINK1
8486 024476                                            END ;OF IF
8487 024476                                            IF I EQ #2
8488 024506  022737  000001  003752                      CMP     #1,PROTYP
8489 024514  001403                                      BEQ     5$
8490 024516  012737  036346  002260                      MOV     #MTP034+6,SUPDOADD
8491 024524  004737  027206                          5$: CALL    SUPD03
8492 024530                                            END ;OF IF
8493 024530                                          END ;OF IF
8494 024530                                        END ;OF FOR BANK
8495 024544                                      END ;OF FOR I
8496 024560  005037  002422                      CLR     NOFSMODE
8497 024564  000207                              RETURN
```

```
                                        MT0030: SUBTST  <<MT0030        SETUP FLUSH OUT DBE'S TEST>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       MT0030  SETUP FLUSH OUT DBE'S TEST
                                        ;*****************************************************************************
 8501 024566  005037  002262                    CLR     PASFLG
 8502 024572                                    SET     FULLREL
 8503 024600  012737  000030  002274  MTA030:   MOV     #30,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
 8504 024606  012737  000001  002074            MOV     #1,NOPAR                ;INDICATE COUNT PARITY ERRORS
 8505 024614  022737  000001  003752            CMP     #1,PROTYP
 8506 024622  001007                            BNE     4$
 8507 024624                                    BMOV    MTP030
 8508 024632  012737  027030  002516            MOV     #SUPD01,LINK1
 8509 024640  000406                            BR      1$
 8510 024642  012737  027206  002516  4$:       MOV     #SUPD03,LINK1
 8511 024650  012737  036122  002260            MOV     #MTP030,SUPDOADD
 8512 024656  104470                    1$:     ECCDIS                          ;DISABLE ERROR CORRECTION
 8513 024660                                    SET     NOFSMODE,NOSCOPE
 8514 024674                                    FOR BANK := #0 TO LASTBANK
 8515 024700  004737  047020                      CALL  EXBANK
 8516 024704                                      IF MKFLAG IS TRUE
 8517 024712                                        IF ACFLAG IS TRUE AND RRFLAG IS FALSE
 8518 024726  012701  040000                          MOV       #SIZE,R1
 8519 024732  012700  060000                          MOV       #FIRST,R0
 8520 024736  004777  155554                          CALL      @LINK1
 8521 024742                                        END ;OF IF ACFLAG
 8522 024742                                      END ;OF IF MKFLAG
 8523 024742                                    END ;OF FOR
 8524 024756                                    IF PASFLG IS FALSE
 8525 024764                                      SET PASFLG
 8526 024772  104502                              CLRCSR                        ;CLEAR CSRS
 8527 024774  004737  045172                      CALL  RELOCATE
 8528 025000                                      ON.ERROR
 8529 025002  104472                                ECCINIT                     ;TRAP ON DOUBLE BIT ERRORS (NORMAL)
 8530 025004                                        CLEAR       NOFSMODE,NOSCOPE,FULLREL
 8531 025020  000207                                RETURN
 8532 025022                                      END ;OF ON.ERROR
 8533 025022  013737  002304  002100              MOV   NEWBANK,BANK
 8534 025030  004737  047020                      CALL  EXBANK
 8535 025034  004737  024600                      CALL  MTA030
 8536 025040  104472                              ECCINIT                       ;TRAP ON DOUBLE BIT ERRORS (NORMAL)
 8537 025042  004737  046056                      CALL  UNRELOCATE
 8538 025046  000207                              RETURN
 8539 025050                                    END ;OF IF PASFLG
 8540 025050  104472                            ECCINIT                 ;TRAP ON DOUBLE BIT ERRORS (NORMAL)
 8541 025052                                    CLEAR   NOFSMODE,NOSCOPE,FULLREL
 8542 025066  000207                            RETURN
```

```
8545 025070                                MT0031: SUBTST  <<MT0031        SETUP SOB-A-LONG TEST>>
                                           ;*****************************************************************************
                                           ;*SUBTEST       MT0031  SETUP SOB-A-LONG TEST
                                           ;*****************************************************************************
                                           ;
8546 025070  004737  026774                        CALL    KAMITEST                ;CHECK FOR KAMIKAZE MODE
8547 025074                                        ON.ERROR THEN $RETURN           ;IF NOT IN KAMIKAZE MODE RETURN
8548 025100                                        SET     NOSCOPE
8549 025106  012737  000031  002274                MOV     #31,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
8550 025114  005037  002074                        CLR     NOPAR                   ;SETUP PARITY ACTION
8551 025120                                        MAP     BANK                    ;MAP FIRST SO BLOCK MOVE WORKS
8552 025134                                        TESTAREA                        ;ENTER TEST MODE
8553 025142                                        BMOV    MTP031,FIRST,SOBLENGTH/2
8554 025154  104417                                KERNEL                          ;ENTER KERNEL MODE
8555 025156  013702  002556                        MOV     SOBK,R2
8556 025162  010200                                MOV     R2,R0
8557 025164  012701  100776                        MOV     #100776,R1              ;COMPLEMENT OF INSTRUCTION "SOB R0,DOT"
8558 025170  012705  060056                        MOV     #FIRST+SOBLENGTH,R5
8559 025174  012737  060002  002260                MOV     #FIRST+2,SUPDOADD
8560 025202  012737  160000  002516                MOV     #LAST+2,LINK1
8561 025210  005737  002452                        TST     NOSUPER
8562 025214  001005                                BNE     1$
8563 025216  023737  172252  172254                CMP     SIPAR5,SIPAR6
8564 025224  001405                                BEQ     2$
8565 025226  000407                                BR      3$
8566 025230  023737  177652  177654  1$:           CMP     UIPAR5,UIPAR6
8567 025236  001003                                BNE     3$
8568 025240  012737  140000  002516  2$:           MOV     #140000,LINK1
8569 025246  004737  027222          3$:           CALL    SUPD04
8570 025252  005037  002434                        CLR     NOSCOPE
8571 025256  000207                                RETURN
```

```
                                        MT0032: SUBTST  <<MT0032        SETUP WRITE RECOVERY TEST>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       MT0032  SETUP WRITE RECOVERY TEST
                                        ;*****************************************************************************
                                        ;
8574 025260                                     CALL    KAMITEST                ;CHECK FOR KAMIKAZE MODE
8575 025260  004737  026774                     ON.ERROR THEN $RETURN           ;IF NOT IN KAMIKAZE MODE RETURN
8576 025264                                     SET     NOSCOPE
8577 025270                                     MOV     #32,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
8578 025276  012737  000032  002274             CLR     NOPAR                   ;SETUP PARITY ACTION
8579 025304  005037  002074                     MAP     BANK                    ;MAP FIRST SO THAT THE BLOCK MOVE WORKS
8580 025310                                     MOV     #10247,R0               ;OP CODE OF INSTRUCTION 'MOV    R2,-(PC)''
8581 025324  012700  010247                     MOV     #177667,R1              ;OP CODE OF COMPLEMENT OF INSTRUCTION ''JMP (R0)''
8582 025330  012701  177667                     MOV     #SIZE/2,R2              ;USED FOR 1/2 BANK LOOP
8583 025334  012702  020000                     MOV     R2,LIN
8584 025340  010237  002516                     MOV     #FIRST,R3
8585 025344  012703  060000                     MOV     #LAST+2,R4
8586 025350  012704  160000                     CLR     LINK2
8587 025354  005037  002520                     TST     NOSUPER
8588 025360  005737  002452                     BNE     1$
8589 025364  001005                            CMP     SIPAR5,SIPAR6
8590 025366  023737  172252  172254            BEQ     2$
8591 025374  001405                            BR      3$
8592 025376  000415                    1$:     CMP     UIPAR5,UIPAR6
8593 025400  023737  177652  177654            BNE     3$
8594 025406  001011                    2$:     MOV     #140000,R4
8595 025410  012704  140000                     MOV     #14000,R2
8596 025414  012702  014000                     MOV     R2,LINK1
8597 025420  010237  002516                     MOV     #1,LINK2
8598 025424  012737  000001  002520
8599
8600 025432                            3$:     TESTAREA                        ;ENTER TEST MODE
8601                                            ;MOVE TEST TO MEMORY UNDER TEST
8602 025440  010023                    4$:     MOV     R0,(R3)+
8603 025442  010144                            MOV     R1,-(R4)
8604 025444  077203                            SOB     R2,4$
8605
8606 025446  022737  000001  003752            CMP     #1,PROTYP
8607 025454  001003                            BNE     5$
8608                                            ;MOVE LAST PART OF TEST TO FASTCITY
8609 025456                                     BMOV    MTP032
8610 025464  104417                    5$:     KERNEL                          ;ENTER KERNEL MODE
8611
8612 025466  012702  005141                     MOV     #5141,R2                ;OP CODE OF INSTRUCTION ''COM    -(R1)''
8613 025472  012700  025610                     MOV     #10$,R0                 ;ADDRESS TO RETURN TO IN R0
8614 025476  012701  160000                     MOV     #LAST+2,R1              ;TOP OF BANK
8615 025502  012737  060000  002260             MOV     #FIRST,SUPDOADD
8616 025510  005737  002520                     TST     LINK2
8617 025514  001402                            BEQ     6$
8618 025516  012701  140000                     MOV     #140000,R1
8619 025522  004737  027222            6$:     CALL    SUPD04
8620 025526  012703  020000                     MOV     #SIZE/2,R3
8621 025532  012705  000110                     MOV     #110,R5
8622 025536  012704  060000                     MOV     #FIRST,R4
8623 025542  005737  002520                     TST     LINK2
8624 025546  001402                            BEQ     7$
8625 025550  012703  014000                     MOV     #14000,R3
8626 025554  022737  000001  003752    7$:     CMP     #1,PROTYP
8627 025562  001406                            BEQ     8$
```

```
 8628 025564  012737  036210  002260          MOV     #MTP032,SUPDOADD
 8629 025572  004737  027222                  CALL    SUPD04
 8630 025576  000402                          BR      9$
 8631 025600  004737  027044          8$:     CALL    SUPD02
 8632 025604  005037  002434          9$:     CLR     NOSCOPE
 8633 025610  000207                  10$:    RETURN                  ;THIS RETURN ACTS AS A NORMAL RETURN FROM MT0032
 8634                                                                 ;ALSO A RETURN FROM THE "CALL   SUPD04" ABOVE
 8635
```

```
 8638 025612                                MT0033: SUBTST  <<MT0033        SETUP BRANCH GOBBLE TEST>>
                                            ;*****************************************************************************
                                            ;*SUBTEST       MT0033  SETUP BRANCH GOBBLE TEST
                                            ;*****************************************************************************
 8639 025612  004737  026774                        CALL    KAMITEST                ;CHECK FOR KAMIKAZE MODE
 8640 025616                                        ON.ERROR THEN $RETURN           ;IF NOT IN KAMIKAZE MODE RETURN
 8641 025622                                        SET     NOSCOPE
 8642 025630  012737  000033  002274                MOV     #33,REALPAT             ;SETUP PATTERN NUMBER FOR TYPEOUT & DISPLAY
 8643 025636  005037  002074                        CLR     NOPAR                   ;SETUP PARITY ACTION
 8644 025642                                        MAP     BANK                    ;MAP FIRST SO THAT BLOCK MOVE WORKS
 8645
 8646 025656                                        TESTAREA                        ;ENTER TEST MODE
 8647 025664                                        BMOV    MTP033,FIRST,GBLENGTH/2
 8648 025676  104417                                KERNEL                          ;ENTER KERNEL MODE
 8649
 8650 025700  012705  060076                        MOV     #FIRST+GBLENGTH,R5
 8651 025704  012737  060004  002260                MOV     #FIRST+4,SUPDOADD
 8652 025712  012701  060002                        MOV     #FIRST+2,R1
 8653 025716  012702  060003                        MOV     #FIRST+3,R2
 8654 025722  012737  160000  002516                MOV     #LAST+2,LINK1
 8655 025730  005737  002452                        TST     NOSUPER
 8656 025734  001005                                BNE     1$
 8657 025736  023737  172252  172254                CMP     SIPAR5,SIPAR6
 8658 025744  001405                                BEQ     2$
 8659 025746  000407                                BR      3$
 8660 025750  023737  177652  177654  1$:           CMP     UIPAR5,UIPAR6
 8661 025756  001003                                BNE     3$
 8662 025760  012737  140000  002516  2$:           MOV     #140000,LINK1
 8663
 8664 025766  004737  027222          3$:           CALL    SUPD04
 8665 025772  005037  002434                        CLR     NOSCOPE
 8666 025776  000207                                RETURN
 8667
 8668 026000                                MT0034: SUBTST  <<MT0034        SOFT ERROR - BACKGROUND PATTERN TEST>>
                                            ;*****************************************************************************
                                            ;*SUBTEST       MT0034  SOFT ERROR - BACKGROUND PATTERN TEST
                                            ;*****************************************************************************
 8669 026000  012737  000034  002274                MOV     #34,REALPAT
 8670 026006  012700  060000                        MOV     #FIRST,R0
 8671 026012  012701  040000                        MOV     #SIZE,R1
 8672 026016  013702  002604                        MOV     SOFTPAT,R2
 8673 026022  010103                                MOV     R1,R3
 8674 026024  013705  002102                        MOV     BANKINDEX,R5
 8675 026030  010004                                MOV     R0,R4
 8676 026032  022737  000001  003752                CMP     #1,PROTYP               ;IS THIS AN 11/44?
 8677 026040  001006                                BNE     1$                      ;BRANCH IF NOT
 8678 026042                                        BMOV    MTP034
 8679 026050  012737  177646  002260                MOV     #UIPAR3,SUPDOADD
 8680 026056                                1$:     IF #BIT13 SET.IN CONFIG+2(R5)
 8681                                                 ;BACKGROUND PATTERN IS VALID
 8682 026066  022737  000001  003752                  CMP   #1,PROTYP
 8683 026074  001403                                  BEQ   2$
 8684 026076  012737  036346  002260                  MOV   #MTP034+6,SUPDOADD
 8685 026104  004737  027206                2$:       CALL  SUPD03                  ;READ IT
 8686 026110                                        ELSE
 8687                                                 ;BACKGROUND PATTERN HAS BEEN INVALIDATED
 8688 026112  022737  000001  003752                  CMP   #1,PROTYP
```

```
8689 026120  001406                          BEQ   3$
8690 026122  012737  036340  002260          MOV   #MTP034,SUPDOADD
8691 026130  004737  027206                  CALL  SUPD03
8692 026134  000402                          BR    4$
8693 026136  004737  027030            3$:   CALL  SUPD01                    ;WRITE IT
8694 026142  052765  020000  002652    4$:   BIS   #BIT13,CONFIG+2(R5)       ;VALIDATE IT
8695 026150                                 END ;OF IF #BIT13
8696 026150  000207                         RETURN
8697
8698 026152                                 MT0035: SUBTST  <<MT0035        SETUP WORST CASE NOISE PARITY TEST>>
                                            ;*****************************************************************************
                                            ;*SUBTEST       MT0035  SETUP WORST CASE NOISE PARITY TEST
                                            ;*****************************************************************************
8699 026152  012737  000035  002274          MOV     #35,REALPAT             ;SET UP TEST NUMBER FOR DISPLAY
8700 026160  013703  002102                  MOV     BANKINDEX,R3
8701 026164  016301  002650                  MOV     CONFIG(R3),R1
8702 026170  000301                          SWAB    R1
8703 026172  042701  177760                  BIC     #^C17,R1
8704 026176  006301                          ASL     R1
8705 026200  010137  002150                  MOV     R1,CSRNO
8706 026204  023737  002150  002526          CMP     CSRNO,PGMCSR
8707 026212  001001                         BNE     1$
8708 026214  000207                         RETURN
8709 026216  012702  052524            1$:   MOV     #52524,R2
8710 026222  004737  041224                  CALL    BACKGND                 ;WRITE BACKROUND OF ALMOST ALT. 1'S AND 0'S
8711 026226  012737  036364  002260          MOV     #MTP035,SUPDOADD
8712 026234  004737  027206                  CALL    SUPD03
8713 026240                                  IF QVFLAG IS TRUE THEN $RETURN
8714 026250  005102                          COM     R2
8715 026252  004737  041224                  CALL    BACKGND                 ;WRITE COMPLEMENT PATTERN INTO MUT
8716 026256  004737  027222                  CALL    SUPD04
8717 026262  000207                         RETURN
```

```
 8719 026264                                MT0036: SUBTST  <<MT0036        SETUP CORRECTION CODE TEST>>
                                            ;*****************************************************************************
                                            ;*SUBTEST       MT0036  SETUP CORRECTION CODE TEST
                                            ;*****************************************************************************
                                                    IF PMEMFLG IS FALSE THEN $RETURN ;IF NOT MS11-P THEN EXIT
 8720 026264                                MOV     #36,REALPAT             ;SET UP TEST NUMBER FOR TYPEOUT AND DISPLAY
 8721 026274  012737  000036  002274       CALL    GETCSR                  ;GET CSR INFO FROM CONFIG TABLE
 8722 026302  004737  041324               CLR     PASFLG                  ;CLEAR LOOP COUNTER
 8723 026306  005037  002262               CLR     R0                      ;GET TEST DATA
 8724 026312  005000                       MOV     #100000,R1              ;GET FIRST ADDRESS IN BANK
 8725 026314  012701  100000               CALL    MAPKERNAL               ;MAP KIPAR5 AND 6 TO BANK
 8726 026320  004737  046706               CALL    MTP036                  ;EXECUTE TEST
 8727 026324  004737  036526               CALL    UNMAP                   ;REMAP KERNAL SPACE
 8728 026330  004737  046774               RETURN
 8729 026334  000207
```

```
                                        MT0037: SUBTST  <<MT0037         SETUP ECC DISABLE TEST>>
                                        ;*****************************************************************************
                                        ;*SUBTEST        MT0037  SETUP ECC DISABLE TEST
                                        ;*****************************************************************************
                                        ;
                                                IF PMEMFLG IS FALSE THEN $RETURN ;RETURN IF NOT A MS11-P
 8732 026336                                    MOV     #37,REALPAT             ;SETUP PATTERN AND NUMBER FOR TYPEOUT AND DISPLAY
 8733 026346  012737  000037  002274            MOV     #100000,R1              ;SET UP TEST ADDRESS
 8734 026354  012701  100000                    CLR     R0                      ;CLEAR DATA TO BE WRITTEN
 8735 026360  005000                            CALL    MAPKERNAL               ;MAP THIS TEST TO KERNEL SPACE
 8736 026362  004737  046706                    CALL    GETCSR                  ;GET CSRINFO FROM CONFIG TABLE
 8737 026366  004737  041324                    CALL    MTP037                  ;CHECK ECC DISABLE
 8738 026372  004737  036752                    CALL    UNMAP                   ;REMAP KERNEL SPACE
 8739 026376  004737  046774                    RETURN
 8740 026402  000207
 8741
 8742
 8743 026404                            MT0040: SUBTST  <<MT0040         >>
                                        ;*****************************************************************************
                                        ;*SUBTEST        MT0040
                                        ;*****************************************************************************
                                        ;
 8744 026404  000207                            RETURN                          ;
```

```
                                        MT0041: SUBTST  <<MT0041        SETUP ADDRESS TO CSR ON DOUBLE BIT ERROR TEST>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       MT0041  SETUP ADDRESS TO CSR ON DOUBLE BIT ERROR TEST
                                        ;*****************************************************************************
 8747 026406                                    IF PMEMFLG IS FALSE THEN $RETURN ;EXIT IF NOT MS11-P
 8748 026416  012737  000041  002274            MOV     #41,REALPAT             ;SETUP PATTERN AND NUMBER FOR TYPEOUT AND DISPLAY
 8749 026424  004737  041324                    CALL    GETCSR                  ;GET CSR NUMBER AND ADDRESS FROM CONFIGURATION TABLE
 8750 026430                                    LET SUPDOADD := #MTP041         ;SET UP TEST ADDRESS
 8751 026436                                    LET R1 := #FIRST                ;SET UP FIRST ADDRESS
 8752 026442  004737  027206                    CALL SUPDO3                     ;EXECUTE ADDDRESS TO CSR TEST IN SUPVISIOR MODE
 8753 026446  000207                            RETURN                          ;
```

```
 8755 026450                                MT0042: SUBTST  <<MT0042        SETUP EXTENDED UNIBUS ADDRESS TO CSR TEST>>
                                            ;*****************************************************************************
                                            ;*SUBTEST       MT0042  SETUP EXTENDED UNIBUS ADDRESS TO CSR TEST
                                            ;*****************************************************************************
                                                    IF PMEMFLG IS FALSE THEN $RETURN ;EXIT IF NOT MS11-P
 8756 026450                                        MOV     #42,REALPAT             ;SETUP PATTERN AND NUMBER FOR TYPEOUT AND DISPLAY
 8757 026460  012737  000042  002274               MOV     #100000,R1              ;SET UP TEST ADDRESS
 8758 026466  012701  100000                        CALL    MAPKERNAL               ;MAP TO KERNEL SPACE
 8759 026472  004737  046706                        CALL    GETCSR                  ;SET UP CSRINFO FROM CONFIGURATION TABLE
 8760 026476  004737  041324                        CALL    MTP042                  ;CHECK EXTENDED UNIBUS ADDRESS TO CSR
 8761 026502  004737  037176                        CALL    UNMAP                   ;REMAP KERNEL SPACE
 8762 026506  004737  046774                        RETURN
 8763 026512  000207
```

```
                                        MT0043: SUBTST  <<MT0043        SETUP WRITE BYTE CLEARS SBE TEST>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       MT0043  SETUP WRITE BYTE CLEARS SBE TEST
                                        ;*****************************************************************************
                                                IF PMEMFLG IS FALSE THEN $RETURN ;EXIT IF NOT MS11-P
 8766 026514
 8767 026524  012737  000043  002274           MOV     #43,REALPAT             ;SET UP TEST NUMBER FOR TYPEOUT AND DISPLAY
 8768 026532  004737  046706                   CALL    MAPKERNAL               ;MAP TO KERNEL SPACE
 8769 026536                                    LET R1 := #100000               ;SET UP TEST ADDRESS
 8770 026542  004737  037432                   CALL MTP043                     ;PERFORM WRITE BYTE TEST
 8771 026546  004737  046774                   CALL UNMAP                      ;REMAP KERNEL SPACE
 8772 026552  000207                           RETURN
 8773 026554                            MT0044: SUBTST  <<MT0044        SETUP SHIFTING 1/0'S THROUGH THE CHECK BITS TEST>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       MT0044  SETUP SHIFTING 1/0'S THROUGH THE CHECK BITS TEST
                                        ;*****************************************************************************
                                                IF PMEMFLG IS FALSE THEN $RETURN ;EXIT IF NOT MS11-P
 8774 026554
 8775 026564  012737  000044  002274           MOV     #44,REALPAT             ;SET UP TEST NUMBER FOR TYPEOUT AND DISPLAY
 8776 026572  004737  046706                   CALL    MAPKERNAL               ;MAP TO KERNEL SPACE
 8777 026576                                    LET R1 := #100000               ;SET UP TEST ADDRESS
 8778 026602  004737  037626                   CALL MTP044                     ;PERFORM SHIFTING 1/0'S THROUGH THE CHECK BITS
 8779 026606  004737  046774                   CALL UNMAP                      ;REMAP KERNEL SPACE
 8780 026612  000207                           RETURN
 8781 026614                            MT0045: SUBTST  <<MT0045        SETUP SYNDROMES TO CSR ON DOUBLE BIT ERROR>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       MT0045  SETUP SYNDROMES TO CSR ON DOUBLE BIT ERROR
                                        ;*****************************************************************************
                                                IF PMEMFLG IS FALSE THEN $RETURN ;EXIT IF NOT MS11-P
 8782 026614
 8783 026624  012737  000045  002274           MOV     #45,REALPAT             ;SET UP TEST NUMBER FOR TYPEOUT AND DISPLAY
 8784 026632  004737  046706                   CALL    MAPKERNAL               ;MAP TO KERNEL SPACE
 8785 026636                                    LET R1 := #100000               ;SET UP TEST ADDRESS
 8786 026642  004737  040142                   CALL MTP045                     ;PERFORM SYNDROMES TO CSR ON DOUBLE BIT ERROR
 8787 026646  004737  046774                   CALL UNMAP                      ;REMAP KERNEL SPACE
 8788 026652  000207                           RETURN                          ;
 8789 026654                            MT0046: SUBTST  <<MT0046        SETUP CHECK SINGLE BIT ERRORS WITH ECC DISABLED TET>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       MT0046  SETUP CHECK SINGLE BIT ERRORS WITH ECC DISABLED TET
                                        ;*****************************************************************************
                                                IF PMEMFLG IS FALSE THEN $RETURN ;EXIT IF NOT MS11-P
 8790 026654
 8791 026664  012737  000046  002274           MOV     #46,REALPAT             ;SET UP TEST NUMBER FOR TYPEOUT AND DISPLAY
 8792 026672  004737  046706                   CALL    MAPKERNAL               ;MAP TO KERNEL SPACE
 8793 026676                                    LET R1 := #100000               ;SET UP TEST ADDRESS
 8794 026702  004737  040330                   CALL MTP046                     ;PERFORM TRAPS DECTECTED ON SBE WITH ECC DISABLED TE
 8795 026706  004737  046774                   CALL UNMAP                      ;REMAP KERNEL SPACE
 8796 026712  000207                           RETURN
 8797 026714                            MT0047: SUBTST  <<MT0047        SETUP NO CSR UPDATE ON SBE WITH EXSISTING DBE TEST>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       MT0047  SETUP NO CSR UPDATE ON SBE WITH EXSISTING DBE TEST
                                        ;*****************************************************************************
                                                IF PMEMFLG IS FALSE THEN $RETURN ;EXIT IF NOT MS11-P
 8798 026714
 8799 026724  012737  000047  002274           MOV     #47,REALPAT             ;SET UP TEST NUMBER FOR TYPEOUT AND DISPLAY
 8800 026732  004737  046706                   CALL    MAPKERNAL               ;MAP TO KERNEL SPACE
 8801 026736                                    LET R1 := #100000               ;SET UP TEST ADDRESS
 8802 026742                                    LET R2 := #120000               ;  "  "  SECOND  TEST ADDRESS
 8803 026746  004737  040670                   CALL MTP047                     ;PERFORM NO UPDATE TO CSR ON SBE WITH DBE
 8804 026752  004737  046774                   CALL UNMAP                      ;REMAP KERNEL SPACE
 8805 026756  000207                           RETURN
```

```
8808 026760                          MT0999: SUBTST  <<MT0999        SETUP NULL TEST>>
                                     ;*****************************************************************************
                                     ;*SUBTEST       MT0999  SETUP NULL TEST
                                     ;*****************************************************************************
8809 026760  005037  002274                 CLR     REALPAT
8810 026764                                 SET     NULLFLAG
8811 026772  000207                         RETURN
8812
8813 026774                          KAMITEST:SUBTST <<CHECK FOR KAMIKAZE MODE>>
                                     ;*****************************************************************************
                                     ;*SUBTEST       CHECK FOR KAMIKAZE MODE
                                     ;*****************************************************************************
8814 026774                                 IF KAMIKAZE IS TRUE OR ACTFLAG IS TRUE OR APTFLAG IS TRUE
8815 027016                                   $RETURN NOERROR                ;RUN THE TEST
8816 027022                                 ELSE
8817 027024                                   $RETURN ERROR                  ;DON'T RUN THE TEST
8818 027030                                 END ;OF IF KAMIKAZE
```

```
 8821 027030                            SUPD01: SUBTST  <<SUBR  EXECUTE PATTERN IN SUPERVISOR>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       SUBR    EXECUTE PATTERN IN SUPERVISOR
                                        ;*****************************************************************************
 8822 027030                                    MAP     BANK                    ;MAP SUPERVISOR SPACE (TEST AREA) TO BANK
 8823 027044  004737  060450            SUPD02: CALL    GETDIS
 8824 027050                                    PUSH    $LPERR,$LPADR
 8825 027060  010037  002156                    MOV     R0,SUPDR0
 8826 027064  012700  002160                    MOV     #SUPDR1,R0
 8827 027070  010120                            MOV     R1,(R0)+
 8828 027072  010220                            MOV     R2,(R0)+
 8829 027074  010320                            MOV     R3,(R0)+
 8830 027076  010420                            MOV     R4,(R0)+
 8831 027100  010520                            MOV     R5,(R0)+
 8832 027102  010620                            MOV     SP,(R0)+
 8833 027104  013700  002156                    MOV     SUPDR0,R0
 8834 027110  012737  027124  002606            MOV     #TAG4$,$LPADR
 8835 027116  013737  002606  002610            MOV     $LPADR,$LPERR
 8836 027124  012700  002174            TAG4$:  MOV     #SUPDR6+2,R0
 8837 027130  014006                            MOV     -(R0),SP
 8838 027132  014005                            MOV     -(R0),R5
 8839 027134  014004                            MOV     -(R0),R4
 8840 027136  014003                            MOV     -(R0),R3
 8841 027140  014002                            MOV     -(R0),R2
 8842 027142  014001                            MOV     -(R0),R1
 8843 027144  014000                            MOV     -(R0),R0
 8844 027146                                    SUPERVISOR              ;ENTER SUPERVISOR MODE
 8845 027154  012706  000740                    MOV     #SUPSTK,SSP
 8846 027160  104424                            CACHOFF                 ;TURN CACHE OFF
 8847 027162  004737  177640                    CALL    FASTCITY                ;CALL TO THE USER INSTRUCTION PAR'S
 8848 027166  104423                            CACHON                  ;TURN CACHE ON
 8849 027170  104417                            KERNEL                  ;ENTER KERNEL MODE
 8850 027172  000004                            SCOPE
 8851 027174                                    POP     $LPADR,$LPERR
 8852 027204  000207                            RETURN
```

```
8855 027206                                SUPD03: MAP     BANK                    ;MAP SUPERVISOR SPACE (TEST AREA) TO BANK
8856 027222  004737  060450                SUPD04: CALL    GETDIS
8857 027226                                        PUSH    $LPERR,$LPADR
8858 027236  010037  002156                        MOV     R0,SUPDR0
8859 027242  012700  002160                        MOV     #SUPDR1,R0
8860 027246  010120                                MOV     R1,(R0)+
8861 027250  010220                                MOV     R2,(R0)+
8862 027252  010320                                MOV     R3,(R0)+
8863 027254  010420                                MOV     R4,(R0)+
8864 027256  010520                                MOV     R5,(R0)+
8865 027260  010620                                MOV     SP,(R0)+
8866 027262  013700  002156                        MOV     SUPDR0,R0
8867 027266  012737  027302  002606                MOV     #TBG4$,$LPADR
8868 027274  013737  002606  002610                MOV     $LPADR,$LPERR
8869 027302  012700  002174              TBG4$:    MOV     #SUPDR6+2,R0
8870 027306  014006                                MOV     -(R0),SP
8871 027310  014005                                MOV     -(R0),R5
8872 027312  014004                                MOV     -(R0),R4
8873 027314  014003                                MOV     -(R0),R3
8874 027316  014002                                MOV     -(R0),R2
8875 027320  014001                                MOV     -(R0),R1
8876 027322  014000                                MOV     -(R0),R0
8877 027324                                        TESTAREA                        ;ENTER SUPERVISOR MODE
8878 027332  005737  002452                        TST     NOSUPER
8879 027336  001403                                BEQ     1$
8880 027340  012706  000700                        MOV     #USESTK,USP
8881 027344  000402                                BR      2$
8882 027346  012706  000740              1$:       MOV     #SUPSTK,SSP
8883 027352  104424                      2$:       CACHOFF                         ;TURN CACHE OFF
8884 027354  004777  152700                        CALL    @SUPDOADD
8885 027360  104423                                CACHON                          ;TURN CACHE ON
8886 027362  104417                                KERNEL                          ;ENTER KERNEL MODE
8887 027364  000004                                SCOPE
8888 027366                                        POP     $LPADR,$LPERR
8889 027376  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)
                ;*******************************************************************************
8902 027400     MTP000: SUBTST  <<MTP000        BASIC DATA TEST>>
                ;*******************************************************************************
                ;*SUBTEST       MTP000  BASIC DATA TEST
                ;*******************************************************************************
8903 027400  010220 1$:     MOV     R2,(R0)+        ;V177640
8904 027402  077102         SOB     R1,MTP000       ;V177642
8905 027404  000240         NOP                     ;V177644
8906 027406  012401 2$:     MOV     (R4)+,R1        ;V177646
8907 027410  020102         CMP     R1,R2           ;V177650
8908 027412  001402         BEQ     3$              ;V177652
8909 027414  104430         PERR02                  ;V177654
8910 027416  000240         NOP                     ;V177656
8911 027420  077306 3$:     SOB     R3,2$           ;V177660
8912 027422  000207         RETURN                  ;V177662
8913 027424     MTP001: SUBTST  <<MTP001        ADDRESS TEST>>
                ;*******************************************************************************
                ;*SUBTEST       MTP001  ADDRESS TEST
                ;*******************************************************************************
8914 027424  010220 3$:     MOV     R2,(R0)+        ;V177640
8915 027426  062702  000002    ADD     #2,R2           ;V177642
8916 027432  077104         SOB     R1,3$           ;V177646
8917 027434  000240         NOP                     ;V177650
8918 027436  012400 1$:     MOV     (R4)+,R0        ;V177652
8919 027440  020005         CMP     R0,R5           ;V177654
8920 027442  001401         BEQ     2$              ;V177656
8921 027444  104427         PERR01                  ;V177660
8922 027446  062705  000002 2$:     ADD     #2,R5           ;V177662
8923 027452  077307         SOB     R3,1$           ;V177666
8924 027454  000207         RETURN                  ;V177672
8925 027456     MTP002: SUBTST  <<MTP002        COMPLEMENT ADDRESS TEST (WRITE DOWN, READ UP)>>
                ;*******************************************************************************
                ;*SUBTEST       MTP002  COMPLEMENT ADDRESS TEST (WRITE DOWN, READ UP)
                ;*******************************************************************************
8926 027456  010540 3$:     MOV     R5,-(R0)        ;V177640
8927 027460  062705  000002    ADD     #2,R5           ;V177642
8928 027464  077104         SOB     R1,3$           ;V177646
8929 027466  000240         NOP                     ;V177650
8930 027470  162702  000002 1$:     SUB     #2,R2           ;V177652
8931 027474  012401         MOV     (R4)+,R1        ;V177656
8932 027476  020102         CMP     R1,R2           ;V177660
8933 027500  001401         BEQ     2$              ;V177662
8934 027502  104430         PERR02                  ;V177664
8935 027504  077307 2$:     SOB     R3,1$           ;V177666
8936 027506  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
027510  010421                 1$:     MOV     R4,(R1)+        ;V177640
027512  010421                         MOV     R4,(R1)+        ;V177642
027514  077203                         SOB     R2,1$           ;V177644
027516  005104                         COM     R4              ;V177646
027520  052704                         BIS     (PC)+,R4        ;V177650
027522  000401                 WARN2:  401                     ;V177652 WARNING LOCATION IS MODIFIED BEFORE LOADING
027524  012702  000004                 MOV     #4,R2           ;V177654
027530  077511                         SOB     R5,1$           ;V177660
027532  005104                         COM     R4              ;V177662
027534  052704                         BIS     (PC)+,R4        ;V177664
027536  000401                 WARN3:  401                     ;V177666 WARNING LOCATION IS MODIFIED BEFORE LOADING
027540  012705  000100                 MOV     #64.,R5         ;V177670
027544  077317                         SOB     R3,1$           ;V177674
027546  000207                         RETURN                  ;V177676
                                        .DSABL  LSB

027550                          MTPB03: SUBTST  <<MTPB03        3 XOR 9 WORST CASE NOISE TEST (READ)>>
                                ;*****************************************************************************
                                ;*SUBTEST        MTPB03  3 XOR 9 WORST CASE NOISE TEST (READ)
                                ;*****************************************************************************
                                        .ENABL  LSB
027550  000137  027610         1$:     JMP     @#MTPC03        ;V177640        GO TO V172360
027554  077203                         SOB     R2,1$           ;V177644
027556  005104                         COM     R4              ;V177646
027560  052704                         BIS     (PC)+,R4        ;V177650
027562  000401                 WARN4:  401                     ;V177652 WARNINC LOCATION IS MODIFIED BEFORE LOADING
027564  012702  000004                 MOV     #4,R2           ;V177654
027570  077511                         SOB     R5,1$           ;V177660
027572  005104                         COM     R4              ;V177662
027574  052704                         BIS     (PC)+,R4        ;V177664
027576  000401                 WARN5:  401                     ;V177666 WARNING LOCATION IS MODIFIED BEFORE LOADING
027600  012705  000100                 MOV     #64.,R5         ;V177670
027604  077317                         SOB     R3,1$           ;V177674
027606  000207                         RETURN                  ;V177676
                                        .DSABL  LSB
```

```
                                        MTPC03: SUBTST  <<MTPC03        TEST DATA SUBPROGRAM>>
                                        ;*****************************************************************************
                                        ;*SUBTEST        MTPC03  TEST DATA SUBPROGRAM
                                        ;*****************************************************************************
                                        ;
 8980 027610                                    CMP     R4,(R1)+        ;V172360
 8981 027610  020421                            BEQ     1$              ;V172362
 8982 027612  001401                            PERR03                  ;V172364
 8983 027614  104431                    1$:     COM     -(R1)           ;V172366
 8984 027616  005141                            COM     (R1)            ;V172370
 8985 027620  005111                            JMP     @#MTPD03        ;V172372        GO TO V172260
 8986 027622  000137  027626
 8987
 8988 027626                            MTPD03: SUBTST  <<MTPD03        TEST DATA SUBSUBPROGRAM>>
                                        ;*****************************************************************************
                                        ;*SUBTEST        MTPD03  TEST DATA SUBSUBPROGRAM
                                        ;*****************************************************************************
                                        ;
 8989 027626  020421                            CMP     R4,(R1)+        ;V172260
 8990 027630  001401                            BEQ     1$              ;V172262
 8991 027632  104431                            PERR03                  ;V172264
 8992 027634  005127                    1$:     COM     (PC)+           ;V172266
 8993 027636  000000                            0                       ;V172270
 8994 027640  001363                            BNE     MTPC03          ;V172272        GO TO V172360
 8995 027642  000137  027554                    JMP     @#MTPB03+4      ;V172274        GO TO V177644
```

```
 8998 027646                            MTPA04: SUBTST  <<MTPA04         ROTATING ZEROS TEST>>
                                        ;*****************************************************************************
                                        ;*SUBTEST        MTPA04  ROTATING ZEROS TEST
                                        ;*****************************************************************************
 8999 027646  012705  000010            1$:     MOV     #8.,R5           ;V177640
 9000 027652  010504                            MOV     R5,R4            ;V177644
 9001 027654  000241                            CLC                      ;V177646
 9002 027656  000137  027702                    JMP     @#MTPB04         ;V177650
 9003 027662  016004  177776                    MOV     -2(R0),R4        ;V177654
 9004 027666  103402                            BCS     2$               ;V177660
 9005 027670  020204                            CMP     R2,R4            ;V177662
 9006 027672  001401                            BEQ     3$               ;V177664
 9007 027674  104432                    2$:     PERR04                   ;V177666
 9008 027676  077115                    3$:     SOB     R1,1$            ;V177670
 9009 027700  000207                            RETURN                   ;V177672
 9010
 9011 027702                            MTPB04: SUBTST  <<MTPB04         SUBR    ROTATING BIT>>
                                        ;*****************************************************************************
                                        ;*SUBTEST        MTPB04  SUBR    ROTATING BIT
                                        ;*****************************************************************************
 9012 027702  106110                    1$:     ROLB    (R0)             ;V172360
 9013 027704  077502                            SOB     R5,1$            ;V172362
 9014 027706  106120                            ROLB    (R0)+            ;V172364
 9015 027710  106110                    2$:     ROLB    (R0)             ;V172366
 9016 027712  077402                            SOB     R4,2$            ;V172370
 9017 027714  106120                            ROLB    (R0)+            ;V172372
 9018 027716  000137  027662                    JMP     @#MTPA04+14      ;V172374
 9019
 9020 027722                            MTP005: SUBTST  <<MTP005         ROTATION ONES TEST>>
                                        ;*****************************************************************************
                                        ;*SUBTEST        MTP005  ROTATION ONES TEST
                                        ;*****************************************************************************
 9021 027722  012705  0C0010            1$:     MOV     #8.,R5           ;V177640
 9022 027726  010504                            MOV     R5,R4            ;V177644
 9023 027730  000261                            SEC                      ;V177646
 9024 027732  000137  027702                    JMP     @#MTPB04         ;V177650
 9025 027736  016004  177776                    MOV     -2(R0),R4        ;V177654
 9026 027742  103002                            BCC     2$               ;V177660 IF THIS HAPPENS THE GOOD & BAD MATCH
 9027 027744  020204                            CMP     R2,R4            ;V177662
 9028 027746  001401                            BEQ     3$               ;V177664
 9029 027750  104432                    2$:     PERR04                   ;V177666
 9030 027752  077115                    3$:     SOB     R1,1$            ;V177670
 9031 027754  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.
 9037 027756  012737  000001  002240          MOV     #1,DATBUF       ;SET THE FIRST TEST BIT
 9038 027764  005037  002242                  CLR     DATBUF+2        ;CLEAR 2ND WORD
 9039 027770  013771  002240  000000  1$:     MOV     DATBUF,@(R1)    ;WRITE TEST WORD 1
 9040 027776  013771  002242  000002          MOV     DATBUF+2,@2(R1) ;AND TEST WORD 2
 9041 030004  017102  000000                  MOV     @(R1),R2
 9042 030010  023702  002240                  CMP     DATBUF,R2       ;NOW READ THEM
 9043 030014  001401                          BEQ     2$              ;BR IF FIRST 16 OK
 9044 030016  104433                          PERR07                  ;ERROR TRAP
 9045
 9046 030020  017102  000002          2$:     MOV     @2(R1),R2
 9047 030024  023702  002242                  CMP     DATBUF+2,R2     ;NOW READ SECOND WORD
 9048 030030  001401                          BEQ     3$              ;BR IF OK
 9049 030032  104434                          PERR10                  ;ERROR TRAP
 9050
 9051 030034  005737  002242          3$:     TST     DATBUF+2        ;HAS LAST BIT BEEN TESTED ?
 9052 030040  100405                          BMI     4$              ;MINUS MEANS BIT 31
 9053 030042                                  DLEFT   DATBUF          ;NO, SHIFT TEST BIT LEFT
 9054 030052  000746                          BR      1$              ;GO WRITE NEW TEST DATA
 9055                                                                 ;NOW GOING TO SHIFT A 0 IN DATA DIRECTION
 9056 030054  012737  177776  002240  4$:     MOV     #177776,DATBUF  ;PUT A 0 IN BIT 0
 9057 030062  012737  177777  002242          MOV     #-1, DATBUF+2   ;AND 1'S IN ALL OTHERS
 9058 030070  013771  002240  000000  5$:     MOV     DATBUF,@(R1)    ;WRITE THE DATA
 9059 030076  013771  002242  000002          MOV     DATBUF+2,@2(R1) ;2 WORDS WORTH
 9060 030104  017102  000000                  MOV     @(R1),R2
 9061 030110  023702  002240                  CMP     DATBUF,R2       ;NOW READ FIRST WORD
 9062 030114  001401                          BEQ     6$              ;BR IF OK
 9063 030116  104433                          PERR07
 9064
 9065 030120  017102  000002          6$:     MOV     @2(R1),R2
 9066 030124  023702  002242                  CMP     DATBUF+2,R2     ;NOW, READ SECOND WORD
 9067 030130  001401                          BEQ     7$              ;BR IF OK
 9068 030132  104434                          PERR10
 9069
 9070 030134  005737  002242          7$:     TST     DATBUF+2        ;TESTED BIT 31 YET?
 9071 030140  100005                          BPL     8$              ;BR IF YES, WE'RE DONE
 9072 030142                                  DLEFT   DATBUF
 9073 030152  000746                          BR      5$              ;KEEP GOING
 9074 030154  000207          8$:     RETURN
```

```
 9077 030156                            MTP007: SUBTST  <<MTP007        ADDRESS BIT TEST>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       MTP007  ADDRESS BIT TEST
                                        ;*****************************************************************************
 9078                                   ;       THIS TEST CHECKS TO SEE THAT EACH ADDRESS
 9079                                   ;       BIT IN EACH 16K BANK CAN BE ASSERTED UNIQUELY.
 9080                                   ;       IT CHECKS FOR ADDRESS BITS THAT MAY BE STUCK
 9081                                   ;       HIGH, STUCK LOW OR STUCK TOGETHER.
 9082 030156  111100                            MOVB    (R1),R0
 9083 030160  105700                            TSTB    R0              ;READ AND COMPARE FOR ZEROS
 9084 030162  001401                            BEQ     1$              ;BR IF OK
 9085 030164  104435                            PERR11
 9086
 9087 030166  105111                    1$:     COMB    (R1)            ;COMPLEMENT THE BYTE
 9088 030170  111100                            MOVB    (R1),R0
 9089 030172  105700                            TSTB    R0              ;READ FOR NON ZEROS
 9090 030174  001001                            BNE     2$              ;BR IF OK
 9091 030176  104436                            PERR12
 9092
 9093 030200  040201                    2$:     BIC     R2,R1           ;MASK OFF THE ASSERTED BIT
 9094 030202  006302                            ASL     R2              ;SHIFT R2 FOR NEXT BIT
 9095 030204  050201                            BIS     R2,R1           ;SET THE NEW BIT INTO R1
 9096 030206  011100                            MOV     (R1),R0
 9097 030210  005700                            TST     R0              ;READ THE NEW ADDRESS
 9098 030212  001401                            BEQ     3$              ;READ FOR ZEROS
 9099 030214  104437                            PERR13
 9100
 9101 030216  005111                    3$:     COM     (R1)            ;COMPL THE WORD
 9102 030220  011100                            MOV     (R1),R0
 9103 030222  005700                            TST     R0              ;READ IT AGAIN
 9104 030224  001001                            BNE     4$
 9105 030226  104440                            PERR14
 9106
 9107 030230  022702  100000            4$:     CMP     #100000,R2
 9108 030234  001407                            BEQ     5$
 9109 030236  022702  010000                    CMP     #10000,R2       ;CHECK FOR MSB IN 4K BANK
 9110 030242  001356                            BNE     2$              ;NOT LAST BIT, BRANCH
 9111 030244  006302                            ASL     R2
 9112 030246  012701  160000                    MOV     #160000,R1
 9113 030252  000752                            BR      2$
 9114 030254  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
030256  010402                                MOV     R4,R2           ;R4 HAS LOWEST ADDRESS
030260  010403                                MOV     R4,R3           ;PUT IT IN R3 ALSO
030262  062702  000004                        ADD     #4,R2           ;POINT R2 TO LAST BYTE +1
030266  012713  177777                        MOV     #-1,(R3)        ;WRITE ALL ONES IN
030272  012763  177777  000002                MOV     #-1,2(R3)       ;THE 4 TEST BYTES
030300  105013                        1$:     CLRB    (R3)            ;CLEAR A BYTE
030302  010401                                MOV     R4,R1           ;INITIALIZE R1 FOR EACH PASS
030304  020201                        2$:     CMP     R2,R1           ;IF EQUAL, JUST READ LAST BYTE
030306  001420                                BEQ     6$              ;BR IF EQUAL
030310  020301                                CMP     R3,R1           ;IS THIS THE BYTE OF ZEROS
030312  001007                                BNE     4$              ;BR IF NOT
030314  111100                                MOVB    (R1),R0
                                                ;WARNING IF YOU OPTOMIZE CHANGE THE PCBUMP FOR THIS ERROR INCASE OF TRAPS
030316  022700  000000                        CMP     #0,R0           ;IT IS, COMPARE FOR ZEROS
030322  001401                                BEQ     3$
030324  104435                                PERR11

030326  005201                        3$:     INC     R1              ;NEXT BYTE
030330  000765                                BR      2$              ;RETURN
030332  111100                        4$:     MOVB    (R1),R0
030334  122700  177777                        CMPB    #-1,R0          ;ITS NOT THE BYTE OF 0'S, READ 1'S
030340  001401                                BEQ     5$
030342  104436                                PERR12

030344  005201                        5$:     INC     R1              ;MOVE TO NEXT BYTE
030346  000756                                BR      2$
030350  112713  177777                6$:     MOVB    #-1,(R3)        ;RESTORE 1'S TO BYTE JUST TESTED
030354  005203                                INC     R3              ;INC TO NEXT BYTE
030356  020302                                CMP     R3,R2           ;WAS THAT JUST THE LAST ONE?
030360  001347                                BNE     1$              ;BR IF NO
030362  000207                                RETURN
```

```
 9153 030364                            MTP011: SUBTST  <<MTP011        SINGLE BIT ERROR TEST>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       MTP011  SINGLE BIT ERROR TEST
                                        ;*****************************************************************************
 9154                                           ;(1)    CREATE A SINGLE BIT ERROR
 9155                                           ;
 9156                                           ;(2)    READ BACK SBE UNCORRECTED (WITH ECC DISABLE)
 9157                                           ;
 9158                                           ;(3)    ENABLE ECC & READ CORRECTED DATA
 9159                                           ;
 9160                                           ;(4)    CHECK THAT THE SBE FLAG WAS SET FROM THE LAST READ
 9161                                           ;
 9162                                           ;(5)    DO (1-4) FOR DATA CONSISTING OF 1 BIT SET IN EACH OF 32
 9163                                           ;       POSITIONS OF A DOUBLE WORD
 9164                                           ;       THEN DO IT AGAIN FOR 1 BIT CLEARED IN EACH OF 32 POSITIONS OF
 9165                                           ;       A DOUBLE WORD
 9166                                           ;       IE (64 TIMES)
 9167                                           ;
 9168                                           ;(6)    DO (1-5) FOR A SBE IN EACH OF 32 BIT POSITIONS
 9169                                           ;       IE (RUN TEST 64 * 32 = 2048 TIMES)
 9170 030364  104503                           CLR1CSR                 ;CLEAR 1 SELECTED CSR
 9171 030366  005737  014004                   TST     PHEBE           ;TEST SPECIAL CASE INDICATOR
 9172 030372  001407                           BEQ     MTLA11          ;BRANCH IF NOT SET
 9173 030374  013702  172246                   MOV     SIPAR3,R2       ;SAVE CONTENTS OF SIPAR #3
 9174 030400  013737  172252  172246           MOV     SIPAR5,@#SIPAR3 ;COPY CONTENTS OF #5 INTO #3
 9175 030406  010237  172252                   MOV     R2,@#SIPAR5     ;COPY CONTENTS OF #3 INTO #5
 9176                                           ;BIG LOOP
 9177 030412  012737  000001  002240  MTLA11:  MOV     #1,DATBUF       ;INITIAL DATA
 9178 030420  005037  002242                   CLR     DATBUF+2        ;32 BITS WORTH
 9179                                           ;MEDIUM LOOP
 9180 030424  012737  000001  002250  MTLB11:  MOV     #1,SBEMSK       ;INITIAL ERROR MASK
 9181 030432  005037  002252                   CLR     SBEMSK+2        ;32 BITS WORTH
 9182                                           ;LITTLE LOOP
 9183 030436  013737  002240  002244  MTLC11:  MOV     DATBUF,TSTDAT   ;
 9184 030444  013737  002242  002246           MOV     DATBUF+2,TSTDAT+2;TO SAVE ORIG DATA
 9185 030452  105737  002262                   TSTB    PASFLG  ;COMP DATA ON SECOND PASSONLY
 9186 030456  001404                           BEQ     4$              ;BR IF FIRST PASS
 9187 030460  005137  002244                   COM     TSTDAT          ;SECOND PASS, COMP BOTH WORDS
 9188 030464  005137  002246                   COM     TSTDAT+2
 9189 030470  013702  002244           4$:     MOV     TSTDAT,R2
 9190 030474  013703  002246                   MOV     TSTDAT+2,R3
 9191 030500  012737  002244  002306           MOV     #TSTDAT,SOURCE  ;SET UP ADDRESS FOR CHKGEN
 9192 030506  004737  044366                   CALL    CHKGEN          ;GEN CHECKBITS ON TSTDAT
 9193                                           ;*********************************
 9194                                           ;** CREATE A SINGLE BIT ERROR **
 9195                                           ;*********************************
 9196 030512  013701  002250                   MOV     SBEMSK,R1
 9197 030516  074137  002244                   XOR     R1,TSTDAT
 9198 030522  013701  002252                   MOV     SBEMSK+2,R1
 9199 030526  074137  002246                   XOR     R1,TSTDAT+2
 9200 030532  013701  002406           MTLD11:  MOV     TESTADD,R1      ;FIRST TEST ADDRESS
 9201 030536  013705  002410                   MOV     TESTADD+2,R5    ;SECOND TEST ADDRESS
 9202 030542  104471                           ECC1DIS                 ;DISABLE ECC ON 1 SELECTED CSR
 9203 030544  013711  002244                   MOV     TSTDAT,(R1)     ;WRITE FIRST 16 BITS
 9204 030550  104475                           CB1CSR                  ;WRITE GENERATED CHECKBITS IN 1 SELECTED CSR
 9205 030552  013715  002246                   MOV     TSTDAT+2,(R5)   ;WRITE SECOND 16 BITS AND
 9206                                                                   ;CHECK BITS. WE NOW HAVE CHECKBITS
```

```
                                                                ;GENERATED ON DATBUF AND DATA WITH
                                                                ;ONE BIT IN ERROR (AS PER SBEMSK).
                                        ECC1DIS                 ;DISABLE ECC ON 1 SELECTED CSR
030556  104471                          MOV     (R1),R0
030560  011100
030562  020037  002244                  CMP     R0,TSTDAT       ;READ THE LOW WORD (UNCORRECTED)
030566  001403                          BEQ     6$              ;BR IF OK
030570  010137  002032                  MOV     R1,ADDRESS
030574  104455                          PERR31

030576  011500                  6$:     MOV     (R5),R0
030600  020037  002246                  CMP     R0,TSTDAT+2     ;READ THE HIGH WORD (UNCORRECTED)
030604  001403                          BEQ     7$              ;BR IF OK
030606  010537  002032                  MOV     R5,ADDRESS
030612  104455                          PERR31

030614                          7$:     IF KFLAG IS FALSE
030622  104426                            READCSR
030624                                    IF #BIT4 OFF.IN CSR OR #BIT15 OFF.IN CSR
030644  104045                              ERROR       +45
030646                                    END; OF IF #BIT4
030646                                  END; OF IF KFLAG
030646  005737  014004                  TST     PHEBE
030652  001001                          BNE     17$
030654  104512                          ERRGEN
030656  104503                  17$:    CLR1CSR                 ;CLEAR 1 SELECTED CSR
030660  011100                          MOV     (R1),R0
030662  020002                          CMP     R0,R2           ;SEE IF ITS BEEN CORRECTED
030664  001401                          BEQ     8$              ;IT SHOULD HAVE BEEN
030666  104456                          PERR32

030670  104510                  8$:     TSTREAD                 ;TEST LOC (R1) & TST FOR SBE (WITHOUT FETCHES)
030672  103411                          BCS     9$              ;BR IF IT IS SET
030674                                  SET     HEADER          ;ENABLE PRINTING OF ERROR HEADER INFO
030702  010137  002032                  MOV     R1,ADDRESS
030706  104460                          PERR34
030710                                  SET     HEADER          ;ENABLE PRINTING OF ERROR HEADER INFO

030716  104503                  9$:     CLR1CSR                 ;CLEAR 1 SELECTED CSR
030720  011500                          MOV     (R5),R0
030722  020003                          CMP     R0,R3           ;SEE IF ITS BEEN CORRECTED
030724  001401                          BEQ     10$             ;BR IF OK
030726  104456                          PERR32

030730  104510                  10$:    TSTREAD                 ;TEST LOC (R1) & TST FOR SBE (WITHOUT FETCHES)
030732  103411                          BCS     11$             ;BR IF YES
030734                                  SET     HEADER          ;ENABLE PRINTING OF ERROR HEADER INFO
030742  010137  002032                  MOV     R1,ADDRESS
030746  104460                          PERR34
030750                                  SET     HEADER          ;ENABLE PRINTING OF ERROR HEADER INFO
030756  104512                  11$:    ERRGEN                  ;TEST ERROR ADDRESS
030760  105737  002262                  TSTB    PASFLG
030764  100452                          BMI     15$
030766  005737  002252                  TST     SBEMSK+2        ;TEST FOR LAST MASK BIT
030772  100405                          BMI     12$             ;MINUS MEANS BIT 31
030774                                  DLEFT   SBEMSK
031004  000614                          BR      MTLC11
031006                          12$:    IF #SW11 SET.IN @SWR THEN GOTO 13$
```

```
                                                    IF QVFLAG IS TRUE THEN GOTO 13$
 9264 031016
 9265 031024  005737  002242                        TST     DATBUF+2        ;LAST DATA BIT ?
 9266 031030  100406                                BMI     13$             ;WHICH IS BIT 31
 9267 031032                                        DLEFT   DATBUF
 9268 031042  000137  030424                        JMP     MTLB11
 9269 031046  105737  002262            13$:      TSTB    PASFLG  ;FIRST OR SECOND PASS ?
 9270 031052  001004                                BNE     14$             ;NON ZERO MEANS WE'RE DONE
 9271 031054  105237  002262                        INCB    PASFLG  ;NOT DONE, GO DO SECOND PASS
 9272 031060  000137  030412                        JMP     MTLA11
 9273 031064  052737  000200  002262  14$:      BIS     #BIT7,PASFLG
 9274 031072  005002                                CLR     R2
 9275 031074  005003                                CLR     R3
 9276 031076  005037  002244                        CLR     TSTDAT
 9277 031102  005037  002246                        CLR     TSTDAT+2
 9278 031106  012704  000040                        MOV     #40,R4
 9279 031112  012737  003740  002310  15$:      MOV     #3740,CHECK
 9280 031120  074437  002310                        XOR     R4,CHECK
 9281 031124  006304                                ASL     R4
 9282 031126  032704  020000                        BIT     #BIT13,R4
 9283 031132  001002                                BNE     16$
 9284 031134  000137  030532                        JMP     MTLD11
 9285                                               ;CLEAR OUT ANY DBE'S OR SBE'S
 9286 031140  104471                      16$:      ECC1DIS                 ;DISABLE ECC ON 1 SELECTED CSR
 9287 031142  013701  002406                        MOV     TESTADD,R1
 9288 031146  013705  002410                        MOV     TESTADD+2,R5
 9289 031152                                        CLEAR   (R1),(R5)
 9290 031156  104503                                CLR1CSR                 ;CLEAR 1 SELECTED CSR
 9291 031160  000207                                RETURN
```

```
 9294 031162                                MTP012: SUBTST  <<MTP012        WRITE BYTE CLEARS SBE TEST>>
                                            ;*****************************************************************************
                                            ;*SUBTEST       MTP012  WRITE BYTE CLEARS SBE TEST
                                            ;*****************************************************************************
 9295                                                       ;SINGLE BIT ERROR TEST TO INSURE THAT A WRITE
 9296                                                       ;BYTE CLEARS SINGLE BIT ERRORS.
 9297 031162  104503                              CLR1CSR                   ;CLEAR 1 SELECTED CSR
 9298 031164  012737  000001  002240              MOV     #1,DATBUF         ;INITIAL DATA
 9299 031172  005037  002242                      CLR     DATBUF+2          ;32 BITS WORTH
 9300 031176  012737  000001  002250  1$:         MOV     #1,SBEMSK         ;INITIAL ERROR MASK
 9301 031204  005037  002252                      CLR     SBEMSK+2          ;32 BITS WORTH
 9302 031210  013737  002240  002244  2$:         MOV     DATBUF,TSTDAT     ;SAVE ORIGINAL DATA
 9303 031216  013737  002242  002246              MOV     DATBUF+2,TSTDAT+2;BOTH WORDS
 9304 031224  012737  002244  002306              MOV     #TSTDAT,SOURCE    ;NEED ADDRESS FOR CHKGEN
 9305 031232  004737  044366                      CALL    CHKGEN            ;GENERATE CHECK BITS
 9306 031236  013701  002250                      MOV     SBEMSK,R1
 9307 031242  074137  002244                      XOR     R1,TSTDAT
 9308 031246  013701  002252                      MOV     SBEMSK+2,R1
 9309 031252  074137  002246                      XOR     R1,TSTDAT+2
 9310 031256  013704  002406                      MOV     TESTADD,R4        ;FIRST TEST ADDRESS
 9311 031262  010401                              MOV     R4,R1             ;PUT IT IN R1 ALSO
 9312 031264  104471                              ECC1DIS                   ;DISABLE ECC ON 1 SELECTED CSR
 9313 031266  013711  002244                      MOV     TSTDAT,(R1)       ;WRITE 16 BITS
 9314 031272  104475                              CB1CSR                    ;WRITE GENERATED CHECKBITS IN 1 SELECTED CSR
 9315 031274  060500                              ADD     R5,R1             ;INDEX UP TO SECOND WORD
 9316 031276  013711  002246                      MOV     TSTDAT+2,(R1)     ;WRITE HIGH WORD+CHECKBITS
 9317 031302  104503                              CLR1CSR                   ;CLEAR 1 SELECTED CSR
 9318                                                                       ;IT'S DANGEROUS IF WE DON'T
 9319 031304  012702  002250                      MOV     #SBEMSK,R2        ;ADDRESS OF ERROR MASK
 9320 031310  160501                              SUB     R5,R1             ;RETURN TO FIRST WORD
 9321 031312  112711  177777              3$:     MOVB    #-1,(R1)          ;WRITE A BYTE OF 1'S
 9322 031316  005737  002524                      TST     KFLAG             ;IS THIS MF11S-K
 9323 031322  001403                              BEQ     4$                ;BRANCH IF NOT - IT'S MS11-M
      03 324  132712  177777                      BITB    #-1,(R2)          ;DID THIS BYTE HAVE THE BAD BIT IN IT?
 9325 031330  001420                              BEQ     6$                ;NO - BRANCH
 9326 031332  104500                      4$:     TSTREAD                   ;TEST LOC (R1) & TST FOR SBE (WITHOUT FETCHES)
 9327 031334  103001                              BCC     5$                ;NO - SKIP
 9328 031336                                      SET     HEADER            ;ENABLE PRINTING OF ERROR HEADER INFO
 9329 031344  010137  002C32                      MOV     R1,ADDRESS
 9330 031350  104017                              ERROR   +17
 9331 031352                                      SET     HEADER            ;ENABLE PRINTING OF ERROR HEADER INFO
 9332
 9333 031360  111100                      5$:     MOVB    (R1),R0
 9334 031362  122700  177777                      CMPB    #-1,R0            ;CHECK DATA
 9335 031366  001414                              BEQ     7$                ;BR IF OK
 9336 031370  104457                              PERR33
 9337
 9338 031372  104510                      6$:     TSTREAD                   ;TEST LOC (R1) & TST FOR SBE (WITHOUT FETCHES)
 9339                                                                       ;READ THE BYTE
 9340                                                                       ;SBE ERROR BIT ONLY SET ?
 9341 031374  103771                              BCS     5$                ;SHOULD BE SET, BR IF OK
 9342 031376                                      SET     HEADER            ;ENABLE PRINTING OF ERROR HEADER INFO
 9343 031404  010137  002032                      MOV     R1,ADDRESS
 9344 031410  104460                              PERR34
 9345 031412                                      SET     HEADER            ;ENABLE PRINTING OF ERROR HEADER INFO
 9346
 9347 031420  132712  177777              7$:     BITB    #-1,(R2)          ;CHECK FOR LAST BYTE
```

```
 9348 031424  001012                          BNE     8$              ;
 9349 031426  005202                          INC     R2
 9350 031430  005201                          INC     R1              ;MOVE TO NEXT BYTE
 9351 031432  013704  002406                  MOV     TESTADD,R4      ;FIRST TEST ADDRESS
 9352 031436  032701  000002                  BIT     #2,R1           ;TEST FOR LOWER WORD
 9353 031442  001723                          BEQ     3$              ;BR IF IT'S LOW 16 BITS
 9354 031444  062704  000002                  ADD     #2,R4           ;ADJUST POINTER FOR ERROR REPT.
 9355 031450  000720                          BR      3$
 9356 031452  005737  002252          8$:     TST     SBEMSK+2        ;LAST ERROR BIT ?
 9357 031456  100405                          BMI     9$              ;MINUS MEANS BIT 31
 9358 031460                                  DLEFT   SBEMSK
 9359 031470  000647                          BR      2$
 9360 031472                          9$:     IF #SW11 SET.IN @SWR THEN GOTO 10$
 9361 031502                                  IF QVFLAG IS TRUE THEN GOTO 10$
 9362 031510  005737  002242                  TST     DATBUF+2        ;LAST DATA BIT?
 9363 031514  100405                          BMI     10$             ;MINUS = BIT 31
 9364 031516                                  DLEFT   DATBUF
 9365 031526  000623                          BR      1$
 9366                                         ;CLEAR OUT ANY DBE'S OR SBE'S
 9367 031530  104471                  10$:    ECC1DIS                 ;DISABLE ECC ON 1 SELECTED CSR
 9368 031532  013701  002406                  MOV     TESTADD,R1
 9369 031536  005011                          CLR     (R1)
 9370 031540  060501                          ADD     R5,R1
 9371 031542  005011                          CLR     (R1)
 9372 031544  104503                          CLR1CSR                 ;CLEAR 1 SELECTED CSR
 9373 031546  000207                          RETURN
```

```
9376 031550                                MTP013: SUBTST  <<MTP013        CREATE DOUBLE BIT ERROR TEST>>
                                           ;*****************************************************************************
                                           ;*SUBTEST       MTP013  CREATE DOUBLE BIT ERROR TEST
                                           ;*****************************************************************************
9377                                                       ;DOUBLE BIT ERROR FORCE TO CHECK DOUBLE ERROR LOGIC
9378 031550  104503                              CLR1CSR                   ;CLEAR 1 SELECTED CSR
9379 031552  012701  002406                      MOV     #TESTADD,R1
9380 031556  005037  002240              1$:     CLR     DATBUF            ;MAKE INITIAL DATA
9381 031562  005037  002242                      CLR     DATBUF+2          ;ALL ZEROS
9382 031566  012737  000001  002250      2$:     MOV     #1,SBEMSK         ;INITIAL SINGLE ERROR MASK
9383 031574  005037  002252                      CLR     SBEMSK+2          ;SECOND WORD
9384 031600  012737  000001  002254      3$:     MOV     #1,DBEMSK         ;INITIAL DOUBLE ERROR MASK
9385 031606  005037  002256                      CLR     DBEMSK+2          ;32 BITS HERE ALSO
9386 031612  013737  002240  002244      4$:     MOV     DATBUF,TSTDAT     ;
9387 031620  013737  002242  002246              MOV     DATBUF+2,TSTDAT+2
9388 031626  105737  002262                      TSTB    PASFLG  ;NO COMPLEMENTING FIRST PASS
9389 031632  001404                              BEQ     5$
9390 031634  005137  002244                      COM     TSTDAT            ;COMP FIRST WORD
9391 031640  005137  002246                      COM     TSTDAT+2          ;SECOND WORD
9392 031644  104503                      5$:     CLR1CSR                   ;CLEAR 1 SELECTED CSR
9393 031646  023737  002250  002254              CMP     SBEMSK,DBEMSK     ;CAN'T HAVE THE SAME ERROR BIT SET
9394 031654  001004                              BNE     6$                ;IN BOTH MASKS
9395 031656  023737  002252  002256              CMP     SBEMSK+2,DBEMSK+2;COULD BE EQUAL IN SECOND WORD
9396 031664  001460                              BEQ     13$               ;GO MAKE THEM NOT EQUAL
9397 031666  012737  002244  002306      6$:     MOV     #TSTDAT,SOURCE    ;SOURCE ADDRESS FOR CHKGEN
9398 031674  004737  044366                      CALL    CHKGEN            ;GO GENERATE CHECK BITS
9399 031700  013702  002250                      MOV     SBEMSK,R2
9400 031704  074237  002244                      XOR     R2,TSTDAT
9401 031710  013702  002252                      MOV     SBEMSK+2,R2
9402 031714  074237  002246                      XOR     R2,TSTDAT+2
9403 031720  013702  002254                      MOV     DBEMSK,R2
9404 031724  074237  002244                      XOR     R2,TSTDAT
9405 031730  013702  002256                      MOV     DBEMSK+2,R2
9406 031734  074237  002246                      XOR     R2,TSTDAT+2
9407 031740  104471                      16$:    ECC1DIS                   ;DISABLE ECC ON 1 SELECTED CSR
9408 031742  013731  002244                      MOV     TSTDAT,@(R1)+     ;WRITE 16 BITS
9409 031746  104475                              CB1CSR                    ;WRITE GENERATED CHECKBITS IN 1 SELECTED CSR
9410 031750  013771  002246  000000              MOV     TSTDAT+2,@(R1)    ;WRITE HIGH WORD
9411 031756  104503                              CLR1CSR                   ;CLEAR 1 SELECTED CSR
9412 031760  162701  000002                      SUB     #2,R1             ;ADJUST TEST ADDRESS
9413 031764  005771  000000                      TST     @(R1)             ;READ THE LOCATION
9414 031770  104501                              WAS1DBE                   ;WAS THERE ANY DOUBLE BIT ERRORS ON 1 SELECTED CSR
9415 031772  103411                              BCS     9$                ;IT SHOULD BE SET
9416 031774                                      SET     HEADER
9417 032002  011137  002032                      MOV     (R1),ADDRESS
9418 032006  104030                              ERROR   +30
9419 032010                                      SET     HEADER
```

```
9422 032016 104512                      9$:     ERRGEN
9423 032020 105737 002262                       TSTB    PASFLG
9424 032024 100452                              BMI     14$
9425 032026 005737 002256               13$:    TST     DBEMSK+2        ;CHECK MASK FOR LAST BIT
9426 032032 100405                              BMI     10$             ;MINUS = BIT31
9427 032034                                     DLEFT   DBEMSK
9428 032044 000662                              BR      4$
9429 032046                             10$:    IF #SW11 SET.IN @SWR THEN GOTO 11$
9430 032056                                     IF QVFLAG IS TRUE THEN GOTO 11$
9431 032064 005737 002252                       TST     SBEMSK+2        ;CHECK SINGLE ERROR MASK TOO
9432 032070 100405                              BMI     11$             ;BR IF DONE
9433 032072                                     DLEFT   SBEMSK
9434 032102 000636                              BR      3$
9435 032104 105737 002262               11$:    TSTB    PASFLG  ;FIRST PASS
9436 032110 001003                              BNE     12$             ;NON ZERO MEANS WE'RE DONE
9437 032112 105237 002262                       INCB    PASFLG  ;FIRST PASS, NOT DONE
9438                                            ;CLEAR OUT ANY DBE'S OR SBE'S
9439 032116 000617                              BR      1$              ;KEEP GOING
9440 032120 052737 000200 002262        12$:    BIS     #BIT7,PASFLG    ;SET UP FOR CHECK BIT TEST
9441 032126 005037 002244                       CLR     TSTDAT
9442 032132 005037 002246                       CLR     TSTDAT+2
9443 032136 012737 000040 002250                MOV     #40,SBEMSK
9444 032144 012737 000100 002254                MOV     #100,DBEMSK
9445 032152 012737 003740 002310        14$:    MOV     #3740,CHECK
9446 032160 013702 002250                       MOV     SBEMSK,R2
9447 032164 074237 002310                       XOR     R2,CHECK
9448 032170 013702 002254                       MOV     DBEMSK,R2
9449 032174 074237 002310                       XOR     R2,CHECK
9450 032200 006337 002254                       ASL     DBEMSK
9451 032204 032737 020000 002254                BIT     #BIT13,DBEMSK
9452 032212 001652                              BEQ     16$
9453 032214 006337 002250                       ASL     SBEMSK
9454 032220 032737 004000 002250                BIT     #BIT11,SBEMSK
9455 032226 001006                              BNE     15$
9456 032230 013737 002250 002254                MOV     SBEMSK,DBEMSK
9457 032236 006337 002254                       ASL     DBEMSK
9458 032242 000743                              BR      14$
9459 032244 104471                      15$:    ECC1DIS                 ;DISABLE ECC ON 1 SELECTED CSR
9460 032246 012701 002406                       MOV     #TESTADD,R1
9461 032252                                     CLEAR   @(R1)+,@(R1)
9462 032260 104503                              CLR1CSR                 ;CLEAR 1 SELECTED CSR
9463 032262 000207                              RETURN
```

```
   9467 032264                          MTP014: SUBTST  <<MTP014        BASIC DOUBLE BIT ERROR TEST>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       MTP014  BASIC DOUBLE BIT ERROR TEST
                                        ;*****************************************************************************
   9468                                 ;
   9469                                 ; THIS TEST CHECKS THAT A DOUBLE ERROR WILL BE DETECTED
   9470                                 ; A BYTE WRITE WITH A DOUBLE ERROR ON A MS11-P
   9471                                 ; WILL BE ABOTRED.
   9472                                 ;
   9473 032264  104424                        CACHOFF                 ;TURN OFF CACHE
   9474 032266                                LET PARCNT := #0        ;CLEAR PARCNT
   9475 032272                                LET NOPAR := #1         ;SET PARITY ACTION
   9476 032300                                LET ADDRESS := #FIRST   ;SET ADDRESS FOR ERROR REPORT
   9477 032306  104513                        CBREG                   ;ENABLE CHECK/SYNDROME BIT REGISTER
   9478 032310                                LET CSR := #3145        ;DBE CHECK BITS FOR CSR
   9479 032316  104425                        LOADCSR                 ;WRITE DBE CHECK BITS TO CSR
   9480 032320                                LET GOOD := #103145     ;GOOD DATA
   9481 032326                                LET (R1) := #0          ;WRITE ZEROS AND DBL ERROR CHK BITS A=0
   9482 032330  005711                        TST (R1)                ;READ A=0 TO GET DOUBLE BIT ERROR
   9483 032332                                IF PARCNT NE #1         ;WAS BUSPBL ASSERTED????
   9484 032342  104055                           ERROR +55            ;ERROR CALL ;;MISSED EXPECTED TRAP
   9485 032344                                END                     ;
   9486 032344  104426                        READCSR                 ;READ CSR FOR CORRECT CHECK BITS AND DBE INDICATOR
   9487 032346  042737  020000  002146        BIC #BIT13,CSR          ;CLEAR INHIBIT MODE POINTER FROM DATA IF IT EXSISTS1
   9488 032354                                IF CSR NE GOOD THEN     ;CHECK IF DOUBLE ERROR BIT IS SET
   9489 032364                                   SET HEADER           ;
   9490 032372                                   LET BAD := CSR       ;BAD DATA
   9491 032400  104065                           ERROR +65            ;
   9492 032402                                END                     ;
   9493 032402  104473                        ECC1INIT                ;ENABLE BUSPBL
   9494 032404  005037  002264                CLR PASSNO              ;CLEAR LOOP COUNTER
   9495 032410                                REPEAT                  ;
   9496 032410  104473                          ECC1INIT              ;ENABLE BUSPBL
   9497 032412  005237  002264                  INC PASSNO            ;INCREMENT LOOP COUNTER
   9498 032416  005037  002070                  CLR PARCNT            ;CLEAR PARITY ACTION COUNTER
   9499 032422                                  LET (R1) :B= #377     ;WRITE BYTE SHOULD BE ABORTED
   9500 032426  105711                          TSTB (R1)             ;READ R1 TO SEE IF IT IS STILL 0
   9501 032430                                  IF PARCNT NE #1       ;WAS WRITE ABORTED???
   9502 032440                                     SET HEADER         ;
   9503 032446                                     LET GOOD := #0     ;GOOD DATA
   9504 032452                                     LET BAD := #377    ;BAD DATA
   9505 032460  104056                             ERROR +56          ;
   9506 032462                                  END                   ;
   9507 032462  005201                          INC   R1              ;AND REPEAT ON HIGH BYTE
   9508 032464                                UNTIL PASSNO EQ #2      ;
   9509 032474  005041                        CLR     -(R1)           ;CLEAR LUT
   9510 032476  104503                        CLR1CSR                 ;CLEAR CSR
   9511 032500  005037  002070                CLR     PARCNT          ;CLEAR PARITY TRAP COUNTER
   9512 032504  104423                        CACHON                  ;TURN ON CACHE
   9513 032506  000207                        RETURN                  ;
```

```
                                        MTP015: SUBTST  <<MTP015        WRITE INHIBIT OF BYTE WITH DBE>>
                                        ;*****************************************************************************
                                        ;
                                        ;*SUBTEST       MTP015  WRITE INHIBIT OF BYTE WITH DBE
                                        ;*****************************************************************************
                                        ;
                                                        ;CHECK FOR WRITE INHIBIT DURING A WRITE BYTE.
                                                        ;CHECKS FOR UNCORRECTED DATA.
032510  005037  002240          1$:     CLR     DATBUF          ;INITIAL DATA
032514  005037  002242                  CLR     DATBUF+2        ;32 BITS WORTH
032520  012737  000001  002250  2$:     MOV     #1,SBEMSK       ;SINGLE ERROR MASK
032526  005037  002252                  CLR     SBEMSK+2        ;
032532  012737  000001  002254  3$:     MOV     #1,DBEMSK       ;DOUBLE ERROR MASK
032540  005037  002256                  CLR     DBEMSK+2        ;
032544  013737  002240  002244  4$:     MOV     DATBUF,TSTDAT   ;PRESERVE ORIG DATA
032552  013737  002242  002246          MOV     DATBUF+2,TSTDAT+2
032560  105737  002262                  TSTB    PASFLG          ;WHICH PASS ?
032564  001404                          BEQ     5$              ;FIRST PASS, NO COMPLEMENTING
032566  005137  002244                  COM     TSTDAT
032572  005137  002246                  COM     TSTDAT+2        ;SECOND PASS, COMPLEMENT TSTDAT
032576  104503                  5$:     CLR1CSR                 ;CLEAR 1 SELECTED CSR
032600  023737  002250  002254          CMP     SBEMSK,DBEMSK   ;CHECK FOR SAME MASKS
032606  001004                          BNE     6$              ;BR IF NOT EQUAL
032610  023737  002252  002256          CMP     SBEMSK+2,DBEMSK+2       ;SECOND WORD ALSO
032616  001474                          BEQ     11$             ;BR TO MAKE THEM NOT EQUAL
032620  012737  002244  002306  6$:     MOV     #TSTDAT,SOURCE  ;ADDRESS FOR CHKGEN
032626  004737  044366                  CALL    CHKGEN          ;GO GENERATE CHECK BITS
032632  013701  002250                  MOV     SBEMSK,R1
032636  074137  002244                  XOR     R1,TSTDAT
032642  013701  002252                  MOV     SBEMSK+2,R1
032646  074137  002246                  XOR     R1,TSTDAT+2
032652  013701  002254                  MOV     DBEMSK,R1
032656  074137  002244                  XOR     R1,TSTDAT
032662  013701  002256                  MOV     DBEMSK+2,R1
032666  074137  002246                  XOR     R1,TSTDAT+2
032672  012701  002406          7$:     MOV     #TESTADD,R1     ;TEST LOCATION
032676  104471                          ECC1DIS                 ;DISABLE ECC ON 1 SELECTED CSR
032700  013731  002244                  MOV     TSTDAT,@(R1)+   ;WRITE FIRST 16 BITS
                                        ;LOAD CSR WITH IMAGE FROM R2
032704  104475                          CB1CSR                  ;WRITE GENERATED CHECKBITS IN 1 SELECTED CSR
032706  013771  002246  000000          MOV     TSTDAT+2,@(R1)  ;WRITE SECOND 16 BITS + CHECKBITS
032714  104503                          CLR1CSR                 ;CLEAR 1 SELECTED CSR
032716  013702  002406                  MOV     TESTADD,R2      ;GET ADDRESS OF TEST LOC
032722  010203                          MOV     R2,R3           ;R2 DESIGNATES FIRST BYTE
032724  062703  000003                  ADD     #3,R3           ;R3 DESIGNATES LAST BYTE
032730  112722  000360          8$:     MOVB    #360,(R2)+      ;TRY WRITING A BYTE
032734  012701  002406                  MOV     #TESTADD,R1
032740  017100  000000                  MOV     @(R1),R0
032744  023700  002244                  CMP     TSTDAT,R0       ;CHECK FOR UNCHANGED DATA
032750  001404                          BEQ     9$              ;BR IF OK
032752  017137  000000  002032          MOV     @(R1),ADDRESS
032760  104455                          PERR31
                                        
032762  017100  000002          9$:     MOV     @2(R1),R0
032766  023700  002246                  CMP     TSTDAT+2,R0     ;READ SECOND WORD
032772  001404                          BEQ     10$             ;BR IF UNCHANGED
032774  017137  000002  002032          MOV     @2(R1),ADDRESS
033002  104455                          PERR31
```

```
9569 033004  020203                    10$:    CMP     R2,R3           ;TESTED LAST BYTE ?
9570 033006  001350                            BNE     8$              ;BR IF NO
9571 033010  105737  002262            11$:    TSTB    PASFLG
9572 033014  100452                            BMI     15$             ;BRANCH IF TESTING CHECK BITS
9573 033016  005737  002256                    TST     DBEMSK+2        ;CHECKING FOR LAST ERROR BIT
9574 033022  100405                            BMI     12$             ;BR IF DONE HERE
9575 033024                                    DLEFT   DBEMSK
9576 033034  000643                            BR      4$
9577 033036                            12$:    IF #SW11 SET.IN @SWR THEN GOTO 13$
9578 033046                                    IF QVFLAG IS TRUE THEN GOTO 13$
9579 033054  005737  002252                    TST     SBEMSK+2        ;LAST SBE MASK
9580 033060  100405                            BMI     13$             ;BR IF DONE WITH THIS PASS
9581 033062                                    DLEFT   SBEMSK
9582 033072  000617                            BR      3$
9583 033074  105737  002262            13$:    TSTB    PASFLG  ;TEST PASS FLAG
9584 033100  001003                            BNE     14$             ;NON ZERO MEANS WE'RE DONE
9585 033102  105237  002262                    INCB    PASFLG  ;NOT DONR
9586 033106  000600                            BR      1$
9587 033110  052737  000200  002262    14$:    BIS     #BIT7,PASFLG
9588 033116  005037  002244                    CLR     TSTDAT
9589 033122  005037  002246                    CLR     TSTDAT+2
9590 033126  012737  000040  002250            MOV     #40,SBEMSK
9591 033134  012737  000100  002254            MOV     #100,DBEMSK
9592 033142  012737  003740  002310    15$:    MOV     #3740,CHECK
9593 033150  013702  002250                    MOV     SBEMSK,R2
9594 033154  074237  002310                    XOR     R2,CHECK
9595 033160  013702  002254                    MOV     DBEMSK,R2
9596 033164  074237  002310                    XOR     R2,CHECK
9597 033170  006337  002254                    ASL     DBEMSK
9598 033174  032737  020000  002254            BIT     #BIT13,DBEMSK
9599 033202  001633                            BEQ     7$
9600 033204  006337  002250                    ASL     SBEMSK
9601 033210  032737  004000  002250            BIT     #BIT11,SBEMSK
9602 033216  001006                            BNE     16$
9603 033220  013737  002250  002254            MOV     SBEMSK,DBEMSK
9604 033226  006337  002254                    ASL     DBEMSK
9605 033232  000743                            BR      15$
9606 033234  104471                    16$:    ECC1DIS                 ;DISABLE ECC ON 1 SELECTED CSR
9607 033236  012701  002406                    MOV     #TESTADD,R1     ;TEST LOCATION
9608 033242                                    CLEAR   @(R1)+,@(R1)    ;TO ERASE ANY DBE'S FROM TESTING
9609                                           ;RESTORE CSR
9610 033250  104503                            CLR1CSR                 ;CLEAR 1 SELECTED CSR
9611 033252  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.
033254  005037  002240          T12A:   CLR     DATBUF          ;BACKGROUND FOR DOUBLE ERRORS
033260  005037  002242                  CLR     DATBUF+2        ;2 WORDS WORTH
033264  012737  000001  002250          MOV     #1,SBEMSK       ;SINGLE FRROR MASK
033272  005037  002252                  CLR     SBEMSK+2        ;
033276  012737  000001  002254  T12B:   MOV     #1,DBEMSK       ;DOUBLE ERROR MASK
033304  005037  002256                  CLR     DBEMSK+2
033310  013737  002240  002244  1$:     MOV     DATBUF,TSTDAT   ;DATA FOR TEST
033316  013737  002242  002246          MOV     DATBUF+2,TSTDAT+2;BOTH WORDS
033324  105737  002262                  TSTB    PASFLG  ;COMP DATA ON SECOND PASS ONLY
033330  001404                          BEQ     2$              ;BR IF FIRST PASS
033332  005137  002244                  COM     TSTDAT          ;COMP FIRST WORD
033336  005137  002246                  COM     TSTDAT+2        ;NOW SECOND WORD
033342  023737  002250  002254  2$:     CMP     SBEMSK,DBEMSK   ;CHECK FOR IDENTICAL MASKS
033350  001004                          BNE     3$              ;BR IF DIFFERENT
033352  023737  002252  002256          CMP     SBEMSK+2,DBEMSK+2;UPPER WORD TOO
033360  001502                          BEQ     8$              ;BR TO MAKE THEM NOT EQUAL
033362  012737  002244  002306  3$:     MOV     #TSTDAT,SOURCE  ;NEED ADDR OF DATA FOR CHKGEN
033370  004737  044366                  CALL    CHKGEN          ;GO GENERATE CHECK BITS
033374  013701  002250                  MOV     SBEMSK,R1
033400  074137  002244                  XOR     R1,TSTDAT
033404  013701  002252                  MOV     SBEMSK+2,R1
033410  074137  002246                  XOR     R1,TSTDAT+2
033414  013701  002254                  MOV     DBEMSK,R1
033420  074137  002244                  XOR     R1,TSTDAT
033424  013701  002256                  MOV     DBEMSK+2,R1
033430  074137  002246                  XOR     R1,TSTDAT+2
033434  012701  002406          4$:     MOV     #TESTADD,R1     ;FIRST TEST ADRRESS
033440  104471                          ECC1DIS                 ;DISABLE ECC ON 1 SELECTED CSR
033442  013731  002244                  MOV     TSTDAT,@(R1)+   ;WRITE FIRST 16 BITS
033446  104475                          CB1CSR                  ;WRITE GENERATED CHECKBITS IN 1 SELECTED CSR
033450  013771  002246  000000          MOV     TSTDAT+2,@(R1)  ;WRITE SECOND 16 BITS + CHECKBITS
033456  105037  002263                  CLRB    UPPFLG          ;SET FOR 2 LOOPS
033462  162701  000002                  SUB     #2,R1           ;POINT TO LOW WORD
033466  104503          5$:     CLR1CSR                         ;CLEAR 1 SELECTED CSR
033470  012771  177400  000000          MOV     #177400,@(R1)   ;TRY WRITING LOCATION
033476  012701  002406                  MOV     #TESTADD,R1
033502  017100  000000                  MOV     @(R1),R0
033506  023700  002244                  CMP     TSTDAT,R0       ;CHECK FOR ORIGINAL DATA
033512  001404                          BEQ     6$              ;SHOULD BE UNCHANGED
033514  017137  000000  002032          MOV     @(R1),ADDRESS
033522  104455                          PERR31
                                        
033524  062701  000002          6$:     ADD     #2,R1
033530  017100  000000                  MOV     @(R1),R0
033534  023700  002246                  CMP     TSTDAT+2,R0     ;THIS SHOULD BE UNCHANGED ALSO
033540  001404                          BEQ     7$
033542  017137  000000  002032          MOV     @(R1),ADDRESS
033550  104455                          PERR31
```

```
9669 033552  105737  002263          7$:     TSTB    UPPFLG          ;WHICH LOOP ?
9670 033556  001003                          BNE     8$              ;SECOND, BR OUT
9671 033560  105237  002263                  INCB    UPPFLG          ;FIRST, KEEP GOING
9672 033564  000740                          BR      5$
9673 033566  105737  002262          8$:     TSTB    PASFLG
9674 033572  100454                          BMI     12$
9675 033574  005737  002256                  TST     DBEMSK+2        ;LAST BIT ?
9676 033600  100405                          BMI     9$              ;MINUS = BIT 31
9677 033602                                  DLEFT   DBEMSK
9678 033612  000636                          BR      1$
9679 033614                          9$:     IF #SW11 SET.IN @SWR THEN GOTO 10$
9680 033624                                  IF QVFLAG IS TRUE THEN GOTO 10$
9681 033632  005737  002252                  TST     SBEMSK+2        ;LAST BIT IN THIS MASK ?
9682 033636  100406                          BMI     10$             ;BR IF LAST BIT
9683 033640                                  DLEFT   SBEMSK
9684 033650  000137  033276                  JMP     T12B
9685 033654  105737  002262          10$:    TSTB    PASFLG  ;FIRST PASS ?
9686 033660  001004                          BNE     11$             ;BR IF SECOND
9687 033662  105237  002262                  INCB    PASFLG  ;INDICATE SECOND PASS COMING
9688 033666  000137  033254                  JMP     T12A
9689 033672  052737  000200  002262  11$:    BIS     #BIT7,PASFLG
9690 033700  005037  002244                  CLR     TSTDAT
9691 033704  005037  002246                  CLR     TSTDAT+2
9692 033710  012737  000040  002250          MOV     #40,SBEMSK
9693 033716  012737  000100  002254          MOV     #100,DBEMSK
9694 033724  012737  003740  002310  12$:    MOV     #3740,CHECK
9695 033732  013702  002250                  MOV     SBEMSK,R2
9696 033736  074237  002310                  XOR     R2,CHECK
9697 033742  013702  002254                  MOV     DBEMSK,R2
9698 033746  074237  002310                  XOR     R2,CHECK
9699 033752  006337  002254                  ASL     DBEMSK
9700 033756  032737  020000  002254          BIT     #BIT13,DBEMSK
9701 033764  001623                          BEQ     4$
9702 033766  006337  002250                  ASL     SBEMSK
9703 033772  032737  004000  002250          BIT     #BIT11,SBEMSK
9704 034000  001006                          BNE     13$
9705 034002  013737  002250  002254          MOV     SBEMSK,DBEMSK
9706 034010  006337  002254                  ASL     DBEMSK
9707 034014  000743                          BR      12$
9708 034016  104471                  13$:    ECC1DIS                 ;DISABLE ECC ON 1 SELECTED CSR
9709 034020  012701  002406                  MOV     #TESTADD,R1     ;RESTORE TEST ADDRESS
9710 034024  005031                          CLR     @(R1)+          ;CLEAR ANY DBE'S FROM TEST
9711 034026  005071  000000                  CLR     @(R1)
9712 034032  104503                          CLR1CSR                 ;CLEAR 1 SELECTED MK11 CSR
9713 034034  000207                          RETURN
```

```
 9716 034036                            MTP017: SUBTST  <<MTP017        HOLDING 1'S & 0'S TEST>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       MTP017  HOLDING 1'S & 0'S TEST
                                        ;*****************************************************************************
 9717                                           ;*(1)   THIS TEST CHECKS THE MEMORY FOR THE CAPABILITY
 9718                                           ;*      OF HOLDING 1'S AND 0'S BY WRITING A BACKGROUND
 9719                                           ;*      OF 000377 AND READING IT
 9720                                           ;*(2)   MEMORY IS WRITTEN USING A BYTE AT A TIME
 9721                                           ;*(3)   STEPS 1 & 2 ARE REPEATED WITH A SWAPPED BACKGROUND PATTERN
 9722                                           ;NOTE:  THIS TEST WRITES BYTES & READS WORDS
 9723 034036  012701  060000                   MOV     #FIRST,R1
 9724 034042  010104                           MOV     R1,R4
 9725 034044  012705  160000                   MOV     #LAST+2,R5
 9726 034050  012700  000377                   MOV     #377,R0         ;GET THE PATTERN INTO R0
 9727 034054  010003                           MOV     R0,R3
 9728 034056  000303                           SWAB    R3
 9729 034060  110021                  1$:      MOVB    R0,(R1)+        ;WRITE A BYTE
 9730 034062  110321                           MOVB    R3,(R1)+        ;WRITE THE MEMORY WITH THE BYTE STORED IN BAKPAT+1
 9731 034064  020105                           CMP     R1,R5           ;COMPARE TEST LOC TO TOP + 2
 9732 034066  103774                           BLO     1$              ;BRANCH IF LOWER
 9733
 9734 034070  014102                  2$:      MOV     -(R1),R2
 9735 034072  020002                           CMP     R0,R2           ;TEST THE MEMORY TO SEE IF IT CONTAINS
 9736                                                                  ;THE WORD STORED IN BAKPAT
 9737 034074  001401                           BEQ     3$
 9738 034076  104446                           PERR22
 9739
 9740 034100  020104                  3$:      CMP     R1,R4           ;KEEP ON TESTING THE MEMORY UNTIL
 9741 034102  101372                           BHI     2$              ;R1 EQUALS THE LOWEST ADDRESS
 9742 034104  000303                           SWAB    R3              ;CHANGE THE DATA PATTERN
 9743 034106  000300                           SWAB    R0
 9744 034110  001763                           BEQ     1$              ;IF THE DATA PATTERN DOES NOT HAVE LOW
 9745                                                                  ; BYTE =0 THEN FALL THRU
 9746 034112  000207                           RETURN
```

```
9748
```

```
 9750 034114                            MTP020: SUBTST  <<MTP020        SYNDROMES TO CSR ON SINGLE BIT ERROR TEST>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       MTP020  SYNDROMES TO CSR ON SINGLE BIT ERROR TEST
                                        ;*****************************************************************************
 9751                                           ;
 9752                                           ;  THIS TEST CHECKS TO SEE IF THE SINGLE BIT ERRORS CAUSE THE SBE
 9753                                           ;  BIT IN THE CSR TO BE SET AND CORRECT SYNDROME BITS ARE GENERATED FOR
 9754                                           ;  ALL 16 DATA BITS.
 9755                                           ;
 9756 034114  104424                           CACHOFF                         ;TURN OFF CACHE
 9757 034116  005000                           CLR  R0                         ;CLEAR DATA
 9758 034120  105037  002262                   CLRB PASFLG                     ;CLEAR PASFLG
 9759 034124  104513                           CBREG                           ;ENABLE CHECK/SYNDROME BIT REGISTER
 9760 034126                                   REPEAT                          ;
 9761 034126                                     LET PASFLG :B= PASFLG + #1     ;INCREMENT LOOP COUNTER
 9762 034132                                     LET R4  := #-1                 ;INDEX TO SINGLE BIT ERROR TABLE
 9763 034136                                     LET BITNO := #0                ;CLEAR INNER LOOP COUNTER
 9764 034142                                     IFB PASFLG EQ #1               ;SELECT DATA TO BE CORRECTED BY PASSNO
 9765 034152                                       LET R5 := #1                 ;DATA=0;BIT TO BE CORRECTED IS A ONE
 9766 034156                                     ELSE                           ;
 9767 034160                                       LET R5 := #177776            ;DATA=177776;BIT TO BE CORRECTED IS A ZERO
 9768 034164                                     END                            ;
 9769 034164                                     REPEAT                         ;
 9770 034164  005237  002320                       INC BITNO                    ;INCREMENT BIT POINTER
 9771 034170                                       LET R4 := R4 + #1            ;POINT TO NEXT SET OF CHECK BITS
 9772 034172                                       LET R2 :B= PTABLE(R4)        ;GET NEXT SET OF CHECK BITS
 9773 034176  072227  000005                       ASH #5,R2                    ;SHIFT TO LINE UP IN CSR
 9774 034202  052702  000004                       BIS #BIT2,R2                 ;ENABLE DIAG MODE
 9775 034206                                       LET CSR := R2                ;GET CHECK BITS TO BE WRITTEN
 9776 034212  104425                               LOADCSR                      ;LOAD CSR WITH DATA
 9777 034214                                       LET (R1) := R0               ;WRITE DATA TO TEST ADDRESS
 9778 034216  104503                               CLR1CSR                      ;CLEAR CSR
 9779 034220  005711                               TST (R1)                     ;CORRECT SBE
 9780 034222  104426                               READCSR                      ;READ CSR FOR CORRECT SBE BIT AND SYNDROMES
 9781 034224  042737  177757  002146               BIC #^C20,CSR                ;CLEAR ALL BUT SBE INDICATOR
 9782 034232                                       IF CSR NE #20                ;WAS DATA CORRECTED???
 9783 034242                                         LET GOOD := #20            ;
 9784 034250                                         LET BAD := CSR             ;
 9785 034256  104060                                 ERROR +60                  ;NO ERROR
 9786 034260                                       END                          ;
 9787 034260  104513                               CBREG                        ;ENABLE SYNDROME BIT REGISTER
 9788 034262  104426                               READCSR                      ;GET SYNDROMES FROM CSR
 9789 034264  042737  174033  002146               BIC #^C3744,CSR              ;MASK SYNDROME BITS
 9790 034272                                       LET R3 :B= SBESYN(R4)        ;GET GOOD SYNDROMES
 9791 034276  072327  000005                       ASH #5,R3                    ;SHIFT INTO POSITION
 9792 034302  052703  000004                       BIS #BIT2,R3                 ;SET DIAG MODE IN DATA
 9793 034306                                       IF R3 NE CSR                 ;DO SYNDROME BITS AGREE
 9794 034314                                         SET HEADER                 ;
 9795 034322                                         LET GOOD := R3             ;
 9796 034326                                         LET BAD := CSR             ;
 9797 034334  104C42                                 ERROR +42                  ;
 9798 034336                                       END                          ;
 9799 034336  005011                               CLR (R1)                     ;CLEAR LUT
 9800 034340                                       IFB PASFLG EQ #1             ;SHIFT NEW DATA DEPENDING ON PASFLG
 9801 034350  006305                                 ASL  R5                    ;SHIFT BITNO TO THE LEFT
 9802 034352                                       ELSE                         ;
 9803 034354  000261                                 SEC                        ;SET CARRY BIT AND......
```

```
9804 034356  006105                              ROL   R5                ;ROTATE LEFT
9805 034360                                    END                       ;
9806 034360                                  UNTIL BITNO EQ #16.         ;UNTIL ALL BITS ARE DONE
9807 034370  005100                            COM R0                    ;COMPLEMENT DATA AND REPEAT
9808 034372                                  UNTILB PASFLG EQ #2         ;UNTIL 2 PASSES ARE COMPLETE!
9809 034402  104503                          CLR1CSR                     ;CLEAR CSR
9810 034404  104423                          CACHON                      ;TURN CACHE
9811 034406  000207                          RETURN
9812                                ;
9813                                ;        MS11-P SINGLE BIT ERROR SYNDROME BIT TABLE
9814                                ;
9815 034410      016     013     023  SBESYN: .BYTE   16,13,23,25,26,31,32,34,43,45,46,51,52,54,61,64
     034413      025     026     031
     034416      032     034     043
     034421      045     046     051
     034424      052     054     061
     034427      064
9816
```

```
 9818 034430                    MTPA21: SUBTST  <<MTPA21        MARCHING 1'S & 0'S PATTERN TEST>>
                                ;*****************************************************************************
                                ;*SUBTEST       MTPA21  MARCHING 1'S & 0'S PATTERN TEST
                                ;*****************************************************************************
 9819                                   ;READ,BYTESWAP-MODIFY,READ,DOWN
 9820 034430  014100            1$:     MOV     -(R1),R0;V177640
 9821 034432  020200                    CMP     R2,R0   ;V177642
 9822 034434  001401                    BEQ     2$      ;V177644
 9823 034436  104443                    PERR17          ;V177646
 9824
 9825 034440  000311            2$:     SWAB    (R1)    ;V177650
 9826 034442  011100                    MOV     (R1),R0 ;V177652
 9827 034444  020300                    CMP     R3,R0   ;V177654
 9828 034446  001401                    BEQ     3$      ;V177656
 9829 034450  104444                    PERR20          ;V177660
 9830
 9831 034452  020401            3$:     CMP     R4,R1   ;V177662        ;DONE?
 9832 034454  001365                    BNE     1$      ;V177664        ;NO - LOOP
 9833 034456  000207                    RETURN          ;V177666        ;YES - RETURN
 9834
 9835 034460                    MTPB21: ;READ,BYTESWAP-MODIFY,READ,UP
 9836 034460  011100            1$:     MOV     (R1),R0 ;V177640
 9837 034462  020300                    CMP     R3,R0   ;V177642
 9838 034464  001401                    BEQ     2$      ;V177644
 9839 034466  104444                    PERR20          ;V177646
 9840
 9841 034470  000311            2$:     SWAB    (R1)    ;V177650
 9842 034472  011100                    MOV     (R1),R0 ;V177652
 9843 034474  020200                    CMP     R2,R0   ;V177654
 9844 034476  001401                    BEQ     3$      ;V177656
 9845 034500  104443                    PERR17          ;V177660
 9846
 9847 034502  062701  000002    3$:     ADD     #2,R1   ;V177662
 9848 034506  020501                    CMP     R5,R1   ;V177666        ;DONE?
 9849 034510  001363                    BNE     1$      ;V177670        ;NO - LOOP
 9850 034512  000207                    RETURN          ;V177672        ;YES - RETURN
 9851
 9852 034514                    MTPC21: ;READ,BYTESWAP-MODIFY,READ,UP
 9853 034514  011100            1$:     MOV     (R1),R0 ;V177640
 9854 034516  020200                    CMP     R2,R0   ;V177642
 9855 034520  001401                    BEQ     2$      ;V177644
 9856 034522  104443                    PERR17          ;V177646
 9857
 9858 034524  000311            2$:     SWAB    (R1)    ;V177650
 9859 034526  011100                    MOV     (R1),R0 ;V177652
 9860 034530  020300                    CMP     R3,R0   ;V177654
 9861 034532  001401                    BEQ     3$      ;V177656
 9862 034534  104444                    PERR20          ;V177660
 9863
 9864 034536  062701  000002    3$:     ADD     #2,R1   ;V177662
 9865 034542  020501                    CMP     R5,R1   ;V177666        ;DONE?
 9866 034544  001363                    BNE     1$      ;V177670        ;NO - LOOP
 9867 034546  000207                    RETURN          ;V177672        ;YES - RETURN
```

```
 9870 034550                    MTPD21: ;READ,BYTESWAP-MODIFY,READ,DOWN
 9871 034550  014100            1$:     MOV     -(R1),R0:V177640
 9872 034552  020300                    CMP     R3,R0   :V177642
 9873 034554  001401                    BEQ     2$      :V177644
 9874 034556  104444                    PERR20          :V177646
 9875
 9876 034560  000311            2$:     SWAB    (R1)    :V177650
 9877 034562  011100                    MOV     (R1),R0 :V177652
 9878 034564  020200                    CMP     R2,R0   :V177654
 9879 034566  001401                    BEQ     3$      :V177656
 9880 034570  104443                    PERR17          :V177660
 9881
 9882 034572  020401            3$:     CMP     R4,R1   :V177662        ;DONE?
 9883 034574  001365                    BNE     1$      :V177664        ;NO - LOOP
 9884 034576  000207                    RETURN          :V177666        ;YES - RETURN
 9885
```

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

                                            ;WRITE LOOP
                                            CACHON                      ;TURN CACHE ON
                                            LET COUNT := STRIPES
                                            LET R1 := #FIRST
                                            WHILE R1 LOS #LAST
                                              IF COUNT LT #0 THEN LET COUNT := #KDIAG-1
                                              IF #374 OFF.IN R1 THEN LET COUNT := COUNT - #1
                                              IF COUNT NE #0
                                                LET (R1) := R2
                                                LET 2(R1) := R2
                                              ELSE
                                                LET (R1) := R3
                                                LET 2(R1) := R3
                                              END ;OF IF COUNT
                                              LET COUNT := COUNT - #1
                                              LET R1 := R1 + #4
                                            END ;OF WHILE
                                            ;END OF WRITE LOOP

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

```
9929 035000                                     WHILE R1 LOS #LAST
9930 035006                                       IF COUNT LT #0 THEN LET COUNT := #KDIAG-1
9931 035022                                       IF #374 OFF.IN R1 THEN LET COUNT := COUNT - #1
9932 035034                                       IF COUNT NE #0
9933 035042                                         LET R0 := (R1)
9934 035044                                         IF R2 NE R0
9935 035050  104443                                   PERR17
9936 035052                                         END ;OF IF R2
9937 035052                                         LET R0 := 2(R1)
9938 035056                                         IF R2 NE R0
9939 035062  104443                                   PERR17
9940 035064                                         END ;OF IF R2
9941 035064                                       ELSE
9942 035066                                         LET R0 := (R1)
9943 035070                                         IF R3 NE R0
9944 035074  104444                                   PERR20
9945 035076                                         END ;OF IF R3
9946 035076                                         LET R0 := 2(R1)
9947 035102                                         IF R3 NE R0
9948 035106  104444                                   PERR20
9949 035110                                         END ;OF IF R3
9950 035110                                       END ;OF IF COUNT
9951 035110                                       LET COUNT := COUNT - #1
9952 035114                                       LET R1 := R1 + #4
9953 035120                                     END ;OF WHILE
9954                                            ;END OF READ LOOP
9955
9956 035122                                   END ;OF FOR STRIPES
9957 035136                                 END ;OF FOR EVEN
9958 035152  000207                           RETURN
9959
9960 035154                           REFRESH:SUBTST  <<SUBR  REFRESH DELAY>>
                                      ;*****************************************************************************
                                      ;*SUBTEST       SUBR    REFRESH DELAY
                                      ;*****************************************************************************
9961                                          ;DISTURB EACH ROW FOR > 3.2 MS
9962 035154                                   FOR R0 := #FIRST TO #FIRST+374 BY #4
9963 035160  004737  035224                     CALL REFSUB
9964 035164                                   END ;OF FOR R0
9965 035176                                   LET R0 := #FIRST+BIT14
9966 035202                                   WHILE R0 LOS #LAST+BIT14+374
9967 035210  004737  035224                     CALL REFSUB
9968 035214                                     LET R0 := R0 + #4
9969 035220                                   END ;OF WHILE
9970 035222  000207                           RETURN
9971 035224  012704  000640           REFSUB: MOV     #640,R4                 ;TIME FOR A > 3.2 MS LOOP
9972 035230  062700  000002                   ADD     #2,R0
9973 035234  005140           1$:     COM     -(R0)
9974 035236  005120                   COM     (R0)+
9975 035240  005110                   COM     (R0)
9976 035242  005110                   COM     (R0)
9977 035244  077405                   SOB     R4,1$
9978 035246  162700  000002          SUB     #2,R0
9979 035252  000207                   RETURN
```

```
 9982 035254                    MTPA24: SUBTST  <<MTPA24        FAST GALLOPING PATTERN TEST>>
                                ;*****************************************************************************
                                ;*SUBTEST        MTPA24  FAST GALLOPING PATTERN TEST
                                ;*****************************************************************************
 9983                                   ;THE TOTAL TEST (INCLUDING SETUP) IS AS FOLLOWS
 9984                                   ;*(1)   THIS TEST WRITES THE MEMORY WITH A BACK GROUND PATTERN
 9985                                   ;*      STORED AT LOCATION BAKPAT
 9986                                   ;*(2)   TEST BEGINS AT LOWEST LOCATION BEING TESTED
 9987                                   ;*      (LETS NAME IT 'A')
 9988                                   ;*(3)   LETS NAME THE 1ST LOCATION IN THE ROW/COLUMN UNDER TEST AS 'B'.
 9989                                   ;*(4)   SWAPS BYTES FOR LOCATION 'A'.
 9990                                   ;*(5)   READS 'A', READS 'B'
 9991                                   ;*(6)   'B' = 'B'+400   (ADDS 64 DOUBLE WORDS TO 'B')
 9992                                   ;*(7)   REPEATS STEPS 5 AND 6 UNTIL 'B' IS GREATER THAN THE
 9993                                   ;*(8)   END OF THE BANK  A+2
 9994                                   ;*(9)   REPEATS  STEPS 3-8 UNTILL 'A' REACHES THE END OF THE BANK
 9995                                   ;*(10)  AFTER EXECUTING THE TEST  DATA IS COMPLEMENTED
 9996                                   ;*      AND STEPS 1-9 ARE REPEATED
 9997                                   ;REGISTERS ARE USED AS FOLLOWS
 9998                                   ;R0     TEST DATA
 9999                                   ;R1     'A'
10000                                   ;R2     'B'
10001                                   ;R3     BAKPAT
10002                                   ;R4     SWAPAT
10003                                   ;R5     LAST
10004
10005                                   ;NOTE THE PATTERN STARTS AT MTPB24!!!!!!!!!!!!!!!
10006
10007                                   ;UIPAR'S
10008 035254  011100           1$:     MOV     (R1),R0         ;V177640        ;READ 'A'
10009 035256  020004                   CMP     R0,R4           ;V177642        ;CHECK 'A'
10010 035260  001401                   BEQ     2$              ;V177644        ;BR IF OK
10011 035262  104447                   PERR23                  ;V177646        ;REPORT ERROR
10012
10013 035264  011200           2$:     MOV     (R2),R0         ;V177650        ;READ 'B'
10014 035266  020003                   CMP     R0,R3           ;V177652        ;CHECK 'B'
10015 035270  001401                   BEQ     3$              ;V177654        ;BR IF OK
10016 035272  104450                   PERR24                  ;V177656        ;REPORT ERROR
10017
10018 035274  062702  000400     3$:     ADD     #400,R2         ;V177660        ;BUMP 'B'
10019 035300  020205                   CMP     R2,R5           ;V177664        ;AT END YET?
10020 035302  101764                   BLOS    1$              ;V177666        ;BR IF NO
10021
10022 035304  062701  000002           ADD     #2,R1           ;V177670        ;BUMP 'A'
10023 035310  000137  035314           JMP     @#MTPB24        ;V177674        ;GOTO V177260
```

```
                                        MTPB24: SUBTST  <<MTPB24        FAST GALLOP PART B>>
                                        ;*********************************************************************************
                                        ;*SUBTEST        MTPB24  FAST GALLOP PART B
                                        ;*********************************************************************************
10027                                           ;SDPAR'S
10028 035314  010411                    MOV     R4,(R1)         ;V172260        ;WRITE 'A'
10029 035316  020105                    CMP     R1,R5           ;V172262        ;DONE?
10030 035320  001001                    BNE     1$              ;V172264        ;BR IF NO
10031 035322  000207                    RETURN                  ;V172266        ;YES - RETURN
10032 035324  000137  035330      1$:   JMP     @#MTPC24        ;V172270        ;GOTO V172360
1003>
10034 035330                            MTPC24: SUBTST  <<MTPC24        FAST GALLOP PART C>>
                                        ;*********************************************************************  ******
                                        ;*SUBTEST        MTPC24  FAST GALLOP PART C
                                        ;*********************************************************************************
10035                                           ;KDPAR'S
10036 035330  010102                    MOV     R1,R2           ;V172360        ;RESET 'B' <--- 'A'
10037 035332  011100                    MOV     (R1),R0         ;V172362        ;READ 'A'
10038 035334  020004                    CMP     R0,R4           ;V172364        ;CHECK 'A'
10039 035336  001401                    BEQ     1$              ;V172366        ;BR IF OK
10040 035340  104447                    PERR23                  ;V172370        ;REPORT ERROR
10041 035342  000137  035274      1$:   JMP     @#MTPA24+20     ;V172372        ;GOTO V177660
```

```
10044 035346                                MTP025: SUBTST  <<MTP025         INTERRUPT ENABLE TEST>>
                                            ;*****************************************************************************
                                            ;*SUBTEST        MTP025  INTERRUPT ENABLE TEST
                                            ;*****************************************************************************
10045 035346  005037  002244                        CLR     TSTDAT          ;GENERATE CHECKBITS ON 0,,0
10046 035352  005037  002246                        CLR     TSTDAT+2
10047 035356  012737  002244  002306                MOV     #TSTDAT,SOURCE
10048 035364  004737  044366                        CALL    CHKGEN
10049 035370  012737  000003  002074                MOV     #3,NOPAR                ;SETUP PARITY ACTION
10050 035376  012701  002406                        MOV     #TESTADD,R1     ;FIRST TEST ADDRESS
10051 035402  012737  035442  002300                MOV     #1$,PARTHERE    ;SETUP TRAP DESTINATION
10052 035410  004737  035664                        CALL    MTPA25          ;WRITE DATA & CHECKBITS
10053 035414  104473                                ECC1INIT                ;INITIALIZE 1 SELECTED MK11 CSR
10054 035416  005771  000000                        TST     @(R1)           ;ACCESS LOCATIONS FOR DBE TRAPS
10055 035422  005771  000002                        TST     @2(R1)
10056                                               ;NONE - GOOD - ACCESS FOR SBE TRAPS
10057 035426  104507                                ENA1SBE                 ;DISABLE TRAPS ON SBE'S FROM 1 SELECTED CSR
10058 035430  005771  000000                        TST     @(R1)
10059 035434  005771  000002                        TST     @2(R1)
10060 035440  000404                                BR      2$              ;NONE - GOOD - SKIP
10061 035442  104426                          1$:   READCSR
10062 035444                                        FATAL   27
10063 035452  005237  002244                  2$:   INC     TSTDAT          ;CHECK FOR CORRECT ACTION ON SBE'S
10064 035456  004737  035612                        CALL    MTPD25          ;IN ALL 4 BYTES
10065 035462  012737  000400  002244                MOV     #400,TSTDAT
10066 035470  004737  035612                        CALL    MTPD25
10067 035474  005037  002244                        CLR     TSTDAT
10068 035500  005237  002246                        INC     TSTDAT+2
10069 035504  004737  035612                        CALL    MTPD25
10070 035510  012737  000400  002246                MOV     #400,TSTDAT+2
10071 035516  004737  035612                        CALL    MTPD25
10072
10073 035522  005037  002246                        CLR     TSTDAT+2        ;CHECK FOR CORRECT ACTION ON DBE'S
10074 035526  012737  000003  002244                MOV     #3,TSTDAT       ;IN ALL 4 BYTES
10075 035534  004737  035634                        CALL    MTPE25
10076 035540  012737  001400  002244                MOV     #1400,TSTDAT
10077 035546  004737  035634                        CALL    MTPE25
10078 035552  005037  002244                        CLR     TSTDAT
10079 035556  012737  000003  002246                MOV     #3,TSTDAT+2
10080 035564  004737  035634                        CALL    MTPE25
10081 035570  012737  001400  002246                MOV     #1400,TSTDAT+2
10082 035576  004737  035634                        CALL    MTPE25
10083 035602  104503                                CLR1CSR                 ;CLEAR 1 SELECTED MK11 CSR
10084 035604  005037  002074                        CLR     NOPAR           ;INDICATE PARITY ACTION
10085 035610  000207                                RETURN
10086
10087 035612  004737  035664          MTPD25:       CALL    MTPA25          ;WRITE DATA & CHECKBITS
10088 035616  104471                                ECC1DIS                 ;DISABLE ECC ON 1 SELECTED CSR
10089 035620  004737  035706                        CALL    MTPB25          ;CHECK FOR NO TRAPS
10090 035624  104507                                ENA1SBE                 ;DISABLE TRAPS ON SBE'S FROM 1 SELECTED CSR
10091 035626  004737  035746                        CALL    MTPC25          ;CHECK FOR EXPECTED TRAP
10092 035632  000207                                RETURN
```

```
10095 035634  004737  035664          MTPE25: CALL    MTPA25          ;WRITE DATA & CHECKBITS
10096 035640  104471                          ECC1DIS                 ;DISABLE ECC ON 1 SELECTED CSR
10097 035642  004737  035706                  CALL    MTPB25          ;CHECK FOR NO TRAPS
10098                                         ;ENABLE DBE TRAPS
10099 035646  104473                          ECC1INIT                ;INITIALIZE 1 SELECTED MK11 CSR
10100 035650  004737  035746                  CALL    MTPC25          ;CHECK FOR EXPECTED TRAP
10101 035654  104507                          ENA1SBE                 ;DISAB'E TRAPS ON SBE'S FROM 1 SELECTED CSR
10102 035656  004737  035746                  CALL    MTPC25          ;CHECK FOR EXPECTED TRAP
10103 035662  000207                          RETURN
10104
10105                                         ;WRITE TSTDAT & TSTDAT+2 & CHECKBITS
10106 035664  104471                  MTPA25: ECC1DIS                 ;DISABLE ECC ON 1 SELECTED CSR
10107 035666  013771  002244  000000          MOV     TSTDAT,@(R1)    ;WRITE FIRST 16 BITS
10108 035674  104475                          CB1CSR                  ;WRITE GENERATED CHECKBITS IN 1 SELECTED CSR
10109 035676  013771  002246  000002          MOV     TSTDAT+2,@2(R1) ;WRITE 2ND 16 BITS & CHECKBITS
10110 035704  000207                          RETURN
10111
10112                                         ;CHECK FOR NO TRAP OCCURING CONDITION
10113 035706  012737  035726  002300  MTPB25: MOV     #1$,PARTHERE    ;SETUP TRAP DESTINATION
10114 035714  005771  000000                  TST     @(R1)           ;ACCESS LOCATIONS
10115 035720  005771  000002                  TST     @2(R1)
10116 035724  000207                          RETURN                  ;NO TRAP - GOOD - RETURN
10117
10118 035726  104426                  1$:     READCSR
10119 035730  011137  002032                  MOV     (R1),ADDRESS    ;SAVE VIRTUAL ADDRESS
10120 035734  104024                          ERROR   +24
10121 035736                                  SET     HEADER
10122 035744  000207                          RETURN
10123
10124                                         ;TRAP SHOULD OCCURE TEST
10125 035746  012737  035762  002300  MTPC25: MOV     #1$,PARTHERE    ;SETUP TRAP DESTINATION
10126 035754  005771  000000                  TST     @(R1)           ;ACCESS 1ST LOCATION
10127 035760  000405                          BR      2$              ;NO TRAP - BAD NEWS - SKIP
10128 035762  012737  036012  002300  1$:     MOV     #3$,PARTHERE    ;SETUP TRAP DESTINATION
10129 035770  005771  000002                  TST     @2(R1)          ;ACCESS 2ND LOCATION
10130 035774  104426                  2$:     READCSR                 ;NO TRAP - BAD NEWS
10131 035776  011137  002032                  MOV     (R1),ADDRESS    ;SAVE VIRTUAL ADDRESS
10132 036002  104025                          ERROR   +25
10133 036004                                  SET     HEADER
10134 036012  000207                  3$:     RETURN
10135
```

```
                                MTPA26: SUBTST  <<MTPA26        RANDOM DATA (WRITE)>>
                                ;*********************************************************************************
                                ;*SUBTEST        MTPA26  RANDOM DATA (WRITE)
                                ;*********************************************************************************
10139 036014  000137  036064   1$:     JMP     @#MTPC26        ;V177640        GOTO V172360
10140 036020  010221           1$:     MOV     R2,(R1)+        ;V177644
10141 036022  010321           1$:     MOV     R3,(R1)+        ;V177646
10142 036024  077005           1$:     SOB     R0,1$           ;V177650
10143 036026  000207           1$:     RETURN                  ;V177652
10144
10145 036030                    MTPB26: SUBTST  <<MTPB26        RANDOM DATA (READ)>>
                                ;*********************************************************************************
                                ;*SUBTEST        MTPB26  RANDOM DATA (READ)
                                ;*********************************************************************************
10146                                   .DSABL  AMA
10147                                   .ENABL  LSB
10148 036030  000137  036064   1$:     JMP     @#MTPC26        ;V177640        GOTO V172360
10149 036034  020221           1$:     CMP     R2,(R1)+        ;V177644
10150 036036  001401           1$:     BEQ     2$              ;V177646
10151 036040  104451           1$:     PERR25                  ;V177650
10152 036042  005127           2$:     COM     (PC)+           ;V177652
10153 036044  000000           RANODD: 0                       ;V177654        FOR ERROR REPORTING
10154 036046  020321           1$:     CMP     R3,(R1)+        ;V177656
10155 036050  001401           1$:     BEQ     3$              ;V177660
10156 036052  104451           1$:     PERR25                  ;V177662
10157 036054  005167  177764   3$:     COM     RANODD          ;V177664
10158 036060  077015           1$:     SOB     R0,1$           ;V177670
10159 036062  000207           1$:     RETURN                  ;V177672
10160                                   .DSABL  LSB
10161                                   .ENABL  AMA
10162
10163 036064                    MTPC26: SUBTST  <<RANDOM NUMBER SUBPROGRAM>>
                                ;*********************************************************************************
                                ;*SUBTEST        RANDOM NUMBER SUBPROGRAM
                                ;*********************************************************************************
10164                           ;       :CALLER MUST SETUP
10165                           ;       ;       MOV     SEEDLO,R3
10166                           ;       ;       MOV     SEEDHI,R2
10167                           ;       ;       MOV     R3,R5
10168                           ;       ;       MOV     R2,R4
10169 036064  073427  000007           ASHC    #7,R4           ;V172360
10170 036070  060305                   ADD     R3,R5           ;V172364
10171 036072  005504                   ADC     R4              ;V172366
10172 036074  060204                   ADD     R2,R4           ;V172370
10173 036076  062705  001057           ADD     #1057,R5        ;V172372
10174 036102  000240                   NOP                     ;V172376        GOTO V172260
10175
10176 036104                    MTPD26: SUBTST  <<RANDOM NUMBER SUBSUBPROGRAM>>
                                ;*********************************************************************************
                                ;*SUBTEST        RANDOM NUMBER SUBSUBPROGRAM
                                ;*********************************************************************************
10177 036104  005504                   ADC     R4              ;V172260
10178 036106  062704  047401           ADD     #47401,R4       ;V172262
10179 036112  010503                   MOV     R5,R3           ;V172266
10180 036114  010402                   MOV     R4,R2           ;V172270
10181 036116  000137  036020           JMP     @#MTPA26+4      ;V172272        GOTO V177644
```

```
10184 036122                            MTP030: SUBTST  <<MT0030        FLUSH OUT DBE'S>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       MT0030  FLUSH OUT DBE'S
                                        ;*****************************************************************************
10185 036122  011002                    1$:     MOV     (R0),R2         ;V177640
10186 036124  010220                            MOV     R2,(R0)+        ;V177642
10187 036126  077103                            SOB     R1,1$           ;V177644
10188 036130  000207                            RETURN                  ;V177646
10189
10190 036132                            MTP031: SUBTST  <<MTP031        SOB-A-LONG TEST>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       MTP031  SOB-A-LONG TEST
                                        ;*****************************************************************************
10191                                           .DSABL  AMA
10192 036132  000000                            0                       ;MOVE TERMINATOR
10193 036134  077001                    1$:     SOB     R0,1$           ;SOB TILL R0 UNDERFLOWS
10194 036136  005167  177772                    COM     1$              ;WRITE COMPLEMENT OF SOB
10195 036142  020167  177766                    CMP     R1,1$           ;READ & CHECK FOR NOT "SOB R0,DOT"
10196 036146  001403                            BEQ     2$              ;OK - SKIP
10197 036150  104454                            PERR30
10198 036152  010167  177756                    MOV     R1,1$
10199 036156  005167  177752            2$:     COM     1$              ;CORRECT SOB INSTRUCTION
10200 036162  010200                            MOV     R2,R0           ;REINITIALIZE SOB CONSTANT
10201                                           ;UPDATE MOVE REGISTERS
10202 036164  010503                            MOV     R5,R3
10203 036166  005725                            TST     (R5)+           ;BUMP (SAFELY) BY 2
10204 036170  010504                            MOV     R5,R4
10205 036172  020537  002516                    CMP     R5,@#LINK1      ;DONE?
10206 036176  001001                            BNE     3$              ;NO - SKIP
10207 036200  000207                            RETURN                  ;YES
10208
10209 036202  014344                    3$:     MOV     -(R3),-(R4)
10210 036204  001376                            BNE     3$
10211 036206  000752                            BR      1$
10212         000056                    SOBLENGTH=.-MTP031
10213                                           .ENABL  AMA
```

```
 10241 036210                           MTP032: SUBTST  <<MTP032        WRITE RECOVERY TEST>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       MTP032  WRITE RECOVERY TEST
                                        ;*****************************************************************************
 10242                                          ;THE TEST ACTUALLY EXECUTED ALREADY IN THE MEMORY UNDER TEST.
 10243                                          ;THIS CODE INSURES THAT IT CHANGED MEMORY TO HAVE
 10244                                          ;1/2 BANK OF #5141      WHICH IS A "COM -(R1)" INSTRUCTION AND
 10245                                          ;1/2 BANK OF #110       WHICH IS A "JMP (R0)"   INSTRUCTION.
 10246
 10247 036210  012401                  1$:     MOV     (R4)+,R1        ;V177640        ;GET DATA FROM LOWER 1/2 BANK
 10248 036212  020102                          CMP     R1,R2           ;V177642        ;IS IT #5141?
 10249 036214  001401                          BEQ     2$              ;V177644        ;YES - SKIP
 10250 036216  104430                          PERR02                  ;V177646        ;NO - TAKE ERROR TRAP
 10251 036220  077305                  2$:     SOB     R3,1$           ;V177650        ;LOOP FOR 1/2 BANK
 10252 036222  013703  002516                  MOV     @#LINK1,R3      ;V177652        ;RESTORE LOOP SIZE
 10253 036226  012400                  3$:     MOV     (R4)+,R0        ;V177656        ;GET DATA FROM UPPER 1/2 BANK
 10254 036230  020005                          CMP     R0,R5           ;V177660        ;IS IT #110?
 10255 036232  001401                          BEQ     4$              ;V177662        ;YES - SKIP
 10256 036234  104427                          PERR01                  ;V177664        ;NO- TAKE ERROR TRAP
 10257 036236  077305                  4$:     SOB     R3,3$           ;V177666        ;LOOP FOR 1/2 BANK
 10258 036240  000207                          RETURN
```

```
10261 036242                            MTP033: SUBTST  <<MTP033         BRANCH GOBBLE TEST>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       MTP033  BRANCH GOBBLE TEST
                                        ;*****************************************************************************
10262                                           .DSABL  AMA
10263 036242  000000                            0                       ;MOVE TERMINATOR
10264 036244  000000                    BGTEST: 0                       ;TEST WORD (TWO BYTES)
10265 036246  000261                    BRGOBB: SEC                     ;SET CARRY (TO BE ADDED TO 'BGTEST')
10266 036250  105511                            ADCB    (R1)            ;INCREMENT LOW BYTE OF 'BGTEST'
10267 036252  100402                            BMI     1$              ;BRANCH WHEN BIT7 IS SET
10268 036254  105212                            INCB    (R2)            ;INCREMENT HIGH BYTE OF 'BGTEST'
10269 036256  000773                            BR      BRGOBB          ;LOOP 128 TIMES
10270
10271                                           ;NOW CHECK FOR CORRECT CONDITION CODES
10272 036260  102401                    1$:     BVS     2$              ;BR IF V-BIT SET (SHOULD BE)
10273 036262  104461                            PERR35                  ;NO - REPORT ERROR AND ABORT TEST
10274                                                                   ;COND CODES NOT EQUAL TO 1010
10275 036264  000242                    2$:     CLV                     ;CLEAR V-BIT
10276 036266  105212                            INCB    (R2)            ;INCREMENT HIGH BYTE OF 'BGTEST' ONCE MORE
10277 036270  103402                            BCS     3$              ;BR IF C-BIT SET (SHOULD NOT BE)
10278 036272  102001                            BVC     3$              ;BR IF V-BIT CLEAR (SHOULD NOT BE)
10279 036274  100401                            BMI     4$              ;BR IF N-BIT SET (SHOULD BE)
10280 036276  104461                    3$:     PERR35                  ;NO - REPORT ERROR AND ABORT TEST
10281                                                                   ;COND CODES NOT EQUAL TO 1010
10282
10283                                           ;UPDATE TEST POINTERS
10284 036300  010701                    4$:     MOV     PC,R1
10285 036302  162701  000036            5$:     SUB     #5$-BGTEST,R1
10286 036306  010102                            MOV     R1,R2
10287 036310  005202                            INC     R2
10288
10289                                           ;UPDATE MOVE REGISTERS
10290 036312  010503                            MOV     R5,R3
10291 036314  005725                            TST     (R5)+           ;BUMP (SAFELY) BY 2
10292 036316  010504                            MOV     R5,R4
10293
10294                                           ;DONE?
10295 036320  020537  002516                    CMP     R5,@#LINK1      ;DONE?
10296 036324  001001                            BNE     6$              ;NO - SKIP
10297 036326  000207                            RETURN                  ;YES - RETURN
10298
10299                                           ;MOVE CODE 1 LOCATION
10300 036330  014344                    6$:     MOV     -(R3),-(R4)
10301 036332  001376                            BNE     6$
10302 036334  005011                            CLR     (R1)            ;CLEAR TEST WORD 'BGTEST'
10303 036336  000743                            BR      BRGOBB          ;RUN MOVED CODE AGAIN
10304         000076                    GBLENGTH=.-MTP033
10305                                           .ENABL  AMA
```

```
10307 036340                    MTP034: SUBTST  <<MTP034         SOFT ERROR - BACKROUND PATTERN TEST>>
                                ;*****************************************************************************
                                ;*SUBTEST        MTP034  SOFT ERROR - BACKROUND PATTERN TEST
                                ;*****************************************************************************
10308 036340  010220            1$:     MOV     R2,(R0)+        ;V177640
10309 036342  077102                    SOB     R1,MTP034       ;V177642
10310 036344  000207                    RETURN                  ;V177644
10311 036346  012401            2$:     MOV     (R4)+,R1        ;V177646
10312 036350  020102                    CMP     R1,R2           ;V177650
10313 036352  001402                    BEQ     3$              ;V177652
10314 036354  104430                    PERR02                  ;V177654
10315 036356  000240                    NOP                     ;V177656
10316 036360  077306            3$:     SOB     R3,2$           ;V177660
10317 036362  000207                    RETURN                  ;V177662
```

```
10319 036364                                MTP035:SUBTST   <<MTP035        WORST CASE NOISE PARITY TEST>>
                                            ;*****************************************************************************
                                            ;*SUBTEST       MTP035  WORST CASE NOISE PARITY TEST
                                            ;*****************************************************************************
10320 036364  012737  000003  002074              MOV     #3,NOPAR        ;SET PARITY TRAPS TO RETURN TO 'PARTHERE'
10321
10322 036372                                      FOR R0 := #FIRST TO #LAST BY #4000
10323 036376  012737  000005  002146                MOV   #BIT2!BIT0,CSR  ;SET WRITE WRONG PARITY & PAR. TRAPS INTO CSR
10324 036404  104425                                LOADCSR
10325 036406  012737  036442  002300                MOV   #1$,PARTHERE
10326 036414  011010                                MOV   (R0),(R0)       ;WWP TEST LOCATION
10327 036416  005710                                TST   (R0)
10328 036420  010037  002032                        MOV   R0,ADDRESS
10329 036424  104050                                ERROR +50
10330 036426  004737  057476                        CALL  PERBNK
10331 036432  032763  002000  002652                BIT   #BIT10,CONFIG+2(R3)
10332 036440  001002                                BNE   2$
10333 036442  104426                          1$:   READCSR
10334 036444  104512                                ERRGEN
10335
10336 036446  104503                          2$:   CLR1CSR
10337 036450  011010                                MOV   (R0),(R0)       ;CLEAR WRONG PARITY IN MEMORY
10338 036452  012737  000001  002146                MOV   #BIT0,CSR
10339 036460  104425                                LOADCSR
10340 036462  012737  036474  002300                MOV   #3$,PARTHERE
10341 036470  005710                                TST   (R0)
10342 036472  000405                                BR    4$
10343 036474  010037  002032                  3$:   MOV   R0,ADDRESS
10344 036500  104050                                ERROR +50
10345 036502  004737  057476                        CALL  PERBNK
10346 036506                                  4$:   END; OF FOR
10347
10348 036520  005037  002074                    CLR     NOPAR           ;RESET PARITY TRAP ACTION
10349 036524  000207                            RETURN
```

```
10351 036526                          MTP036: SUBTST          <<MTP036        CORRECTION CODE TEST>>
                                      ;*****************************************************************************
                                      ;*SUBTEST       MTP036  CORRECTION CODE TEST
                                      ;*****************************************************************************
10352                                         ;
10353                                         ; THIS TEST CHECKS TO SEE THAT EACH BIT OF A DATA WORD
10354                                         ; CAN BE CORRECTED INDIVIDUALLY FROM A ZERO TO A ONE AND
10355                                         ; VISA VERSA.
10356 036526  104424                        CACHOFF                         ;TURN OFF CACHE
10357 036530  105037  002262                CLRB PASFLG                     ;CLEAR PASFLG
10358 036534  104513                        CBREG                           ;ENABLE CHECK/SYNDROME BIT REGISTER
10359 036536                                REPEAT                          ;
10360 036536                                  LET PASFLG :B= PASFLG + #1     ;INCREMENT LOOP COUNTER
10361 036542                                  LET R4  := #-1                ;INDEX TO SINGLE BIT ERROR TABLE
10362 036546                                  LET BITNO := #0               ;CLEAR INNER LOOP COUNTER
10363 036552                                  IFB PASFLG EQ #1              ;SELECT DATA TO BE CORRECTED BY PASSNO
10364 036562                                     LET R5 := #1               ;DATA=0;BIT TO BE CORRECTED IS A ONE
10365 036566                                  ELSE                          ;
10366 036570                                     LET R5 := #177776          ;DATA=177776;BIT TO BE CORRECTED IS A ZERO
10367 036574                                  END                           ;
10368 036574                                  REPEAT                        ;
10369
10370 036574  005237  002320                   INC BITNO                    ;INCREMENT BIT POINTER
10371 036600                                   LET R4 := R4 + #1            ;POINT TO NEXT SET OF CHECK BITS
10372 036602                                   LET R2 :B= PTABLE(R4)        ;GET NEXT SET OF CHECK BITS
10373 036606  072227  000005                   ASH #5,R2                    ;SHIFT TO LINE UP IN CSR
10374 036612  052702  000004                   BIS #BIT2,R2                 ;ENABLE DIAG MODE
10375 036616                                   LET CSR := R2                ;GET CHECK BITS TO BE WRITTEN
10376 036622  104425                           LOADCSR                      ;LOAD CSR WITH DATA
10377 036624                                   LET (R1) := R0               ;WRITE DATA TO TEST ADDRESS
10378 036626  005711                           TST (R1)                     ;CORRECT SBE
10379 036630                                   IF (R1) NE R5                ;WAS DATA CORRECTED???
10380 036634                                      LET ADDRESS := #60000     ;MOV ERROR INFORMATION IN
10381 036642                                      LET CHECK := R2           ;
10382 036646                                      LET TSTDAT := R5          ;
10383 036652                                      LET TSTDAT+2 := (R1)      ;
10384 036656  104052                              ERROR +52                 ;NO ERROR
10385 036660                                   END                          ;
10386 036660  005011                           CLR (R1)                     ;CLEAR LUT
10387 036662                                   IFB PASFLG EQ #1             ;SHIFT NEW DATA DEPENDING ON PASFLG
10388 036672  006305                              ASL  R5                   ;SHIFT BITNO TO THE LEFT
10389 036674                                   ELSE                         ;
10390 036676  000261                              SEC                       ;SET CARRY BIT AND......
10391 036700  006105                              ROL   R5                  ;ROTATE LEFT
10392 036702                                   END                          ;
10393 036702                                 UNTIL BITNO EQ #16.            ;UNTIL ALL BITS ARE DONE
10394 036712  005100                           COM R0                       ;COMPLEMENT DATA AND REPEAT
10395 036714                                UNTILB PASFLG EQ #2             ;UNTIL 2 PASSES ARE COMPLETE!
10396 036724  104503                        CLR1CSR                         ;CLEAR CSR
10397 036726  104423                        CACHON                          ;TURN CACHE
10398 036730  000207                        RETURN                          ;
10399                                         ;
10400                                         ;       MS11-P SINGLE BIT ERROR CHECK BIT TABLE
10401                                         ;
10402 036732     002     007     037  PTABLE: .BYTE   2,7,37,31,32,25,26,20,57,51,52,45,46,40,75,70
      036735     031     032     025
      036740     026     020     057
```

MTP036 CORRECTION CODE TEST

```
        036743      051     052     045
        036746      046     040     075
        036750      070
```

```
 10404 036752                          MTP037: SUBTST  <<MTP037        CHECK ECC DISABLE TEST>>
                                       ;*****************************************************************************
                                       ;*SUBTEST       MTP037  CHECK ECC DISABLE TEST
                                       ;*****************************************************************************
 10405                                         ;
 10406                                         ; THIS TEST CHECKS THAT ECC CAN BE DISABLED AND THAT
 10407                                         ; NO CORRECTION TAKES PLACE WITH ECC DISABLED.
 10408                                         ;
 10409 036752  104424                          CACHOFF                 ;TURN OFF CACHE
 10410 036754                                  LET GOOD := #0          ;GOOD DATA FOR ERROR PRINT OUT
 10411 036760                                  LET CHECK := #0         ;CLEAR CHECK BIT FIELD
 10412 036764  104475                          CB1CSR                  ;ENABLE SYNDROME/CHECK BIT REGISTER
 10413 036766                                  LET CHECK := #100        ;SBE CHECK BITS
 10414 036774  104475                          CB1CSR                  ;WRITE CHECK BITS TO CB REGISTER
 10415 036776                                  LET (R1) := #0          ;WRITE CHECK BITS TO MEMORY
 10416 037000                                  IF (R1) NE #0           ;WAS CORRECTION MADE????
 10417 037004                                     LET BAD := (R1)      ;YES IT WAS.....ERROR
 10418 037010                                     LET ADDRESS := #60000 ;
 10419 037016  104037                             ERROR +37            ;
 10420 037020                                  END                     ;
 10421 037020  104423                          CACHON                  ;TURN ON CACHE
 10422 037022  000207                          RETURN                  ;
```

```
10425 037024                            MTP041: SUBTST  <<MTP041        ADDRESS TO CSR ON DOUBLE BIT ERROR TEST>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       MTP041  ADDRESS TO CSR ON DOUBLE BIT ERROR TEST
                                        ;*****************************************************************************
10426                                   ;
10427                                   ; THIS TEST CHECKS TO SEE IF THE CORRECT ADDRESS APPEARS
10428                                   ; IN CSR BITS 5-11 ON A DOUBLE ERROR.
10429                                   ;
10430 037024                                  LET R4 := BANK            ;GET STARTING BANK NUMBER
10431 037030  072427  000011                  ASH #9.,R4                ;SHIFT INTO POSTION TO MATCH ADDRESS IN CSR
10432 037034  042704  170037                  BIC #^C7740,R4            ;CLEAR OFF EXTRANEOUS BITS
10433 037040                                  LET R0 := #-40            ;INIT CSR ADDRESS TO 0 - 1K (BIT 5 = 1K ADD.)
10434 037044                                  LET R1 := #FIRST - #4000 ;GET LOW ADDRESS IN BANK
10435 037054                                  LET PASFLG :B= #0         ;INIT PASFLG
10436 037060  005037  002310                  CLR CHECK                 ;CLEAR CHECK BIT FIELD TO BE LOADED
10437 037064  104475                          CB1CSR                    ;ENABLE CHECK/SYNDROME BIT REGISTER
10438 037066                                  REPEAT
10439 037066  105237  002262                    INCB  PASFLG            ;INC LOOP COUNTER
10440 037072                                    LET R0 := R0 + #40      ;INC CSR ADDRESS TO BE EXPECTED
10441 037076                                    LET R1 := R1 + #4000                                          ;
10442 037102                                    LET CHECK := #1340      ;DOUBLE ERROR CHECK BITS
10443 037110  104475                            CB1CSR                  ;WRITE DOUBLE ERROR CHECK BITS
10444 037112                                    LET (R1) := #0          ;WRITE DATA AND D.E. CHK BITS AT A=0
10445 037114  104503                            CLR1CSR                 ;CLEAR CSR
10446 037116  005711                            TST (R1)                ;READ ADDRESS TO GET DOUBLE ERROR
10447 037120  104426                            READCSR                 ;READ CSR FOR CORRECT ADDRESS
10448 037122                                    LET R5 := CSR           ;
10449 037126  042705  170037                    BIC #^C7740,R5          ;
10450 037132                                    LET R2 := R0            ;GET CORRECT ADDRESS
10451 037134  060402                            ADD R4,R2               ;ADD STARTING BANK TO DOUBLE BIT ADDRESS
10452 037136  000240                            NOP                     ;DEBUG AIDE
10453 037140                                    IF R2 NE R5             ;DO ADDRESSES AGREE?
10454 037144                                       LET BAD := R2        ;
10455 037150                                       LET GOOD := R5       ;
10456 037154  104455                               PERR31               ;NO ERROR
10457 037156                                    END                     ;
10458 037156                                    LET (R1) := #0          ;
10459 037160  104475                            CB1CSR                  ;ENABLE CHECK/SYNDROME BIT REGISTER
10460 037162                                UNTILB PASFLG EQ #16.       ;DO 16K AT A TIME
10461 037172  104503                        CLR1CSR                     ;
10462 037174  000207                        RETURN                      ;
10463
```

```
10466 037176                            MTP042: SUBTST  <<MTP042        EXTENDED ADDRESS TO CSR ON ERROR TEST>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       MTP042  EXTENDED ADDRESS TO CSR ON ERROR TEST
                                        ;*****************************************************************************
10467                                   ;
10468                                   ; THIS TESTS THE EXTENDED UNIBUS ADDRESS IN THE
10469                                   ; CSR BY CAUSING A SINGLE ERROR, ENABLING BIT # 14, THEN CHECKING
10470                                   ; FOR THE PROPER ADDRESS IN THE CSR.
10471                                   ;
10472 037176  104424                          CACHOFF                 ;TURN OFF CACHE MEMORY
10473 037200                                  LET R4 := BANK          ;GET BANK NUMBER TO FIGURE OUT EXTENDED ADDRESS
10474 037204  042704  177607                  BIC #^C170,R4           ;CLEAR OFF LOWER BITS
10475 037210  072427  000002                  ASH #2,R4               ;SHIFT TO LINE UP WITH CSR
10476 037214  052704  040000                  BIS #BIT14,R4           ;SET EXTENDED ADDRESS BIT
10477 037220  062737  000400  172352          ADD #400,KIPAR5         ;SET UP PAR TO POINT TO TOP OF A BANK
10478 037226                                  LET PASFLG :B= #0       ;INIT LOOP COUNTER
10479 037232                                  LET R5 := BANK          ;R5 GETS THE BANK NUMBER
10480 037236  042705  177770                  BIC #^C7,R5             ;CLEAR ALL BUT THE LOWER BITS
10481 037242  072527  000011                  ASH #9.,R5              ;ROTATE INTO POSTION
10482 037246  052705  000020                  BIS #BIT4,R5            ;SET UP SBE INDICATOR ;;DATA TO BE EXPECTED
10483 037252  104513                          CBREG                   ;ENABLE CHECK/SYNDPOME BIT REGISTER
10484 037254                                  REPEAT
10485 037254  105237  002262                      INCB PASFLG         ;INCR LOOP COUNTER
10486 037260                                      LET CSR := #104     ;WRITE CHECK BITS TO CSR WITH DIAG MODE
10487 037266  104425                              LOADCSR             ;LOAD CSR WITH DATA
10488 037270                                      LET (R1) := #0      ;WRT ZEROS AT A=0 AND SINGLE ERROR BITS
10489 037272  104503                              CLR1CSR             ;CLEAR CSR
10490 037274  005711                              TST (R1)            ;READ A=0;DATA BIT 0 SHOULD BE CORRECTED TO A 1
10491 037276  104426                              READCSR             ;READ CSR FOR DATA
10492 037300  042737  020000  002146              BIC #BIT13,CSR      ;CLEAR POSSIBLE INHIBIT MODE IN DATA "CSR"
10493 037306                                      IF CSR NE R5 THEN   ;HAS SINGLE ERROR BITS SET IN CSR?
10494 037314                                         LET BAD := CSR   ;
10495 037322                                         LET GOOD := R5   ;
10496 037326  104023                                 ERROR +23        ;
10497 037330                                      END                 ;
10498 037330                                      LET CSR := #40000   ;WRITE EUB BIT TO CSR
10499 037336  104425                              LOADCSR             ;
10500 037340  104426                              READCSR             ;READ FOR CORRECT EXTENDED UNIBUS ADDRESS
10501 037342  042737  020000  002146              BIC #BIT13,CSR      ;CLEAR INHIBIT MODE POINTER IN DATA
10502 037350                                      IF CSR NE R4 THEN   ;READ EUB ADDRESS
10503 037356                                         LET BAD := CSR   ;
10504 037364                                         LET GOOD := R4   ;
10505 037370                                         SET HEADER       ;
10506 037376  104023                                 ERROR +23        ;
10507 037400                                      END                 ;
10508 037400                                      LET (R1) := #0      ;CLEAR LUT
10509 037402                                      LET R1 := #137776   ;SET UP NEW ADDRESS
10510 037406  062705  000740                      ADD #740,R5         ;ADD TO GET NEW ADDRESS
10511 037412  104513                              CBREG               ;ENABLE CHECK/SYNDROME BIT REGISTER
10512 037414                                  UNTILB PASFLG EQ #2     ;LOOP 2 TIMES
10513 037424  104503                          CLR1CSR                 ;CLEAR CSR
10514 037426  104423                          CACHON                  ;TURN  ON CACHE
10515 037430  000207                          RETURN
10516
10517
```

```
10519 037432                            MTP043: SUBTST <<MTP043 WRITE BYTE CLEARS SINGLE BIT ERROR TEST>>
                                        ;*********************************************************************************
                                        ;*SUBTEST       MTP043  WRITE BYTE CLEARS SINGLE BIT ERROR TEST
                                        ;*********************************************************************************
10520                                           ;
10521                                           ; THIS TEST CHECKS TO SEE IF A SINGLE BIT ERROR WILL BE CORRECTED DURING
10522                                           ; THE READ PORTION OF A WRITE BYTE AND THAT THE CORRECT CHECK BITS WILL
10523                                           ; BE GENERATED ON A WRITE.
10524                                           ;
10525 037432  104424                            CACHOFF                 ;TURN OFF CACHE
10526 037434  104513                            CBREG                   ;ENABLE CHECK/SYNDROME BIT REGISTER
10527 037436  105037  002262                    CLRB PASFLG             ;CLEAR LOOP COUNTER
10528 037442                                    LET R2 := R1 + #1       ;R2 POINTS TO HIGH BYTE
10529 037446                                    LET R4 := #1            ;INITIAL DATA = 1
10530 037452                                    REPEAT                  ;
10531 037452  105237  002262                        INCB PASFLG         ;INCREMENT LOOP COUNTER
10532 037456                                        LET CSR := #604     ;WRITE CHECK BITS CORRESPONDING TO DATA OF 0
10533 037464  104425                                LOADCSR             ;WRITE CSR
10534 037466                                        LET (R1) := R4      ;WRITE DATA OF 1 CREATING A SINGLE BIT ERROR
10535 037470  104503                                CLR1CSR             ;WRITE CSR TO NORMAL MODE
10536 037472                                        LET (R2) :B= #377   ;WRITE BYTE OF WORD
10537 037476  104426                                READCSR             ;READ CSR
10538 037500  042737  177757  002146                BIC #^C20,CSR       ;SEE IF SBE INDICATOR IS SET
10539 037506                                        IF CSR NE #20       ;IS SBE SET?????
10540 037516                                            LET GOOD := #20 ;
10541 037524                                            LET BAD := CSR  ;
10542 037532  104060                                    ERROR +60       ;
10543 037534                                        END                 ;
10544 037534  104513                                CBREG               ;WRITE CSR TO DIAG MODE
10545 037536  005711                                TST (R1)            ;READ SA0 FOR CORRECT CHECK BITS
10546 037540  104426                                READCSR             ;READ CSR
10547 037542  042737  174037  002146                BIC #^C3740,CSR     ;MASK OUT CHECK BIT FIELD
10548 037550                                        IF CSR NE #300      ;WERE CORRECT CHECK BITS GENERATED?????
10549 037560                                            SET HEADER      ;
10550 037566                                            LET GOOD := #300;
10551 037574                                            LET BAD := CSR  ;
10552 037602  104061                                    ERROR +61       ;
10553 037604                                        END                 ;
10554 037604  005302                                DEC R2              ;POINT TO HIGH BYTE AND REPEAT
10555 037606                                        LET R4 := #400      ;BIT 0 OF HIGH BYTE
10556 037612                                    UNTILB PASFLG EQ #2     ;DO HIGH AND LOW BYTE
10557 037622  104423                            CACHON                  ;TURN ON CACHE
10558 037624  000207                            RETURN                  ;
```

```
10560 037626                            MTP044: SUBTST  <<MTP044        SHIFTING CHECK BITS THROUGH THE CSR TEST>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       MTP044  SHIFTING CHECK BITS THROUGH THE CSR TEST
                                        ;*****************************************************************************
                                        ;
10561                                   ; THIS TEST CHECKS THE ABILITY TO READ AND WRITE CHECKBITS INTO MEMORY
10562                                   ; BY SHIFTING A ONE BIT THROUGH A FIELD OF ZEROS. THE CSR IS READ FOR THE
10563                                   ; CORRECT PATTERNS. THE TEST IS THEN REPEATED ON A ZERO BIT THROUGH A
10564                                   ; FIELD OF ALL ONES.
10565                                   ;
10566
10567 037626  104424                        CACHOFF                         ;TURN OFF CACHE
10568 037630                                LET PASFLG :B= #0               ;INIT PASFLG
10569 037634                                LET R5 := #174037               ;CHECK BIT MASK FOR CSR
10570 037640  104475                        CB1CSR                          ;ENABLE CHECK/SYNDROME BIT REGISTER
10571 037642                                LET R2 := #46                   ;SET UP INITIAL CSR DATA
10572 037646                                REPEAT
10573 037646                                   LET PASFLG :B= PASFLG + #1    ;INC LOOP COUNTER
10574                                                                       ;CHK BITS = 1
10575                                                                       ;DISABLE ECC;DIAG CHK SET
10576 037652                                   LET PASSNO := #0             ;INIT PASSNO(INNER LOOP COUNTER)
10577 037656                                   REPEAT
10578 037656                                     LET PASSNO := PASSNO + #1  ;INC LOOP COUNTER
10579 037662                                     LET R4 := R2               ;COPY R2 TO R4
10580 037664                                     LET CSR := R2              ;GET CSR DATA TO BE WRITTEN
10581 037670  104425                             LOADCSR                    ;WRITE SBE CHECK BITS TO CSR
10582 037672                                     LET (R1) := #0             ;WRITE DATA AND CHECK BITS AT A=0
10583 037674  005105                             COM R5                     ;COMPLEMENT MASK
10584 037676  010546                             MOV R5,-(SP)               ;SAVE R5 ON STACK
10585 037700  040416                             BIC R4,(SP)                ;CREATE AN XOR FUNCTION
10586 037702  040504                             BIC R5,R4                  ;
10587 037704  052604                             BIS (SP)+,R4               ;
10588 037706                                     LET CSR := R4              ;
10589 037712  104425                             LOADCSR                    ;LOAD CSR WITH COMPLEMENT CHECK BITS
10590 037714  104426                             READCSR                    ;READ CSR FOR COMPLEMENT CHECK BITS
10591 037716                                     LET R3 := CSR              ;COPY CSR DATA TO R3
10592 037722  042703  020000                     BIC #BIT13,R3              ;CLEAR ANY POSSIBLE INHIBIT MODE POINTER
10593 037726                                     IF R3 NE R4 THEN           ;READ CSR FOR PROPER CHECK BITS
10594 037732                                        LET ADDRESS := #FIRST   ;
10595 037740                                        LET GOOD := R4          ;
10596 037744                                        LET BAD := R3           ;
10597 037750                                        SET HEADER              ;
10598 037756  104053                                ERROR +53               ;ERROR CALL
10599 037760                                     END
10600 037760  005105                             COM R5                     ;COMPLEMENT MASK
10601 037762  005711                             TST (R1)                   ;READ CHECK BITS AT A=0 INTO CSR
10602 037764  000240                             NOP                        ;
10603 037766  104426                             READCSR                    ;READ CSR FOR CORRECT CHECK BITS
10604 037770  040537  002146                     BIC R5,CSR                 ;MASK OUT CHECK BIT FIELD
10605 037774                                     LET R4 := R2               ;GET CHECK BITS THAT WERE WRITTEN
10606 037776  040504                             BIC R5,R4                  ;MASK OUT CHECK BIT FIELD
10607 040000                                     IF R4 NE CSR               ;ARE CHECK BITS THE SAME?
10608 040006                                        LET GOOD := R4          ;
10609 040012                                        LET BAD := CSR          ;
10610 040020                                        LET ADDRESS := #FIRST   ;
10611 040026                                        SET HEADER              ;
10612 040034  104054                                ERROR +54               ;ERROR CALL
10613 040036                                     END
```

```
10614 040036  040502                    BIC R5,R2                ;SHIFT CHECK BITS AND CREATE NEW DATA FOR CSR
10615 040040                            IFB PASFLG EQ #1          ;SELECT FUNCTION
10616                                                             ;DO A FIELD OF ZEROS--->ONES
10617 040050  006302                      ASL R2                  ;SHIFT CHECK BITS
10618 040052                            ELSE                      ;DO A FIELD OF ONES --->ZEROS
10619 040054  005105                      COM R5                  ;
10620 040056  010546                      MOV R5,-(SP)            ;TAKE OUT CHECK BIT FIELD
10621 040060  040216                      BIC R2,(SP)             ;
10622 040062  040502                      BIC R5,R2               ;
10623 040064  052602                      BIS (SP)+,R2            ;
10624 040066  006302                      ASL R2                  ;SHIFT CHECK BITS
10625 040070  010546                      MOV R5,-(SP)            ;PUT BACK CHECK BIT FIELD
10626 040072  040216                      BIC R2,(SP)             ;
10627 040074  040502                      BIC R5,R2               ;
10628 040076  052602                      BIS (SP)+,R2            ;
10629 040100  005105                      COM R5                  ;COMPLEMENENT DATA PATTERN
10630 040102                            END
10631 040102                            LET R2 := R2 + #6         ;ADD 6 SO THAT WRITE ON CSR WILL ENABLE DIAG MODE
10632 040106                          UNTILB PASSNO EQ #6         ;DO ALL CHECK BITS
10633 040116                          LET R2 := #3706             ;REPEAT WITH FIELD OF ONES
10634 040122                        UNTILB PASFLG EQ #2
10635 040132  104503                CLR1CSR                       ;
10636 040134  005011                CLR (R1)                      ;
10637 040136  104423                CACHON                        ;TURN ON CACHE
10638 040140  000207                RETURN                        ;
```

```
                                MTP045: SUBTST  <<MTP045        SYNDROMES TO CSR ON DOUBLE BIT ERROR TEST>>
                                ;*****************************************************************************
                                ;*SUBTEST       MTP045  SYNDROMES TO CSR ON DOUBLE BIT ERROR TEST
                                ;*****************************************************************************
                                ;
                                ;  THIS TEST CHECKS TO SEE IF THE DOUBLE BIT ERROR INDICATOR IS SET
                                ;  ON A DOUBLE BIT ERROR AND THE CORRECT SYNDROMES ARE LATCHED INTO THE
                                ;  CSR.  THIS TEST IS THEN REPEATED WITH MULTIPLE ERROR CHECK/SYNDROME BITS
                                ;
10646 040142  104424          CACHOFF                   ;TURN OFF CACHE
10647 040144  104513          CBREG                     ;ENABLE CHECK/SYNDROME BIT REGISTER
10648 040146                  LET PASSNO := #0          ;CLEAR LOOP COUNTER
10649 040152                  LET GOOD := #3744         ;GOOD DATA
10650 040160                  LET CSR := #3144          ;DBE CHECK BITS FOR CSR
10651 040166                  REPEAT                    ;
10652 040166  005237  002264      INC PASSNO            ;
10653 040172  104425          LOADCSR                   ;WRITE DBE CHECK BITS TO CSR
10654 040174                  LET (R1) := #0            ;WRITE ZEROS AND DBL ERROR CHK BITS A=0
10655 040176  104503          CLR1CSR                   ;CLEAR CSR OUT
10656 040200  005711          TST (R1)                  ;READ A=0 TO GET DOUBLE BIT ERROR
10657 040202  104426          READCSR                   ;WAS UNCORRECABLE ERROR BIT SET???
10658 040204                  IF #BIT15 OFF.IN CSR ;
10659 040214                     SET HEADER        ;
10660 040222                     LET BAD := CSR    ;
10661 040230  104063             ERROR +63         ;BIT NOT SET
10662 040232                  END                       ;
10663 040232  104513          CBREG                     ;ENABLE SYNDROME BIT REGISTER
10664 040234  104426          READCSR                   ;READ CSR FOR CORRECT SYNDROME BITS
10665 040236  000240          NOP                       ;DEBUG AIDE
10666 040240  042737  174033  002146  BIC #^C3744,CSR   ;MASK SYNDROMES OUT
10667 040246                  IF CSR NE GOOD THEN       ;CHECK IF DOUBLE ERROR BIT IS SET
10668 040256                     LET BAD := CSR         ;BAD DATA
10669 040264                     SET HEADER             ;
10670 040272  104042             ERROR +42              ;
10671 040274                  END                       ;
10672 040274  005011          CLR (R1)                  ;CLEAR LUT
10673 040276                  LET GOOD := #3604         ;REPEAT WITH MULTIPLE ERROR SYNDROMES
10674 040304                  LET CSR := #3004          ;MULTIPLE ERROR CHECK BITS
10675 040312                UNTIL PASSNO EQ #2          ;
10676 040322  104503          CLR1CSR                   ;
10677 040324  104423          CACHON                    ;
10678 040326  000207          RETURN
```

```
10680 040330                        MTP046: SUBTST  <<MTP046        CHECK SINGLE BIT ERRORS WITH ECC DISABLED>>
                                    ;*****************************************************************************
                                    ;*SUBTEST       MTP046  CHECK SINGLE BIT ERRORS WITH ECC DISABLED
                                    ;*****************************************************************************
10681                                       ;
10682                                       ; THIS TEST CHECKS TO SEE THAT FOR EACH BIT OF A DATA WORD THAT A SBE
10683                                       ; IS TREATED LIKE A UNCORRECTABLE ERROR WITH ECC DISABLED AND TRAPS
10684                                       ; ARE DETECTED.
10685                                       ;
10686 040330  005037  002264          CLR  PASSNO                         ;CLEAR OUTER LOOP COUNTER
10687 040334  104424                  CACHOFF                             ;TURN OFF CACHE
10688 040336                           REPEAT                              ;
10689 040336                           LET PASSNO := PASSNO + #1           ;
10690 040342  005000                   CLR  R0                             ;CLEAR DATA
10691 040344  105037  002262           CLRB PASFLG                         ;CLEAR PASFLG
10692 040350  104513                   CBREG                               ;ENABLE CHECK/SYNDROME BIT REGISTER
10693 040352                           REPEAT                              ;
10694 040352                                LET PASFLG :B= PASFLG + #1      ;INCREMENT LOOP COUNTER
10695 040356                                LET R4  := #-1                  ;INDEX TO SINGLE BIT ERROR TABLE
10696 040362                                LET NOPAR := #1                 ;ENABLE PARITY ACTION
10697 040370                                LET BITNO := #0                 ;CLEAR INNER LOOP COUNTER
10698 040374                                IFB PASFLG EQ #1                ;SELECT DATA TO BE CORRECTED BY PASSNO
10699 040404                                   LET R5 := #1                 ;DATA=0;BIT TO BE CORRECTED IS A ONE
10700 040410                                ELSE                            ;
10701 040412                                   LET R5 := #177776            ;DATA=177776;BIT TO BE CORRECTED IS A ZERO
10702 040416                                END                             ;
10703 040416                                REPEAT                          ;
10704 040416                                  LET PARCNT := #0              ;CLEAR PARITY COUNTER
10705 040422  005237  002320                  INC BITNO                     ;INCREMENT BIT POINTER
10706 040426                                  LET R4 := R4 + #1             ;POINT TO NEXT SET OF CHECK BITS
10707 040430                                  LET R2 :B= PTABLE(R4)         ;GET NEXT SET OF CHECK BITS
10708 040434  072227  000005                  ASH #5,R2                     ;SHIFT TO LINE UP IN CSR
10709 040440  052702  000006                  BIS #BIT2!BIT1,R2             ;ENABLE DIAG MODE
10710 040444                                  LET CSR := R2                 ;GET CHECK BITS TO BE WRITTEN
10711 040450  104425                          LOADCSR                       ;LOAD CSR WITH DATA
10712 040452                                  LET (R1) := R0                ;WRITE DATA TO TEST ADDRESS
10713 040454                                  IF PASSNO EQ #1               ;WRITE CSR
10714 040464  104471                             ECC1DIS                    ;FIRST PASS  ;ECC DISABLE,NO PBL
10715 040466                                  ELSE                          ;
10716 040470  104507                             ENA1SBE                    ;SECOND PASS ;ECC DISABLE,PBL ENABLED
10717 040472                                  END                           ;
10718 040472  005711                          TST (R1)                      ;CORRECT SBE
10719 040474  004737  040614                  CALL CHKTRP                   ;CHECK FOR CORRECT TRAP
10720 040500  104426                          READCSR                       ;READ THE CSR FOR UNCORRECTABLE ERROR
10721 040502                                  IF #BIT15 OFF.IN CSR          ;IS UNCORRECTABLE ERROR BIT SET????
10722 040512                                     LET BAD := CSR             ;
10723 040520                                     SET HEADER                 ;
10724 040526  104045                             ERROR +45                  ;
10725 040530                                  END                           ;
10726 040530  104503                          CLR1CSR                       ;
10727 040532  005011                          CLR (R1)                      ;CLEAR LUT
10728 040534                                  IFB PASFLG EQ #1              ;SHIFT NEW DATA DEPENDING ON PASFLG
10729 040544  006305                             ASL  R5                    ;SHIFT BITNO TO THE LEFT
10730 040546                                  ELSE                          ;
10731 040550  000261                             SEC                        ;SET CARRY BIT AND.....
10732 040552  006105                             ROL   R5                   ;ROTATE LEFT
10733 040554                                  END                           ;
```

```
10734 040554                                        UNTIL BITNO EQ #16.          ;UNTIL ALL BITS ARE DONE
10735 040564  005100                                COM R0                       ;COMPLEMENT DATA AND REPEAT
10736 040566                                      UNTILB PASFLG EQ #2            ;UNTIL 2 PASSES ARE COMPLETE!
10737 040576                                    UNTIL PASSNO EQ #2               ;
10738 040606  104503                            CLR1CSR                          ;
10739 040610  104423                            CACHON                           ;TURN CACHE
10740 040612  000207                            RETURN                           ;
10741
10742
10743 040614                                CHKTRP: IF PASSNO EQ #1              ;PASS 1 CHECK FOR NO TRAP
10744 040624                                          IF PARCNT EQ #1            ;
10745 040634                                             SET HEADER              ;
10746 040642  104057                                     ERROR +57               ;
10747 040644                                          END                        ;
10748 040644                                        ELSE                         ;
10749 040646                                          IF PARCNT NE #1            ;
10750 040656                                             SET HEADER              ;
10751 040664  104062                                     ERROR +62               ;
10752 040666                                          END                        ;
10753 040666                                        END                          ;
10754 040666  000207                                RETURN                       ;
```

```
10756 040670                          MTP047: SUBTST  <<MTP047        NO CSR UPDATE ON SBE WITH EXSISTING DBE>>
                                      ;*****************************************************************************
                                      ;*SUBTEST       MTP047  NO CSR UPDATE ON SBE WITH EXSISTING DBE
                                      ;*****************************************************************************
10757                                         ;
10758                                         ;  THIS TEST CHECKS TO SEE THAT THE CSR CONTENTS WILL NOT CHANGE
10759                                         ;  WITH A SINGLE BIT ERROR WHEN A DOUBLE BIT ERROR ALREADY
10760                                         ;  EXISTS.
10761                                         ;
10762 040670  104424                        CACHOFF                         ;TURN OFF CACHE
10763 040672                                LET R4 := BANK                  ;GET BANK NUMBER
10764 040676  072427  000011                ASH #9.,R4                      ;SHIFT INTO PLACE
10765 040702  042704  170037                BIC #^C7740,R4                  ;MASK OUT UNWANTED BITS
10766 040706  052704  100000                BIS #BIT15,R4                   ;SET UP GOOD DATA
10767 040712  104513                        CBREG                           ;ENABLE CHECK/SYNDROME BIT REGISTER
10768 040714                                LET CSR := #3144                ;CHECK BITS FOR DOUBLE BIT ERROR
10769 040722  104425                        LOADCSR                         ;
10770 040724                                LET (R1) := #0                  ;WRITE DBE CHECK BITS
10771 040726                                LET CSR := #104                 ;WRITE SBE CHECK BITS
10772 040734  104425                        LOADCSR                         ;
10773 040736                                LET (R2) := #0                  ;WRITE SBE CHECK BITS AT ADDRESS + 4K
10774 040740  104503                        CLR1CSR                         ;CLEAR CSR
10775 040742  005711                        TST (R1)                        ;READ DBE LOCATION
10776 040744  104426                        READCSR                         ;READ FOR CSR DBE INDICATOR
10777 040746  042737  020000  002146        BIC #BIT13,CSR                  ;CLEAR INHIBIT MODE POINTER
10778 040754                                IF CSR NE R4                    ;
10779 040762                                   LET BAD := CSR               ;
10780 040770                                   LET GOOD := R4               ;
10781 040774                                   SET HEADER                   ;
10782 041002  104063                           ERROR +63                    ;
10783 041004                                END                             ;
10784 041004  052704  000020                BIS #20,R4                      ;SET BIT IN GOOD DATA
10785 041010  005712                        TST (R2)                        ;READ SBE
10786 041012  104426                        READCSR                         ;READ CSR FOR NO CHANGE
10787 041014  042737  020000  002146        BIC #BIT13,CSR                  ;CLEAR INHIBIT MODE POINTER
10788 041022                                IF CSR NE R4                    ;
10789 041030                                   LET BAD := CSR               ;
10790 041036                                   LET GOOD := R4               ;
10791 041042                                   SET HEADER                   ;
10792 041050  104051                           ERROR +51                    ;
10793 041052                                END                             ;
10794 041052  104503                        CLR1CSR                         ;CLEAR 1 CSR
10795 041054  005011                        CLR (R1)                        ;
10796 041056  005012                        CLR (R2)                        ;
10797 041060  104423                        CACHON                          ;TURN ON CACHE
10798 041062  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
```

```
10835 041224                                  BACKGND:SUBTST  ;<SUBR  WRITE BACKGROUND>>
                                              ;*****************************************************************************
                                              ;*SUBTEST       SUBR    WRITE BACKGROUND
                                              ;*****************************************************************************
10836                                                 ;WRITES DATA FROM R2
10837 041224  104415                                  SAVREG
10838 041226  012700  060000                          MOV     #FIRST,R0
10839 041232  012701  040000                          MOV     #SIZE,R1
10840 041236  022737  000001  003752                  CMP     #1,PROTYP
10841 041244  001415                                  BEQ     WARN6B
10842 041246  012737  000207  027404  WARN6A:         MOV     #207,MTP000+4           ;WARNING PUTTING 'RETURN'' AFTER WRITE
10843 041254  012737  027400  002260                  MOV     #MTP000,SUPD0ADD
10844 041262  004737  027206                          CALL    SUPD03
10845 041266  012737  000240  027404                  MOV     #240,MTP000+4           ;RESTORE 'NOP' AFTER WRITE
10846 041274  104416                                  RESREG
10847 041276  000207                                  RETURN
10848 041300                                  WARN6B: BMOV    MTP000
10849 041306  012737  000207  177644  WARN6:         MOV     #207,UIPAR2             ;WARNING PUTTING 'RETURN'' INSTRUCTION AFTER WRITE
10850 041314  004737  027030                          CALL    SUPD01
10851 041320  104416                                  RESREG
10852 041322  000207                                  RETURN
```

```
 10854 041324                           GETCSR: SUBTST  <<SUBR  GET CSR INFORMATION FROM CONFIGURATION TABLE>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       SUBR    GET CSR INFORMATION FROM CONFIGURATION TABLE
                                        ;*****************************************************************************
 10855                                          ;INPUTS : NONE
 10856                                          ;
 10857                                          ;OUTPUT : CSRNO = CSR NUMBER
 10858                                          ;
 10859 041324  013702  002102                   MOV     BANKINDEX,R2            ;GET INDEX INTO CONFIG TABLE
 10860 041330  016203  002650                   MOV     CONFIG(R2),R3           ;MOV IT INTO R3
 10861 041334  000303                           SWAB    R3                      ;
 10862 041336  006303                           ASL     R3                      ;
 10863 041340  042703  177741                   BIC     #^C36,R3                ;CLEAR OFF SOME BITS
 10864 041344  010337  002150                   MOV     R3,CSRNO                ;SAVE CSR NUMBER
 10865 041350  000207                           RETURN
```

```
10868 041352                                PCONFIG:SUBTST  <<SUBR  PRINT CONFIGURATION MAP>>
                                            ;*****************************************************************************
                                            ;*SUBTEST       SUBR    PRINT CONFIGURATION MAP
                                            ;*****************************************************************************
10869 041352                                        PUSH    TKVEC,TKVEC+2,R0
10870 041364  010637  041652                        MOV     SP,PCONFS               ;SAVE LAST GOOD SP
10871 041370  012737  041620  000060                MOV     #PCONF2,TKVEC
10872 041376  012737  000340  000062                MOV     #340,TKVEC+2
10873 041404  017700  141220                        MOV     @$TKB,R0                ;KILL ANY OLD INTERRUPT
10874 041410  042737  000200  177776                BIC     #BIT7,PSW               ;LOWER CPU PRIORITY TO 140
10875 041416  052777  000100  141202                BIS     #BIT6,@$TKS             ;ENABLE KEYBOARD INTERRUPTS
10876
10877 041424                                        TYPE    MSG001
10878 041430                                        TYPE    MSG002
10879 041434                                        TYPE    MSG003
10880 041440  022737  000060  002552                CMP     #60,LASTBANK
10881 041446  002006                                BGE     NOOJ
10882                                               ;IF FAT PAPER ON TERMINAL GOTO 1$
10883 041450                                        IF #SW4 SET.IN @SWR THEN JUMPTO PCONF1
10884 041464  012700  000074          NOOJ:         MOV     #60.,R0
10885 041470  010004                                MOV     R0,R4
10886 041472                                        CLEAR   R1,R3
10887 041476                                        TYPE    MSG004
10888 041502  004737  041654                        CALL    TCONFIG                 ;GO TYPE CONFIGURATION (1ST HALF)
10889 041506  022737  000060  002552                CMP     #60,LASTBANK
10890 041514  002041                                BGE     PCONF2
10891 041516                                        TYPE    $CRLF
10892 041522                                        TYPE    MSG017                  ;PRINT SPACE(S)
10893 041526                                        TYPE    MSG011
10894 041532                                        TYPE    $CRLF
10895 041536                                        TYPE    MSG017                  ;PRINT SPACE(S)
10896 041542                                        TYPE    MSG012
10897 041546  012701  000360                        MOV     #60.*2*2,R1
10898 041552  010103                                MOV     R1,R3
10899 041554  004737  041654                        CALL    TCONFIG
10900 041560  000417                                BR      PCONF2
10901
10902 041562  012700  000170          PCONF1:       MOV     #120.,R0
10903 041566  010004                                MOV     R0,R4
10904 041570                                        CLEAR   R1,R3
10905 041574                                        TYPE    MSG014                  ;SPACE
10906 041600                                        TYPE    MSG011
10907 041604                                        TYPE    MSG004
10908 041610                                        TYPE    MSG012
10909 041614  004737  041654                        CALL    TCONFIG
10910
10911 041620  013706  041652          PCONF2:       MOV     PCONFS,SP               ;RESTORE STACK
10912 041624  042777  000100  140774                BIC     #BIT6,@$TKS
10913 041632  117700  140772                        MOVB    @$TKB,R0                ;READ CHAR TO KILL FLAG
10914 041636                                        POP     R0,TKVEC+2,TKVEC
10915 041650  000207                                RETURN
10916
10917 041652  000000          PCONFS:       0                                       ;STACK SAVED HERE!
```

```
                                        SUBTST  <<SUBR  TYPE CONFIGURATION>>
                                ;*****************************************************************************
                                ;*SUBTEST        SUBR    TYPE CONFIGURATION
                                ;*****************************************************************************
                                ;*****************************************************************************
10921                           ;CALL:  MOV     #N,R0           ;N=NUMBER OF CHARACTERS
10922                           ;       MOV     R0,R4           ;BACKUP
10923                           ;       MOV     #K.R1           ;INDEX CONSTANT
10924                           ;       MOV     R1,R3           ;BACKUP
10925                           ;       CALL    TCONFIG         ;ACTUAL CALL
10926                           ;       RETURN                  ;ONLY RETURN
10927                           ;*****************************************************************************
10928
10929                                   ;************
10930                                   ;** ERROR **
10931                                   ;************
10932   041654  012737  000340  177776  TCONFIG:MOV     #340,PSW        ;DISABLE INTERUPTS
10933   041662                          TYPE    MSG005
10934   041666  032761  000001  002650  1$:     BIT     #BIT0,CONFIG(R1)        ;ERROR ON THIS BANK?
10935   041674  001403                  BEQ     2$                      ;NO - SKIP
10936   041676                          TYPE    MSG013                  ;PRINT 'X'
10937   041702  000402                  BR      3$
10938   041704                  2$:     TYPE    MSG014                  ;PRINT SPACE
10939   041710  062701  000004          3$:     ADD     #4,R1                   ;BUMP POINTER
10940   041714  077014                  SOB     R0,1$                   ;LOOP UNTIL DONE
10941   041716  010400                  MOV     R4,R0
10942   041720  010301                  MOV     R3,R1
```

```
10946                                           ;*****************
10947                                           ;** INTERLEAVE **
10948                                           ;*****************
10949 041722                                    TYPE    MSG007
10950                                           ;THIS IS AN ENTRY POINT FROM ERROR REPORTS
10951 041726  012737  000340  177776  TCFIG1:   MOV     #340,PSW        ;DISABLE INTERUPTS
10952 041734  032761  010000  002652            BIT     #BIT12,CONFIG+2(R1)
10953 041742  001014                            BNE     1$
10954 041744  032761  000002  002650            BIT     #BIT1,CONFIG(R1)        ;IS THERE ANY MEMORY HERE?
10955 041752  001004                            BNE     18$                     ;BRANCH IF MEMORY PRESENT.
10956 041754  112737  000040  074704            MOVB    #' ,MSG015              ;MOVE A BLANK IN TO BE PRINTED
10957 041762  000424                            BR      16$                     ;BRANCH TO TYPE ROUTINE
10958 041764  112737  000055  074704  18$:      MOVB    #'-,MSG015
10959 041772  000420                            BR      16$
10960 041774  016105  002650          1$:       MOV     CONFIG(R1),R5
10961 042000  042705  007777                    BIC     #^C170000,R5            ;GET CSR INTERLEAVE
10962 042004  000305                            SWAB    R5
10963 042006  072527  177774                    ASH     #-4,R5
10964 042012  022705  000011                    CMP     #9.,R5
10965 042016  100002                            BPL     2$
10966 042020  062705  000007                    ADD     #7,R5
10967 042024  062705  000060          2$:       ADD     #60,R5                  ;MAKE ASCII
10968 042030  110537  074704                    MOVB    R5,MSG015               ;PLUG INTO MEMORY
10969 042034                          16$:      TYPE    MSG015
10970 042040                                    IF NOTAB NE #0 THEN $RETURN
10971 042050  062701  000004                    ADD     #4,R1                   ;BUMP POINTER
10972 042054  077054                            SOB     R0,TCFIG1               ;LOOP UNTIL DONE
10973 042056  010400                            MOV     R4,R0
10974 042060  010301                            MOV     R3,R1
10975
10976                                           ;******************
10977                                           ;** MEMORY TYPE **
10978                                           ;******************
10979                                           .ENABL  LSB
10980 042062                                    TYPE    MSG009
10981 042066  033761  002104  002650  TCFIG2:   BIT     CPUBIT,CONFIG(R1)
10982 042074  001432                            BEQ     17$
10983 042076  016105  002652                    MOV     CONFIG+2(R1),R5
10984 042102  000305                            SWAB    R5                      ;GET MEMORY TYPE
10985 042104  042705  177770                    BIC     #^C7,R5                 ;CLEAR NON INTERESTING BITS
10986 042110  020527  000003                    CMP     R5,#3                   ;IS IT A LEGAL MEMORY TYPE
10987 042114  003022                            BGT     17$                     ;IF IF SO BRANCH!!!!!!!
10988 042116                                    IF #BIT0 SET.IN R5              ;IS IT AN ECC MEMORY????
10989 042124                                      IF #BIT1 SET.IN R5            ;IS IT A MS11-P OR A MS11-M???
10990 042132  112737  000120  074704                MOVB        #'P,MSG015    ;IT IS A MS11-P
10991 042140                                      ELSE                          ;
10992 042142  112737  000115  074704                MOVB        #'M,MSG015    ;IT IS A MS11-M
10993 042150                                      END                           ;
10994 042150                                    ELSE                            ;
10995 042152  112737  000114  074704                MOVB        #'L,MSG015    ;IT IS A MS11-L
10996 042160                                    END                             ;
10997 042160  000403                            BR      8$
10998 042162  112737  000040  074704  17$:      MOVB    #' ,MSG015
10999 042170                          8$:       TYPE    MSG015
11000 042174                                    IF NOTAB NE #0 THEN $RETURN
11001 042204  062701  000004                    ADD     #4,R1                   ;BUMP POINTER
11002 042210  077052                            SOB     R0,TCFIG2               ;LOOP UNTIL DONE
```

```
                                                MOV     R4,R0
                                                MOV     R3,R1
                                                .DSABL  LSB
                                                ;*********
                                                ;** CSR **
                                                ;*********
                                                TYPE    MSG016
042222  112737  000040  074704  TCFIG3: MOVB    #' ,MSG015
042230  016105  002650                  MOV     CONFIG(R1),R5
042234  032705  000002                  BIT     #BIT1,R5
042240  001414                          BEQ     16$
042242  042705  170377                  BIC     #^C7400,R5
042246  000305                          SWAB    R5
042250  022705  000011                  CMP     #9.,R5
042254  100002                          BPL     10$
042256  062705  000007                  ADD     #7,R5
042262  062705  000060          10$:    ADD     #60,R5                  ;MAKE ASCII
042266  110537  074704                  MOVB    R5,MSG015               ;PLUG INTO MEMORY
042272                          16$:    TYPE    MSG015
042276                                  IF NOTAB NE #0 THEN $RETURN
042306  062701  000004                  ADD     #4,R1                   ;BUMP POINTER
042312  077035                          SOB     R0,TCFIG3
042314  010400                          MOV     R4,R0
042316  010301                          MOV     R3,R1

                                                ;****************
                                                ;** PROTECTED **
                                                ;****************
042320                                  TYPE    MSG010
042324  105761  002650          11$:    TSTB    CONFIG(R1)              ;BANK PROTECTED?
042330  100004                          BPL     12$                     ;NO - SKIP
042332  112737  000120  074704          MOVB    #'P,MSG015
042340  000407                          BR      13$
042342  032761  000100  002650  12$:    BIT     #BIT6,CONFIG(R1)        ;PROTECTED REGION OF ECC?
042350  001406                          BEQ     14$                     ;NO - SKIP
042352  112737  000111  074704          MOVB    #'I,MSG015
042360                          13$:    TYPE    MSG015
042364  000402                          BR      15$
042366                          14$:    TYPE    MSG014                  ;PRINT SPACE
042372  062701  000004          15$:    ADD     #4,R1                   ;BUMP POINTER
042376  077026                          SOB     R0,11$                  ;LOOP UNTIL DONE
042400  010400                          MOV     R4,R0
042402  010301                          MOV     R3,R1
042404  000207                          RETURN
```

```
                                        .SBTTL  TRAP    PARITY ERROR HANDLER
                                ;*******************************************************************************
                                ;VECTOR TO HERE FROM TRAPS TO 114
                                ;IGNORE ERRORS BUT COUNT IF NOPAR FLAG = 1.
                                ;*******************************************************************************
                                ;
                                ;       CODE    ACTION
                                ;
                                ;       --0-    PRINT UNEXPECTED PARITY TRAP
                                ;         1     COUNT ERROR
                                ;         2     SET "ABORT" / SETUP "BADPC" / RETURN VIA PCBUMP
                                ;         3     RETURN VIA "PARTHERE"
                                ;
042406  022737  000001  002074  PARITY: CMP     #1,NOPAR                ;COUNTING PARITY ERRORS?
042414  001003                          BNE     1$                      ;NO - SKIP
042416  005237  002070                  INC     PARCNT                  ;PARITY ERROR COUNTER + 1
042422  000002                          RTI
042424  022737  000002  002074  1$:     CMP     #2,NOPAR                ;ACTION CODE = 2 ?
042432  001013                          BNE     2$                      ;NO - SKIP
042434                                  SET     ABORTFLAG               ;YES
042442  004737  042614                  CALL    BADSTACK                ;FIND BAD SP,PC,PSW OFF STACK
042446  063716  002322                  ADD     PCBUMP,(SP)             ;UPDATE RETURN PC
042452  042766  000004  000002          BIC     #BIT2,2(SP)             ;SHOW FAILURE BY .NE.
042460  000002                          RTI
042462  022737  000003  002074  2$:     CMP     #3,NOPAR                ;ACTION CODE = 3 ?
042470  001003                          BNE     3$                      ;NO - SKIP
042472  013716  002300                  MOV     PARTHERE,(SP)
042476  000002                          RTI
042500  004737  042614          3$:     CALL    BADSTACK                ;FIND BAD SP,PC,PSW OFF STACK
042504                                  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
                                ;*******************************************************************************
                                ;
042512  022737  000001  002076  NONEXIST:CMP    #1,NONEM                ;COUNTING NON-EXISTANT MEMORY ERRORS?
042520  001011                          BNE     2$                      ;NO - SKIP
042522  005237  002066                  INC     NEMCNT                  ;BUMP NON-EXISTANT MEMORY COUNTER
042526  022737  000001  002066          CMP     #1,NEMCNT               ;FIRST ERROR?
042534  001002                          BNE     1$                      ;NO - SKIP
042536  010037  002032                  MOV     R0,ADDRESS              ;ASSUME R0 CONTAINS THE ADDRESS ACCESSED
042542  000002                  1$:     RTI
042544  005237  002066          2$:     INC     NEMCNT                  ;BUMP NON-EXISTANT MEMORY COUNTER
042550  012701  000001                  MOV     #1,R1                   ;DUMMY UP R1 FOR A FORCED SOB EXIT
042554  000002                          RTI
                                ;*******************************************************************************
                                        .SBTTL  TRAP    TIMEOUT (TRAP TO 4) HANDLER
042556  004737  042614          TIMEOUT:CALL    BADSTACK                ;FIND BAD SP,PC,PSW OFF STACK
042562                                  FATAL   6
                                ;*******************************************************************************
                                        .SBTTL  TRAP    MEMORY MANAGEMENT (TRAP TO 250) HANDLER
042570  004737  042614          MMTRAP: CALL    BADSTACK                ;FIND BAD SP,PC,PSW OFF STACK
042574                                  FATAL   7
                                        .SBTTL  TRAP    RESERVED INSTRUCTION HANDLER
042602  004737  042614          PDP1105:CALL    BADSTACK                ;FIND BAD SP,PC,PSW OFF STACK
042606                                  FATAL   5

042614                          BADSTACK:SUBTST <<FIND BAD SP, PC, & PSW FROM STACK>>
                                ;*******************************************************************************
                                ;*SUBTEST       FIND BAD SP, PC, & PSW FROM STACK
                                ;*******************************************************************************
                                ;
042614  010637  002024                  MOV     SP,BADSP
042620  062737  000002  002024          ADD     #2,BADSP
042626  016637  000002  002020          MOV     2(SP),BADPC
042634  016637  000004  002030          MOV     4(SP),BADPSW
042642  000207                          RETURN
```

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

042644  042766  140000  000002  $KERNEL:        BIC     #140000,2(SP)
042652  000002                  RTI
                                        ;*******************************************************************************
                                                .SBTTL  TRAP    ENERGIZE TRAP HANDLER
042654  052737  000001  177572  $ENERGIZE:BIS   #BIT0,MMR0
042662  000002                  RTI
                                        ;*******************************************************************************
                                                .SBTTL  TRAP    DEENERGIZE TRAP HANDLER
042664  042737  000001  177572  $DEENERGIZE:BIC #BIT0,MMR0
042672  000002                  RTI
                                        ;*******************************************************************************
                                                .SBTTL  TRAP    CACHON TRAP HANDLER
042674  005737  002540          $CACHN: TST     CACHKN          ;IS THERE A CACHE
042700  001406                  BEQ     1$              ;NO - RETURN
042702  013737  002540  177746        MOV     CACHKN,CONTRL   ;SETUP CACHE AS PER CONSTANT (USUALLY 1 = FULLY ON)
042710  052737  000001  177746        BIS     #BIT0,CONTRL    ;DISABLE TRAPS (BUT NOT ABORTS)
042716  000002                  1$:     RTI
                                        ;*******************************************************************************
                                                .SBTTL  TRAP    CACHOFF TRAP HANDLER
042720  005737  002540          $CACHF: TST     CACHKN          ;IS THERE A CACHE?
042724  001403                  BEQ     1$              ;NO - RETURN
                                        ;DISABLE TRAPS (NOT ABORTS), FORCE MISSES, FLUSH, BYPASS
042726  053737  002544  177746        BIS     CACHKF,CONTRL
042734  000002                  1$:     RTI
```

```
                                                        .SBTTL  TRAP    LOAD CSR TRAP HANDLER
                                                        ;LOAD CORRECT CSR WITH DATA IN CSR
                                                        ;PROGRAM CSR'S ASSERT INHIBIT MODE POINTER WHEN LOADED
042736                                          $LOADC: PUSH    R0,R1                   ;SAVE REGISTERS
042742  013700  002150                                MOV     CSRNO,R0                ;CREATE CSR ADDRESS
042746                                                IF INHECC IS TRUE THEN GOTO 3$  ;DON'T WANT INH. MODE POINTER ON
042754  005737  002526                                TST     PGMCSR                  ;PROGRAM IN INTERLEAVED SPACE?
042760  100007                                        BPL     1$                      ;BRANCH IF NOT
042762  113701  002527                                MOVB    PGMCSR+1,R1             ;CHECK SECOND CSR
042766  042701  177740                                BIC     #^C37,R1                ;CLEAR UNNECESSARY BITS
042772  020137  002150                                CMP     R1,CSRNO                ;IS THIS THE CURRENT CSR?
042776  001404                                        BEQ     2$                      ;BRANCH IF IT IS
043000  123737  002526  002150  1$:                   CMPB    PGMCSR,CSRNO            ;IS THIS THE CURRENT CSR?
043006  001003                                        BNE     3$                      ;BRANCH IF NOT
043010  052737  020000  002146  2$:                   BIS     #BIT13,CSR              ;SET THE INHIBIT MODE POINTER TO  1ST 16K
043016  013760  002146  172100  3$:                   MOV     CSR,CSRADD(R0)          ;LOAD THE CSR
043024                                                POP     R1,R0                   ;RESTORE REGISTERS
043030  000002                                        RTI

                                                        .SBTTL  TRAP    READ CSR TRAP HANDLER
                                                        ;READ THE CORRECT CSR INTO LOCATIONS CSR
043032                                          $READC: PUSH    R0
043034  013700  002150                                MOV     CSRNO,R0
043040  016037  172100  002146                        MOV     CSRADD(R0),CSR          ;READ IT
043046                                                POP     R0
043050  000002                                        RTI
```

```
                                                .SBTTL  TRAP    TEST (R1) & READ CSR CAREFULLY
043052                                  $TSTRD: PUSH    R0,R2,R3
043060  012700  172100                          MOV     #CSRADD,R0              ;CREATE CSR ADDRESS
043064  063700  002150                          ADD     CSRNO,R0
043070  005002                                  CLR     R2
043072  005737  002526                          TST     PGMCSR
043076  100007                                  BPL     1$
043100  113703  002527                          MOVB    PGMCSR+1,R3
043104  042703  000200                          BIC     #BIT7,R3
043110  020337  002150                          CMP     R3,CSRNO
043114  001404                                  BEQ     2$
043116  123737  002526  002150          1$:     CMPB    PGMCSR,CSRNO
043124  001002                                  BNE     3$
043126  012702  020000                  2$:     MOV     #BIT13,R2
043132  022737  000001  003752          3$:     CMP     #1,PROTYP       ;IS THIS AN 11/44?
043140  001403                                  BEQ     4$              ;BRANCH IF IT IS
043142  004737  043230                          CALL    TSTRD1
043146  000405                                  BR      5$
043150                                  4$:     BMOV    TSTRD1
043156  004737  177640                          CALL    FASTCITY                ;CALL TO THE USER INSTRUCTION PAR'S
                                                ;IF SINGLE BIT ERROR ONLY - SET CARRY BIT
043162                                  5$:     POP     R3,R2,R0
043170                                          IF #BIT4 SET.IN CSR AND #BIT15 OFF.IN CSR
043210  052766  000001  000002                    BIS   #BIT0,2(SP)
043216                                          ELSE
043220  042766  000001  000002                    BIC   #BIT0,2(SP)
043226                                          END ;OF IF #BIT4
043226  000002                                  RTI

043230  010210                          TSTRD1: MOV     R2,(R0)                 ;V177640
043232                                          TESTAREA                        ;V177642 ;ENTER SUPERVISOR MODE
043240  105711                                  TSTB    (R1)                    ;V177646
043242  042737  140000  177776                  BIC     #BIT15!BIT14,PSW        ;V177650
043250  011037  002146                          MOV     (R0),CSR                ;V177656
043254  000207                                  RETURN                          ;V177662
```

```
11222                                                   .SBTTL  TRAP    ECC DISABLE ALL CSR'S TRAP HANDLER
11223 043256  012737  000002  002146  $ECCDIS:MOV       #BIT1,CSR
11224 043264  004737  044002                           CALL    CSROUT
11225 043270  000002                                   RTI
11226                                                   .SBTTL  TRAP    ECC DISABLE OF 1 SELECTED CSR TRAP HANDLER
11227 043272  012737  000002  002146  $ECC1DIS:MOV      #BIT1,CSR
11228 043300  104425                                   LOADCSR
11229 043302  000002                                   RTI
11230                                                   .SBTTL  TRAP    INITIALIZE ALL CSR'S TRAP HANDLER
11231 043304  012737  000001  002146  $ECCINIT:MOV      #BIT0,CSR
11232 043312  004737  044002                           CALL    CSROUT
11233 043316  000002                                   RTI
11234                                                   .SBTTL  TRAP    INITIALIZE 1 SELECTED CSR TRAP HANDLER
11235 043320  012737  000001  002146  $ECC1INIT:MOV     #BIT0,CSR
11236 043326  104425                                   LOADCSR
11237 043330  000002                                   RTI
11238                                                   .SBTTL  TRAP    ENABLE SBE PARITY TRAPS ON ALL CSR'S
11239 043332  012737  000003  002146  $ENASBE:MOV      #BIT0!BIT1,CSR
11240 043340  004737  044002                           CALL    CSROUT
11241 043344  000002                                   RTI
11242                                                   .SBTTL  TRAP    ENABLE SBE PARITY TRAPS ON 1 SELECTED CSR
11243 043346  012737  000003  002146  $ENA1SBE:MOV     #BIT0!BIT1,CSR
11244 043354  104425                                   LOADCSR
11245 043356  000002                                   RTI
11246                                                   .SBTTL  TRAP    WRITE CHECKBITS THRU ALL CSR'S TRAP HANDLER
11247 043360  013737  002310  002146  $CBCSR: MOV      CHECK,CSR               ;BITS 11-5
11248 043366  052737  000006  002146          BIS      #BIT1!BIT2,CSR          ;CHECK MODE
11249 043374  004737  044002                   CALL CSROUT
11250 043400  000002                           RTI
11251                                           .SBTTL  TRAP    WRITE CHECKBITS THRU 1 SELECTED CSR TRAP HANDLER
11252 043402  013737  002310  002146  $CB1CSR:MOV      CHECK,CSR               ;BITS 11-5
11253 043410  052737  000006  002146          BIS      #BIT1!BIT2,CSR          ;CHECK MODE
11254 043416  104425                           LOADCSR
11255 043420  000002                           RTI
```

```
                                                .SBTTL  TRAP    WAS THERE A SBE ON ANY CSR TRAP HANDLER
                                        $WASSBE:PUSH    R1,R4
                                                MOV     TOTCSRS,R1      ;GET CSR'S BYTE
                                                CLR     R4
                                                BEGIN LWSBE
                                                  FOR CSRNO := #0 TO #36 BY #2
                                                    ASL         R1
                                                    ON.ERROR
                                                      READCSR
                                                      IF #BIT4 SET.IN CSR
                                                        SET     R4
                                                        LEAVE LWSBE
                                                      END ;OF IF #BIT4
                                                    END ;OF ON.ERROR
                                                    IF R1 EQ #0 THEN LEAVE LWSBE
                                                  END ;OF FOR CSRNO
                                                END LWSBE
                                                ROR     R4              ;SET C BIT FOR ERROR
                                                POP     R4,R1
                                                ON.ERROR
                                                  BIS   #BIT0,2(SP)
                                                ELSE
                                                  BIC   #BIT0,2(SP)
                                                END ;OF ON.ERROR
                                                RTI
                                                .SBTTL  TRAP    WAS THERE A SBE IN 1 SELECTED CSR TRAP HANDLER
                                                ;ON RETURN IF CARRY IS SET THERE WAS A SBE
                                        $WAS1SBE:READCSR
                                                BIC     #BIT0,2(SP)     ;CLR C BIT ON STACK
                                                BIT     #BIT4,CSR
                                                BEQ     1$
                                                BIS     #BIT0,2(SP)     ;SET C BIT ON STACK
                                        1$:     RTI
```

```
11293                                                   .SBTTL  TRAP    WAS THERE A DBE ON ANY CSR TRAP HANDLER
11294 043566                                    $WASDBE:PUSH    R1,R4
11295 043572  013701  002222                            MOV     TOTCSRS,R1      ;GET CSR'S BYTE
11296 043576  005004                                    CLR     R4
11297 043600                                            BEGIN LWDBE
11298 043600                                              FOR CSRNO := #0 TO #36 BY #2
11299 043604  006301                                        ASL         R1
11300 043606                                              ON.ERROR
11301 043610  104426                                        READCSR
11302 043612                                                IF #BIT15 SET.IN CSR
11303 043622                                                  SET     R4
11304 043626                                                  LEAVE LWDBE
11305 043630                                                END ;OF IF #BIT4
11306 043630                                              END ;OF ON.ERROR
11307 043630                                            IF R1 EQ #0 THEN LEAVE LWDBE
11308 043634                                            END ;OF FOR CSRNO
11309 043652                                            END LWDBE
11310 043652  006004                                    ROR     R4              ;SET C BIT FOR ERROR
11311 043654                                            POP     R4,R1
11312 043660                                            ON.ERROR
11313 043662  052766  000001  000002                      BIS   #BIT0,2(SP)
11314 043670                                            ELSE
11315 043672  042766  000001  000002                      BIC   #BIT0,2(SP)
11316 043700                                            END ;OF ON.ERROR
11317 043700  000002                                    RTI
11318                                                   .SBTTL  TRAP    WAS THERE A DBE ON 1 SELECTED CSR TRAP HANDLER
11319                                                   ;ON RETURN IF CARRY IS SET THERE WAS A DBE
11320 043702  104426                            $WAS1DBE:READCSR
11321 043704  005737  002146                            TST     CSR             ;DBE?
11322 043710  100004                                    BPL     3$              ;NO - SKIP
11323 043712  052766  000001  000002                    BIS     #BIT0,2(SP)     ;SET C BIT ON STACK
11324 043720  000002                                    RTI
11325 043722  042766  000001  000002  3$:               BIC     #BIT0,2(SP)     ;CLR C BIT ON STACK
11326 043730  000002                                    RTI
```

```
                                                        .SBTTL  TRAP    CLEAR ALL ECC CSR'S TRAP HANDLER
043732                                          $CLRCSR:CLEAR   CSR
043736  004737  044002                                CALL    CSROUT
043742  000002                                        RTI
                                                        .SBTTL  TRAP    CLEAR 1 SELECTED CSR TRAP HANDLER
043744                                          $CLR1CSR:CLEAR  CSR
043750  104425                                        LOADCSR
043752  000002                                        RTI
                                                        .SBTTL  TRAP    ECC DISABLE, CHECK MODE, & WRITE CHECKBITS IN ALL CSR'S TRAP HANDLER
                                                        ;CHECKBITS ALREADY IN LOC ''CSR''
043754  052737  000006  002146                $CHKDIS:BIS     #BIT1!BIT2,CSR          ;ECC DISABLE & DIAG CHECK MODE
043762  004737  044002                                CALL    CSROUT
043766  000002                                        RTI
                                                        .SBTTL  TRAP    ECC DISABLE, CHECK MODE, & WRITE CHECKBITS IN 1 SELECTED CSR
                                                        ;CHECKBITS ALREADY IN LOC ''CSR''
043770  052737  000006  002146                $CHK1DIS:BIS    #BIT1!BIT2,CSR          ;ECC DISABLE & DIAG CHECK MODE
044000  104425                                        LOADCSR
044002  000002                                        RTI
```

```
 11349 044002                                CSROUT: SUBTST  <<SUBR  WRITE IN ALL CSR'S>>
                                             ;*****************************************************************************
                                             ;*SUBTEST        SUBR    WRITE IN ALL CSR'S
                                             ;*****************************************************************************
 11350 044002                                        PUSH    R1
 11351 044004  013701  002222                        MOV     TOTCSRS,R1      ;GET CSR'S BYTE
 11352 044010                                        BEGIN LCSROUT
 11353 044010                                          FOR CSRNO := #0 TO #36 BY #2
 11354 044014  006301                                   ASL         R1
 11355 044016                                           ON.ERROR
 11356 044020  104425                                      LOADCSR
 11357 044022                                           END ;OF ON.ERROR
 11358 044022                                         IF R1 EQ #0 THEN LEAVE LCSROUT
 11359 044026                                         END ;OF FOR CSRNO
 11360 044044                                        END LCSROUT
 11361 044044                                        POP     R1
 11362 044046  000207                                RETURN
 11363
 11364 044050                                $INVALID:       SUBTST  <<TRAP  INVALIDATE BACKGROUND PATTERN>>
                                             ;*****************************************************************************
                                             ;*SUBTEST        TRAP    INVALIDATE BACKGROUND PATTERN
                                             ;*****************************************************************************
 11365 044050                                        PUSH    R0,R1
 11366 044054  013701  002100                        MOV     BANK,R1
 11367 044060  006301                                ASL     R1
 11368 044062  006301                                ASL     R1
 11369 044064  042761  020000  002652                BIC     #BIT13,CONFIG+2(R1)
 11370 044072                                        POP     R1,R0
 11371 044076  000002                                RTI
```

```
 11373 044100                           $ERRGEN:        SUBTST<<TRAP    GENERATE AND TEST ERROR ADDRESS>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       TRAP    GENERATE AND TEST ERROR ADDRESS
                                        ;*****************************************************************************
                                        ;
 11374 044100                                   PUSH    R0,R1,R2,R3
 11375 044110  013703  002102                   MOV     BANKINDEX,R3
 11376 044114  005737  002452                   TST     NOSJPER
 11377 044120  001003                           BNE     6$
 11378 044122  013700  172246                   MOV     SIPAR3,R0               ;GENERATE WHAT ERROR ADDR SHOULD BE
 11379 044126  000402                           BR      7$
 11380 044130  013700  177646           6$:     MOV     UIPAR3,R0
 11381 044134  072027  177773           7$:     ASH     #-5,R0
 11382 044140  005737  002130                   TST     EUFLAG
 11383 044144  001002                           BNE     1$
 11384 044146  042700  177600                   BIC     #^C177,R0
 11385 044152  000301                   1$:     SWAB    R1                      ;GET CURRENT ADDRESS BITS 11 AND 12
 11386 044154  006201                           ASR     R1
 11387 044156  006201                           ASR     R1
 11388 044160  006201                           ASR     R1
 11389 044162  042701  177775                   BIC     #^C2,R1
 11390 044166  060100                           ADD     R1,R0                   ;ADD THEM TO THE ADJUSTED PAR VALUE
 11391                                          ;GET ERROR ADDRESS FROM CSR UNDER TEST
 11392 044170  013701  002146                   MOV     CSR,R1
 11393 044174  072127  177773                   ASH     #-5,R1
 11394 044200  042701  177600                   BIC     #^C177,R1
 11395 044204  005737  002450                   TST     N022BIT                 ;IS THIS AN 11/44 OR 11/24?
 11396 044210  001024                           BNE     2$                      ;BRANCH IF NOT NECESSARY
 11397 044212  005737  002130                   TST     EUFLAG                  ;IS IT EUB?
 11398 044216  001421                           BEQ     2$                      ;BRANCH IF NOT
 11399 044220                                   PUSH    R0                      ;SAVE GENERATED ERROR ADDRESS
 11400 044222  013702  002150                   MOV     CSRNO,R2                ;GET CSR NUMBER
 11401 044226  052762  040000  172100           BIS     #BIT14,CSRADD(R2)       ;TURN ON EUB BIT CAREFULLY
 11402 044234  016200  172100                   MOV     CSRADD(R2),R0           ;GET CSR CONTENTS
 11403 044240  042762  040000  172100           BIC     #BIT14,CSRADD(R2)       ;TURN OFF EUB BIT CAREFULLY
 11404 044246  042700  177037                   BIC     #^C740,R0               ;CLEAR EVERYTHING BUT ERROR ADDR
 11405 044252  006300                           ASL     R0
 11406 044254  006300                           ASL     R0                      ;SHIFT ADDR BITS 18-21 INTO POSITION
 11407 044256  060001                           ADD     R0,R1                   ;ADD TO CURRENT ERROR ADDRESS
 11408 044260                                   POP     R0
 11409 044262  020001                   2$:     CMP     R0,R1                   ;COMPARE REAL AND GENERATED ERR. ADDR.
 11410 044264  001420                           BEQ     5$                      ;BRANCH IF THEY ARE THE SAME
 11411 044266  005737  002134                   TST     INTFLAG                 ;INTERLEAVED?
 11412 044272  001411                           BEQ     3$                      ;NO - WE HAVE AN ERROR
 11413 044274  062700  000100                   ADD     #100,R0
 11414 044300  005737  002136                   TST     INT64K                  ;64K INTERLEAVED MEMORY?
 11415 044304  001002                           BNE     4$
 11416 044306  062700  000100                   ADD     #100,R0
 11417 044312  020001                   4$:     CMP     R0,R1
 11418 044314  001404                           BEQ     5$
 11419 044316  005737  002064           3$:     TST     SKPERR                  ;ARE WE SUPPOSED TO SKIP ERROR P.O.?
 11420 044322  001001                           BNE     5$                      ;YES - SKIP ERROR PRINTOUT
 11421 044324  104462                           PERR36                          ;ELSE PRINT ERROR ADDRESS ERROR
 11422 044326  010137  002454           5$:     MOV     R1,ERRADD               ;SAVE CSR'S ERROR ADDRESS
 11423 044332  005037  002064                   CLR     SKPERR                  ;ENABLE THE ERROR PRINTOUT AGAIN
 11424 044336                                   POP     R3,R2,R1,R0             ;RESTORE REGISTERS
 11425 044346  000002                           RTI
 11426
```

```
 11427 044350                           $CBREG: SUBTST  <<TRAP          ENABLE CHECK/SYNDROME BIT REGISTER>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       TRAP            ENABLE CHECK/SYNDROME BIT REGISTER
                                        ;*****************************************************************************
 11428 044350  005037  002146                      CLR     CSR                             ;
 11429 044354  052737  000004  002146              BIS     #BIT2,CSR                       ;ENABLE DIAGNOSTIC MODE
 11430 044362  104425                              LOADCSR                                 ;LOAD CSR REGISTER
 11431 044364  000002                              RTI                                     ;
```

```
 11434 044366                           CHKGEN: SUBTST<<SUBR    GENERATE CHECK BITS>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       SUBR    GENERATE CHECK BITS
                                        ;*****************************************************************************
 11435                                          ;CHECK BIT GENERATOR ROUTINE
 11436                                          ;CALLING SEQUENCE IS:
 11437                                          ;       MOV     #WORD1,SOURCE   ;SOURCE = ADDRESS OF DATA
 11438                                          ;       CALL    CHKGEN
 11439                                          ;
 11440                                          ;CHECK BITS RETURNED IN BITS 11-5 OF LOCATION CHECK
 11441                                          ;
 11442 044366                                   PUSH    R0,R1,R2,R3,R4,R5
 11443 044402  012702  000077                   MOV     #77,R2          ;DEFAULT CHECKBITS FOR DOUBLE WORD OF ZEROS
 11444 044406  012703  044474                   MOV     #CHKTAB,R3      ;ADDRESS OF CHECKBIT TABLE
 11445 044412  013705  002306                   MOV     SOURCE,R5       ;GET SOURCE ADDRESS
 11446 044416  012501                           MOV     (R5)+,R1        ;GET LSB'S
 11447 044420  011500                           MOV     (R5),R0         ;GET MSB'S
 11448
 11449 044422  006704                   1$:     SXT     R4              ;EXTEND SIGN OF DOUBLE WORD TO R4
 11450 044424  142304                           BICB    (R3)+,R4        ;ELIMINATE BITS THAT DON'T COUNT
 11451 044426  074402                           XOR     R4,R2           ;COMPLEMENT MASKED BITS IN CHECKBITS
 11452 044430  073027  000001                   ASHC    #1,R0           ;DOUBLE PRECISION LEFT SHIFT R0,,R1
 11453 044434  001372                           BNE     1$              ;LOOP TILL ALL BITS ARE CHECKED
 11454
 11455 044436  042702  177600                   BIC     #^C177,R2               ;KILL ALL JUNK BITS
 11456 044442  000302                           SWAB    R2              ;POSITION CHECKBITS IN BITS 11-5
 11457 044444  006202                           ASR     R2
 11458 044446  006202                           ASR     R2
 11459 044450  006202                           ASR     R2
 11460 044452  010237  002310                   MOV     R2,CHECK
 11461 044456                                   POP     R5,R4,R3,R2,R1,R0
 11462 044472  000207                           RETURN
```

```
11465 044474                    CHKTAB: ;BYTE #3
11466 044474  200                       .BYTE   ^C177           ;BIT 31
11467 044475  301                       .BYTE   ^C076           ;BIT 30
11468 044476  302                       .BYTE   ^C075           ;BIT 29
11469 044477  203                       .BYTE   ^C174           ;BIT 28
11470 044500  304                       .BYTE   ^C073           ;BIT 27
11471 044501  205                       .BYTE   ^C172           ;BIT 26
11472 044502  206                       .BYTE   ^C171           ;BIT 25
11473 044503  307                       .BYTE   ^C070           ;BIT 24
11474                                   ;BYTE #2
11475 044504  310                       .BYTE   ^C067           ;BIT 23
11476 044505  211                       .BYTE   ^C166           ;BIT 22
11477 044506  212                       .BYTE   ^C165           ;BIT 21
11478 044507  313                       .BYTE   ^C064           ;BIT 20
11479 044510  214                       .BYTE   ^C163           ;BIT 19
11480 044511  315                       .BYTE   ^C062           ;BIT 18
11481 044512  316                       .BYTE   ^C061           ;BIT 17
11482 044513  217                       .BYTE   ^C160           ;BIT 16
11483                                   ;BYTE #1
11484 044514  320                       .BYTE   ^C057           ;BIT 15
11485 044515  221                       .BYTE   ^C156           ;BIT 14
11486 044516  222                       .BYTE   ^C155           ;BIT 13
11487 044517  323                       .BYTE   ^C054           ;BIT 12
11488 044520  224                       .BYTE   ^C153           ;BIT 11
11489 044521  325                       .BYTE   ^C052           ;BIT 10
11490 044522    6                       .BYTE   ^C051           ;BIT  9
11491 044523  227                       .BYTE   ^C150           ;BIT  8
11492                                   ;BYTE #0
11493 044524  340                       .BYTE   ^C037           ;BIT  7
11494 044525  241                       .BYTE   ^C136           ;BIT  6
11495 044526  242                       .BYTE   ^C135           ;BIT  5
11496 044527  343                       .BYTE   ^C034           ;BIT  4
11497 044530  244                       .BYTE   ^C133           ;BIT  3
11498 044531  345                       .BYTE   ^C032           ;BIT  2
11499 044532  346                       .BYTE   ^C031           ;BIT  1
11500 044533  247                       .BYTE   ^C130           ;BIT  0
```

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

                                        ;SET SUPERVISOR/USER UP FOR 1 TO 1 MAP
044534                          MAPPER: PUSH    R0,R1,R2,R4,R5
044546  012700  172340                  MOV     #KIPAR0,R0              ;FIRST AREA TO MAP TO
044552  012701  172240                  MOV     #SIPAR0,R1              ;FIRST ADDRESS REGISTER
044556  012704  172200                  MOV     #SIPDR0,R4              ;FIRST DESCRIPTOR REGISTER
044562  005737  002452                  TST     NOSUPER                 ;CAN WE USE SUPERVISOR MODE?
044566  001404                          BEQ     4$                      ;YES, BRANCH
044570  012701  177640                  MOV     #UIPAR0,R1              ;FIRST ADDRESS REGISTER
044574  012704  177600                  MOV     #UIPDR0,R4              ;FIRST DESCRIPTOR REGISTER
044600  012702  077406          4$:     MOV     #77406,R2               ;CONSTANT FOR 4K PAGE, UP, R/W
044604  012705  000010                  MOV     #8.,R5                  ;COUNTER
044610  012021                  1$:     MOV     (R0)+,(R1)+             ;PUT IN SUPERVISOR ADDRESS
044612  010224                          MOV     R2,(R4)+                ;PUT IN SUPERVISOR DESCRIPTOR
044614  077503                          SOB     R5,1$                   ;LOOP TILL DONE
044616  012741  177600                  MOV     #177600,-(R1)           ;CORRECT LAST FIELD FOR PERIPHERALS PAGE

                                        ;SET UP SUPERVISOR/USER FOR TEST AREA
044622  022703  000170                  CMP     #120.,R3                ;MAP NOTHING (1 TO 1)?
044626  001516                          BEQ     3$                      ;YES - SKIP
044630  072327  000011                  ASH     #9.,R3                  ;BANK 1 STARTS AT 100,000 LESS 6 LSB'S
                                                                        ;FOR MEMORY MANAGEMENT = 1000
044634  012701  172246                  MOV     #SIPAR3,R1              ;SETUP FOR AUTO INCREMENTING
044640  005737  002452                  TST     NOSUPER                 ;DO WE HAVE SUPERVISOR MODE?
044644  001402                          BEQ     5$                      ;YES - BRANCH
044646  012701  177646                  MOV     #UIPAR3,R1              ;SETUP FOR AUTO INCREMENTING
044652  012702  000004          5$:     MOV     #4,R2                   ;COUNTER
044656  010321                  2$:     MOV     R3,(R1)+                ;PLUG IN PAR INFO
044660  062703  000200                  ADD     #200,R3                 ;BUMP ADDRESS 4K
044664  077204                          SOB     R2,2$                   ;LOOP TILL DONE
044666  005737  002236                  TST     SPLTCSR
044672  001442                          BEQ     9$
044674  162701  000010                  SUB     #10,R1
044700  010102                          MOV     R1,R2
044702  062702  000004                  ADD     #4,R2
044706  022737  000001  002236          CMP     #1,SPLTCSR
044714  001403                          BEQ     10$
044716  010200                          MOV     R2,R0
044720  010102                          MOV     R1,R2
044722  010001                          MOV     R0,R1
044724  012122                  10$:    MOV     (R1)+,(R2)+
044726  011112                          MOV     (R1),(R2)
044730  013700  002102                  MOV     BANKINDEX,R0
044734  005737  002136                  TST     INT64K
044740  001403                          BEQ     11$
```

```
11557 044742  012700  004000                    MOV     #4000,R0
11558 044746  000402                            BR      12$
11559 044750  012700  010000            11$:    MOV     #10000,R0
11560 044754  005737  002452            12$:    TST     NOSUPER
11561 044760  001403                            BEQ     13$
11562 044762  012701  177652                    MOV     #UIPAR5,R1
11563 044766  000402                            BR      14$
11564 044770  012701  172252            13$:    MOV     #SIPAR5,R1
11565 044774  060021                    14$:    ADD     R0,(R1)+
11566 044776  060011                            ADD     R0,(R1)
11567                                           ;IF WE ONLY HAVE AN 124K SYSTEM, WE DON'T WANT TO TEST THE
11568                                           ;LAST 4K, WHERE THE UNIBUS DEVICE PAGE IS. INSTEAD, THE
11569                                           ;PROGRAM WILL REMAP THE LAST 4K TO 8-12K. ALSO, IF THERE
11570                                           ;IS A BANK 177 ON AN 11/44, THE PROGRAM WILL REMAP THE LAST
11571                                           ;4K TO 8-12K FOR THE SAME REASON.
11572 045000  022737  000007  002552    9$:     CMP     #7,LASTBANK
11573 045006  001010                            BNE     7$
11574 045010  005737  002450                    TST     NO22BIT                 ;11/44 OR 24?
11575 045014  001423                            BEQ     3$                      ;BRANCH IF SO
11576 045016  022737  000007  002100            CMP     #7,BANK                 ;BANK 7?
11577 045024  001017                            BNE     3$                      ;NO - BRANCH
11578 045026  000404                            BR      8$
11579 045030  022737  000177  002552    7$:     CMP     #177,LASTBANK
11580 045036  001012                            BNE     3$
11581 045040  005737  002452            8$:     TST     NOSUPER
11582 045044  001404                            BEQ     6$
11583 045046  013737  177652  177654            MOV     UIPAR5,UIPAR6
11584 045054  000403                            BR      3$
11585 045056  013737  172252  172254    6$:     MOV     SIPAR5,SIPAR6
11586 045064                            3$:     POP     R5,R4,R2,R1,R0
11587 045076  000207                            RETURN
11588                                           .SBTTL  TRAP    MAP KERNEL (ALMOST 1 TO 1) TRAP HANDLER
11589 045100                            $KMAP:  PUSH    R0,R1,R2,R3,R4
11590 045112  005000                            CLR     R0                      ;1ST AREA TO MAP TO
11591 045114  012701  172340                    MOV     #KIPAR0,R1              ;FIRST ADDRESS
11592 045120  012702  077406                    MOV     #77406,R2               ;CONSTANT FOR 4K PAGE,UP,R/W
11593 045124  012703  172300                    MOV     #KIPDR0,R3              ;1ST PAGE DESCRIPTOR REGISTER
11594 045130  012704  000010                    MOV     #8.,R4                  ;COUNTER
11595 045134  010021                    1$:     MOV     R0,(R1)+                ;PUT IN KERNEL ADDRESS
11596 045136  010223                            MOV     R2,(R3)+                ;PUT IN KERNEL DISCRIPTOR
11597 045140  062700  000200                    ADD     #200,R0                 ;ADD ADDRESS CONSTANT FOR 4K CHANGE
11598 045144  077405                            SOB     R4,1$                   ;LOOP TILL DONE
11599 045146  012741  177600                    MOV     #177600,-(R1)           ;THE PERIPHERALS PAGE TO KIPAR7
11600 045152  012741  177400                    MOV     #177400,-(R1)           ;AND NEXT LOWER PAGE TO KIPAR6
11607 045156                                    POP     R4,R3,R2,R1,R0
11608 045170  000002                            RTI
```

```
11611 045172                                RELOCATE:SUBTST <<RELOCATE PROGRAM>>
                                            ;*****************************************************************************
                                            ;*SUBTEST       RELOCATE PROGRAM
                                            ;*****************************************************************************
11612 045172                                        IF #SW12 SET.IN @SWR THEN $RETURN ERROR
11613 045206                                        IF APTFLAG IS TRUE OR ACTFLAG IS TRUE
11614 045222                                          IF $PASS NE #0 THEN $RETURN ERROR
11615 045234                                        END; OF IF APTFLAG
11616 045234                                        BEGIN LOADERBANK
11617 045234                                          FOR BANK := #1 TO LASTBANK
11618 045242  004737  047020                            CALL EXBANK
11619 045246                                            IF ACFLAG IS TRUE AND PFLAG IS FALSE AND BMFLAG IS FALSE
11620 045270  013700  002100                              MOV       BANK,R0
11621 045274  010037  002426                              MOV       R0,LOADBANK
11622 045300  013701  002562                              MOV       LOADHOME,R1
11623 045304  004737  046410                              CALL      BANKMOV
11624 045310  004737  046742                              CALL      NEWLOAD          ;MAP NEW LOADER BANK IN KERNEL
11625 045314  013701  002102                              MOV       BANKINDEX,R1
11626 045320  052761  100000  002652                      BIS       #BIT15,CONFIG+2(R1)      ;MARK LOADER
11627 045326  042761  020000  002652                      BIC       #BIT13,CONFIG+2(R1)      ;INVALIDATE BACKGROUND PATTERN
11628 045334                                              LEAVE LOADERBANK
11629 045336                                            END ;OF IF ACFLAG
11630 045336                                          END ;OF FOR BANK
11631 045352                                          IF #SW13 OFF.IN @SWR
11632 045362                                            TYPE        MSG075           ;RELOCATION NOT POSSIBLE
11633 045366                                          END ;OF IF #SW13
11634 045366                                          $RETURN ERROR
11635 045372                                        END LOADERBANK
11636 045372                                        BEGIN FINDBANK
11637 045372  013702  002552                        MOV     LASTBANK,R2
11638 045376  006302                                ASL     R2
11639 045400  006302                                ASL     R2                ;R2 <- R2 * 4
11640 045402                                          FOR R1 := #2*2 TO R2 BY #4
11641 045406                                            IF #BIT7!BIT0 OFF.IN CONFIG(R1) ;IF NO ERRORS & NOT PROGRAM SPACE
11642 045416                                              IF #BIT15 OFF.IN CONFIG+2(R1) ;IF NOT LOADER BANK
11643 045426                                                IF CPUBIT SET.IN CONFIG(R1) ;IF ACCESSABLE
11644 045436                                                  IF #BIT8 OFF.IN CONFIG+2(R1) THEN LEAVE FINDBANK ;IF PARITY
11645 045446                                                    IF #BIT6 SET.IN CONFIG(R1) AND #BIT7 OFF.IN CONFIG(R1)
11646                                                             ;IF 1ST PROTECTABLE ECC BANK
11647 045466                                                    LEAVE FINDBANK
11648 045470                                                  END ;OF IF #BIT6
11649 045470                                                  IF INHECC IS FALSE
11650 045476                                                    SET   INHECC
11651 045504  010137  002534                                  MOV   R1,INHBANK
11652 045510                                                  END; OF IF INHECC
11653 045510                                                END ;OF IF CPUBIT
11654 045510                                              END ;OF IF #BIT15
11655 045510                                            END ;OF IF #BIT7
11656 045510                                          END ;OF FOR
11657 045520                                          IF FULLREL IS FALSE
11658 045526                                            IF INHECC IS TRUE
11659 045534  013701  002534                              MOV    INHBANK,R1
11660 045540  023727  002274  000030                      CMP    REALPAT,#30       ;IS THIS PATTERN 30?
11661 045546  001421                                      BEQ    RELENT1           ;YES - SKIP MESSAGE
11662 045550  000420                                      BR     RELENT1
11663 045552                                            END; OF IF INHECC
11664 045552                                          END; OF IF FULLREL
```

```
11665 045552 005037 002532                    CLR   INHECC                    ;MAKE SURE FLAG IS TURNED OFF!
11666 045556                                  IF #SW13 OFF.IN @SWR
11667 045566 023727 002274 000030               CMP         REALPAT,#30     ;IS THIS PATTERN 30?
11668 045574 001402                             BEQ         SKUB            ;YES - SKIP MESSAGE
11669 045576                                    TYPE        MSG075          ;RELOCATION NOT POSSIBLE
11670 045602                                  END ;OF IF #SW13
11671 045602                          SKUB:   $RETURN ERROR
11672 045606                                END FINDBANK
11673 045606                                CLEAR   INHECC                    ;IF WE RELOCATED PROPERLY, THIS SHOULD BE OFF!
11674 045612 042761 020000 002652 RELENT1: BIC     #BIT13,CONFIG+2(R1)       ;INVALIDATE BACKGROUND PATTERN
11675 045620 005000                         CLR     R0
11676 045622 071027 000004                  DIV     #4,R0
11677 045626                          RELOC1: LET NEWBANK := R0
11678 045632 013737 002526 002530           MOV     PGMCSR,PGMCSR+2           ;SAVE CURRENT PGM. CSR
11679 045640 004737 046556                  CALL    USERMAP                   ;MAP NEWBANK TO USER PAR
11680 045644                                USER                              ;ENTER USER MODE
11681 045652                                BMOV    0,100000,SIZE             ;MOVE PROGRAM
11682 045664 104417                         KERNEL                            ;ENTER KERNEL MODE
11683 045666 022737 000001 003752           CMP     #1,PROTYP                 ;IS THIS AN 11/44 ?
11684 045674 001021                         BNE     JMPRL1                    ;JUMP IF NOT
11685 045676 042737 000040 172516           BIC     #BIT5,MMR3                ;TURN OFF UNIBUS MAP
11686 045704 013700 002304                  MOV     NEWBANK,R0
11687 045710 006200                         ASR     R0
11688 045712                                ON.ERROR
11689 045714 012737 100000 170200             MOV   #BIT15,MAPL0
11690 045722                                END ;OF ON.ERROR
11691 045722 010037 170202                  MOV     R0,MAPH0
11692 045726 004737 046344                  CALL    LOWMAP                    ;SETUP LOWER 16K IN UNIBUS MAP
11693 045732 052737 000040 172516           BIS     #BIT5,MMR3                ;ENERGIZE UNIBUS MAP
11694 045740 042737 000001 177572 JMPRL1: BIC     #BIT0,MMR0                ;DEENERGIZE MEMORY MANAGEMENT
11695 045746 004737 046640                  CALL    NEWKERNEL
11696 045752 013700 002304                  MOV     NEWBANK,R0
11697 045756 006300                         ASL     R0
11698 045760 006300                         ASL     R0                        ;R0 <- R0 * 4
11699 045762 016002 002650                  MOV     CONFIG(R0),R2
11700 045766 000302                         SWAB    R2
11701 045770 042702 177760                  BIC     #^C17,R2
11702 045774 006302                         ASL     R2
11703 045776 052737 000001 177572           BIS     #BIT0,MMR0                ;ENERGIZE MEMORY MANAGEMENT
11704 046004 010237 002526                  MOV     R2,PGMCSR                 ;PUT NEW PGM. CSR INTO PGMCSR
11705 046010 032760 010000 002652           BIT     #BIT12,CONFIG+2(R0)       ;IS THE NEW BANK INTERLEAVED?
11706 046016 001412                         BEQ     1$                        ;BRANCH IF NOT INTERLEAVED
11707 046020 016002 002650                  MOV     CONFIG(R0),R2
11708 046024 042702 007777                  BIC     #^C170000,R2
11709 046030 072227 177775                  ASH     #-3,R2
11710 046034 052702 100000                  BIS     #BIT15,R2
11711 046040 050237 002526                  BIS     R2,PGMCSR
11712 046044                          1$:   SET     RLFLAG
11713 046052                                $RETURN NOERROR
```

```
 11716 046056                           UNRELOCATE:SUBTST       <<UNRELOCATE PROGRAM>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       UNRELOCATE PROGRAM
                                        ;*****************************************************************************
                                        ;
 11717                                          ;RESTORE LOADERS
 11718 046056                                   PUSH    R0
 11719 046060  013701  002426                   MOV     LOADBANK,R1
 11720 046064  013700  002562                   MOV     LOADHOME,R0
 11721 046070  004737  046410                   CALL    BANKMOV
 11722 046074  004737  046742                   CALL    NEWLOAD                 ;MAP NEW LOADER BANK IN KERNEL SPACE
 11723 046100                                   PUSH    BANK
 11724 046104  013737  002426  002100         MOV     LOADBANK,BANK
 11725 046112  004737  047020                   CALL    EXBANK
 11726 046116  013701  002102                   MOV     BANKINDEX,R1
 11727 046122  042761  100000  002652         BIC     #BIT15,CONFIG+2(R1)     ;CLEAR LOADER FLAG
 11728 046130  013737  002562  002100         MOV     LOADHOME,BANK
 11729 046136  004737  047020                   CALL    EXBANK
 11730 046142  013701  002102                   MOV     BANKINDEX,R1
 11731 046146  042761  020000  002652         BIC     #BIT13,CONFIG+2(R1)     ;INVALIDATE BACKGROUND PATTERN
 11732 046154                                   POP     BANK
 11733 046160                                   CLEAR   INHECC                  ;MAKE SURE ECC TESTS ARE NOT INHIBITED!
 11734
 11735                                          ;RESTORE BANK 0
 11736 046164  042737  020000  002652         BIC     #BIT13,CONFIG+2         ;INVALIDATE BACKGROUND PATTERN
 11737 046172                                   LET NEWBANK := #0
 11738 046176  004737  046556                   CALL    USERMAP                 ;MAP NEWBANK TO USER PAR
 11739 046202                                   USER                            ;ENTER USER MODE
 11740 046210                                   BMOV    0,100000,SIZE           ;MOVE PROGRAM
 11741 046222  104417                           KERNEL                          ;ENTER KERNEL MODE
 11742 046224  042737  000001  177572         BIC     #BIT0,MMR0              ;DEENERGIZE MEMORY MANAGEMENT
 11743 046232  004737  046640                   CALL    NEWKERNEL
 11744 046236  013737  002530  002526         MOV     PGMCSR+2,PGMCSR         ;RESTORE PREVIOUS PGM. CSR
 11745 046244  052737  000001  177572         BIS     #BIT0,MMR0              ;ENERGIZE MEMORY MANAGEMENT
 11746 046252  005037  002124                   CLR     RLFLAG
 11747 046256  022737  000001  003752         CMP     #1,PROTYP               ;IS THIS AN 11/44 ?
 11748 046264  001014                           BNE     1$
 11749 046266  042737  000040  172516         BIC     #BIT5,MMR3              ;TURN OFF UNIBUS MAP
 11750 046274                                   CLEAR   MAPL0,MAPH0
 11751 046304  004737  046344                   CALL    LOWMAP                  ;SETUP LOWER 16K OF UNIBUS MAP
 11752 046310  052737  000040  172516         BIS     #BIT5,MMR3              ;ENERGIZE UNIBUS MAP
 11753 046316  012700  002652         1$:     MOV     #CONFIG+2,R0    ;MOVE 2ND WORD OF CONFIG TO R0
 11754 046322  042710  020000         2$:     BIC     #BIT13,(R0)     ;CLEAR BACKGROUND VALID BIT
 11755 046326  062700  000004                   ADD     #4,R0           ;INCREMENT TO NEXT BANK
 11756 046332  020027  003620                   CMP     R0,#3620        ;DONE?
 11757 046336  003771                           BLE     2$              ;NO - BRANCH
 11758 046340                                   POP     R0
 11759 046342  000207                           RETURN
 11760
 11761 046344                           LOWMAP: SUBTST  <<SETUP LOWER 16K OF UNIBUS MAP>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       SETUP LOWER 16K OF UNIBUS MAP
                                        ;*****************************************************************************
                                        ;
 11762 046344                                   PUSH    R0,R1,R2
 11763 046352  012700  170200                   MOV     #MAPL0,R0
 11764 046356  012701  170204                   MOV     #MAPL1,R1
 11765 046362  012702  000003                   MOV     #3,R2
 11766 046366  012011                   1$:     MOV     (R0)+,(R1)
```

```
CZMSPA0 MS11-L/M/P MEMORY DIAG. MACRO M1113  26-APR-82 09:41  PAGE 365-1
SETUP LOWER 16K OF UNIBUS MAP

 11767 046370  062721  020000                ADD     #BIT13,(R1)+
 11768 046374  012021                        MOV     (R0)+,(R1)+
 11769 046376  077205                        SOB     R2,1$
 11770 046400                                POP     R2,R1,R0
 11771 046406  000207                        RETURN
```

```
 11774 046410                           BANKMOV:SUBTST  <<MOVE BANKS>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       MOVE BANKS
                                        ;*****************************************************************************
 11775                                          ;MOVE 3/4 OF A BANK
 11776                                          ;CALLING SEQUENCE
 11777                                          ;R0 = DESTINATION BANK
 11778                                          ;R1 = SOURCE BANK
 11779 046410  104415                           SAVREG
 11780 046412  004737  046556                   CALL    USERMAP
 11781 046416  104416                           RESREG
 11782 046420  104415                           SAVREG
 11783 046422  072027  000011                   ASH     #9.,R0
 11784 046426  072127  000011                   ASH     #9.,R1
 11785 046432  012702  177650                   MOV     #UIPAR4,R2
 11786 046436  012703  000200                   MOV     #200.R3
 11787
 11788 046442  010122                           MOV     R1,(R2)+                ;MAP 1ST HALF BANK
 11789 046444  060301                           ADD     R3,R1                   ;BUMP BY 4K
 11790 046446  010122                           MOV     R1,(R2)+
 11791 046450  060301                           ADD     R3,R1
 11792
 11793 046452  010022                           MOV     R0,(R2)+
 11794 046454  060300                           ADD     R3,R0
 11795 046456  010022                           MOV     R0,(R2)+
 11796 046460  060300                           ADD     R3,R0
 11797
 11798 046462                                   USER
 11799 046470                                   BMOV    100000,140000,SIZE/2    ;MOV 1ST HALF BANK
 11800 046502  104417                           KERNEL                          ;ENTER KERNEL MODE
 11801
 11802 046504  012702  177650                   MOV     #UIPAR4,R2
 11803
 11804 046510  010122                           MOV     R1,(R2)+                ;MAP 2ND HALF BANK
 11805 046512  060301                           ADD     R3,R1                   ;BUMP BY 4K
 11806 046514  010122                           MOV     R1,(R2)+
 11807 046516  060301                           ADD     R3,R1
 11808
 11809 046520  010022                           MOV     R0,(R2)+
 11810 046522  060300                           ADD     R3,R0
 11811 046524  010022                           MOV     R0,(R2)+
 11812 046526  060300                           ADD     R3,R0
 11813
 11814 046530                                   USER
 11815 046536                                   BMOV    100000,140000,SIZE/4    ;MOV 3ND FOURTH OF BANK
 11816 046550  104417                           KERNEL                          ;ENTER KERNEL MODE
 11817
 11818 046552  104416                           RESREG
 11819 046554  000207                           RETURN
```

```
 11822 046556                           USERMAP:SUBTST  <<SUBR  MAP USER TO NEW BANK>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       SUBR    MAP USER TO NEW BANK
                                        ;*****************************************************************************
 11823 046556  012701  177640                  MOV     #UIPAR0,R1              ;COPY KERNEL PAR'S & PDR'S (0-3)
 11824 046562  012702  172340                  MOV     #KIPAR0,R2
 11825 046566  012703  177600                  MOV     #UIPDR0,R3
 11826 046572  012704  172300                  MOV     #KIPDR0,R4
 11827 046576  012705  000004                  MOV     #4,R5
 11828 046602  012221           1$:            MOV     (R2)+,(R1)+
 11829 046604  011423                          MOV     (R4),(R3)+
 11830 046606  077503                          SOB     R5,1$
 11831
 11832 046610  013700  002304                  MOV     NEWBANK,R0
 11833 046614  072027  000011                  ASH     #9.,R0                  ;BANK 1 STARTS AT 100,000 LESS 6 LSB'S
 11834                                                                         ;FOR MEMORY MANAGEMENT = 1000
 11835 046620  012705  000004                  MOV     #4,R5
 11836 046624  010021           2$:            MOV     R0,(R1)+                ;SETUP UIPAR(4-7)
 11837 046626  062700  000200                  ADD     #200,R0                 ;BUMP ADDRESS 4K
 11838 046632  011423                          MOV     (R4),(R3)+              ;SETUP UIPDR(4-7)
 11839 046634  077505                          SOB     R5,2$
 11840 046636  000207                          RETURN
 11841
 11842 046640                           NEWKERNEL:SUBTST        <<SUBR  SETUP KERNEL PAR'S FOR NEW BANK>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       SUBR    SETUP KERNEL PAR'S FOR NEW BANK
                                        ;*****************************************************************************
 11843 046640                                  PUSH    R0,R1,R5
 11844 046646  012700  172340                  MOV     #KIPAR0,R0
 11845 046652  013701  002304                  MOV     NEWBANK,R1
 11846 046656  072127  000011                  ASH     #9.,R1                  ;BANK 1 STARTS AT 100,000 LESS 6 LSB'S
 11847                                                                         ;FOR MEMORY MANAGEMENT = 1000
 11848 046662  012705  000004                  MOV     #4,R5
 11849 046666  010120           1$:            MOV     R1,(R0)+                ;SETUP KIPAR(0-3)
 11850 046670  062701  000200                  ADD     #200,R1
 11851 046674  077504                          SOB     R5,1$
 11852 046676                                  POP     R5,R1,R0
 11853 046704  000207                          RETURN
 11854
 11855 046706                           MAPKERNAL:SUBTST <<SUBR MAP KERNAL PARS 4 AND 5 TO A BANK>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       SUBR    MAP KERNAL PARS 4 AND 5 TO A BANK
                                        ;*****************************************************************************
 11856
 11857 046706  013705  002100                  MOV BANK,R5             ;MOV BANK NUMBER TO R5
 11858 046712  072527  000011                  ASH #9.,R5              ;R5 ENTERS 100000 LESS SHIFT TO CREATE MAPPING
 11859 046716  013737  172350  002266          MOV KIPAR4,SAVPAR       ;SAVE OLD PAR
 11860 046724  010537  172350                  MOV R5,KIPAR4           ;GET NEW PAR'S
 11861 046730  062705  000200                  ADD #200,R5             ;
 11862 046734  010537  172352                  MOV R5,KIPAR5           ;
 11863 046740  000207                          RETURN                  ;
 11864
 11865 046742                           NEWLOAD:SUBTST  <<SUBR  SETUP KERNEL PAR'S FOR NEW LOADER BANK>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       SUBR    SETUP KERNEL PAR'S FOR NEW LOADER BANK
                                        ;*****************************************************************************
 11866                                          ;R0 CONTAINS THE DESTINATION BANK
```

```
 11867 046742                                   PUSH    R0,R1
 11868 046746  012701  172350                  MOV     #KIPAR4,R1
 11869 046752  072027  000011                  ASH     #9.,R0                  ;BANK 1 STARTS AT 100000 LESS 6 LSB'S (1000)
 11870 046756  010021                          MOV     R0,(R1)+                ;SETUP KIPAR4
 11871 046760  062700  000200                  ADD     #200,R0
 11872 046764  010021                          MOV     R0,(R1)+                ;SETUP KIPAR5
 11873 046766                                   POP     R1,R0
 11874 046772  000207                          RETURN
 11875
 11876 046774                           UNMAP:  SUBTST  <<SUBR  UNMAP KERNAL PAR'S 4 AND 5>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       SUBR    UNMAP KERNAL PAR'S 4 AND 5
                                        ;*****************************************************************************
 11877 046774  013737  002266  172350          MOV     SAVPAR,KIPAR4           ;RESTORE KIPAR4
 11878 047002  062737  000200  002266          ADD     #200,SAVPAR             ;ADD 200 FOR NEXT PAR
 11879 047010  013737  002266  172352          MOV     SAVPAR,KIPAR5           ;RESTORE KIPAR5
 11880 047016  000207                          RETURN                          ;
```

```
 11883 047020                           EXBANK: SUBTST  <<SUBR  EXAMINE BANK>>
                                        ;*****************************************************************************
                                        ;*SUBTEST        SUBR    EXAMINE BANK
                                        ;*****************************************************************************
 11884                                  ;DOES THE FOLLOWING:
 11885                                  ;(1)   SETS UP "BANKINDEX" AND R1 BASED ON VALUE OF "BANK".
 11886                                  ;(2)   SETS THE "MKFLAG" IF THE BANK IS ECC.
 11887                                  ;(3)   SETS THE "KPFLAG" IF THE BANK IS THE PROTECTED REGION OF ECC MEMORY.
 11888                                  ;(4)   SETS THE "ACFLAG" IF THE BANK CAN BE ACCESSED BY THIS CPU.
 11889                                  ;(5)   SETS THE "PFLAG" IF THE BANK IS IN PROGRAM SPACE.
 11890                                  ;(6)   SETS THE  "RRFLAG" IF RELOCATION IS REQUIRED TO TEST THIS BANK; HOWEVER,
 11891                                  ;      IT COMPLEMENTS THIS FLAG IF THE RELOCATION FLAG "RLFLAG" IS SET (THIS IS
 11892                                  ;      NECESSARY FOR THE USE OF THE RECURSIVE "MODE" SUBROUTINES). THE "RRFLAG"
 11893                                  ;      IS ALWAYS SET TO DISABLE TESTING IF FIELD SERVICE MODE  "SELECTED BANKS"
 11894                                  ;      ARE BEING TESTED AND THIS BANK IS NOT SELECTED.
 11895                                  ;(7)   SETS THE "BMFLAG" IF THE BANK IS A BAD MEMORY;  HOWEVER, IT  COMPLEMENTS
 11896                                  ;      THIS FLAG IF THE "WORST" FLAG IS NOT SET  (THIS IS NECESSARY FOR THE USE
 11897                                  ;      OF THE RECURSIVE "MODE" SUBROUTINES).
 11898                                  ;(8) SETS THE "INTFLAG" IF THE BANK IS INTERLEAVED.
 11899                                  ;(9) SETS THE "INT64K" FLAG IF THE BANK IS INTERLEAVED ON 64K WORD BOUNDS.
 11900                                  ;(10) SETS THE "SKIPMK" FLAG IF THIS BANK IS INTERLEAVED, AND HAS ALREADY
 11901                                  ;      BEEN TESTED.
 11902                                  ;
 11903                                  ;(11) SETS THE "PMEMFLG" IF THE ECC MEMORY UNDER TEST IS A MS11-P
 11904 047020                                   PUSH    R0,R1,R2
 11905 047026                                   CLEAR   MKFLAG,KPFLAG,PMEMFLAG
 11906 047042                                   SET     ACFLAG
 11907 047050                                   CLEAR   PFLAG,RRFLAG,BMFLAG
 11908 047064                                   CLEAR   INTFLAG,INT64K,SKIPMK
 11909 047100  013701  002100                    MOV     BANK,R1
 11910 047104  006301                           ASL     R1
 11911 047106  006301                           ASL     R1                      ;R1 <- R1 * 4
 11912 047110  010137  002102                    MOV     R1,BANKINDEX
 11913 047114  032761  000100  002650           BIT     #BIT6,CONFIG(R1)        ;PROTECTED REGION OF ECC MEMORY?
 11914 047122  001403                           BEQ     1$                      ;NO - SKIP
 11915 047124                                   SET     KPFLAG
 11916 047132  012700  000002          1$:      MOV     #BIT1,R0
 11917 047136                                   IF R0 SET.IN CPUBIT AND R0 OFF.IN CONFIG(R1)
 11918 047152  005037  002114                     CLR   ACFLAG
 11919 047156                                   END :OF IF R0
 11920 047156  005737  002114                    TST     ACFLAG                  ;ACTIVE MEMORY?
 11921 047162  001415                           BEQ     2$                      ;BRANCH IF NOT
 11922 047164  016102  002652                    MOV     CONFIG+2(R1),R2
 11923 047170  000302                           SWAB    R2
 11924 047172  042702  177770                    BIC     #^C7,R2                 ;ISOLATE MEM TYPE BITS
 11925 047176  020227  000003                    CMP     R2,#3                   ;IS THIS AN ILLEGAL MEM TYPE?
 11926 047202  003405                           BLE     2$                      ;BRANCH IF NOT
 11927 047204                                   SET     BMFLAG                  ;SET BAD BANK FLAG
 11928 047212  000137  047436                    JMP     ENEXBK                  ;JUMP OVER REST OF FLAG TESTS
 11929 047216  032761  000400  002652  2$:      BIT     #BIT8,CONFIG+2(R1)      ;IS THERE ECC THERE?
 11930 047224  001412                           BEQ     3$                      ;NO - SKIP
 11931 047226                                   SET     MKFLAG                  ;YES - SET MKFLAG
 11932 047234  032761  001000  002652           BIT     #BIT9,CONFIG+2(R1)      ;IS IT A MS11-P????
 11933 047242  001403                           BEQ     3$                      ;NO SKIP!!!!
 11934 047244                                   SET     PMEMFLG                 ;SET MS11-P FLAG
 11935 047252  032761  000200  002650  3$:      BIT     #BIT7,CONFIG(R1)        ;BANK = PROGRAM SPACE?
 11936 047260  001406                           BEQ     5$                      ;NO - SKIP
```

```
11937 047262                                    SET     PFLAG,RRFLAG
11938 047276  005737  002124          5$:       TST     RLFLAG                  ;IS PROGRAM RELOCATED?
11939 047302  001402                            BEQ     6$                      ;NO - SKIP
11940 047304  005137  002122                    COM     RRFLAG                  ;YES - COMPLEMENT RELOCATION REQUIRED FLAG
11941 047310  032761  000001  002650  6$:       BIT     #BIT0,CONFIG(R1)        ;ERRORS PRESENT IN THIS BANK?
11942 047316  001403                            BEQ     8$                      ;NO - SKIP
11943 047320                                    SET     BMFLAG
11944 047326  005737  002564          8$:       TST     WORST                   ;IS THIS A WORST FIRST PASS?
11945 047332  001002                            BNE     9$                      ;YES - SKIP
11946 047334  005137  002126                    COM     BMFLAG                  ;NO - COMPLEMENT BAD MEMORY FLAG
11947 047340                            9$:     IF SELONLY IS TRUE AND #BIT14 OFF.IN CONFIG+2(R1)
11948 047356                              SET   RRFLAG
11949 047364                            END ;OF IF SELONLY
11950 047364  032761  010000  002652              BIT     #BIT12,CONFIG+2(R1)     ;IS THIS BANK INTERLEAVED?
11951 047372  001421                            BEQ     ENEXBK                  ;BRANCH IF IT IS NOT
11952 047374                                    SET     INTFLAG
11953 047402  032761  004000  002652              BIT     #BIT11,CONFIG+2(R1)     ;IS THIS BANK INTERLEAVED WITH 64K BOARDS?
11954 047410  001403                            BEQ     10$                     ;BRANCH IF IT IS NOT
11955 047412                                    SET     INT64K
11956 047420  032761  000040  002650  10$:      BIT     #BIT5,CONFIG(R1)        ;SHOULD THIS BANK BE TESTED?
11957 047426  001403                            BEQ     ENEXBK                  ;BRANCH IF IT SHOULD
11958 047430                                    SET     SKIPMK
11959 047436                            ENEXBK: POP     R2,R1,R0                ;RESTORE REGISTERS
11960 047444  000207                            RETURN
```

```
11963 047446                          BANKOK: SUBTST  <<SUBR  BANK OK?>>
                                      ;*****************************************************************************
                                      ;*SUBTEST       SUBR    BANK OK?
                                      ;*****************************************************************************
11964                                         ;TEST TO INSURE THAT THE TYPE OF MEMORY IN THE PRESENT BANK
11965                                         ;IS OF THE TYPE WE ARE TESTING 'TMFLAG'.
11966                                         ;RESULT IS RETURNED IN THE CONDITION CODES (OK = (=0)).
11967 047446  013700  002132                  MOV     TMFLAG,R0
11968 047452  005100                          COM     R0
11969 047454  013701  002116                  MOV     MKFLAG,R1
11970 047460  074001                          XOR     R0,R1
11971 047462  000207                          RETURN                          ;OK = (=OK)
11972
11973 047464                          INCRPT:
11974 047464                          INCPAT: SUBTST  <<SUBR  INCREMENT PATTERN TESTING >>
                                      ;*****************************************************************************
                                      ;*SUBTEST       SUBR    INCREMENT PATTERN TESTING
                                      ;*****************************************************************************
11975                                         ;INCREMENT THE PATTERN & SET UP THE CONDITION CODES
11976                                         ;RESULT - Z BIT SET INDICATES OVERFLOW
11977 047464  005237  002110                  INC     PATTERN
11978 047470  022737  000030  002110          CMP     #30,PATTERN             ;SET UP CONDITION CODES
11979 047476  000207                          RETURN                          ;NOT EQUAL TO ZERO IS GOOD (NO OVERFLOW)
11980
11981 047500                          SETPAT:
11982 047500                          HIPAT:  SUBTST  <<SUBR  SET HIGHEST PATTERN TESTING TYPE>>
                                      ;*****************************************************************************
                                      ;*SUBTEST       SUBR    SET HIGHEST PATTERN TESTING TYPE
                                      ;*****************************************************************************
11983 047500  012737  000027  002110          MOV     #27,PATTERN             ;SET HIGHEST PATTERN
11984 047506  000207                          RETURN
11985
11986 047510                          INCBNK: SUBTST  <<SUBR  INCREMENT BANK & TEST>>
                                      ;*****************************************************************************
                                      ;*SUBTEST       SUBR    INCREMENT BANK & TEST
                                      ;*****************************************************************************
11987                                         ;RESULTS RETURNED IN CONDITION CODES
11988 047510  005237  002100                  INC     BANK
11989 047514  023737  002552  002100          CMP     LASTBANK,BANK           ;TOO FAR?
11990 047522  000207                          RETURN
```

```
 11993 047524                           BOOT:   SUBTST  <<BOOTSTRAP ROUTINE>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       BOOTSTRAP ROUTINE
                                        ;*****************************************************************************
 11994                                          ;INITIALIZE ALL CSR'S
 11995                                          ;UNRELOCATE IF NECESSARY
 11996                                          ;FLUSH OUT ANY DBE'S
 11997                                          ;TURN OFF MEMORY MANAGEMENT
 11998                                          ;TURN OFF THE UNIBUS MAP
 11999                                          ;BOOT RK0 OR RK1
 12000 047524  104472                           ECCINIT         ;TRAP ON DOUBLE BIT ERRORS (NORMAL)
 12001 047526                                   SET4    #BOOT1          ;TRAPS TO 4 GOTO BOOT1
 12002 047534                                   IF RLFLAG IS TRUE THEN $CALL UNRELOCATE
 12003 047546  004737  024566                   CALL    MT0030          ;FLUSH OUT DBE'S
 12004 047552  104421                           DEENERGIZE              ;TURN OFF MEMORY MANAGEMENT
 12005 047554  005737  002450                   TST     N022BIT                 ;IS THIS AN 11/44 OR 11/24?
 12006 047560  001003                           BNE     BOOT1
 12007 047562  042737  000040  172516         BIC     #BIT5,MMR3                      ;TURN OFF THE UNIBUS MAP
 12008 047570  005001                   BOOT1:  CLR     R1
 12009 047572  000005                   1$:     RESET
 12010 047574  012700  177406                   MOV     #177406,R0
 12011 047600  010160  000004                   MOV     R1,4(R0)
 12012 047604  012710  177400                   MOV     #177400,(R0)
 12013 047610  012740  000005                   MOV     #5,-(R0)
 12014 047614  105710                   2$:     TSTB    (R0)
 12015 047616  100376                           BPL     2$
 12016 047620  062701  020000                   ADD     #BIT13,R1
 12017 047624  005710                           TST     (R0)
 12018 047626  100761                           BMI     1$
 12019 047630  005007                           CLR     PC
```

```
12022 047632                            EXIT:   SUBTST  <<HALT PROGRAM>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       HALT PROGRAM
                                        ;*****************************************************************************
12023 047632  004737  047664                  CALL    SHUTUP
12024 047636                            EXIT2:  IF APTFLAG IS TRUE OR ACTFLAG IS TRUE
12025 047652  000777                            BR      .
12026 047654                                    ELSE
12027 047656  000000                    $EXHALT:  HALT
12028 047660  000137  003654                     JMP     START
12029 047664                                    END ;OF IF APTFLAG
12030
12031 047664                            SHUTUP: SUBTST  <<SHUTDOWN DIAGNOSTIC>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       SHUTDOWN DIAGNOSTIC
                                        ;*****************************************************************************
12032                                           ;INITIALIZE ALL CSR'S
12033                                           ;UNRELOCATE
12034                                           ;FLUSH OUT DBE'S
12035                                           ;RESTORE LOADERS
12036                                           ;TURN OFF MEMORY MANAGEMENT
12037                                           ;UNMAP THE UNIBUS MAP
12041 047664  104472                          ECCINIT         ;TRAP ON DOUBLE BIT ERRORS (NORMAL)
12042 047666                                  IF RLFLAG IS TRUE THEN $CALL UNRELOCATE
12043 047700                                  IF QUICK IS FALSE
12044 047706  004737  024566                    CALL  MT0030          ;FLUSH OUT DBE'S
12045 047712                                  END ;OF IF QUICK
12046 047712  012700  000001                  MOV     #1,R0           ;DESTINATION BANK
12047 047716  013701  002562                  MOV     LOADHOME,R1     ;SOURCE BANK
12048 047722  004737  046410                  CALL    BANKMOV
12049 047726  104421                          DEENERGIZE              ;TURN OFF MEMORY MANAGEMENT
12050 047730  005737  002450                  TST     NO22BIT         ;DOES THIS PDP-11 HAVE 22-BIT ADDR?
12051 047734  001003                          BNE     1$              ;BRANCH IF NOT
12052 047736  042737  000040  172516          BIC     #BIT5,MMR3              ;TURN OFF UNIBUS MAP
12056 047744  000207                  1$:     RETURN
12057
12058 047746                            APTDOWN:SUBTST  <<APT SHUTDOWN SEQUENCE>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       APT SHUTDOWN SEQUENCE
                                        ;*****************************************************************************
12059 047746                                  MAP     #0                      ;MAP SUPERVISOR SPACE (TEST AREA) TO BANK #0
12060 047762                                  TESTAREA                ;ENTER TEST MODE
12061 047770  012737  047746  060024          MOV     #APTDOWN,FIRST+24
12062 047776  012737  000340  060026          MOV     #340,FIRST+26
12063 050004  012737  000000  127746          MOV     #0,FIRST+APTDOWN
12064 050012  104417                          KERNEL                  ;ENTER KERNEL MODE
12065 050014  000000                    APTHLT: HALT
```

```
 12068 050016                                   SUBTST  <<BLOCK MOVE SUBROUTINE>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       BLOCK MOVE SUBROUTINE
                                        ;*****************************************************************************
                                        ;
 12069                                          ;BLOCK3 HAS 3 ARGUEMENTS
 12070                                          ;BLOCK2 HAS 2 ARGUEMENTS
 12071                                          ;BLOCK1 HAS 1 ARGUEMENTS
 12072                                          ;
 12073                                          ;ALL ARE CALLED BY THE BMOV MACRO
 12074                                          .ENABL  LSB
 12075 050016                           BLOCK1: PUSH    R0,R1,R2
 12076 050024  012702  177640                   MOV     #FASTCITY,R2
 12077 050030  012701  000020                   MOV     #16.,R1
 12078 050034  000413                           BR      3$
 12079
 12080 050036                           BLOCK2: PUSH    R0,R1,R2
 12081 050044  012701  000020                   MOV     #16.,R1
 12082 050050  000404                           BR      2$
 12083
 12084 050052                           BLOCK3: PUSH    R0,R1,R2
 12085 050060  012501                           MOV     (R5)+,R1
 12086 050062  012502                   2$:     MOV     (R5)+,R2
 12087 050064  012500                   3$:     MOV     (R5)+,R0
 12088
 12089 050066  012022                   1$:     MOV     (R0)+,(R2)+
 12090 050070  077102                           SOB     R1,1$
 12091 050072                                   POP     R2,R1,R0
 12092 050100  000205                           RTS     R5
 12093                                          .DSABL  LSB
```

```
12095                                           .SBTTL  FIELD SERVICE MODE
12096
12097 050102                                FIELDSERVICE:SUBTST     <<SUBR  FIELD SERVICE COMMAND MODE>>
                                            ;*********************************************************************************
                                            ;*SUBTEST       SUBR    FIELD SERVICE COMMAND MODE
                                            ;*********************************************************************************
12098 050102  104415                        ;       SAVREG
12099 050104                                        TYPE    MSG020          ;FIELD SERVICE COMMAND MODE
12100
12101 050110                                        IF RLFLAG IS TRUE OR NOFSMODE IS TRUE
12102 050124                                          TYPE  MSG048          ;NOT AVAILABLE NOW - TRY LATER!
12103 050130  104416                                  RESREG
12104 050132  000207                                  RETURN
12105 050134                                        END ;OF IF RLFLAG
12106 050134  005737  002540                        TST     CACHKN
12107 050140  001402                                BEQ     1$
12108 050142                                        PUSH    CONTRL          ;SAVE CACHE STATUS
12109 050146                                1$:     PUSH    CSRNO,KAMIKAZE  ;SAVE CSR & KAMIKAZE STATUS
12110 050156  104424                                CACHOFF                 ;TURN CACHE OFF
12111 050160                                        SET     KAMIKAZE
12112 050166                                FS1:    TYPE    MSG026          ;COMMAND:
12113 050172  104414                                RDDEC                   ;READ A DECIMAL NUMBER
12114 050174                                        POP     R0              ;COMMAND --> R0
12115 050176  020027  000022                        CMP     R0,#18.
12116 050202  101403                                BLOS    1$
12117 050204                                        TYPE    MSG021
12118 050210  000766                                BR      FS1
12119 050212                                1$:     CASE R0
12120 050222  050300                                  FSCMD0                ;EXIT FIELD SERVICE COMMANDS
12121 050224  050402                                  FSCMD1                ;READ CSR
12122 050226  050512                                  FSCMD2                ;LOAD CSR
12123 050230  050660                                  FSCMD3                ;EXAMINE MEMORY
12124 050232  051134                                  FSCMD4                ;MODIFY MEMORY
12125 050234  051454                                  FSCMD5                ;SELECT BANK & PATTERN
12126 050236  052372                                  FSCMD6                ;TYPE CONFIGURATION MAP
12127 050240  052400                                  FSCMD7                ;SOB-A-LONG TEST
12128 050242  052672                                  FSCMD8                ;ERROR SUMMARY
12129 050244  053076                                  FSCMD9                ;REFRESH TEST
12130 050246  053370                                  FCMD10                ;SET FILL COUNT
12131 050250  053416                                  FCMD11                ;ENTER KAMIKAZE MODE
12132 050252  053440                                  FCMD12                ;EXIT KAMIKAZE MODE
12133 050254  053460                                  FCMD13                ;TURN CACHE OFF
12134 050256  053502                                  FCMD14                ;TURN CACHE ON
12139 050260  053520                                  FCMD15                ;TEST ONLY SELECTED BANKS
12140 050262  053604                                  FCMD16                ;RESUME TESTING ALL BANKS
12141 050264  053646                                  FCMD17                ;ENABLE TRACE
12142 050266  053662                                  FCMD18                ;DISABLE TRACE
12143 050270                                        END ;OF CASE
12144 050276  000733                                BR      FS1
```

```
12147 050300                          FSCMD0: SUBTST  <<COMMAND 0     EXIT>>
                                      ;*****************************************************************************
                                      ;*SUBTEST       COMMAND 0       EXIT
                                      ;*****************************************************************************
12148 050300                                  TYPE    MSG103          ;LEAVING FIELD SERVICE MODE
12149 050304  062706  000002                    ADD     #2,SP
12150 050310                                  IF SKIPKAMI IS TRUE
12151 050316  062706  000002                    ADD   #2,SP                     ;THROW AWAY OLD KAMIKAZE FLAG
12152 050322  005037  002006                    CLR   SKIPKAMI
12153 050326                                  ELSE
12154 050330                                    POP   KAMIKAZE                  ;RESTORE OLD KAMIKAZE FLAG
12155 050334                                  END ;OF IF SKIPKAMI
12156 050334                                  POP     CSRNO
12157 050340  005737  002540                    TST     CACHKN
12158 050344  001414                            BEQ     RES0
12159 050346                                  IF CACHKN EQ CACHKF             ;IF CACHE IS OFF
12160 050356  062706  000002                    ADD   #2,SP                     ;THROW AWAY CACHE STATUS
12161 050362                                  ELSE
12162 050364  005737  002540                    TST   CACHKN
12163 050370  001402                            BEQ   RES0
12164 050372                                    POP   CONTRL                    ;RESTORE CACHE STATUS
12165 050376                                  END ;OF IF CACHKN
12166 050376  104416                    RES0: RESREG
12167 050400  000207                          RETURN
12168
12169 050402                          FSCMD1: SUBTST  <<FS    COMMAND 1       READ CSR>>
                                      ;*****************************************************************************
                                      ;*SUBTEST       FS      COMMAND 1       READ CSR
                                      ;*****************************************************************************
12170 050402  004737  053674                  CALL    WHICHCSR
12171 050406  010637  002302                  MOV     SP,FSSTACK
12172 050412                                  SET4    #RES1           ;TRAPS TO 4 GOTO RES1
12173 050420  104426                          READCSR
12174 050422                                  SET     NOERROR
12175 050430  104026                          ERROR   +26             ;USE ERROR ROUTINE FOR PRINTOUT
12176 050432                                  RES4                    ;RESET TRAPS TO 4 TO DEFAULT
12177 050454  000207                          RETURN
12178 050456                          RES1:   TYPE    MSG025          ;THIS CSR DOES NOT EXIST
12179 050462  013706  002302                  MOV     FSSTACK,SP
12180 050466                                  RES4                    ;RESET TRAPS TO 4 TO DEFAULT
12181 050510  000207                          RETURN
```

```
 12184 050512                          FSCMD2: SUBTST  <<FS     COMMAND 2       LOAD CSR>>
                                       ;*****************************************************************************
                                       ;*SUBTEST       FS      COMMAND 2       LOAD CSR
                                       ;*****************************************************************************
                                       ;
 12185 050512  004737  053674                  CALL    WHICHCSR
 12186 050516  010637  002302                  MOV     SP,FSSTACK
 12187 050522                                  SET4    #RES2           ;TRAPS TO 4 GOTO RES2
 12188 050530  104426                          READCSR
 12189 050532                                  TYPE    M^G027
 12190 050536                                  SET     NOERROR
 12191 050544  104026                          ERROR   +26             ;USE ERROR ROUTINE FOR PRINTOUT
 12192 050546                                  RES4                    ;RESET TRAPS TO 4 TO DEFAULT
 12193 050570                                  TYPE    MSG023          ;FIRST CSR WORD
 12194 050574  104413                          RDOCT                   ;READ AN OCTAL NUMBER
 12195 050576                                  POP     CSR             ;PUT IN IN LOC ''CSR''
 12196 050602  104425                          LOADCSR
 12197 050604  104426                          READCSR
 12198 050606                                  TYPE    MSG028
 12199 050612                                  SET     NOERROR
 12200 050620  104026                          ERROR   +26             ;USE FOR PRINTOUT - NOT AN ERROR
 12201 050622  000207                          RETURN
 12202 050624                          RES2:   TYPE    MSG025          ;THIS CSR DOES NOT EXIST
 12203 050630  013706  002302                  MOV     FSSTACK,SP
 12204 050634                                  RES4                    ;RESET TRAPS TO 4 TO DEFAULT
 12205 050656  000207                          RETURN
```

```
12208 050660                                FSCMD3: SUBTST  <<FS     COMMAND 3       EXAMINE MEMORY>>
                                                    ;*****************************************************************************
                                                    ;*SUBTEST        FS      COMMAND 3       EXAMINE MEMORY
                                                    ;*****************************************************************************
                                                    ;
12209 050660                                        PUSH    BANK,NOPAR,PARTHERE,4
12210 050700  012737  000002  002074                MOV     #2,NOPAR        ;INDICATE PARITY ACTION
12211 050706                                        TYPE    MSG029          ;EXAMINE MEMORY
12212 050712                                1$:     TYPE    MSG031          ;PHYSICAL ADDRESS (0-17775776)??
12213 050716  104413                                RDOCT                   ;READ OCTAL NUMBER ONTO STACK & $HIOCT
12214 050720  013737  065140  002100                MOV     $HIOCT,BANK     ;PUT MSB'S IN BANK
12215 050726                                        POP     R0              ;PUT LSB'S IN R0
12216 050730  000241                                CLC
12217 050732  006100                                ROL     R0
12218 050734  006137  002100                        ROL     BANK
12219 050740  000241                                CLC
12220 050742  006000                                ROR     R0
12221 050744  023737  002100  002552                CMP     BANK,LASTBANK   ;CHECK FOR BANK TOO HIGH
12222 050752  003357                                BGT     1$              ;BRANCH IF TRUE
12223 050754  062700  060000                        ADD     #FIRST,R0
12224 050760  032700  000001                        BIT     #BIT0,R0        ;CHECK FOR ODD ADDRESS
12225 050764  001352                                BNE     1$              ;BRANCH IF ODD ADDRESS
12226 050766  020027  157776                        CMP     R0,#LAST        ;CHECK FOR ADDRESS OVER 16K
12227 050772  101347                                BHI     1$              ;BRANCH IF OVER 16K
12228 050774  012737  051046  002300                MOV     #3$,PARTHERE    ;INCASE OF ABORTS
12229 051002                                        SET4    #4$             ;TRAPS TO 4 GOTO 4$
12230 051010                                        MAP     BANK                    ;MAP SUPERVISOR SPACE (TEST AREA) TO BANK
12231 051024                                        TESTAREA                ;ENTER TEST MODE
12232 051032  011001                                MOV     (R0),R1
12233 051034  104417                                KERNEL                  ;ENTER KERNEL MODE
12234 051036                                        TYPOCS  R1
12235 051044  000410                                BR      EXCMD3
12236
12237 051046                                3$:     TYPE    MSG032                  ;PARITY ABORT
12238 051052  000405                                BR      EXCMD3
12239
12240 051054  062706  000004                4$:     ADD     #4,SP                   ;FIX STACK
12241 051060                                        TYPE    MSG033                  ;TIMEOUT TRAP
12242 051064  000400                                BR      EXCMD3
12243
12244 051066  104417                        EXCMD3: KERNEL                  ;ENTER KERNEL MODE
12245 051070                                        POP     4,PARTHERE,NOPAR,BANK
12246 051110                                        RES4                    ;RESET TRAPS TO 4 TO DEFAULT
12247 051132  000207                                RETURN
```

```
12250 051134                                FSCMD4: SUBTST  <<FS    COMMAND 4       MODIFY MEMORY>>
                                                    ;*****************************************************************************
                                                    ;*SUBTEST       FS      COMMAND 4       MODIFY MEMORY
                                                    ;*****************************************************************************
12251 051134                                        PUSH    BANK,NOPAR,PARTHERE,4
12252 051154  012737  000003  002074                MOV     #3,NOPAR                ;INDICATE PARITY ACTION
12253 051162                                        TYPE    MSG036                  ;MODIFY MEMORY
12254 051166                                1$:     TYPE    MSG031          ;PHYSICAL ADDRESS (0-17775776)??
12255 051172  104413                              RDOCT                   ;READ OCTAL NUMBER ONTO STACK & $HIOCT
12256 051174  013737  065140  002100              MOV     $HIOCT,BANK     ;PUT MSB'S IN BANK
12257 051202                                      POP     R0              ;PUT LSB'S IN R0
12258 051204  000241                              CLC
12259 051206  006100                              ROL     R0
12260 051210  006137  002100                      ROL     BANK
12261 051214  000241                              CLC
12262 051216  006000                              ROR     R0
12263 051220                                      IF BANK GT LASTBANK THEN GOTO 1$  ;CHECK FOR BANK TOO HIGH
12264 051230  062700  060000                      ADD     #FIRST,R0
12265 051234                                      IF #BIT0 SET.IN R0 THEN GOTO 1$ ;CHECK FOR ODD ADDRESS
12266 051242                                      IF R0 HI #LAST THEN GOTO 1$     ;CHECK FOR ADDRESS OVER 16K
12267 051250  012737  051316  002300              MOV     #3$,PARTHERE            ;INCASE OF ABORTS
12268 051256                                      SET4    #4$             ;TRAPS TO 4 GOTO 4$
12269 051264                                      MAP     BANK                    ;MAP SUPERVISOR SPACE (TEST AREA) TO BANK
12270 051300  104511                              INVALIDATE
12271 051302                                      TESTAREA                ;ENTER TEST MODE
12272 051310  011001                              MOV     (R0),R1
12273                                             ;GETTING HERE MEANS WE GOT LUCKY - NO TRAPS
12274 051312  104417                              KERNEL                  ;ENTER KERNEL MODE
12275 051314  000410                              BR      5$
12276
12277 051316                                3$:     TYPE    MSG032                  ;PARITY ABORT
12278 051322  000431                              BR      EXCMD4                  ;EXIT
12279
12280 051324  062706  000004                4$:     ADD     #4,SP           ;FIX STACK
12281 051330                                      TYPE    MSG033          ;TIMEOUT TRAP
12282 051334  000424                              BR      EXCMD4                  ;EXIT
12283
12284 051336                                5$:     TYPE    MSG037                  ;OLD DATA WAS
12285 051342                                      TYPOCS  R1                      ;PRINT IT
12286 051350                                      TYPE    MSG039                  ;INPUT NEW DATA
12287 051354  104413                              RDOCT                           ;READ ON OCTAL NUMBER ONTO THE STACK
12288 051356                                      POP     R1                      ;GET NEW NUMBER
12289 051360                                      TESTAREA                ;ENTER TEST MODE
12290 051366  010110                              MOV     R1,(R0)                 ;PUT IT IN MEMORY
12291 051370  011001                              MOV     (R0),R1                 ;READ IT AGAIN
12292 051372  104417                              KERNEL                  ;ENTER KERNEL MODE
12293 051374                                      TYPE    MSG038                  ;DATA IS NOW
12294 051400                                      TYPOCS  R1                      ;PRINT IT
12295
12296 051406  104417                        EXCMD4: KERNEL                  ;ENTER KERNEL MODE
12297 051410                                      POP     4,PARTHERE,NOPAR,BANK
12298 051430                                      RES4                    ;RESET TRAPS TO 4 TO DEFAULT
12299 051452  000207                              RETURN
```

```
12302 051454                            FSCMD5: SUBTST  <<FS     COMMAND 5       SELECT BANK & PATTERN>>
                                        ;*****************************************************************************
                                        ;*SUBTEST        FS      COMMAND 5       SELECT BANK & PATTERN
                                        ;*****************************************************************************
                                        ;
12303 051454                                    PUSH    BANK,PATTERN,TESTADD,PCBUMP,TKVEC,TKVEC+2
12304 051504  010637  002302                      MOV     SP,FSSTACK              ;SAVE LAST GOOD STACK POINTER
12305 051510                                      TYPE    MSG040                  ;SELECT BANK & PATTERN TEST
12306 051514                            1$:       TYPE    MSG030                  ;BANK(0-177)?
12307 051520  104413                              RDOCT                           ;READ AN OCTAL NUMBER ONTO THE STACK
12308 051522                                      POP     BANK                    ;PUT IT IN BANK
12309 051526                                      IF BANK GT LASTBANK THEN GOTO 1$  ;CHECK FOR BANK TOO HIGH
12310
12311 051536  013701  002100                      MOV     BANK,R1
12312 051542  006301                              ASL     R1
12313 051544  006301                              ASL     R1
12314 051546                                      IF CPUBIT OFF.IN CONFIG(R1)
12315 051556                                        TYPE  MSG041                  ;BANK NOT ACCESSABLE
12316 051562                                        GOTO 1$
12317 051564                                      END ;OF IF
12318
12319 051564                            2$:       TYPE    MSG042                  ;PATTERN(0-45)?
12320 051570  104413                              RDOCT                           ;READ AN OCTAL NUMBER ONTO THE STACK
12321 051572                                      POP     PATTERN                 ;PUT IT IN PATTERN
12322 051576                                      IF PATTERN GT #47 THEN GOTO 2$  ;CHECK FOR PATTERN TO HIGH
12323 051606                                      IF PATTERN EQ #0
12324 051614                                        TYPE  MSG043                  ;PATTERN 0 DATA IS?
12325 051620  104413                                RDOCT                         ;READ AN OCTAL NUMBER ONTO THE STACK
12326 051622                                        POP   R2                      ;PUT IT IN R2
12327 051624                                      END ;OF IF
12328
12329
12330 051624                                    MAP     BANK                    ;MAP SUPERVISOR SPACE (TEST AREA) TO BANK
12331 051640  104511                            INVALIDATE
12332 051642  004737  047020                    CALL    EXBANK                  ;SET NEW MARGINS
12333 051646                                    IF RRFLAG IS TRUE
12334 051654                                      TYPE  MSG049                  ;BANK REQUIRES RELOCATION
12335 051660  000137  052274                      JMP CMD5C
12336 051664                                    END ;OF IF RRFLAG
12337 051664                                    TYPE    MSG046          ;TO ESCAPE TYPE ANY KEY!
12338 051670  013737  002150  002152            MOV     CSRNO,SAVCSR    ;SAVE OLD CSR NUMBER
12339 051676  013702  002100                    MOV     BANK,R2         ;
12340 051702  072227  000002                    ASH     #2,R2           ;GENERATE INDEX INTO CONFIGURATION TABLE
12341 051706  016203  002650                    MOV     CONFIG(R2),R3   ;R3 = LOW WORD OF CONFIGURATION TABLE FOR THIS BANK
12342 051712  072327  177770                    ASH     #-10,R3         ;POSITION CSR CODE IN BITS 0-3
12343 051716  042703  177760                    BIC     #^C17,R3        ;CLEAR ALL BUT THE CSR CODE
12344 051722  006303                            ASL     R3              ;ADJUST CSR NUMBER
12345 051724  010337  002150                    MOV     R3,CSRNO        ;
12346 051730  012737  052274  000060            MOV     #CMD5C,TKVEC
12347 051736  012737  000340  000062            MOV     #340,TKVEC+2
12348 051744  017700  130660                    MOV     @$TKB,R0                ;KILL ANY OLD INTERRUPT
12349 051750  042737  000200  177776            BIC     #BIT7,PSW               ;LOWER CPU PRIORITY TO 140
12350 051756  052777  000100  130642            BIS     #BIT6,@$TKS             ;ENABLE KEYBOARD INTERRUPTS
12351
12352
12353 051764                                    SET     HEADER,MUT
12354 052000  013701  002100            CMD5B:  MOV     BANK,R1
12355 052004  006301                            ASL     R1
```

```
FS      COMMAND 5       SELECT BANK & PATTERN

 12356 052006  006301                          ASL     R1
 12357 052010  005037  002236                  CLR     SPLTCSR
 12358 052014  005037  002262                  CLR     PASFLG
 12359 052020  012737  060000  002406          MOV     #FIRST,TESTADD
 12360 052026  012737  060002  002410          MOV     #FIRST+2,TESTADD+2
 12361 052034                                  IF #BIT12 SET.IN CONFIG+2(R1)
 12362 052044  005237  002236                    INC   SPLTCSR
 12363 052050                                    MAP   BANK
 12364 052064  012737  120000  002410            MOV   #120000,TESTADD+2
 12365 052072                                  END; OF IF #BIT12
 12366 052072                                  IF #SW0 SET.IN @SWR
 12367 052102  104470                            ECCDIS                      ;DISABLE ERROR CORRECTION
 12368 052104                                  ELSE
 12369 052106                                    PUSH  CSRNO
 12370 052112  104502                            CLRCSR                      ;CLEAR CSRS
 12371 052114                                    POP   CSRNO
 12372 052120                                  END ;OF IF
 12373 052120  012737  000002  002074          MOV     #2,NOPAR              ;PARITY ACTION
 12374 052126  012737  000002  002322          MOV     #2,PCBUMP             ;TRAPS ADD 2 TO PC
 12375 052134  013700  002110                  MOV     PATTERN,R0
 12376 052140  006300                          ASL     R0
 12377 052142  004770  052154                  CALL    @FSPAT(R0)
 12378 052146  005037  002074                  CLR     NOPAR
 12379 052152  000712                          BR      CMD5B                 ;LOOP TILL KEYBOARD INTERRUPT
 12380
 12381 052154  020700          FSPAT:  MT0000  ;<1 SEC       DATA PATTERN TEST
 12382 052156  020760                  MT0001  ;<1 SEC       ADDRESS TEST
 12383 052160  021100                  MT0002  ;<1 SEC       COMPLEMENT ADDRESS TEST
 12384 052162  021240                  MT0003  ; 1 SEC       3 XOR 9 WORST CASE NOISE TEST
 12385 052164  021472                  MT0004  ; 1 SEC       ROTATING ZEROS TEST
 12386 052166  021614                  MT0005  ; 1 SEC       ROTATING ONES TEST
 12387 052170  021750                  MT0006  ;<1 SEC       INITIAL DATA TEST
 12388 052172  022004                  MT0007  ;<1 SEC       ADDRESS BIT TEST
 12389 052174  022046                  MT0010  ;<1 SEC       BYTE ADDRESSING TEST
 12390 052176  022102                  MT0011  ;<2 SEC       CREATE SINGLE BIT ERROR TEST
 12391 052200  022160                  MT0012  ;<1 SEC       WRITE BYTE CLEARS SBE TEST
 12392 052202  022264                  MT0013  ; 1 SEC       CREATE DOUBLE BIT ERROR TEST
 12393 052204  022350                  MT0014  ; 1 SEC       BASIC DOUBLE BIT ERROR TEST
 12394 052206  022440                  MT0015  ; 1 SEC       WRITE INHIBIT OF BYTE WITH DBE
 12395 052210  022516                  MT0016  ;<1 SEC       WRITE INHIBIT OF WORD WITH DBE
 12396 052212  022574                  MT0017  ;<1 SEC       HOLDING 1'S & 0'S TEST
 12397 052214  022616                  MT0020  ; 1 SEC       SYNDROMES TO CSR ON SBE TEST
 12398 052216  022706                  MT0021  ; 1 SEC       MARCHING 0'S & 1'S TEST
 12399 052220  023160                  MT0022  ;10 SEC       REFRESH & SHIFTING DIAGONAL TEST
 12400 052222  023212                  MT0023  ;10 SEC       SHIFTING DIAGONAL TEST
 12401 052224  023256                  MT0024  ;20 SEC       FAST GALLOPING PATTERN TEST
 12402 052226  023522                  MT0025  ;<1 SEC       INTERRUPT ENABLE TEST
 12403 052230  023600                  MT0026  ;<1 SEC       RANDOM DATA TEST
 12404 052232  024102                  MT0027  ; 1 SEC       UNIQUE BANK TEST
 12405 052234  024566                  MT0030  ; 1 SEC       FLUSH OUT DBE'S TEST
 12406 052236  025070                  MT0031  ; 3 SEC       SOB-A-LONG TEST
 12407 052240  025260                  MT0032  ;<1 SEC       WRITE RECOVERY TEST
 12408 052242  025612                  MT0033  ;35 SEC       BRANCH GOBBLE TEST
 12409 052244  026000                  MT0034  ; 1 SEC       SOFT ERROR TEST
 12410 052246  026152                  MT0035  ;<1 SEC       WORST CASE NOISE PARITY TEST
 12411 052250  026264                  MT0036  ; 1 SEC       CORRECTION CODE TEST
 12412 052252  026336                  MT0037  ;<1 SEC       ECC DISABLE TEST
```

FS COMMAND 5 SELECT BANK & PATTERN

```
12413 052254  026404                          MT0040  ; 1 SEC          NO WRITE ABORT WITH ECC DISABLED TEST
12414 052256  026406                          MT0041  ; 1 SEC          ADDRESS TO CSR ON DOUBLE BIT ERROR TEST
12415 052260  026450                          MT0042  ;<1 SEC          EXTENDED UNIBUS ADDRESS TEST
12416 052262  026514                          MT0043  ; 1 SEC          WRITE BYTE CLEARS SBE TEST
12417 052264  026554                          MT0044  ; 1 SEC          SHIFTING 1/0'S THROUGH THE CHECK BITS
12418 052266  026614                          MT0045  ; 1 SEC          SYNDROMES TO CSR ON DBE TEST
12419 052270  026654                          MT0046  ; 1 SEC          CHECK SINGLE BIT ERROR WITH ECC DISABLED TEST
12420 052272  026714                          MT0047  ;<1 SEC          NO CSR UPDATE WITH SBE ON DBE TEST
12421
12422 052274  013706  002302          CMD5C:  MOV     FSSTACK,SP               ;RECOVER OLD STACK POINTER
12423 052300  042777  000100  130320          BIC     #BIT6,@$TKS
12424 052306                                  POP     TKVEC+2,TKVEC
12425 052316  117700  130306                  MOVB    @$TKB,R0                 ;GET CHARACTER TO GET RID OF FLAG
12426 052322                                  POP     PCBUMP,TESTADD
12427 052332                                  POP     PATTERN,BANK
12428 052342                                  MAP     BANK                     ;REMAP OLD BANK
12429 052356  004737  047020                  CALL    EXBANK
12430 052362  013737  002152  002150          MOV     SAVCSR,CSRNO             ;RESTORE CSRNO.
12431 052370  000207                          RETURN
```

```
 12433 052372                          FSCMD6: SUBTST  <<FS     COMMAND 6       TYPE CONFIGURATION MAP>>
                                       ;*****************************************************************************
                                       ;*SUBTEST       FS       COMMAND 6       TYPE CONFIGURATION MAP
                                       ;*****************************************************************************
                                       ;
 12434 052372  004737  041352                  CALL    PCONFIG
 12435 052376  000207                          RETURN
 12436
```

```
12439 052400                                FSCMD7: SUBTST  <<FS    COMMAND 7       SOB-A-LONG TEST>>
                                                    ;*****************************************************************************
                                                    ;*SUBTEST       FS      COMMAND 7       SOB-A-LONG TEST
                                                    ;*****************************************************************************
12440 052400                                        PUSH    BANK,PATTERN,TKVEC,TKVEC+2,NOPAR
12441 052424  010637  002302                        MOV     SP,FSSTACK              ;SAVE LAST GOOD STACK POINTER
12442 052430                                        TYPE    MSG055                  ;SOB-A-LONG TEST
12443
12444 052434                                        IF #SW0 SET.IN @SWR
12445 052444  104470                                  ECCDIS                        ;DISABLE ERROR CORRECTION
12446 052446                                        ELSE
12447 052450  104502                                  CLRCSR                        ;CLEAR CSRS
12448 052452                                        END ;OF IF
12449 052452                                        TYPE    MSG056                  ;BELL = EACH PASS COMPLETE
12450
12451 052456                                        TYPE    MSG046                  ;TO ESCAPE TYPE ANY KEY!
12452 052462  012737  052606  000060                MOV     #CMD7C,TKVEC
12453 052470  012737  000340  000062                MOV     #340,TKVEC+2
12454 052476  017700  130126                        MOV     @$TKB,R0                ;KILL ANY OLD INTERRUPT
12455 052502  042737  000200  177776                BIC     #BIT7,PSW               ;LOWER CPU PRIORITY TO 140
12456 052510  052777  000100  130110                BIS     #BIT6,@$TKS             ;ENABLE KEYBOARD INTERRUPTS
12457
12458
12459 052516                                        SET     HEADER,MUT
12460
12461 052532                                CMD7B:  FOR BANK := #0 TO LASTBANK
12462 052536  004737  047020                          CALL EXBANK
12463 052542                                          IF ACFLAG IS TRUE AND RRFLAG IS FALSE
12464 052556  104511                                    INVALIDATE
12465 052560  004737  025070                            CALL MT0031
12466 052564                                          END ;OF IF ACFLAG
12467 052564                                        END ;OF FOR BANK
12468 052600                                        TYPE    $BELL                   ;RING BELL
12469 052604                                        GOTO    CMD7B
12470
12471 052606  013706  002302                CMD7C:  MOV     FSSTACK,SP                      ;RECOVER OLD STACK POINTER
12472 052612  042777  000100  130006                BIC     #BIT6,@$TKS
12473 052620  117700  130004                        MOVB    @$TKB,R0                        ;READ CHAR TO KILL FLAG
12474 052624                                        POP     NOPAR,TKVEC+2,TKVEC,PATTERN,BANK
12475 052650                                        MAP     BANK                    ;MAP SUPERVISOR SPACE (TEST AREA) TO BANK
12476 052664  004737  047020                        CALL    EXBANK
12477 052670  000207                                RETURN
```

```
 12480 052672                                FSCMD8: SUBTST  <<FS     COMMAND 8       ERROR SUMMARY>>
                                             ;*********************************************************************
                                             ;*SUBTEST       FS      COMMAND 8       ERROR SUMMARY
                                             ;*********************************************************************
 12481 052672                                        PUSH    R0,R2,R3,BANK
 12482 052704  013737  065646  002430                MOV     $PASS,TEMP
 12483 052712  005337  002430                        DEC     TEMP
 12484 052716                                        TYPDEC  TEMP
 12485 052724                                        TYPE    MSG125                  ;PASSES COMPLETED
 12486 052730                                        TYPDEC  $ERTTL
 12487 052736                                        TYPE    MSG079                  ;ERROR(S) DETECTED
 12488 052742                                        IF $ERTTL NE #0
 12489 052750  005037  002330                          CLR   SUCCESS
 12490 052754                                          FOR BANK := #0 TO LASTBANK
 12491 052760  013703  002100                            MOV BANK,R3
 12492 052764  070327  000004                            MUL #4,R3
 12493 052770                                            IFB CONFIG+2(R3) NE #0
 12494 052776                                              IF SUCCESS IS FALSE
 12495 053004                                                TYPE    MSG076          ;BANK  ERRORS
 12496 053010                                                SET SUCCESS
 12497 053016                                              END ;OF IF SUCCESS
 12498 053016                                              TYPOCS    BANK,3
 12499 053026  116300  002652                            MOVB      CONFIG+2(R3),R0
 12500 053032  042700  177400                            BIC       #^C377,R0
 12501 053036                                            TYPDEC    R0
 12502 053042                                            TYPE      $CRLF
 12503 053046                                          END ;OF IFB CONFIG(R3)
 12504 053046                                        END ;OF FOR BANK
 12505 053062                                      END ;OF IF $ERTTL
 12506 053062                                      POP     BANK,R3,R2,R0
 12507 053074  000207                              RETURN
```

```
 12510 053076                                FSCMD9: SUBTST  <<FS     COMMAND 9       REFRESH TEST>>
                                                     ;*****************************************************************************
                                                     ;*SUBTEST       FS      COMMAND 9       REFRESH TEST
                                                     ;*****************************************************************************
 12511 053076                                        PUSH    BANK,PATTERN,TKVEC,TKVEC+2,NOPAR
 12512 053122  010637  002302                        MOV     SP,FSSTACK                ;SAVE LAST GOOD STACK POINTER
 12513 053126                                        TYPE    MSG073                    ;REFRESH TEST
 12514
 12515 053132                                        IF #SW0 SET.IN @SWR
 12516 053142  104470                                  ECCDIS                          ;DISABLE ERROR CORRECTION
 12517 053144                                        ELSE
 12518 053146  104502                                  CLRCSR                          ;CLEAR CSRS
 12519 053150                                        END ;OF IF
 12520 053150                                        TYPE    MSG056                    ;BELL = EACH PASS COMPLETE
 12521
 12522 053154                                        TYPE    MSG046                    ;TO ESCAPE TYPE ANY KEY!
 12523 053160  012737  053304  000060                MOV     #CMD9C,TKVEC
 12524 053166  012737  000340  000062                MOV     #340,TKVEC+2
 12525 053174  017700  127430                        MOV     @$TKB,R0                  ;KILL ANY OLD INTERRUPT
 12526 053200  042737  000200  177776                BIC     #BIT7,PSW                 ;LOWER CPU PRIORITY TO 140
 12527 053206  052777  000100  127412                BIS     #BIT6,@$TKS               ;ENABLE KEYBOARD INTERRUPTS
 12528
 12529 053214                                        SET     HEADER,MUT
 12530
 12531 053230                                CMD98:  FOR BANK := #0 TO LASTBANK
 12532 053234  004737  047020                          CALL EXBANK
 12533 053240                                          IF ACFLAG IS TRUE AND RRFLAG IS FALSE
 12534 053254  104511                                    INVALIDATE
 12535 053256  004737  023160                          CALL MT0022
 12536 053262                                        END ;OF IF ACFLAG
 12537 053262                                    END ;OF FOR BANK
 12538 053276                                    TYPE    $BELL                     ;RING BELL
 12539 053302                                    GOTO    CMD98
 12540
 12541 053304  013706  002302          CMD9C:    MOV     FSSTACK,SP                ;RECOVER OLD STACK POINTER
 12542 053310  042777  000100  127310            BIC     #BIT6,@$TKS
 12543 053316  117700  127306                    MOVB    @$TKB,R0                  ;READ CHAR TO KILL FLAG
 12544 053322                                    POP     NOPAR,TKVEC+2,TKVEC,PATTERN,BANK
 12545 053346                                    MAP     BANK                      ;MAP SUPERVISOR SPACE (TEST AREA) TO BANK
 12546 053362  004737  047020                    CALL    EXBANK
 12547 053366  000207                            RETURN
 12548
```

```
12551 053370                            FCMD10: SUBTST  <<FS     COMMAND 10      SET FILL COUNT>>
                                        ;*****************************************************************************
                                        ;*SUBTEST        FS      COMMAND 10      SET FILL COUNT
                                        ;*****************************************************************************
12552 053370                                    PUSH    R0
12553 053372                                    TYPE    MSG085                  ;FILL COUNT(OCTAL)?
12554 053376  104413                              RDOCT
12555 053400                                    POP     R0
12556 053402  042700  177760                    BIC     #^C17,R0
12557 053406  110037  002353                    MOVB    R0,$FILLS
12558 053412                                    POP     R0
12559 053414  000207                            RETURN
12560
12561 053416                            FCMD11: SUBTST  <<FS     COMMAND 11      ENTER KAMIKAZE MODE>>
                                        ;*****************************************************************************
                                        ;*SUBTEST        FS      COMMAND 11      ENTER KAMIKAZE MODE
                                        ;*****************************************************************************
12562 053416                                    TYPE    MSG101          ;ENTERING KAMIKAZE MODE
12563 053422                                    SET     KAMIKAZE,SKIPKAMI
12564 053436  000207                            RETURN
12565
12566 053440                            FCMD12: SUBTST  <<FS     COMMAND 12      EXIT KAMIKAZE MODE>>
                                        ;*****************************************************************************
                                        ;*SUBTEST        FS      COMMAND 12      EXIT KAMIKAZE MODE
                                        ;*****************************************************************************
12567 053440                                    TYPE    MSG102          ;LEAVING KAMIKAZE MODE
12568 053444  005037  002004                    CLR     KAMIKAZE
12569 053450                                    SET     SKIPKAMI
12570 053456  000207                            RETURN
12571
12572 053460                            FCMD13: SUBTST  <<FS     COMMAND 13      TURN CACHE OFF>>
                                        ;*****************************************************************************
                                        ;*SUBTEST        FS      COMMAND 13      TURN CACHE OFF
                                        ;*****************************************************************************
12573 053460                                    TYPE    MSG106                  ;CACHE IS OFF
12574 053464  104424                            CACHOFF                         ;TURN CACHE OFF
12575 053466  013737  002540  002542            MOV     CACHKN,CACHKN+2         ;SAVE OLD CACHE ON STATE
12576 053474  005037  002540                    CLR     CACHKN                  ;KEEP CACHE OFF
12577 053500  000207                            RETURN
12578
12579 053502                            FCMD14: SUBTST  <<FS     COMMAND 14      TURN CACHE ON>>
                                        ;*****************************************************************************
                                        ;*SUBTEST        FS      COMMAND 14      TURN CACHE ON
                                        ;*****************************************************************************
12580 053502                                    TYPE    MSG107                  ;CACHE IS ON (EXCEPT DURING ACTUAL PATTERNS)
12581 053506  013737  002542  002540            MOV     CACHKN+2,CACHKN         ;RESTORE OLD CACHE ON STATE
12582 053514  104423                            CACHON                          ;TURN CACHE ON
12583 053516  000207                            RETURN
12584
```

```
12597
12598 053520                            FCMD15: SUBTST  <<FS    COMMAND 15      TEST ONLY SELECTED BANKS>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       FS      COMMAND 15      TEST ONLY SELECTED BANKS
                                        ;*****************************************************************************
12599 053520                                    TYPE    MSG105          ;ENTER BANKS IN OCTAL - USE NUMBER OUTSIDE RANGE TO TERMINAT
12600 053524  004737  053614                    CALL    CMD16A          ;ERASE OLD SELECTIONS
12601 053530                                    BEGIN CMD16LOOP
12602 053530                                      REPEAT
12603 053530                                        TYPE        MSG030      ;BANK(0-177)?
12604 053534  104413                              RDOCT                   ;READ AN OCTAL NUMBER ONTO THE STACK
12605 053536                                      POP R1                    ;PUT IT IN R1
12606 053540                                      IF R1 GT #177 OR R1 LT #0
12607 053552                                        LEAVE CMD16LOOP
12608 053554                                      END ;OF IF R1
12609 053554  006301                              ASL  R1
12610 053556  006301                              ASL  R1                   ;R1 <- R1 * 4
12611 053560  052761  040000  002652              BIS #BIT14,CONFIG+2(R1)
12612 053566                                     END ;OF REPEAT
12613 053570                                    END CMD16LOOP
12614 053570                                    TYPE    MSG110              ;ONLY SELECTED BANKS WILL BE TESTED
12615 053574                                    SET     SELONLY
12616 053602  000207                            RETURN
12617
12618 053604                            FCMD16: SUBTST  <<FS    COMMAND 16      RESUME TESTING ALL BANKS>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       FS      COMMAND 16      RESUME TESTING ALL BANKS
                                        ;*****************************************************************************
12619 053604                                    TYPE    MSG111              ;ALL BANKS WILL BE TESTED
12620 053610  005037  002000                    CLR     SELONLY
12621
12622                                           ;ENTRY POINT FROM CMD15
12623 053614  013702  002552            CMD16A: MOV     LASTBANK,R2
12624 053620  006302                            ASL     R2
12625 053622  006302                            ASL     R2
12626 053624                                    FOR R1 := #0 TO R2 BY #4
12627 053626  042761  040000  002652              BIC   #BIT14,CONFIG+2(R1)
12628 053634                                    END ;OF FOR R1
12629 053644  000207                            RETURN
```

FS	COMMAND 16	RESUME TESTING ALL BANKS

```
12632 053646                            FCMD17: SUBTST  <<FS     COMMAND 17      ENABLE TRACE>>
                                        ;*****************************************************************************
                                        ;*SUBTEST        FS      COMMAND 17      ENABLE TRACE
                                        ;*****************************************************************************
12633 053646                                    TYPE    MSG127
12634 053652  012737  177777  006204            MOV     #-1,TRACE
12635 053660  000207                            RETURN
```

FS COMMAND 17 ENABLE TRACE

```
 12638 053662                           FCMD18: SUBTST  <<FS    COMMAND 18      DISABLE TRACE>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       FS      COMMAND 18      DISABLE TRACE
                                        ;*****************************************************************************
                                        ;
 12639 053662                                   TYPE    MSG128
 12640 053666  005037  006204                   CLR     TRACE
 12641 053672  000207                           RETURN
```

FS COMMAND 18 DISABLE TRACE

```
 12644 053674                           WHICHCSR:SUBTST <<SUBR  DETERMINE CORRECT CSR>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       SUBR    DETERMINE CORRECT CSR
                                        ;*****************************************************************************
                                        ;
 12645 053674  013700  002222                  MOV     TOTCSRS,R0      ;GET CSR'S FLAG
 12646 053700  022700  100000                  CMP     #BIT15,R0       ;CSR 0?
 12647 053704  001003                          BNE     1$              ;NO - SKIP
 12648 053706  005037  002150                  CLR     CSRNO           ;YES - SET IT UP
 12649 053712  000207                          RETURN
 12650
 12651 053714                           1$:    TYPE    MSG022          ;WHICH CSR(0-F)
 12652 053720  104412                          RDLIN                   ;GET CHARACTER
 12653 053722                                  POP     R0              ;PUT IN R0
 12654 053724  011000                          MOV     (R0),R0         ;PUT CHAR IN R0
 12655 053726  020027  000106                  CMP     R0,#106         ;CHECK LIMIT
 12656 053732  101370                          BHI     1$              ;IF BAD LOOP TILL HE TYPES IT RIGHT
 12657 053734  022700  000101                  CMP     #'A,R0
 12658 053740  103002                          BHIS    2$
 12659 053742  162700  000007                  SUB     #7,R0
 12660 053746  162700  000060           2$:    SUB     #60,R0
 12661 053752  006300                          ASL     R0
 12662 053754  010037  002150                  MOV     R0,CSRNO
 12663 053760  000207                          RETURN
```

```
13212                                   .SBTTL  ERROR DATA (SUPERVISOR) SETUP STUFF
13213 053762                    $PER25: LET ADDRESS := R1 - #2
13214 053774                            IF ABORTFLAG IS FALSE
13215 054002                              TESTAREA              ;ENTER TEST MODE
13216 054010                              LET BAD := -2(R1)
13217 054016  104417                      KERNEL                ;ENTER KERNEL MODE
13218 054020                            END ;OF IF ABORTFLAG
13219 054020                            IF 177654 EQ #0
13220 054026                              LET GOOD := R2
13221 054032                            ELSE
13222 054034                              LET GOOD := R3
13223 054040                            END ;OF IF
13224 054040  000137  057242            JMP     PERRAW
13225
13226 054044                    PERRA3: SUBTST  <<DATA WAS 3 WORDS>>
                                ;*****************************************************************************
                                ;*SUBTEST       DATA WAS 3 WORDS
                                ;*****************************************************************************
13227 054044                            IF BADPC EQ #0 THEN $CALL BADSTACK
13228 054056                            PUSH    R0
13229 054060  005037  002146            CLR     CSR             ;MAKE SURE CSR BIT HOLDER IS CLEAR
13230 054064  104505                    CHK1DIS                 ;DISABLE ECC & WRITE CHECKBITS FROM 1 SELECTED CSR
13231 054066                            TESTAREA
13232 054074  005711                    TST     (R1)            ;READ LOCATION TO READ CHECKBITS INTO CSR
13233 054076  104417                    KERNEL
13234 054100  104426                    READCSR                 ;GET CSR CONTENTS
13235 054102  013700  002146            MOV     CSR,R0          ;SAVE CSR CONTENTS IN R0
13236 054106  104503                    CLR1CSR                 ;RETURN CSR TO NORMAL MODE
13237 054110  072027  177773            ASH     #-5,R0          ;MOVE CHECK BITS TO BOTTOM OF WORD
13238 054114  042700  177600            BIC     #^C177,R0       ;CLEAR OFF EXTRANEOUS GARBAGE
13239 054120                            LET ADDRESS := R1       ;SAVE VIRTUAL ADDRESS FOR PRINTOUT
13240 054124  005037  002042            CLR     GOOD            ;FIRST TEST WORD WRITTEN SHOULD ALWAYS BE ZERO
13241 054130                            TESTAREA                ;ENTER TEST MODE
13242 054136  011137  002050            MOV     (R1),BAD        ;GET BAD DATA FROM MUT - FIRST WORD
13243 054142  011437  002052            MOV     (R4),BAD2       ;AND SECOND WORD
13244 054146  104417                    KERNEL                  ;ENTER KERNEL MODE
13245 054150  110037  002054            MOVB    R0,BAD3         ;MOVE BAD CHECKBITS FOR PRINTOUT
13246 054154  105037  002055            CLRB    BAD3+1          ;CLEAR OFF THE OTHER UNUSED BITS
13247 054160  004737  057476            CALL    PERBNK          ;MARK BANK AS BAD IN CONFIG TABLE
13248 054164  104033                    ERROR   +33
13249 054166                            POP     R0              ;RESTORE R0
13250 054170                            IF #SW0 SET.IN @SWR
13251 054200  104506                      ENASBE                ;TRAP ON SINGLE BIT ERRORS
13252 054202                            ELSE
13253 054204  104472                      ECCINIT               ;TRAP ON UNCORRECTABLE ERRORS
13254 054206                            END; OF IF #SW0
13255 054206  000002                    RTI
```

DATA WAS 3 WORDS

```
 13258 054210                           $PER30: LET GOOD := R1
 13259 054214                                   LET ADDRESS := (SP) - 16
 13260 054226                                   IF ABORTFLAG IS FALSE
 13261 054234                                     TESTAREA              ;ENTER TEST MODE
 13262 054242                                     LET BAD := @ADDRESS
 13263 054250  104417                             KERNEL                ;ENTER KERNEL MODE
 13264 054252                                   END ;OF IF ABORTFLAG
 13265 054252  000137  057242                   JMP PERRAW
 13266
 13267 054256                           GETDATA:SUBTST  <<GET DATA FROM ABORTED AREA IF POSSIBLE>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       GET DATA FROM ABORTED AREA IF POSSIBLE
                                        ;*****************************************************************************
 13268 054256                                   PUSH    R0,4,114
 13269 054270  010637  054354                   MOV     SP,GETDA1
 13270 054274  012737  054334  000004           MOV     #1$,4
 13271 054302  012737  054334  000114           MOV     #1$,114
 13272 054310  013700  002032                   MOV     ADDRESS,R0
 13273 054314                                   TESTAREA
 13274 054322  011037  002050                   MOV     (R0),BAD
 13275 054326  104417                           KERNEL
 13276 054330  005037  002142                   CLR     ABORTFLAG
 13277 054334  013706  054354           1$:     MOV     GETDA1,SP             ;RESTORE KNOWN GOOD STACK POINTER
 13278 054340                                   POP     114,4,R0
 13279 054352  000207                           RETURN
 13280 054354  000000                   GETDA1: 0
```

```
                                                .SBTTL  POWER FAIL AUTO RESTART
                                                .SBTTL  ROUTINE POWER DOWN AND UP
                                        ;*******************************************************************************
                                        ;POWER DOWN ROUTINE
054356                                  $PWRDN:
                                                ;SAVE CACHE STATUS
054356  005737  002540                          TST     CACHKN
054362  001403                                  BEQ     5$
054364                                          PUSH    CONTRL
054370  104423                                  CACHON                          ;TURN CACHE ON
054372  012737  055330  000024        5$:     MOV     #$ILLUP,PWRVEC ;;SET FOR FAST UP
054400  012737  000340  000026                MOV     #340,PWRVEC+2   ;;PRIO:7
054406                                          PUSH    R0,R1,R2,R3,R4,R5,CSRNO
                                                ;SAVE USER PAR'S & PDR'S
054426  012700  177700                          MOV     #177700,R0
054432  012701  000021                          MOV     #17.,R1
054436                                  1$:     PUSH    -(R0)
054440  077102                                  SOB     R1,1$
                                                ;SAVE SUPERVISOR PAR'S
054442  005737  002452                          TST     NOSUPER
054446  001013                                  BNE     PD1
054450  012700  172300                          MOV     #172300,R0
054454  012701  000020                          MOV     #16.,R1
054460                                  2$:     PUSH    -(R0)
054462  077102                                  SOB     R1,2$
054464                                          IF RLFLAG IS TRUE THEN $CALL WOOPS
                                                ;COPY KERNEL MAP TO USER & SUPERVISOR
054476  012700  172300                  PD1:    MOV     #KIPDR0,R0
054502  012701  177600                          MOV     #UIPDR0,R1
054506  012702  172200                          MOV     #SIPDR0,R2
054512  012703  000040                          MOV     #32.,R3
054516  011021                          3$:     MOV     (R0),(R1)+
054520  012022                                  MOV     (R0)+,(R2)+
054522  077303                                  SOB     R3,3$
```

```
13325                                                   ;SAVE USER & SUPERVISOR STACK POINTERS
13326 054524                                            USER
13327 054532  010600                                    MOV     USP,R0
13328 054534  104417                                    KERNEL                  ;ENTER KERNEL MODE
13329 054536                                            PUSH    R0
13330 054540  005737  002452                            TST     NOSUPER
13331 054544  001006                                    BNE     7$
13332 054546                                            SUPERVISOR              ;ENTER SUPERVISOR MODE
13333 054554  010600                                    MOV     SSP,R0
13334 054556  104417                                    KERNEL                  ;ENTER KERNEL MODE
13335 054560                                            PUSH    R0
13336                                                   ;SAVE ECC REGISTERS
13337 054562  013701  002222            7$:             MOV     TOTCSRS,R1      ;GET CSR'S
13338 054566                                            BEGIN   LCSRSAVE
13339 054566                                              FOR CSRNO := #0 TO #36 BY #2
13340 054572  006301                                        ASL         R1
13341 054574                                              ON.ERROR
13342 054576  104426                                        READCSR
13343 054600                                                PUSH      CSR
13344 054604                                              END ;OF ON.ERROR
13345 054604                                            IF R1 EQ #0 THEN LEAVE LCSRSAVE
13346 054610                                            END ;OF FOR CSRNO
13347 054626                                            END LCSRSAVE
13348                                                   ;SAVE MMR0,1,2,3
13349 054626                                            PUSH    MMR0,MMR1,MMR2
13350 054642  005737  002452                            TST     NOSUPER
13351 054646  001002                                    BNE     8$
13352 054650                                            PUSH    MMR3
13353                                                   ;SAVE KERNEL PAR'S
13354 054654  012700  172400            8$:             MOV     #172400,R0
13355 054660  012701  000020                            MOV     #16.,R1
13356 054664                            4$:             PUSH    -(R0)
13357 054666  077102                                    SOB     R1,4$
13358                                                   ;SAVE UNIBUS MAP REGISTERS
13359 054670  022737  000001  003752                    CMP     #1,PROTYP               ;IS THIS AN 11/44?
13360 054676  001004                                    BNE     9$                      ;BRANCH IF NOT
13361 054700                                            PUSH    MAPH0,MAPL0
13362                                                   ;SAVE POSSIBLE SOFTWARE SWITCH REGISTER
13363 054710                            9$:             PUSH    @SWR
13364                                                   ;SAVE STACK POINTER
13365 054714  010637  055334                            MOV     SP,$SAVR6       ;;SAVE SP
13366                                                   ;NOW SET UP REAL VECTOR
13367 054720  012737  054732  000024                    MOV     #$PWRUP,PWRVEC  ;;SET UP VECTOR
13368 054726  000000                    $DOWN:          HALT
13369 054730  000776                                    BR      $DOWN           ;;HANG UP
```

```
13372                                   ;*****************************************************************************************
13373                                   ;POWER UP ROUTINE
13374 054732                            $PWRUP:
13378 054732  012737  055330  000024              MOV     #$ILLUP,PWRVEC ;;SET FOR FAST DOWN
13379                                             ;RESTORE STACK POINTER
13380 054740  013706  055334                      MOV     $SAVR6,SP       ;;GET SP
13381 054744  005037  055334                      CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
13382 054750  005237  055334              1$:     INC     $SAVR6          ;;WAIT FOR THE INC
13383 054754  001375                              BNE     1$              ;;OF A WORD
13384                                             ;RESTORE POSSIBLE SOFTWARE SWITCH REGISTER
13385 054756                                      POP     @SWR
13386                                             ;RESTORE UNIBUS MAP
13387 054762  022737  000001  003752              CMP     #1,PROTYP               ;IS THIS AN 11/44?
13388 054770  001006                              BNE     10$
13389 054772                                      POP     MAPL0,MAPH0
13390 055002  004737  046344                      CALL    LOWMAP          ;SETUP LOWER 16K OF UNIBUS MAP
13391                                             ;RESTORE KERNEL PAR'S & PDR'S
13392 055006  012700  172340              10$:    MOV     #172340,R0
13393 055012  012702  172300                      MOV     #KIPDR0,R2
13394 055016  012701  000020                      MOV     #16.,R1
13395 055022                              6$:     POP     (R0)+
13396 055024  012722  077406                      MOV     #77406,(R2)+
13397 055030  077104                              SOB     R1,6$
13398                                             ;RESTORE MMR3,2,1,0
13399 055032  005737  002452                      TST     NOSUPER
13400 055036  001002                              BNE     11$
13401 055040                                      POP     MMR3
13402 055044                              11$:    POP     MMR2,MMR1,MMR0
13403                                             ;RESTORE ECC REGISTERS
13404 055060  013701  002222                      MOV     TOTCSRS,R1      ;GET CSR'S
13405 055064  042701  177400                      BIC     #177400,R1
13406 055070                                      BEGIN LCSRRESTORE
13407 055070                                        FOR CSRNO := #36 DOWNTO #0 BY #2
13408 055076  006201                                  ASR         R1
13409 055100                                          ON.ERROR
13410 055102                                            POP       CSR
13411 055106  104425                                    LOADCSR
13412 055110                                          END ;OF ON.ERROR
13413 055110                                        IF R1 EQ #0 THEN LEAVE LCSRRESTORE
13414 055114                                        END ;OF FOR CSRNO
13415 055132                                      END LCSRRESTORE
13416                                             ;COPY KERNEL MAP TO USER & SUPERVISOR
13417 055132  012700  172300                      MOV     #KIPDR0,R0
13418 055136  012701  177600                      MOV     #UIPDR0,R1
13419 055142  012702  172200                      MOV     #SIPDR0,R2
13420 055146  012703  000040                      MOV     #32.,R3
13421 055152  011021                      3$:     MOV     (R0),(R1)+
13422 055154  012022                              MOV     (R0)+,(R2)+
13423 055156  077303                              SOB     R3,3$
```

```
13425
13426 055160  005737  002452                    ;RESTORE SUPERVISOR & USER STACK POINTERS
13427 055164  001006                    TST     NOSUPER
13428 055166                            BNE     13$
13429 055170                            POP     R0
13430 055176  010006                    SUPERV'SOR              ;ENTER SUPERVISOR MODE
13431 055200  104417                    MOV     R0,SSP
13432 055202                            KERNEL                  ;ENTER KERNEL MODE
13433 055204                    13$:    POP     R0
13434 055212  010006                    USER
13435 055214  104417                    MOV     R0,USP
13436                                   KERNEL                  ;ENTER KERNEL MODE
13437 055216  012700  172240            ;RESTORE SUPERVISOR PAR'S
13438 055222  012701  000020            MOV     #172240,R0
13439 055226                            MOV     #16.,R1
13440 055230  077102              7$:   POP     (R0)+
13441                                   SOB     R1,7$
13442 055232  012700  177636            ;RESTORE USER PAR'S & PDR7
13443 055236  012701  000021            MOV     #177636,R0
13444 055242                            MOV     #17.,R1
13445 055244  077102              8$:   POP     (R0)+
13446                                   SOB     R1,8$
13447 055246  013777  002010  125350    ;RESTORE POSSIBLE SOFTWARE DISPLAY REGISTER
13448 055254  013737  002010  000174    MOV     $PATMAR,@DISPLAY
13449 055262                            MOV     $PATMAR,DISPREG
13450 055302  012737  054356  000024    POP     CSRNO,R5,R4,R3,R2,R1,R0
13451 055310                            MOV     #$PWRDN,PWRVEC  ;;SET UP THE POWER DOWN VECTOR
13452                                   TYP'    MSG051          ;REPORT THE POWER FAILURE
13453 055314  005737  002540            ;R ,TORE CACHE STATUS
13454 055320  001402                    T .     CACHKN
13455 055322                            [ :Q    9$
13456 055326  000002                    FOP     CONTRL
13457 055330  000000              9$:   RTI
13458 055332  000776              $ILLUP: HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
13459 055334  000000                    BR      $ILLUP          ;; BEFORE THE POWER DOWN WAS COMPLETE
13460                             $SAVR6: 0                       ;;PUT THE SP HERE
                                          .EVEN
```

```
 13472 055336                                WOOPS:  SUBTST  <<POWER FAIL WHILE RELOCATED>>
                                             ;*****************************************************************************
                                             ;*SUBTEST       POWER FAIL WHILE RELOCATED
                                             ;*****************************************************************************
 13473 055336                                        PUSH    BANK
 13474 055342  005037  002100                        CLR     BANK
 13475 055346                                        MAP     BANK                    ;MAP SUPERVISOR SPACE (TEST AREA) TO BANK
 13476 055362                                        SUPERVISOR                      ;ENTER SUPERVISOR MODE
 13477 055370  013737  060024  055734                MOV     FIRST+PWRVEC,WOOPSAV
 13478 055376  013737  060026  055736                MOV     FIRST+PWRVEC+2,WOOPSAV+2
 13479 055404                                        BMOV    FIRST+WOOPUP,WOOPSAV+4,WOOPEND-WOOPUP/2+12.
 13480 055416  012737  055522  060024                MOV     #WOOPUP,FIRST+PWRVEC
 13481 055424  012737  000340  060026                MOV     #340,FIRST+PWRVEC+2
 13482 055432                                        BMOV    WOOPUP,FIRST+WOOPUP,WOOPEND-WOOPUP/2
 13483 055444  012700  172340                        MOV     #KIPAR0,R0
 13484 055450  012701  135704                        MOV     #FIRST+WOOPEND,R1
 13485 055454  012702  000010                        MOV     #8.,R2
 13486 055460  012021                        1$:     MOV     (R0)+,(R1)+
 13487 055462  077202                                SOB     R2,1$
 13488 055464  005737  002452                        TST     NOSUPER
 13489 055470  001002                                BNE     2$
 13490 055472  013721  172516                        MOV     MMR3,(R1)+
 13491 055476  013721  177576                2$:     MOV     MMR2,(R1)+
 13492 055502  013721  177574                        MOV     MMR1,(R1)+
 13493 055506  013721  177572                        MOV     MMR0,(R1)+
 13494 055512  104417                                KERNEL                          ;ENTER KERNEL MODE
 13495 055514                                        POP     BANK
 13496 055520  000207                                RETURN
```

```
13499 055522                                WOOPUP: SUBTST  <<POWER UP FROM BANK 0 TO RELOCATION>>
                                            ;*****************************************************************************
                                            ;*SUBTEST       POWER UP FROM BANK 0 TO RELOCATION
                                            ;*****************************************************************************

13500 055522  012700  055704                        MOV     #WOOPEND,R0
13501 055526  012701  172340                        MOV     #KIPAR0,R1
13502 055532  012703  172300                        MOV     #KIPDR0,R3
13503 055536  012702  000010                        MOV     #8.,R2
13504 055542  012021                        1$:     MOV     (R0)+,(R1)+
13505 055544  012723  077406                        MOV     #77406,(R3)+
13506 055550  077204                                SOB     R2,1$
13507 055552  005737  002452                        TST     NOSUPER
13508 055556  001002                                BNE     3$
13509 055560  012037  172516                        MOV     (R0)+,MMR3
13510 055564  012037  177576                3$:     MOV     (R0)+,MMR2
13511 055570  012037  177574                        MOV     (R0)+,MMR1
13512 055574  012037  177572                        MOV     (R0)+,MMR0
13513 055600  013706  055334                        MOV     $SAVR6,SP
13514 055604                                        PUSH    BANK
13515 055610  005037  002100                        CLR     BANK
13516 055614                                        MAP     BANK                    ;MAP SUPERVISOR SPACE (TEST AREA) TO BANK
13517 055630                                        SUPERVISOR                      ;ENTER SUPERVISOR MODE
13518 055636  013737  055734  060024                MOV     WOOPSAV,FIRST+PWRVEC
13519 055644  013737  055736  060026                MOV     WOOPSAV+2,FIRST+PWRVEC+2
13520                                               ;SIMULATE THE FOLLOWING BLOCK MOV BUT WITH NO STACK ACCESSES
13521                                               ;BMOV   WOOPSAV+4,FIRST+WOOPUP,WOOPEND-WOOPUP/2+12.
13522 055652  012700  055740                        MOV     #WOOPSAV+4,R0
13523 055656  012701  000105                        MOV     #WOOPEND-WOOPUP/2+12.,R1
13524 055662  012702  135522                        MOV     #FIRST+WOOPUP,R2
13525 055666  012022                        2$:     MOV     (R0)+,(R2)+
13526 055670  077102                                SOB     R1,2$
13527
13528 055672  104417                                KERNEL                          ;ENTER KERNEL MODE
13529 055674                                        POP     BANK
13530 055700  000137  054732                        JMP     $PWRUP
13531 055704  000014                  WOOPEND:.REPT 12.
13534 055734  000107                  WOOPSAV:.REPT WOOPEND-WOOPUP/2+12.+2
```

```
13539                                           .SBTTL  IO SUBROUTINES
13540
13541                                           .SBTTL  ROUTINE TYPE
13542
13543                                   ;*******************************************************************************
13544                                   ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
13545                                   ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
13546                                   ;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
13547                                   ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
13548                                   ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
13549                                   ;*
13550                                   ;*CALL:
13551                                   ;*1) USING A TRAP INSTRUCTION
13552                                   ;*      TYPE    MESADR          ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
13553                                   ;*OR
13554                                   ;*      TYPE
13555                                   ;*      MESADR
13556                                   ;*
13557
13558 056152  105737  002054    $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
13559 056156  100407                    BMI     6$              ;BR IF NO
13560 056160  010046            1$:     MOV     R0,-(SP)        ;;SAVE R0
13561 056162  017600  000002            MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
13562 056166  112046            4$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
13563 056170  001005                    BNE     7$              ;;BR IF IT ISN'T THE TERMINATOR
13564 056172  005726                    TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
13565 056174  012600            5$:     MOV     (SP)+,R0        ;;RESTORE R0
13566 056176  062716  000002    6$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
13567 056202  000002                    RTI                     ;;RETURN
13568 056204  122716  000011    7$:     CMPB    #HT,(SP)        ;BRANCH IF NOT <HT>
13569 056210  001002                    BNE     11$
13570 056212  112716  000040            MOVB    #' ,(SP)        ;REPLACE TAB WITH SPACE
13571 056216  122716  000200    11$:    CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
13572 056222  001006                    BNE     8$
13573 056224  005726                    TST     (SP)+           ;;POP  <CR><LF> EQUIV
13574 056226                            TYPE                    ;;TYPE A CR AND LF
13575 056230  002644                    $CRLF
13576 056232  105037  056464            CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
13577 056236  000753                    BR      4$              ;;GET NEXT CHARACTER
13578 056240  004737  056300    8$:     CALL    $TYPEC          ;;GO TYPE THIS CHARACTER
13579 056244  123726  002636    9$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
13580 056250  001346                    BNE     4$              ;;IF NO GO GET NEXT CHAR.
13581 056252  013746  002352            MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
13582                                                           ;;AND THE NULL CHAR.
13583 056256  105366  000001    10$:    DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
13584 056262  002770                    BLT     9$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
13585 056264  004737  056300            CALL    $TYPEC          ;;GO TYPE A NULL
13586 056270  105337  056464            DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
13587 056274  000770                    BR      10$             ;;LOOP
13588 056276  000000            XOCHAR: .WORD 0
13589 056300                    $TYPEC: PUSH    R1
13590 056302  116601  000004            MOVB    4(SP),R1
13591 056306  005737  002540            TST     CACHKN
13592 056312  001402                    BEQ     2$
13593 056314                            PUSH    CONTRL
13594 056320                    2$:     PUSH    R0
13595 056322  104424                    CACHOFF                 ;TURN CACHE OFF
```

```
13620 056324  105777  124302          3$:       TSTB   @$TPS           ;;WAIT UNTIL PRINTER IS READY
13621 056330  100375                            BPL    3$
13622 056332  005037  056276                    CLR    XOCHAR
13623 056336  105777  124264                    TSTB   @$TKS           ;;CHECK FOR XOFF
13624 056342  100032                            BPL    NC              ;;SKIP IF NO CHARACTER
13625 056344  117737  124260  056276            MOVB   @$TKB,XOCHAR    ;;SAVE THE CHARACTER
13626 056352  042737  177600  056276            BIC    #^C177,XOCHAR   ;;STRIP OFF ASCII
13627 056360  023727  056276  000023            CMP    XOCHAR,#023     ;;WAS IT A CONTROL S?
13628 056366  001020                            BNE    NC              ;;BRANCH IF NOT
13629 056370  105777  124232          CONTS3:   TSTB   @$TKS           ;;WAIT FOR CHARACTER
13630 056374  100375                            BPL    CONTS3
13631 056376  117737  124226  056276            MOVB   @$TKB,XOCHAR    ;;GET CHARACTER
13632 056404  042737  177600  056276            BIC    #^C177,XOCHAR   ;;STRIP OFF ASCII
13633 056412                                    IF XOCHAR EQ #21        ;; IF IT IS A ^Q
13634 056422  000402                              BR   NC
13635 056424                                    ELSE
13636 056426  000760                              BR   CONTS3
13637 056430                                    END ;OF IF XOCHAR
13638 056430  110177  124200          NC:       MOVB   R1,@$TPB        ;;LOAD CHAR TO BE TYPED INTO DATA REG.
13642 056434  122766  000015  000002            CMPB   #CR,2(SP)        ;;IS CHARACTER A CARRIAGE RETURN?
13643 056442  001003                            BNE    1$               ;;BRANCH IF NO
13644 056444  105037  056464                    CLRB   $CHARCNT         ;;YES--CLEAR CHARACTER COUNT
13645 056450  000406                            BR     $TYPEX           ;;EXIT
13646 056452  122766  000012  000002  1$:       CMPB   #LF,2(SP)        ;;IS CHARACTER A LINE FEED?
13647 056460  001402                            BEQ    $TYPEX           ;;BRANCH IF YES
13648 056462  105227                            INCB   (PC)+            ;;COUNT THE CHARACTER
13649 056464  000000          $CHARCNT:.WORD    0                       ;;CHARACTER COUNT STORAGE
13650 056466                  $TYPEX:   POP     R0
13651 056470  005737  002540                    TST    CACHKN           ;IS THERE A CACHE?
13652 056474  001402                            BEQ    2$               ;BRANCH IF NOT
13653 056476                                    POP    CONTRL           ;POP CACHE STATUS
13654 056502                  2$:       POP     R1
13655 056504  000207                            RETURN
13656 056506                  SUPLIMIT:;!!!!!!!!!!!!!THIS IS THE LIMIT ON SUPERVISOR MAPPED TO MUT SPACE
```

```
14334                                   .SBTTL  ERROR DATA SETUP
14335
14336                           ;USE THIS       IF THIS CONDITION DISCRIBES THE ERROR
14337                           ;
14338                           ;PERR01         TRAP
14339                           ;               BAD DATA IN R0 UNLESS ABORTED
14340                           ;                       THEN BAD DATA IS POINTED TO BY -(R4)
14341                           ;               GOOD DATA IN R5
14342                           ;
14343                           ;PERR02         TRAP
14344                           ;               BAD DATA IN R1 UNLESS ABORTED
14345                           ;                       THEN BAD DATA IS POINTED TO BY -(R4)
14346                           ;               GOOD DATA IN R2
14347                           ;
14348                           ;PERR03         TRAP
14349                           ;               BAD DATA IS POINTED TO BY -(R1)
14350                           ;               GOOD DATA IN R4
14351                           ;
14352                           ;PERR04         TRAP
14353                           ;               BAD DATA IN R4 UNLESS ABORTED
14354                           ;                       THEN BAD DATA IS POINTED TO BY -2(R0)
14355                           ;               GOOD DATA IN R2
14356                           ;
14357                           ;PERR05         JSR     PC
14358                           ;               BAD DATA IS POINTED TO BY -(R0)
14359                           ;               GOOD DATA IN R2
14360                           ;               RETURN AFTER SETTING UP GOOD,BAD,ADDRESS
14361                           ;
14362                           ;PERR06         JSR     PC
14363                           ;               BAD DATA IS POINTED TO BY -(R0)
14364                           ;               GOOD DATA IS ZERO
14365                           ;               RETURN AFTER SETTING UP GOOD,BAD,ADDRESS
14366                           ;
14367                           ;PERR07         TRAP
14368                           ;               BAD DATA IN R2 UNLESS ABORTED
14369                           ;                       THEN BAD DATA IS POINTED TO BY (R1)
14370                           ;               GOOD DATA IN DATBUF
14371                           ;
14372                           ;PERR10         TRAP
14373                           ;               BAD DATA IN R2 UNLESS ABORTED
14374                           ;                       THEN BAD DATA IS POINTED TO BY 2(R1)
14375                           ;               GOOD DATA IN DATBUF+2
14376                           ;
14377                           ;PERR11         TRAP
14378                           ;               BYTE TEST
14379                           ;               BAD DATA IN RIGHT BYTE OF R0 UNLESS ABORTED
14380                           ;                       THEN BAD DATA IS POINTED TO BY (R1)
14381                           ;               GOOD DATA IS A ZERO BYTE
14382                           ;
14383                           ;PERR12         TRAP
14384                           ;               BYTE TEST
14385                           ;               BAD DATA IN RIGHT BYTE OF R0 UNLESS ABORTED
14386                           ;                       THEN BAD DATA IS POINTED TO BY (R1)
14387                           ;               GOOD DATA IS A BYTE OF ONES
14388                           ;
14389                           ;PERR13         TRAP
14390                           ;               BAD DATA IN R0 UNLESS ABORTED
```

ERROR DATA SETUP

```
                                        THEN BAD DATA IS POINTED TO BY (R1)
                                GOOD DATA IS ZERO

PERR14          TRAP
                BAD DATA IN R0 UNLESS ABORTED
                                        THEN BAD DATA IS POINTED TO BY (R1)
                GOOD DATA IS ONES

PERR15          TRAP
                BAD DATA IN R0 UNLESS ABORTED
                                        THEN BAD DATA IS POINTED TO BY (R1)
                GOOD DATA IN TSTDAT

PERR16          TRAP
                BAD DATA IN R0 UNLESS ABORTED
                                        THEN BAD DATA IS POINTED TO BY (R1)
                GOOD DATA IN TSTDAT+2

PERR17          TRAP
                BAD DATA IN R0 UNLESS ABORTED
                                        THEN BAD DATA IS POINTED TO BY (R1)
                GOOD DATA IN R2

PERR20          TRAP
                BAD DATA IN R0 UNLESS ABORTED
                                        THEN BAD DATA IS POINTED TO BY (R1)
                GOOD DATA IN R3

PERR21          TRAP
                7 BIT BYTE TEST
                BAD DATA IN RIGHT BYTE OF R0 UNLESS ABORTED
                                        THEN BAD DATA IS POINTED TO BY (R1)
                GOOD DATA IS A 7 BIT BYTE ON ONES

PERR22          TRAP
                BAD DATA IN R2 UNLESS ABORTED
                                        THEN BAD DATA IS POINTED TO BY (R1)
                GOOD DATA IN R0

PERR23          TRAP
                BAD DATA IN R0 UNLESS ABORTED
                                        THEN BAD DATA IS POINTED TO BY (R1)
                GOOD DATA IN R4

PERR24          TRAP
                BAD DATA IN R0 UNLESS ABORTED
                                        THEN BAD DATA IS POINTED TO BY (R2)
                GOOD DATA IN R3

PERR25          TRAP
                BAD DATA POINTED TO BY -(R1)
                GOOD DATA IN R2 UNLESS LOC V177654 IS SET
                                        THEN GOOD DATA IS IN R3

PERR26          TRAP
                BAD DATA IS DOUBLE WORD POINTED TO BY R1 AND IN LOW 7 BITS OF R0
                GOOD DATA IS 000000,,100000,,100
```

ERROR DATA SETUP

```
14448                   ;
14449                   ;PERR27         TRAP
14450                   ;               BAD DATA IS DOUBLE WORD POINTED TO BY R1 AND IN LOW 7 BITS OF R0
14451                   ;               GOOD DATA IS 000000,,000000,,077
14452                   ;
14453                   ;PERR30         TRAP
14454                   ;               BAD DATA IS POINTED TO BY -16(SP)
14455                   ;               GOOD DATA IS IN R1
14456                   ;
14457                   ;PERR31         TRAP
14458                   ;               SPECIAL ECC FAILURE HANDLER
14459                   ;
14460                   ;PERR32         TRAP
14461                   ;               SPECIAL ECC FAILURE HANDLER
14462                   ;
14463                   ;PERR33         TRAP
14464                   ;               SPECIAL ECC FAILURE HANDLER
14465                   ;
14466                   ;PERR34         TRAP
14467                   ;               SPECIAL ECC FAILURE HANDLER
14468                   ;
14469                   ;PERR35         TRAP
14470                   ;               SPECIAL BRANCH GOBBLE FAILURE HANDLER
14471                   ;
14472                   ;       CALLING SEQUENCE FOR TRAP TYPES
14473                   ;       BEQ     2$              ;NO - ERROR,BRANCH FOR CARD
14474                   ;       PERRXX                  ;TRAP TO ERROR ROUTINE
14475                   ;2$:    NEXT    INSTRUCTION     ;CONTINUE TESTING
```

ERROR DATA SETUP

```
14478 056506  010437  002032          $PER01: MOV     R4,ADDRESS
14479 056512  162737  000002  002032          SUB     #2,ADDRESS
14480 056520  010037  002050                  MOV     R0,BAD
14481 056524  010537  002042                  MOV     R5,GOOD
14482 056530  000137  057242                  JMP PERRAW
14483
14484 056534  010437  002032          $PER02: MOV     R4,ADDRESS
14485 056540  162737  000002  002032          SUB     #2,ADDRESS
14486 056546  010137  002050                  MOV     R1,BAD
14487 056552  010237  002042                  MOV     R2,GOOD
14488 056556  000137  057242                  JMP PERRAW
14489
14490 056562  010137  002032          $PER03: MOV     R1,ADDRESS
14491 056566  162737  000002  002032          SUB     #2,ADDRESS
14492 056574  010437  002042                  MOV     R4,GOOD
14493 056600  016137  177776  002050          MOV     -2(R1),BAD
14494 056606  000137  057242                  JMP PERRAW
14495
14496 056612  010037  002032          $PER04: MOV     R0,ADDRESS
14497 056616  162737  000002  002032          SUB     #2,ADDRESS
14498 056624  010437  002050                  MOV     R4,BAD
14499 056630  010237  002042                  MOV     R2,GOOD
14500 056634  000137  057242                  JMP     PERRAW
14501
14502 056640  010237  002042          PERR05: MOV     R2,GOOD
14503 056644  014037  002050          PERA05: MOV     -(R0),BAD
14504 056650  010037  002032                  MOV     R0,ADDRESS
14505 056654  062700  000002                  ADD     #2,R0           ;RESTORE R0
14506 056660  004737  042614                  CALL    BADSTACK
14507 056664  000207                          RETURN
14508
14509 056666  005037  002042          PERR06: CLR     GOOD
14510 056672  000764                          BR      PERA05
14511
14512 056674  010137  002032          $PER07: MOV     R1,ADDRESS
14513 056700  010237  002050                  MOV     R2,BAD
14514 056704  013737  002240  002042          MOV     DATBUF,GOOD
14515 056712  000137  057242                  JMP     PERRAW
14516
14517 056716                          $PER10: LET ADDRESS := R1 + #2
14518 056730                                  LET BAD := R2
14519 056734                                  LET GOOD := DATBUF+2
14520 056742  000137  057242                  JMP     PERRAW
14521
14522 056746                          $PER11: LET ADDRESS := R1
14523 056752                                  LET BAD := R0
14524 056756                                  LET GOOD := #0
14525 056762  000137  057314                  JMP     PERRAB
14526
14527 056766                          $PER12: LET ADDRESS := R1
14528 056772                                  LET BAD := R0
14529 056776                                  LET GOOD := #377
14530 057004  000137  057314                  JMP     PERRAB
```

ERROR DATA SETUP

```
14533 057010                          $PER13: LET ADDRESS := R1
14534 057014                                  LET BAD := R0
14535 057020                                  LET GOOD := #0
14536 057024  000137  057242                  JMP     PERRAW
14537
14538 057030                          $PER14: LET ADDRESS := R1
14539 057034                                  LET BAD := R0
14540 057040                                  LET GOOD := ONES
14541 057046  000137  057242                  JMP     PERRAW
14542
14543 057052                          $PER15: LET ADDRESS := R1
14544 057056                                  LET BAD := R0
14545 057062                                  LET GOOD := TSTDAT
14546 057070  000137  057242                  JMP     PERRAW
14547
14548 057074                          $PER16: LET ADDRESS := R1
14549 057100                                  LET BAD := R0
14550 057104                                  LET GOOD := TSTDAT+2
14551 057112  000453                          BR      PERRAW
14552
14553 057114                          $PER17: LET ADDRESS := R1
14554 057120                                  LET BAD := R0
14555 057124                                  LET GOOD := R2
14556 057130  000444                          BR      PERRAW
14557
14558 057132                          $PER20: LET ADDRESS := R1
14559 057136                                  LET BAD := R0
14560 057142                                  LET GOOD := R3
14561 057146  000435                          BR      PERRAW
14562
14563 057150                          $PER21: LET ADDRESS := R1
14564 057154                                  LET BAD := R0
14565 057160                                  LET GOOD := #177
14566 057166  000477                          BR      PERRA7
14567
14568 057170                          $PER22: LET ADDRESS := R1
14569 057174                                  LET BAD := R2
14570 057200                                  LET GOOD := R0
14571 057204  000416                          BR      PERRAW
14572
14573 057206                          $PER23: LET ADDRESS := R1
14574 057212                                  LET BAD := R0
14575 057216                                  LET GOOD := R4
14576 057222  000407                          BR      PERRAW
14577
14578 057224                          $PER24: LET ADDRESS := R2
14579 057230                                  LET BAD := R0
14580 057234                                  LET GOOD := R3
14581 057240  000400                          BR      PERRAW
```

ERROR DATA SETUP

```
 14583 057242                           PERRAW: SUBTST  <<DATA WAS A WORD>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       DATA WAS A WORD
                                        ;*****************************************************************************
 14584 057242  004737  057476                  CALL    PERBNK
 14585 057246                                  IF ABORTFLAG IS TRUE THEN $CALL GETDATA
 14586 057260                                  IF BADPC EQ #0 THEN $CALL BADSTACK
 14587 057272  004737  057452                  CALL    PERXOR
 14588 057276                                  IF ABORTFLAG IS FALSE
 14589 057304  104011                            ERROR +11
 14590 057306                                  ELSE
 14591 057310  104012                            ERROR +12
 14592 057312                                  END ;OF IF ABORTFLAG
 14593 057312  000002                          RTI
 14594
 14595 057314                           PERRAB: SUBTST  <<DATA WAS A BYTE>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       DATA WAS A BYTE
                                        ;*****************************************************************************
 14596 057314  004737  057476                  CALL    PERBNK
 14597 057320                                  IF ABORTFLAG IS TRUE THEN $CALL GETDATA
 14598 057332                                  IF BADPC EQ #0 THEN $CALL BADSTACK
 14599 057344  004737  057452                  CALL    PERXOR
 14600 057350                                  IF ABORTFLAG IS FALSE
 14601 057356  104014                            ERROR +14
 14602 057360                                  ELSE
 14603 057362  104015                            ERROR +15
 14604 057364                                  END ;OF IF ABORTFLAG
 14605 057364  000002                          RTI
```

```
 14608 057366                           PERRA7: SUBTST  <<DATA WAS A 7 BIT BYTE>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       DATA WAS A 7 BIT BYTE
                                        ;*****************************************************************************
                                        ;
 14609 057366                                   IF BADPC EQ #0 THEN $CALL BADSTACK
 14610 057400  004737  057452                   CALL    PERXOR
 14611 057404  004737  057476                   CALL    PERBNK
 14612 057410  104022                           ERROR   +22
 14613 057412  000002                           RTI
 14614
 14615 057414                           $PER26: LET GOOD2 := #100000
 14616 057422                                   LET GOOD3 := #100
 14617 057430  000137  054044                   JMP PERRA3
 14618
 14619 057434  005037  002044           $PER27: CLR     GOOD2
 14620 057440                                   LET GOOD3 := #077
 14621 057446  000137  054044                   JMP PERRA3
 14622
 14623 057452                           PERXOR: SUBTST  <<DETERMINE XOR OF GOOD & BAD>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       DETERMINE XOR OF GOOD & BAD
                                        ;*****************************************************************************
                                        ;
 14624 057452                                   PUSH    R0
 14625 057454  013700  002042                   MOV     GOOD,R0
 14626 057460  013737  002050  002056           MOV     BAD,BADXOR
 14627 057466  074037  002056                   XOR     R0,BADXOR
 14628 057472                                   POP     R0
 14629 057474  000207                           RETURN
```

DETERMINE XOR OF GOOD & BAD

```
14632 057476                                PERBNK: SUBTST<<LOG ERROR ON BAD BANK>>
                                            ;*****************************************************************************
                                            ;*SUBTEST       LOG ERROR ON BAD BANK
                                            ;*****************************************************************************
14633                                               ;WHILE WE'RE HERE LET'S MARK THE BAD BANK IN THE CONFIGURATION TABLE
14634 057476                                        PUSH    R0,R1
14635 057502  013701  002100                        MOV     BANK,R1
14636 057506  006301                                ASL     R1
14637 057510  006301                                ASL     R1
14638 057512  052761  000001  002650                BIS     #BIT0,CONFIG(R1)
14639 057520  105261  002652                        INCB    CONFIG+2(R1)            ;BUMP BANK COUNTER
14640 057524  001002                                BNE     12$                     ;NO OVERFLOW - SKIP
14641 057526  105361  002652                        DECB    CONFIG+2(R1)            ;SET BACK TO 255.
14642 057532  126137  002652  002550  12$:         CMPB    CONFIG+2(R1),ERRMAX     ;IS IT PAST MAX?
14643 057540  101403                                BLOS    11$                     ;NO - SKIP
14644 057542                                        SET     TOOMANY                 ;YES
14645 057550                                11$:    POP     R1,R0
14646 057554  000207                                RETURN
14647
14648 057556  010037  002050                PERECC: MOV     R0,BAD
14649 057562                                        IF ADDRESS EQ TESTADD
14650 057572  013737  002244  002042                  MOV TSTDAT,GOOD
14651 057600                                        ELSE
14652 057602  013737  002246  002042                  MOV   TSTDAT+2,GOOD
14653 057610                                        END ;OF IF (R1)
14654 057610  004737  057452                        CALL    PERXOR
14655 057614                                        SET     HEADER
14656 057622  000207                                RETURN
14657
14658 057624                                $PER31: IF REALPAT EQ #41
14659 057634  104023                                  ERROR +23
14660 057636                                        END
14661 057636                                        IF BADPC EQ #0 THEN $CALL BADSTACK
14662 057650  004737  057556                        CALL    PERECC
14663 057654                                        IF REALPAT EQ #11
14664 057664  104037                                  ERROR +37
14665 057666                                        END ;OF IF REALPAT
14666 057666                                        IF REALPAT EQ #15
14667 057676  104043                                  ERROR +43
14668 057700                                        END ;OF IF REALPAT
14669 057700                                        IF REALPAT EQ #16
14670 057710  104044                                  ERROR +44
14671 057712                                        END ;OF IF REALPAT
14672 057712                                        SET HEADER
14673 057720  000002                                RTI
```

LOG ERROR ON BAD BANK

```
14676 057722                                $PER32: IF BADPC EQ #0 THEN $CALL BADSTACK
14677 057734  010137  002032                      MOV     R1,ADDRESS
14678 057740  010037  002050                      MOV     R0,BAD
14679 057744  010237  002042                      MOV     R2,GOOD
14680 057750                                      SET     HEADER
14681 057756  104040                              ERROR   +40
14682 057760                                      SET     HEADER
14683 057766  000002                              RTI
14684
14685 057770                                $PER33: IF BADPC EQ #0 THEN $CALL BADSTACK
14686 060002  010137  002032                      MOV     R1,ADDRESS
14687 060006  010037  002050                      MOV     R0,BAD
14688 060012  105037  002051                      CLRB    BAD+1
14689 060016  012737  000377  002042              MOV     #377,GOOD
14690 060024  004737  057452                      CALL    PERXOR
14691 060030                                      SET     HEADER
14692 060036  104041                              ERROR   +41
14693 060040                                      SET     HEADER
14694 060046  000002                              RTI
14695
14696 060050                                $PER34:IF BADPC EQ #0 THEN $CALL BADSTACK
14697 060062                                      IF #BIT15!BIT4 OFF.IN CSR
14698 060072  104016                                ERROR +16                       ;NO SBE OR DBE
14699 060074                                      ELSE
14700 060076  104001                                ERROR +1                        ;EXPECTED SBE SO DBE MUST HAVE GOTTEN SET
14701 060100                                      END ;OF IF #BIT15!BIT4
14702 060100  000002                              RTI
14703
14704                                             ;DURING BRANCH GOBBLE THE CONDITION CODES WERE WRONG
14705 060102  004737  057476              $PER35: CALL    PERBNK
14706 060106  004737  042614                      CALL    BADSTACK
14707 060112  013737  002030  002050              MOV     BADPSW,BAD
14708 060120  012737  000012  002042              MOV     #12,GOOD
14709 060126  104047                              ERROR   +47
14710 060130  062706  000004                      ADD     #4,SP                   ;FIX STACK FROM TRAP
14711 060134  000207                              RETURN                          ;ABORTING TEST
14712
14713 060136  010037  002042              $PER36: MOV     R0,GOOD
14714 060142  010137  002050                      MOV     R1,BAD
14715 060146                                      SET     HEADER
14716 060154  104023                              ERROR   +23
14717 060156                                      SET     HEADER
14718 060164  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
060166  005237  065650          $SCOPE: INC     $DEVCT          ;TELL APT WE ARE ALIVE
060172                                  IF RESULT IS LT
060174  005037  065650                    CLR   $DEVCT
060200  105237  065652                    INCB  $UNIT
060204                                  END ;OF IF RESULT
060204  104410                          CKSWR                   ;;TEST  FOR CHANGE IN SOFT-SWR
060206  005737  006204                  TST     TRACE
060212  001402                          BEQ     NOTRCE
060214  004737  064144                  CALL    CONTT           ;TRACE
060220                          NOTRCE:
060220  005737  061322                  TST     CPERRF          ;IS THERE A CPU ERROR REGISTER?         ;R-C
060224  001410                          BEQ     SKJ             ;BRANCH IF NOT                          ;R-C
060226  013737  177766  061320          MOV     @#177766,CPSAVE ;GET CONTENTS OF ERROR REGISTER         ;R-C
060234  032737  000001  061320          BIT     #BIT0,CPSAVE    ;IS THE POWER FAIL MONITOR BIT SET?     ;R-C
060242  001401                          BEQ     SKJ             ;BRANCH IF NOT                          ;R-C
060244  104177                          ERROR   +177            ;REPORT IF SO                           ;R-C
060246                          SKJ:    IF STOPOK IS TRUE AND #SW8 SET.IN @SWR                          ;R-C
060264  005037  002414                    CLR   STOPOK
060270  000137  047632                    JMP   EXIT
060274                                  END ;OF IF STOPOK
060274                                  IF NOSCOPE IS TRUE
060302  000002                            RTI
060304                                  END ;OF IF NOSCOPE
060304                          1$:     IF #SW14 SET.IN @SWR THEN GOTO $OVER
                                ;#####START OF CODE FOR THE XOR TESTER#####
060314  000425                  $XTSTR: BR      2$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
                                                                ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
060316  013746  000004                  MOV     ERRVEC,-(SP)    ;;SAVE THE CONTENTS OF THE ERROR VECTOR
060322  012737  060342  000004          MOV     #1$,ERRVEC      ;;SET FOR TIMEOUT
060330  005737  177060                  TST     177060  ;;TIMF OUT ON XOR?
060334  012637  000004                  MOV     (SP)+,ERRVEC    ;;RESTORE THE ERROR VECTOR
060340  000430                          BR      $SVLAD          ;;GO TO THE NEXT TEST
060342  062706  000004          1$:     ADD     #4,SP           ;FIX STACK FROM TRAP
060346  022737  000001  00375^          CMP     #1,PROTYP       ;IS THIS AN 11/44?
060354  001002                          BNE     6$              ;BRANCH IF NOT
060356  005.37  177766                  CLR     CPUERR          ;RESET CPU ERROR REGISTER
060362  012637  000004          6$:     MOV     (SP)+,ERRVEC    ;;RESTORE THE ERROR VECTOR
060366  000407                          BR      4$              ;;LOOP ON THE PRESENT TEST
060370                          2$:;#####END OF CODE FOR THE XOR TESTER#####
060370  105737  002012          3$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
060374  001412                          BEQ     $SVLA^          ;;BR IF NO
060376  032777  001000  122216          BIT     #SW9,@SWR       ;;LOOP ON ERROR?
060404  001404                          BEQ     5$              ;;BR IF NO
060406  013737  002610  002606  4$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
060414  000410                          BR      $OVER
060416  105037  002012          5$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
060422  011637  002606          $SVLAD: MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
060426  011637  002610                  MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
```

```
 14786 060432  005037  002356                    CLR     $ESCAPE          ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
 14787 060436  004737  060450          $OVER:    CALL    GETDIS
 14788 060442  013716  002606                    MOV     $LPADR,(SP)      ;;FUDGE RETURN ADDRESS
 14789 060446  000002                            RTI                      ;;FIXES PS
```

```
 14791 060450                           GETDIS: SUBTST  <<SUBR  DISPLAY>>
                                        ;*****************************************************************************
                                        ;*SUBTEST        SUBR    DISPLAY
                                        ;*****************************************************************************
                                        ;
 14792 060450  113737  002100  002011          MOVB    BANK,$BANK
 14793 060456  113737  002274  002010          MOVB    REALPAT,$PATMAR
 14794 060464                                  PUSH    R0
 14795 060466  005737  002124                  TST     RLFLAG                  ;ARE WE RELOCATED?
 14796 060472  001403                          BEQ     1$                      ;NO - SKIP
 14797 060474  052737  100000  002010          BIS     #BIT15,$PATMAR          ;YES - SET MSB
 14798 060502                           1$:
 14802 060502  013777  002010  122114          MOV     $PATMAR,@DISPLAY
 14803 060510  013737  002010  000174          MOV     $PATMAR,DISPREG         ;SOFTWARE DISPLAY REGISTER
 14804 060516                                  POP     R0
 14805 060520  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
060522  105037  061316          $ERROR: CLRB    IBSAVE                                          ;R-C
060526                                  IF NOERROR IS FALSE
060534  104410                            CKSWR                 ;;TEST FOR CHANGE IN SOFT-SWR
060536                          BACK:                                                           ;R-C
060536  105237  002012          1$:       INCB  $ERFLG          ;;SET THE ERROR FLAG
060542  001775                            BEQ   1$              ;;DON'T LET THE FLAG GO TO ZERO
060544  004737  060450                    CALL  GETDIS          ;SETUP DISPLAY STUFF
060550  013737  002010  065644            MOV   $PATMAR,$TESTN  ;FOR APT
060556  032777  002000  122036            BIT   #SW10,@SWR      ;;BELL ON ERROR?
060564  001404                            BEQ   2$              ;;NO - SKIP
060566                                    TYPE  $BELL           ;;RING BELL
060572                                    TYPE  MSG014          ;CONTROL Z
060576  005237  002614          2$:       INC   $ERTTL          ;;COUNT THE NUMBER OF ERRORS
060602                                    IF RESULT IS MI
060604  012737  077777  002614              MOV #77777,$ERTTL
060612                                    END ;OF IF RESULT
060612                                  END ;OF IF NOERROR
060612  011637  002016                  MOV     (SP),ERRPC    ;;GET ADDRESS OF ERROR INSTRUCTION
060616  162737  000002  002016          SUB     #2,ERRPC
060624  010637  002022                  MOV     SP,ERRSP
060630  016637  000002  002026          MOV     2(SP),ERRPSW
060636  117737  121154  002013          MOVB    @ERRPC,$ITEMB ;;STRIP AND SAVE THE ERROR ITEM CODE
060644
060644  122737  000177  002013          CMPB    #177,$ITEMB   ;IS THIS THE POWER FAIL CALL?           ;R-C
060652  001431                          BEQ     1001$         ;BRANCH IF SO                           ;R-C
060654  105737  061316                  TSTB    IBSAVE        ;2ND ERROR CALL?                        ;R-C
060660  001024                          BNE     1000$         ;BRANCH IF SO                           ;R-C
060662  005737  061322                  TST     CPERRF        ;IS THERE A CPU ERROR REGISTER?         ;R-C
060666  001423                          BEQ     1001$         ;BRANCH IF NOT                          ;R-C
060670  013737  177766  061320          MOV     177766,CPSAVE ;SAVE CONTENTS                          ;R-C
060676  032737  000001  061320          BIT     #BIT0,CPSAVE  ;POWER MONITOR BIT SET?                 ;R-C
060704  001414                          BEQ     1001$         ;BRANCH IF NOT                          ;R-C
060706  042737  000001  177766          BIC     #BIT0,177766  ;CLEAR THE BIT                          ;R-C
060714  112737  002013  061316          MOVB    #$ITEMB,IBSAVE ;MAKE IBSAVE NON-ZERO FOR DUAL CALL    ;R-C
060722  112737  000177  002013          MOVB    #177,$ITEMB   ;SET $ITEMB TO POWER FAIL POINTER       ;R-C
060730  000402                          BR      1001$                                                 ;R-C
060732  105037  061316          1000$:  CLRB    IBSAVE                                                ;R-C
060736                          1001$:                                                                ;R-C
060736                                  IF NOERROR IS FALSE
060744                                    IF BADPC NE #0
060752  013737  002020  002016              MOV BADPC,ERRPC
060760  162737  000002  002016              SUB #2,ERRPC
060766  013737  002024  002022              MOV BADSP,ERRSP
```

```
CZMSPA0 MS11-L/M/P MEMORY DIAG. MACRO M1113  26-APR-82 09:41  PAGE 465-1
ROUTINE ERROR HANDLER

 14865 060774  013737  002030  002026              MOV BADPSW,ERRPSW
 14866 061002  005037  002020                      CLR BADPC
 14867 061006                                    END ;IF
 14868 061006  013737  002016  065642            MOV   ERRPC,$FATAL     ;FOR APT
 14869 061014  004737  057476                    CALL PERBNK            ;;LOG ERROR ON BANK
 14870 061020                                    IF #SW13 SET.IN @SWR
 14871 061030  000420                              BR  3$
 14872 061032                                    END ;OF IF #SW13
 14873 061032                                    IF #SW5 SET.IN @SWR AND TOOMANY IS TRUE
 14874 061050                                      GOTO 3$
 14875 061052                                    END ;OF IF #SW5
 14876 061052                                  END ;OF IF NOERROR
 14877 061052  004737  061324                  CALL    $ERRTYP          ;;GO TO USER ERROR ROUTINE
 14878 061056                                  IF MONFLG IS TRUE        ;;SHOULD WE RETURN TO XXDP MONITOR???
 14879 061064  013706  002270                    MOV  SAVMON,SP         ;;GET MONITOR ADDRESS
 14880 061070  000207                            RTS  PC                ;;GO TO MONITOR
 14881 061072                                  END                      ;;
```

```
14883 061072                                3$:     IF NOERROR IS FALSE
14884 061100  005777  121516                          TST   @SWR              ;;HALT ON ERROR
14885 061104  100002                                  BPL   7$                ;;SKIP IF CONTINUE
14886 061106  000000                        $HALT:    HALT                    ;;HALT ON ERROR!
14887 061110  104410                                  CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
14888 061112                                7$:       IF NOSCOPE IS FALSE AND #SW9 SET.IN @SWR
14889 061130  013716  002610                            MOV $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
14890 061134                                          END ;OF IF NOSCOPE
14891 061134  005737  002356                          TST   $ESCAPE           ;;CHECK FOR AN ESCAPE ADDRESS
14892 061140  001402                                  BEQ   9$                ;;BR IF NONE
14893 061142  013716  002356                          MOV   $ESCAPE,(SP)      ;;FUDGE RETURN ADDRESS FOR ESCAPE
14894 061146                                9$:       IF DETFLAG IS FALSE
14895 061154  022737  000001  003752                    CMP #1,PROTYP                 ;IS THIS AN 11/44?
14896 061162  001002                                    BNE 11$
14897 061164  005037  177766                            CLR CPUERR
14898 061170                                      11$:  IF ACTFLAG IS TRUE OR APTFLAG IS TRUE OR FATAL$ IS TRUE
14899 061212  012737  000001  065640                      MOV       #1,$MSGTY       ;FOR APT
14900 061220  000137  047632                            JMP       EXIT
14901 061224                                          END ;OF IF ACTFLAG
14902 061224                                          IF XXDPCHAIN IS TRUE AND $ERTTL HI #20
14903 061242                                            TYPE      MSG066          ;ERROR COUNT EXCEEDED 20 - ABORTING FOR XXDP CHAIN
14904 061246  013700  000042                            MOV       42,R0
14905 061252  005037  000042                            CLR       42
14906 061256  000137  015212                            JMP       $ZAP42
14907 061262                                          END ;OF IF XXDPCHAIN
14908 061262                                        END ;OF IF DETFLAG
14909 061262                                      ELSE
14910 061264                                        SET   HEADER
14911 061272                                      END ;OF IF NOERROR
14912 061272                                10$:    CLEAR   TOOMANY,NOERROR
14913 061302  105737  061316                      TSTB    IBSAVE                ;POWER FAIL ERROR CALL? ;R-C
14914 061306  001402                              BEQ     213$                                          ;R-C
14915 061310  000137  060536                      JMP     BACK                  ;JUMP IF SO             ;R-C
14916 061314  000002                        213$:   RTI                         ;;RETURN
14917 061316  000000                        IBSAVE: .WORD 0                                             ;R-C
14918 061320  000000                        CPSAVE: .WORD 0                                             ;R-C
14919 061322  000000                        CPERRF: .WORD 0                                             ;R-C
14920                                               .DSABL  LSB
```

```
                                        .SBTTL  ROUTINE ERROR MESSAGE TYPEOUT

                                ;*******************************************************************************
                                ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
                                ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
                                ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.

061324  104415                  $ERRTYP:SAVREG
061326                                  TYPE    $CRLF           ;;"CARRIAGE RETURN" & "LINE FEED"
061332  005000                          CLR     R0              ;;PICKUP THE ITEM INDEX
061334  153700  002013                  BISB    $ITEMB,R0
061340  001004                          BNE     1$              ;;IF ITEM NUMBER IS ZERO, JUST
                                                                ;;TYPE THE PC OF THE ERROR
061342                                  TYPOCT  ERRPC,<ERROR ADDRESS>
061350  000511                          BR      11$             ;;GET OUT
061352  122700  000177          1$:     CMPB    #177,R0         ;POWER MONITOR CALL?                    ;R-C
061356  001003                          BNE     100$            ;BRANCH IF NOT                          ;R-C
061360  012700  061634                  MOV     #PFECWS,R0      ;MOV ADDRESS OF PFE BIT ERROR TO R0     ;R-C
061364  000406                          BR      110$                                                    ;R-C
061366  005300                  100$:   DEC     R0              ;;ADJUST THE INDEX SO THAT IT WILL      ;R-C
061370  006300                          ASL     R0              ;;      WORK FOR THE ERROR TABLE
061372  006300                          ASL     R0
061374  006300                          ASL     R0
061376  062700  066262                  ADD     #$ERRTB,R0      ;;FORM TABLE POINTER
061402  012037  061440          110$:   MOV     (R0)+,3$        ;;PICKUP "ERROR MESSAGE" POINTER        ;R-C
061406  001417                          BEQ     4$              ;;SKIP TYPEOUT IF NO POINTER
061410  005737  002424                  TST     NOERROR         ;IS THIS REALLY AN ERROR?
061414  001003                          BNE     12$             ;YES - SKIP
061416  005737  002576                  TST     HEADER          ;TYPE HEADER?
061422  100011                          BPL     4$              ;NO - SKIP
061424  005737  002062          12$:    TST     FATAL$          ;WAS IT A FATAL ERROR?
061430  001402                          BEQ     2$              ;NO - SKIP
061432                                  TYPE    MSG067          ;FATAL
061436                          2$:     TYPE                    ;;TYPE THE "ERROR MESSAGE"
061440  000000                  3$:     .WORD   0               ;;"ERROR MESSAGE" POINTER GOES HERE
061442                                  TYPE    $CRLF           ;;"CARRIAGE RETURN" & "LINE FEED"
061446  012037  061472          4$:     MOV     (R0)+,5$        ;;PICKUP "DATA HEADER" POINTER
061452  001412                          BEQ     6$              ;;SKIP TYPEOUT IF 0
061454  005737  002424                  TST     NOERROR         ;IS THIS REALLY AN ERROR?
061460  001003                          BNE     13$             ;YES - SKIP
061462  005737  002576                  TST     HEADER          ;TYPE HEADER?
061466  100004                          BPL     6$              ;NO - SKIP
061470                          13$:    TYPE                    ;;TYPE THE "DATA HEADER"
061472  000000                  5$:     .WORD   0               ;;"DATA HEADER" POINTER GOES HERE
061474                                  TYPE    $CRLF           ;;"CARRIAGE RETURN" & "LINE FEED"
061500  012001                  6$:     MOV     (R0)+,R1        ;;PICKUP "DATA TABLE" POINTER
061502  001427                          BEQ     10$             ;;BR IF NO DATA TO BE TYPED
061504  012002                          MOV     (R0)+,R2        ;;PICKUP "DATA FORMAT" POINTER
```

```
14973 061506  112203                  7$:     MOVB    (R2)+,R3
14974 061510  006303                          ASL     R3                      ;MAKE IT A WORD ADDRESS
14975 061512  004773  061520                    CALL    @8$(R3)
14976 061516  000412                          BR      9$
14977 061520  061744                  8$:     TAG70$
14978 061522  061754                          TAG71$
14979 061524  061764                          TAG72$
14980 061526  062034                          TAG73$
14981 061530  062074                          TAG74$
14982 061532  062106                          TAG75$
14983 061534  062120                          TAG76$
14984 061536  062164                          TAG77$
14985 061540  062172                          TAG78$
14986 061542  062252                          TAG79$
14991 061544  062701  000002            9$:     ADD     #2,R1                   ;UPDATE DATA TABLE POINTER
14992 061550  005711                          TST     (R1)                    ;;IS THERE ANOTHER NUMBER?
14993 061552  001403                          BEQ     10$                     ;;BR IF NO
14994 061554                                  TYPE    MSG018                  ;TYPE 2 SPACES
14995 061560  000752                          BR      7$                      ;;LOOP
14996
14997 061562  005737  002106            10$:    TST     MUT                     ;IS THERE A MEMORY UNDER TEST
14998 061566  001402                          BEQ     11$                     ;NO - SKIP
14999 061570  005237  002576                    INC     HEADER                  ;YES - BUMP HEADER FLAG
15000 061574  104416                  11$:    RESREG
15001 061576                                  IF #SW7 SET.IN @SWR AND DETFLAG IS FALSE AND NOERROR IS FALSE
15002 061622  004737  062274                      CALL  DETAIL
15003 061626                                  END ;OF IF #SW7
15004 061626                                  TYPE    MSG104                  ;CONTROL Z
15005 061632  000207                          RETURN
15006                                         .EVEN                                           ;R-C
15007 061634  061644  061700  061730  PFECWS: .WORD   PFECEM,PFECDH,PFECDT,PFECDF           ;R-C
      061642  061740
15008 061644    120     117     127  PFECEM: .ASCIZ  'POWER MONITOR BIT FOUND SET'          ;R-C
      061647    105     122     040
      061652    115     117     116
      061655    111     124     117
      061660    122     040     102
      061663    111     124     040
      061666    106     117     125
      061671    116     104     040
      061674    123     105     124
      061677    000
15009 061700    124     105     123  PFECDH: .ASCIZ  'TESTNO  ERR PC  CPUERR'           ;R-C
      061703    124     116     117
      061706    040     040     105
      061711    122     122     040
      061714    120     103     040
      061717    040     103     120
      061722    125     105     122
      061725    122     000
15010                                         .EVEN                                           ;R-C
15011 061730  065644  002016  061320  PFECDT: .WORD   $TESTN,ERRPC,CPSAVE,0            ;R-C
      061736  000000
15012 061740    000     000     000  PFECDF: .BYTE   0,0,0,0                         ;R-C
      061743    000
15013
```

```
15016                                   ;*****************************************************************************
15017                                   ;*** OCTAL ***
15018                                   ;*****************************************************************************
15019 061744                            TAG70$: TYPOCT  @(R1)           ;;TYPE AN OCTAL NUMBER
15020 061752  000207                            RETURN
15021
15022                                   ;*****************************************************************************
15023                                   ;*** DECIMAL ***
15024                                   ;*****************************************************************************
15025 061754                            TAG71$: TYPDEC  @(R1)           ;;TYPE A DECIMAL NUMBER
15026 061762  000207                            RETURN
15027
15028                                   ;*****************************************************************************
15029                                   ;*** INTERLEAVE ***
15030                                   ;*****************************************************************************
15031 061764                            TAG72$: PUSH    R1,R5
15032 061770  013701  002100                    MOV     BANK,R1
15033 061774  070127  000004                    MUL     #4,R1
15034 062000                                    SET     NOTAB                   ;INDICATE NO TABLE TO BE PRINTED - NOW
15035 062006                                    TYPE    MSG014
15036 062012  004737  041726                    CALL    TCFIG1
15037 062016  005037  002366                    CLR     NOTAB
15038 062022                                    POP     R5,R1
15039 062026                                    TYPE    MSG014                  ;1 SPACE
15040 062032  000207                            RETURN
15041
15042                                   ;*****************************************************************************
15043                                   ;*** CSR ***
15044                                   ;*****************************************************************************
15045 062034                            TAG73$: PUSH    R1,R5
15046 062040  013701  002100                    MOV     BANK,R1
15047 062044  070127  000004                    MUL     #4,R1
15048 062050                                    SET     NOTAB
15049 062056  004737  042222                    CALL    TCFIG3
15050 062062  005037  002366                    CLR     NOTAB
15051 062066                                    POP     R5,R1
15052 062072  000207                            RETURN
15053
15054                                   ;*****************************************************************************
15055                                   ;*** PATTERN ***
15056                                   ;*****************************************************************************
15057 062074                            TAG74$: TYPOCS  REALPAT,<TYPE (0-77)>,2,2
15058 062104  000207                            RETURN
15059
15060                                   ;*****************************************************************************
15061                                   ;*** BANK ***
15062                                   ;*****************************************************************************
15063 062106                            TAG75$: TYPOCS  BANK,<TYPE (0-167)>,3
15064 062116  000207                            RETURN
```

```
15066                                   ;*****************************************************************************
15067                                   ;*** MTYPE ***
15068                                   ;*****************************************************************************
15069 062120                            TAG76$: PUSH    R1,R5
15070 062124  013701  002100                    MOV     BANK,R1
15071 062130  070127  000004                    MUL     #4,R1
15072 062134                                    SET     NOTAB
15073 062142                                    TYPE    MSG019
15074 062146  004737  042066                    CALL    TCFIG2
15075 062152  005037  002366                    CLR     NOTAB
15076 062156                                    POP     R5,R1
15077 062162  000207                            RETURN
15078
15079                                   ;*****************************************************************************
15080                                   ;*** UNKNOWN DATA ***
15081                                   ;*****************************************************************************
15082 062164                            TAG77$: TYPE    MSG061
15083 062170  000207                            RETURN
15084
15085                                   ;*****************************************************************************
15086                                   ;*** PHYSICAL ADDRESS ***
15087                                   ;*****************************************************************************
15088 062172  013737  002032  002036    TAG78$: MOV     ADDRESS,PHYADD
15089 062200  162737  060000  002036            SUB     #FIRST,PHYADD
15090 062206  013737  002100  002040            MOV     BANK,PHYADD+2
15091 062214  006237  002040                    ASR     PHYADD+2
15092 062220  103003                            BCC     1$
15093 062222  052737  100000  002036            BIS     #BIT15,PHYADD
15094 062230  012746  002036          1$:       MOV     #PHYADD,-(SP)           ;POINTER TO DOUBLE WORD ON STACK
15095 062234  004737  065520                    CALL    $DB2O                   ;CALL DOUBLE PRECISION CONVERSION ROUTINE
15096 062240  062706  000002                    ADD     #2,SP                   ;FIX STACK
15097 062244                                    TYPE    $OCT8
15098 062250  000207                            RETURN
15099
15100                                   ;*****************************************************************************
15101                                   ;*** OCTAL BYTE ***
15102                                   ;*****************************************************************************
15103 062252                            TAG79$: TYPE    MSG018          ;2 SPACES
15104 062256                                    TYPOCS  @(R1),<TYPE BYTE>,3,Z
15105 062266                                    TYPE    MSG014          ;SPACE
15106 062272  000207                            RETURN
```

```
 15150 062274                                DETAIL: SUBTST  <<SUBR  DETAILED ERROR REPORT>>
                                             ;*****************************************************************************
                                             ;*SUBTEST        SUBR    DETAILED ERROR REPORT
                                             ;*****************************************************************************
 15151 062274  005237  002216                        INC     DETFLAG
 15152 062300  022737  000003  002216                CMP     #3,DETFLAG
 15153 062306  101473                                BLOS    4$
 15154 062310  022737  000002  002216                CMP     #2,DETFLAG
 15155 062316  001435                                BEQ     2$
 15156 062320                                        PUSH    HEADER,MUT
 15157 062330                                        SET     HEADER
 15158 062336  005037  002106                        CLR     MUT
 15159 062342  010037  002176                        MOV     R0,DETR0
 15160 062346  012700  002200                        MOV     #DETR1,R0
 15161 062352  010120                                MOV     R1,(R0)+
 15162 062354  010220                                MOV     R2,(R0)+
 15163 062356  010320                                MOV     R3,(R0)+
 15164 062360  010420                                MOV     R4,(R0)+
 15165 062362  010520                                MOV     R5,(R0)+
 15166 062364  013720  002022                        MOV     ERRSP,(R0)+
 15167 062370  013720  002026                        MOV     ERRPSW,(R0)+
 15168 062374  013700  002176                        MOV     DETR0,R0
 15169 062400                                        SET     NOERROR
 15170 062406  104013                                ERROR   +13
 15171 062410  000423                                BR      1$
 15172 062412                              2$:       PUSH    HEADER,MUT
 15173 062422                                        SET     HEADER
 15174 062430  005037  002106                        CLR     MUT
 15175 062434                                        SET     NOERROR
 15176 062442  104031                                ERROR   +31
 15177 062444  022737  000001  003752                CMP     #1,PROTYP               ;IS THIS AN 11/44?
 15178 062452  001002                                BNE     1$
 15179 062454  005037  177766                        CLR     CPUERR
 15180 062460                              1$:       POP     MUT,HEADER
 15181                                               ;WARNING RECURSIVE
 15182 062470  004737  062274                        CALL    DETAIL
 15183 062474  000207                                RETURN
```

```
15186
15187 062476  004737  064144          4$:     CALL    CONTT                   ;DISPLAY 'DISPLAY'' INFO
15188
15189                                         ;TYPE CONTENTS OF ALL CSR'S
15190 062502                                  PUSH    CSR,CSRNO,R1
15191 062514                                  TYPE    MSG058
15192 062520                                  TYPE    $CRLF
15193 062524  013701  002222                  MOV     TOTCSRS,R1
15194 062530                                  BEGIN DUMPCSRLOOP
15195 062530                                    FOR CSRNO := #0 TO #36 BY #2
15196 062534  006301                              ASL     R1
15197 062536                                      ON.ERROR
15198 062540  104426                                READCSR
15199 062542                                        TYPOCT  CSR
15200 062550                                        TYPE    MSG018          ;2 SPACES
15201 062554                                      END ;OF ON.ERROR
15202 062554                                      IF R1 EQ #0 THEN LEAVE DUMPCSRLOOP
15203 062560                                    END ;OF FOR CSRNO
15204 062576                                  END DUMPCSRLOOP
15205 062576                                  POP     R1,CSRNO,CSR
15206
15207                                         ;TYPE STACKS
15208 062610                                  PUSH    R0,R1
15209 062614                                  TYPE    MSG088          ;KERNEL STACK
15210 062620  013701  002560                  MOV     KSTACK,R1
15211 062624  162701  000002                  SUB     #2,R1
15212 062630                                  FOR     R0 := SP TO R1 BY #2
15213 062632                                    TYPE  $CRLF
15214 062636                                    TYPOCT        R0
15215 062642                                    TYPE          MSG018          ;2 SPACES
15216 062646                                    TYPOCT        (R0)
15217 062652                                  END ;OF FOR R0
15218                                         ;SET PREVIOUS MODE TO SUPERVISOR
15219 062662  005737  002452                  TST     NOSUPER
15220 062666  001036                          BNE     DET1
15221 062670  042737  030000  177776          BIC     #BIT13!BIT12,PSW
15222 062676  052737  010000  177776          BIS     #BIT12,PSW
15223 062704  006506                          MFPI    SSP
15224 062706                                  POP     R1,R0
15225 062712                                  TYPE    MSG089          ;SUPERVISOR STACK
15226 062716                                  IF R0 LT #SUPSTK
15227 062724                                    FOR R0 := R0 TO #SUPSTK-2 BY #2
15228 062724                                      TYPE        $CRLF
15229 062730                                      TYPOCT      R0
15230 062734                                      TYPE        MSG018          ;2 SPACES
15231 062740                                      TYPOCT      (R0)
15232 062744                                    END ;OF FOR R0
15233 062756                                  ELSE
15234 062760                                    TYPE  MSG091          ;IS EMPTY
15235 062764                                  END ;OF IF R0
15236                                         ;SET PREVIOUS MODE TO USER
15237 062764  052737  030000  177776  DET1:   BIS     #BIT13!BIT12,PSW
15238 062772  006506                          MFPI    USP
15239 062774                                  POP     R0
15240 062776                                  TYPE    MSG090          ;USER STACK
15241 063002                                  IF R0 LT #USESTK
15242 063010                                    FOR R0 := R0 TO #USESTK-2 BY #2
```

```
15243 063010                                            TYPE        $CRLF
15244 063014                                            TYPOCT      R0
15245 063020                                            TYPE        MSG018          ;2 SPACES
15246 063024                                            TYPOCT      (R0)
15247 063030                                          END ;OF FOR R0
15248 063042                                        ELSE
15249 063044                                          TYPE  MSG091          ;IS EMPTY
15250 063050                                        END ;OF IF R0
15251 063050                                        TYPE    $CRLF
15252 063054  005037  002216                        CLR     DETFLAG
15253 063060                                        POP     R0
15254 063062  000207                                RETURN
```

```
                                        .SBTTL  ROUTINE BINARY TO OCTAL (ASCII) AND TYPE

                                ;*******************************************************************************
                                ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
                                ;*OCTAL (ASCII) NUMBER AND TYPE IT.
                                ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
                                ;*CALL:
                                ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                                ;*      TYPOS                   ;;CALL FOR TYPEOUT
                                ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
                                ;*      .BYTE   M               ;;M=1 OR 0
                                ;*                                      ;;1=TYPE LEADING ZEROS
                                ;*                                      ;;0=SUPPRESS LEADING ZEROS
                                ;*
                                ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
                                ;*$TYPOS OR $TYPOC
                                ;*CALL:
                                ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                                ;*      TYPON                   ;;CALL FOR TYPEOUT
                                ;*
                                ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
                                ;*CALL:
                                ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                                ;*      TYPOC                   ;;CALL FOR TYPEOUT

063064  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
063070  116637  000001  063307          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
063076  112637  063311                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
063102  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
063106  000406                          BR      $TYPON
063110  112737  000001  063307  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
063116  112737  000006  063311          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
063124  112737  000005  063306  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
063132  010346                          MOV     R3,-(SP)        ;;SAVE R3
063134  010446                          MOV     R4,-(SP)        ;;SAVE R4
063136  010546                          MOV     R5,-(SP)        ;;SAVE R5
063140  113704  063311                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
063144  005404                          NEG     R4
063146  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
063152  110437  063310                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
063156  113704  063307                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
063162  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
063166  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
063170  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
063172  000404                          BR      3$              ;;GO DO MSB
063174  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
063176  006105                          ROL     R5
063200  006105                          ROL     R5
063202  010503                          MOV     R5,R3
063204  006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
063206  105337  063310                  DECB    $OMODE          ;;TYPE THIS DIGIT?
063212  100016                          BPL     6$              ;;BR IF NO
063214  042703  177770                  BIC     #177770,R3      ;;GET RID OF JUNK
063220  001002                          BNE     4$              ;;TEST FOR 0
063222  005704                          TST     R4              ;;SUPPRESS THIS 0?
063224  001403                          BEQ     5$              ;;BR IF YES
063226  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
```

```
CZMSPA0 MS11-L/M/P MEMORY DIAG. MACRO M1113  26-APR-82 09:41  PAGE 481-1
ROUTINE BINARY TO OCTAL (ASCII) AND TYPE

 15349 063230  052703  000060              BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
 15350 063234  052703  000040      5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
 15351 063240  110337  063304              MOVB    R3,8$           ;;SAVE FOR TYPING
 15352 063244                              TYPE    8$              ;;GO TYPE THIS DIGIT
 15353 063250  105337  063306      6$:     DECB    $OCNT           ;;COUNT BY 1
 15354 063254  003347                      BGT     2$              ;;BR IF MORE TO DO
 15355 063256  002402                      BLT     7$              ;;BR IF DONE
 15356 063260  005204                      INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
 15357 063262  000744                      BR      2$              ;;GO DO THE LAST DIGIT
 15358 063264  012605              7$:     MOV     (SP)+,R5        ;;RESTORE R5
 15359 063266  012604                      MOV     (SP)+,R4        ;;RESTORE R4
 15360 063270  012603                      MOV     (SP)+,R3        ;;RESTORE R3
 15361 063272  016666  000002  000004      MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
 15362 063300  012616                      MOV     (SP)+,(SP)
 15363 063302  000002                      RTI                     ;;RETURN
 15364 063304     000      8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
 15365 063305     000              .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
 15366 063306     000      $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
 15367 063307     000      $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
 15368 063310  000000      $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
```

```
                                        .SBTTL  ROUTINE CONVERT BINARY TO DECIMAL AND TYPE
                                ;**************************************************************************
                                ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
                                ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
                                ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
                                ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
                                ;*REPLACED WITH SPACES.
                                ;*CALL:
                                ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
                                ;*      TYPDS                   ;;GO TO THE ROUTINE
063312                          $TYPDS: PUSH    R0,R1,R2,R3,R5
063324  012746  020200                  MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
063330  016605  000020                  MOV     20(SP),R5       ;;GET THE INPUT NUMBER
063334  100004                          BPL     1$              ;;BR IF INPUT IS POS.
063336  005405                          NEG     R5              ;;MAKE THE BINARY NUMBER POS.
063340  112766  000055  000001          MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
063346  005000                  1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
063350  012703  063526                  MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
063354  112723  000040                  MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
063360  005002                  2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
063362  016001  063516                  MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
063366  160105                  3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
063370  002402                          BLT     4$              ;;BR IF DONE
063372  005202                          INC     R2              ;;INCREASE THE BCD DIGIT BY 1
063374  000774                          BR      3$
063376  060105                  4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
063400  005702                          TST     R2              ;;CHECK IF BCD DIGIT=0
063402  001002                          BNE     5$              ;;FALL THROUGH IF 0
063404  105716                          TSTB    (SP)            ;;STILL DOING LEADING 0'S?
063406  100407                          BMI     7$              ;;BR IF YES
063410  106316                  5$:     ASLB    (SP)            ;;MSD?
063412  103003                          BCC     6$              ;;BR IF NO
063414  116663  000001  177777          MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
063422  052702  000060          6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
063426  052702  000040          7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
063432  110223                          MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
063434  005720                          TST     (R0)+           ;;JUST INCREMENTING
063436  020027  000010                  CMP     R0,#10          ;;CHECK THE TABLE INDEX
063442  002746                          BLT     2$              ;;GO DO THE NEXT DIGIT
063444  003002                          BGT     8$              ;;GO TO EXIT
063446  010502                          MOV     R5,R2           ;;GET THE LSD
063450  000764                          BR      6$              ;;GO CHANGE TO ASCII
063452  105726                  8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
063454  100003                          BPL     9$              ;;BR IF NO
063456  116663  177777  177776          MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
063464  105013                  9$:     CLRB    (R3)            ;;SET THE TERMINATOR
063466                                  POP     R5,R3,R2,R1,R0
063500                                  TYPE    $DBLK           ;;NOW TYPE THE NUMBER
063504  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
063512  012616                          MOV     (SP)+,(SP)
063514  000002                          RTI                     ;;RETURN TO USER
063516  023420                  $DTBL:  10000.
063520  001750                          1000.
063522  000144                          100.
063524  000012                          10.
063526  000000  000000  000000  $DBLK:  .WORD   0,0,0,0
063534  000000
```

```
15427                                           .SBTTL  ROUTINE TTY INPUT
15428                                   ;*******************************************************************************
15429                                   ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
15430                                   ;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
15431                                   ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
15432                                   ;*WHEN OPERATING IN TTY FLAG MODE.
15433                                           .ENABLE LSB
15434 063536                           $CKSWR:
15440 063536  005737  056276                     TST     XOCHAR          ;;SOMETHING THERE?
15441 063542  001406                             BEQ     NOCH            ;; GO ON IF NOT
15442 063544  013746  056276                     MOV     XOCHAR,-(SP)    ;; USE IT
15443 063550  005037  056276                     CLR     XOCHAR
15444 063554  000137  063576                     JMP     CONTS1
15445 063560  105777  117042           NOCH:     TSTB    @$TKS           ;;CHAR THERE?
15446 063564  100130                             BPL     12$             ;;IF NO, DON'T WAIT AROUND
15447 063566  117746  117036                     MOVB    @$TKB,-(SP)     ;;SAVE THE CHAR
15448 063572  042716  177600                     BIC     #^C177,(SP)     ;;STRIP-OFF THE ASCII
15449 063576  022716  000006           CONTS1:   CMP     #6,(SP)         ;IS IT CONTROL F?
15450 063602  001002                             BNE     1$              ;NO SKIP
15451 063604  004737  050102                     CALL    FIELDSERVICE
15452 063610  022716  000024           1$:       CMP     #24,(SP)        ;IS IT CONTROL T?
15453 063614  001002                             BNE     16$             ;NO - SKIP
15454 063616  004737  064144                     CALL    CONTT           ;YES - CALL CONTROL T ROUTINE
15455 063622  022716  000003           16$:      CMP     #3,(SP)         ;IS IT CONTROL C?
15456 063626  001454                             BEQ     5$              ;YES EXIT *****NOTE***** STACK IS SCREWED UP!
15457 063630  022716  000023           2$:       CMP     #23,(SP)        ;IS IT CONTROL S?
15458 063634  001002                             BNE     17$             ;NO - SKIP
15459 063636  004737  064220                     CALL    CONTS           ;YES - CALL CONTROL S ROUTINE
15460 063642  022716  000013           17$:      CMP     #13,(SP)        ;IS IT CONTROL K?
15461 063646  001005                             BNE     6$              ;NO - SKIP
15462 063650                                     TYPE    $CNTLK          ;TYPE A ^K
15463 063654  013706  002144                     MOV     CTLKVEC,SP      ;RESET KSP TO AFTER PATTERN EXEC ROUTINE
15464 063660  000207                             RETURN                  ;RETURN TO PATTERN EXEC ROUTINE
15465 063662  022737  000176  002622  6$:       CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED?
15466 063670  001067                             BNE     CKEND           ;;BRANCH IF NO
15467 063672  022716  000007                     CMP     #7,(SP)         ;;IS IT A CONTROL G?
15468 063676  001064                             BNE     CKEND           ;;NO, RETURN TO USER
15469 063700  005737  002060                     TST     $AUTO           ;ARE WE RUNNING IN AUTO-MODE?
15470 063704  001061                             BNE     CKEND           ;BRANCH IF YES
15471 063706                                     TYPE    $CNTLG          ;;ECHO THE CONTROL-G (^G)
15472 063712                           $GTSWR:   TYPE    $MSWR           ;;TYPE CURRENT CONTENTS
15473 063716                                     TYPOCT  @SWR            ;;OF THE SWR
15474 063724                                     TYPE    $MNEW           ;;PROMPT FOR NEW SWR
15475 063730  005046                     3$:     CLR     -(SP)           ;;CLEAR COUNTER
15476 063732  005046                             CLR     -(SP)           ;;THE NEW SWR
15477 063734  105777  116666           4$:       TSTB    @$TKS           ;;CHAR THERE?
15478 063740  100375                             BPL     4$              ;;IF NOT TRY AGAIN
15479 063742  117746  116662                     MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
15480 063746  042716  177600                     BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
15481 063752  021627  000003                     CMP     (SP),#3         ;;IS IT A CONTROL-C?
15482 063756  001006                             BNE     7$              ;;BRANCH IF NOT
15483 063760                           5$:       TYPE    $CNTLC          ;;YES, ECHO CONTROL-C (^C)
15484 063764  062706  000006                     ADD     #6,SP           ;;CLEAN UP STACK
15485 063770  000137  047524                     JMP     BOOT            ;;CONTROL-C RESTART
15486 063774  021627  000025           7$:       CMP     (SP),#25        ;;IS IT A CONTROL-U?
15487 064000  001005                             BNE     9$              ;;BRANCH IF NOT
15488 064002                                     TYPE    $CNTLU          ;;YES, ECHO CONTROL-U (^U)
```

```
15489 064006  062706  000006          8$:     ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
15490 064012  000746                  BR      3$              ;;LET'S TRY IT AGAIN
15491 064014  021627  000015          9$:     CMP     (SP),#15        ;;IS IT A <CR>?
15492 064020  001016                  BNE     13$             ;;BRANCH IF NO
15493 064022  005766  000004                  TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
15494 064026  001403                  BEQ     10$             ;;BRANCH IF YES
15495 064030  016677  000002  116564          MOV     2(SP),@SWR      ;;SAVE NEW SWR
15496 064036  062706  000006          10$:    ADD     #6,SP           ;;CLEAR UP STACK
15497 064042                          TYPE    $CRLF           ;;ECHO <CR> AND <LF>
15498 064046  000002                  12$:    RTI                     ;;RETURN
15499 064050  062706  000002          CKEND:  ADD     #2,SP           ;FIX STACK
15500 064054  000002                  RTI                     ;RETURN
15501 064056  004737  056300          13$:    CALL    $TYPEC          ;;ECHO CHAR
15502 064062  021627  000060                  CMP     (SP),#60        ;;CHAR < 0?
15503 064066  002420                  BLT     15$             ;;BRANCH IF YES
15504 064070  021627  000067                  CMP     (SP),#67        ;;CHAR > 7?
15505 064074  003015                  BGT     15$             ;;BRANCH IF YES
15506 064076  042726  000060                  BIC     #60,(SP)+       ;;STRIP-OFF ASCII
15507 064102  005766  000002                  TST     2(SP)           ;;IS THIS THE FIRST CHAR
15508 064106  001403                  BEQ     14$             ;;BRANCH IF YES
15509 064110  006316                  ASL     (SP)            ;;NO, SHIFT PRESENT
15510 064112  006316                  ASL     (SP)            ;;    CHAR OVER TO MAKE
15511 064114  006316                  ASL     (SP)            ;;    ROOM FOR NEW ONE.
15512 064116  005266  000002          14$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
15513 064122  056616  177776                  BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
15514 064126  000702                  BR      4$              ;;GET THE NEXT ONE
15515 064130                          15$:    TYPE    $QUES           ;;TYPE ?<CR><LF>
15516 064134  000724                  BR      8$              ;;SIMULATE CONTROL-U
15517 064136     136     113     015  $CNTLK: .ASCIZ   /^K/<15><12>    ;CONTROL K ASCII STRING
      064141     012     000
15518                                 .EVEN
15519                                 .DSABL  LSB
```

```
 15522 064144                           CONTT:  SUBTST  <<CONTROL T>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       CONTROL T
                                        ;*****************************************************************************
 15523 064144                                   PUSH    R0
 15524 064146                                   TYPE    $CRLF
 15534 064152                                     IF RLFLAG IS TRUE
 15535 064160                                       TYPE        MSG092          ;RELOCATED
 15536 064164                                     END ;OF IF RLFLAG
 15537 064164                                     TYPE  MSG093                   ;BANK=
 15538 064170                                     TYPOCS        FANK,,3          ;TYPE 3 DIGITS
 15539 064200                                     TYPE  MSG095                   ;PAT=
 15540 064204                                     TYPOCS        REAL, ' 2        ;TYPE 2 DIGITS
 15544 064214                                   POP     R0
 15545 064216  000207                           RETURN
 15546
 15547 064220                           CONTS:  SUBTST  <<CONTROL S & CONTROL Q>>
                                        ;*****************************************************************************
                                        ;*SUBTEST       CONTROL S & CONTROL Q
                                        ;*****************************************************************************
 15548 064220                                   POP     R0              ;GET RID OF RETURN ADDRESS FROM STACK
 15549 064222  105777  116400           CONTS2: TSTB    @$TKS           ;WAIT FOR CHARACTER
 15550 064226  100375                           BPL     CONTS2
 15551 064230  117716  116374                   MOVB    @$TKB,(SP)      ;REPLACE OVER OLD CHARACTER ON STACK
 15552 064234  042716  177600                   BIC     #^C177,(SP)     ;STRIP ALL BUT ASCII
 15553 064240                                   IF (SP) EQ #21          ;IF IT IS A CONTROL Q
 15554 064246  000137  063576                     JMP   CONTS1
 15555 064252                                   ELSE
 15556 064254  000762                             BR    CONTS2
 15557 064256                                   END ;OF IF (SP)
```

```
15559                                   ;*************************************************************************************
15560                                   ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
15561                                   ;*CALL:
15562                                   ;*      RDCHR                       ;;INPUT A SINGLE CHARACTER FROM THE TTY
15563                                   ;*      RETURN HERE                 ;;CHARACTER IS ON THE STACK
15564                                   ;*                                  ;;WITH PARITY BIT STRIPPED OFF
15565                                   ;
15566
15567 064256  011646                   $RDCHR: MOV     (SP),-(SP)          ;;PUSH DOWN THE PC
15568 064260  016666  000004  000002          MOV     4(SP),2(SP)         ;;SAVE THE PS
15569 064266  105777  116334          1$:     TSTB    @$TKS               ;;WAIT FOR
15570 064272  100375                          BPL     1$                  ;;A CHARACTER
15571 064274  117766  116330  000004          MOVB    @$TKB,4(SP)         ;;READ THE TTY
15572 064302  042766  177600  000004          BIC     #^C<177>,4(SP)      ;;GET RID OF JUNK IF ANY
15573 064310  026627  000004  000023          CMP     4(SP),#23           ;;IS IT A CONTROL-S?
15574 064316  001013                          BNE     3$                  ;;BRANCH IF NO
15575 064320  105777  116302          2$:     TSTB    @$TKS               ;;WAIT FOR A CHARACTER
15576 064324  100375                          BPL     2$                  ;;LOOP UNTIL ITS THERE
15577 064326  117746  116276                  MOVB    @$TKB,-(SP)         ;;GET CHARACTER
15578 064332  042716  177600                  BIC     #^C177,(SP)         ;;MAKE IT 7-BIT ASCII
15579 064336  022627  000021                  CMP     (SP)+,#21           ;;IS IT A CONTROL-Q?
15580 064342  001366                          BNE     2$                  ;;IF NOT DISCARD IT
15581 064344  000750                          BR      1$                  ;;YES, RESUME
15582 064346  026627  000004  000021  3$:     CMP     4(SP),#21           ;;IS IT A RANDOM CONTROL-Q?     ;R-C
15583 064354  001744                          BEQ     1$                  ;;BRANCH BACK IF SO             ;R-C
15584 064356  026627  000004  000140          CMP     4(SP),#140          ;;IS IT UPPER CASE?
15585 064364  002407                          BLT     4$                  ;;BRANCH IF YES
15586 064366  026627  000004  000175          CMP     4(SP),#175          ;;IS IT A SPECIAL CHAR?
15587 064374  003003                          BGT     4$                  ;;BRANCH IF YES
15588 064376  042766  000040  000004          BIC     #40,4(SP)           ;;MAKE IT UPPER CASE
15589 064404  000002                  4$:     RTI                         ;;GO BACK TO USER
15590                                   ;*************************************************************************************
15591                                   ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
15592                                   ;*CALL:
15593                                   ;*      RDLIN                       ;;INPUT A STRING FROM THE TTY
15594                                   ;*      RETURN HERE                 ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
15595                                   ;*                                  ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
15596 064406  010346                   $RDLIN: MOV     R3,-(SP)            ;;SAVE R3
15597 064410  005046                          CLR     -(SP)               ;;CLEAR THE RUBOUT KEY
15598 064412  012703  064704          1$:     MOV     #$TTYIN,R3          ;;GET ADDRESS
15599 064416  022703  064730          2$:     CMP     #$TTYIN+20.,R3      ;;BUFFER FULL?
15600 064422  101477                          BLOS    8$                  ;;BR IF YES
15601 064424  104411                          RDCHR                       ;;GO READ ONE CHARACTER FROM THE TTY
15602 064426  112613                          MOVB    (SP)+,(R3)          ;;GET CHARACTER
15603 064430  122713  000003                  CMPB    #3,(R3)             ;;IS IT A CONTROL-C?
15604 064434  001016                          BNE     3$                  ;;BRANCH IF NO
15605 064436                                  TYPE    $CNTLC              ;;TYPE A CONTROL-C (^C)
15606 064442  005726                          TST     (SP)+               ;;CLEAN RUBOUT KEY OFF OF THE STACK
15607 064444  012603                          MOV     (SP)+,R3            ;;RESTORE R3
15608 064446  032777  000400  116146          BIT     #BIT8,@SWR          ;;IS THERE A HALT FLAG SET IN THE SWR?
15609 064454  001404                          BEQ     11$                 ;;BRANCH IF NOT TO BOOT ROUTINE
15610 064456  005037  002414                  CLR     STOPOK              ;;GET READY TO HALT PROGRAM
15611 064462  000137  047632                  JMP     EXIT                ;;GO HALT PROGRAM
15612 064466  000137  047524          11$:    JMP     BOOT                ;;GOTO CONTROL-C RESTART
15613 064472  122713  000177          3$:     CMPB    #177,(R3)           ;;IS IT A RUBOUT
15614 064476  001022                          BNE     5$                  ;;BR IF NO
15615 064500  005716                          TST     (SP)                ;;IS THIS THE FIRST RUBOUT?
```

```
15616 064502 001007                      BNE     4$              ;;BR IF NO
15617 064504 112737 000134 064702        MOVB    #'\,10$         ;;TYPE A BACK SLASH
15618 064512                             TYPE    10$
15619 064516 012716 177777               MOV     #-1,(SP)        ;;SET THE RUBOUT KEY
15620 064522 005303              4$:     DEC     R3              ;;BACKUP BY ONE
15621 064524 020327 064704               CMP     R3,#$TTYIN      ;;STACK EMPTY?
15622 064530 103434                      BLO     8$              ;;BR IF YES
15623 064532 112337 064702               MOVB    (R3),10$                ;;SETUP TO TYPEOUT THE DELETED CHAR.
15624 064536                             TYPE    10$             ;;GO TYPE
15625 064542 000755                      BR      2$              ;;GO READ ANOTHER CHAR.
15626 064544 005716              5$:     TST     (SP)            ;;RUBOUT KEY SET?
15627 064546 001406                      BEQ     6$              ;;BR IF NO
15628 064550 112737 000134 064702        MOVB    #'\,10$         ;;TYPE A BACK SLASH
15629 064556                             TYPE    10$
15630 064562 005016                      CLR     (SP)            ;;CLEAR THE RUBOUT KEY
15631 064564 122713 000025       6$:     CMPB    #25,(R3)        ;;IS CHARACTER A CTRL U?
15632 064570 001003                      BNE     7$              ;;BR IF NO
15633 064572                             TYPE    $CNTLU          ;;TYPE A CONTROL 'U'
15634 064576 000705                      BR      1$              ;;GO START OVER
15635 064600 122713 000022       7$:     CMPB    #22 (R3)        ;;IS CHARACTER A '^R'?
15636 064604 001011                      BNE     9$              ;;BRANCH IF NO
15637 064606 105013                      CLRB    (R3)            ;;CLEAR THE CHARACTER
15638 064610                             TYPE    $CRLF           ;;TYPE A "CR" & "LF"
15639 064614                             TYPE    $TTYIN          ;;TYPE THE INPUT STRING
15640 064620 000676                      BR      2$              ;;GO PICKUP ANOTHER CHACTER
15641 064622                     8$:     TYPE    $QUES           ;;TYPE A '?'
15642 064626 000721                      BR      1$              ;;CLEAR THE BUFFER AND LOOP
15643 064630 112337 064702       9$:     MOVB    (R3),10$                ;;ECHO THE CHARACTER
15644 064634                             TYPE    10$
15645 064640 122723 000015               CMPB    #15,(R3)+       ;;CHECK FOR RETURN
15646 064644 001264                      BNE     2$              ;;LOOP IF NOT RETURN
15647 064646 105063 177777               CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
15648 064652                             TYPE    $LF             ;;TYPE A LINE FEED
15649 064656 005726                      TST     (SP)+           ;;CLEAN RUBOUT KEY FROM THE STACK
15650 064660 012603                      MOV     (SP)+,R3        ;;RESTORE R3
15651 064662 011646                      MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
15652 064664 016666 000004 000002        MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
15653 064672 012766 064704 000004        MOV     #$TTYIN,4(SP)
15654 064700 000002                      RTI                     ;;RETURN
15655 064702    000              10$:    .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
15656 064703    000                      .BYTE   0               ;;TERMINATOR
15657 064704 000024              $TTYIN: .REPT   20.             ;;RESERVE SIZE BYTES FOR TTY INPUT
15660 064730    136    103    015 $CNTLC: .ASCIZ  /^C/<15><12>    ;;CONTROL 'C'
      064733    012    000
15661 064735    136    125    015 $CNTLU: .ASCIZ  /^U/<15><12>    ;;CONTROL 'U'
      064740    012    000
15662 064742    136    107    015 $CNTLG: .ASCIZ  /^G/<15><12>    ;;CONTROL 'G'
      064745    012    000
15663 064747    015    012    123 $MSWR:  .ASCIZ  <15><12>/SWR = /
      064752    127    122    040
      064755    075    040    000
15664 064760    040    040    116 $MNEW:  .ASCIZ  /  NEW = /
      064763    105    127    040
      064766    075    040    000
15665                            .EVEN
```

```
15667                                           .SBTTL  ROUTINE READ AN OCTAL NUMBER FROM THE TTY
15668                                   ;*******************************************************************************
15669                                   ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
15670                                   ;*CHANGE IT TO BINARY.
15671                                   ;*THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL
15672                                   ;*OCTAL DIGITS. IF AN ILLEGAL CHARACTER IS READ A '?' WILL BE TYPED
15673                                   ;*FOLLOWED BY A CARRIAGE RETURN-LINE FEED. THE COMPLETE NUMBER MUST
15674                                   ;*THEN BE RETYPED. THE INPUT IS TERMINATED BY TYPING A CARRIAGE RETURN.
15675                                   ;*CALL:
15676                                   ;*      RDOCT                           ;;READ AN OCTAL NUMBER
15677                                   ;*      RETURN HERE                     ;;LOW ORDER BITS ARE ON TOP OF THE STACK
15678                                   ;*                                      ;;HIGH ORDER BITS ARE IN $HIOCT
15679 064772  011646                  $RDOCT: MOV     (SP),-(SP)              ;;PROVIDE SPACE FOR THE
15680 064774  016666  000004  000002          MOV     4(SP),2(SP)             ;;INPUT NUMBER
15681 065002                                  PUSH    R0,R1,R2
15682 065010  104412                  1$:     RDLIN                           ;;READ AN ASCIZ LINE
15683 065012  012600                          MOV     (SP)+,R0                ;;GET ADDRESS OF 1ST CHARACTER
15684 065014  010037  065120                  MOV     R0,5$                   ;;AND SAVE IT
15685 065020  005001                          CLR     R1                      ;;CLEAR DATA WORD
15686 065022  005002                          CLR     R2
15687 065024  112046                  2$:     MOVB    (R0)+,-(SP)             ;;PICKUP THIS CHARACTER
15688 065026  001420                          BEQ     3$                      ;;IF ZERO GET OUT
15689 065030  122716  000060                  CMPB    #'0,(SP)                ;;MAKE SURE THIS CHARACTER
15690 065034  003026                          BGT     4$                      ;;IS AN OCTAL DIGIT
15691 065036  122716  000067                  CMPB    #'7,(SP)
15692 065042  002423                          BLT     4$
15693 065044  006301                          ASL     R1                      ;;*2
15694 065046  006102                          ROL     R2
15695 065050  006301                          ASL     R1                      ;;*4
15696 065052  006102                          ROL     R2
15697 065054  006301                          ASL     R1                      ;;*8
15698 065056  006102                          ROL     R2
15699 065060  042716  177770                  BIC     #^C7,(SP)               ;;STRIP THE ASCII JUNK
15700 065064  062601                          ADD     (SP)+,R1                ;;ADD IN THIS DIGIT
15701 065066  000756                          BR      2$                      ;;LOOP
15702 065070  005726                  3$:     TST     (SP)+                   ;;CLEAN TERMINATOR FROM STACK
15703 065072  010166  000012                  MOV     R1,12(SP)               ;;SAVE THE RESULT
15704 065076  010237  065140                  MOV     R2,$HIOCT
15705 065102                                  POP     R2,R1,R0
15706 065110  000002                          RTI                             ;;RETURN
15707 065112  005726                  4$:     TST     (SP)+                   ;;CLEAN PARTIAL FROM STACK
15708 065114  105010                          CLRB    (R0)                    ;;SET A TERMINATOR
15709 065116                                  TYPE                            ;;TYPE UP THRU THE BAD CHAR.
15710 065120  000000                  5$:     .WORD   0
15711 065122                                  TYPE    MSG062                  ;INPUT MUST BE A
15712 065126                                  TYPE    MSG063                  ;N OCTAL
15713 065132                                  TYPE    MSG064                  ;NUMBER
15714 065136  000724                          BR      1$                      ;;TRY AGAIN
15715 065140  000000                  $HIOCT: .WORD   0                       ;;HIGH ORDER BITS GO HERE
15716                                         .SBTTL  ROUTINE READ A DECIMAL NUMBER FROM THE TTY
15717
15718                                   ;*******************************************************************************
15719                                   ;*THIS ROUTINE WILL READ A DECIMAL (ASCII) NUMBER FROM THE TTY AND
15720                                   ;*CHANGE IT TO BINARY. IF TOO MANY CHARACTERS OR ANY ILLEGAL CHARACTERS
15721                                   ;*ARE READ A '?' FOLLOWED BY A CARRIAGE RETURN-LINE FEED WILL BE TYPED.
15722                                   ;*THE COMPLETE NUMBER MUST BE RETYPED. THE INPUT IS TERMINATED BY THE
15723                                   ;*USER TYPING A CARRIAGE RETURN. THE RANGE OF THE INPUT NUMBER IS
```

```
15724                                   ;*POSITIVE 32767 TO NEGATIVE 32768.
15725                                   ;*CALL:
15726                                   ;*      RDDEC                   ;;READ A DECIMAL NUMBER
15727                                   ;*      RETURN HERE             ;;NUMBER IS ON TOP OF THE STACK
15728                                   ;
15729
15730 065142  011646                   $RDDEC: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR
15731 065144  016666  000004  000002           MOV     4(SP),2(SP)     ;;THE INPUT NUMBER
15732 065152                                   PUSH    R0,R1,R2
15733 065160  104412                   1$:     RDLIN                   ;;READ AN ASCIZ LINE
15734 065162  012600                           MOV     (SP)+,R0        ;;ADDRESS OF 1ST CHAR.
15735 065164  010037  065310                   MOV     R0,6$           ;;SAVE INCASE OF BAD INPUT
15736 065170  005046                           CLR     -(SP)           ;;CLEAR DATA WORD
15737 065172  005002                           CLR     R2              ;;SIGN SET POSITIVF
15738 065174  122710  000055                   CMPB    #'-,(R0)        ;;SEE IF A MINUS SIGN WAS TYPED
15739 065200  001001                           BNE     2$              ;;BR IF NO MINUS SIGN
15740 065202  112002                           MOVB    (R0)+,R2        ;;SAVE FOR LATER USE
15741 065204  112001                   2$:     MOVB    (R0)+,R1        ;;PICKUP THIS CHARACTER
15742 065206  001424                           BEQ     3$              ;;GET OUT IF ZERO
15743 065210  122701  000060                   CMPB    #'0,R1          ;;MAKE SURE THIS CHARACTER
15744 065214  003032                           BGT     5$              ;;IS A DIGIT BETWEEN 0 & 9
15745 065216  122701  000071                   CMPB    #'9,R1
15746 065222  002427                           BLT     5$
15747 065224  032716  170000                   BIT     #^C7777,(SP)    ;;DON'T LET NUMBER GET TO BIG
15748 065230  001024                           BNE     5$              ;;BR IF NUMBER WOULD OVERFLOW
15749 065232  006316                           ASL     (SP)            ;;*2
15750 065234  011646                           MOV     (SP),-(SP)      ;;SAVE FOR LATER
15751 065236  006316                           ASL     (SP)            ;;*4
15752 065240  006316                           ASL     (SP)            ;;*8.
15753 065242  062616                           ADD     (SP)+,(SP)      ;;*10.
15754 065244  102416                           BVS     5$              ;;OVERFLOW ISN'T ALLOWED
15755 065246  162701  000060                   SUB     #'0,R1          ;;STRIP AWAY THE ASCII JUNK
15756 065252  060116                           ADD     R1,(SP)         ;;ADD IN THIS DIGIT
15757 065254  102412                           BVS     5$              ;;OVERFLOW ISN'T ALLOWED
15758 065256  000752                           BR      2$              ;;LOOP
15759 065260  005702                   3$:     TST     R2              ;;CHECK IF NUMBER IS NEG
15760 065262  001401                           BEQ     4$              ;;BR IF NO
15761 065264  005416                           NEG     (SP)            ;;YES--NEGATE THE NUMBER
15762 065266  012666  000012           4$:     MOV     (SP)+,12(SP)    ;;SAVE THE RESULT
15763 065272                                   POP     R2,R1,R0
15764 065300  000002                           RTI                     ;;RETURN
15765
15766 065302  005726                   5$:     TST     (SP)+           ;;CLEAN PARTIAL NUMBER FROM STACK
15767 065304  105010                           CLRB    (R0)            ;;SET A TERMINATOR
15768 065306                                   TYPE                    ;;TYPE THE INPUT UP TO BAD CHAR.
15769 065310  000000                   6$:     .WORD   0               ;;POINTER GOES HERE
15770 065312                                   TYPE    MSG062          ;INPUT MUSST BE A
15771 065316                                   TYPE    MSG065          ;DECIMAL
15772 065322                                   TYPE    MSG064          ;NUMBER
15773 065326  000714                           BR      1$              ;;TRY AGAIN
```

```
                                        .SBTTL  ROUTINE SAVE AND RESTORE R0-R5

                                ;*******************************************************************************
                                ;*SAVE R0-R5
                                ;*CALL:
                                ;*      SAVREG
                                ;*UPON RETURN  FROM $SAVREG THE STACK WILL LOOK LIKE:
                                ;*
                                ;*TOP---(+16)
                                ;* +2---(+18)
                                ;* +4---R5
                                ;* +6---R4
                                ;* +8---R3
                                ;*+10---R2
                                ;*+12---R1
                                ;*+14---R0

065330                          $SAVREG:
065330                                  PUSH    R0,R1,R2,R3,R4,R5
065344  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PS OF MAIN FLOW
065350  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PC OF MAIN FLOW
065354  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PS OF CALL
065360  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PC OF CALL
065364  000002                          RTI

                                ;*RESTORE R0-R5
                                ;*CALL:
                                ;*      RESREG
065366                          $RESREG:
065366  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF CALL
065372  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF CALL
065376  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
065402  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
065406                                  POP     R5,R4,R3,R2,R1,R0
065422  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

065424                                  $RAND:  PUSH    R0,R1,R2
065432  013700  002570                          MOV     SEEDLO,R0       ;SET R0 WITH LOW
065436  013701  002566                          MOV     SEEDHI,R1       ;SET R1 WITH HIGH
065442  012702  000007                          MOV     #7,R2           ;SET SHIFT COUNT
065446  006300                          1$:     ASL     R0              ;;SHIFT R0 LEFT AND
065450  006101                                  ROL     R1              ;;ROTATE CARRY INTO R1 AND
065452  077203                                  SOB     R2,1$
065454  063700  002570                          ADD     SEEDLO,R0       ;ADD NUMBER TO MAKE X 129
065460  005501                                  ADC     R1              ;;PROPOGATE CARRY
065462  063701  002566                          ADD     SEEDHI,R1       ;ADD NUMBER TO MAKE X 129
065466  062700  001057                          ADD     #1057,R0        ;;ADD LOW CONSTANT
065472  005501                                  ADC     R1              ;;PROPOGATE CARRY
065474  062701  047401                          ADD     #47401,R1       ;;ADD HIGH CONSTANT
065500  010037  002570                          MOV     R0,SEEDLO       ;SAVE R0
065504  010137  002566                          MOV     R1,SEEDHI       ;SAVE R1
065510                                          POP     R2,R1,R0
065516  000207                                  RETURN
```

```
15841                                           .SBTTL  ROUTINE DOUBLE LENGTH BINARY TO OCTAL ASCII CONVERT
15842                                   ;*******************************************************************************
15843                                   ;*THIS ROUTINE WILL CONVERT A 32-BIT UNSIGNED BINARY NUMBER TO AN
15844                                   ;*UNSIGNED OCTAL ASCIZ NUMBER.
15845                                   ;*CALL
15846                                   ;*      MOV     #PNTR,-(SP)     ;;POINTER TO LOW WORD OF BINARY NUMBER
15847                                   ;*      CALL    $DB20           ;;CALL THE ROUTINE
15848                                   ;*      RETURN                  ;;THE ADDRESS OF THE FIRST ASCIZ CHAR. IS ON THE STACK
15849
15850
15851 065520  104415                   $DB20:  SAVREG                  ;;SAVE ALL REGISTERS
15852 065522  016601  000002                   MOV     2(SP),R1        ;;PICKUP THE POINTER TO LOW WORD
15853 065526  012705  065637                   MOV     #$OCTVL+13.,R5  ;;POINTER TO DATA TABLE
15854 065532  012704  000014                   MOV     #12.,R4         ;;DO ELEVEN CHARACTERS
15855 065536  012703  177770                   MOV     #^C7,R3         ;;MASK
15856 065542  012100                           MOV     (R1)+,R0        ;;LOWER WORD
15857 065544  012101                           MOV     (R1)+,R1        ;;HIGH WORD
15858 065546  005002                           CLR     R2              ;;TERMINATOR
15859 065550  110245                   1$:     MOVB    R2,-(R5)        ;;PUT CHARACTER IN DATA TABLE
15860 065552  010002                           MOV     R0,R2           ;;GET THIS DIGIT
15861 065554  005304                           DEC     R4              ;;COUNT THIS CHARACTER
15862 065556  003007                           BGT     3$              ;;BR IF NOT THE LAST DIGIT
15863 065560  001405                           BEQ     2$              ;;BR IF IT IS THE LAST DIGIT
15864 065562  005205                           INC     R5              ;;ALL DIGITS DONE-ADJUST POINTER FOR FIRST
15865 065564  010566  000002                   MOV     R5,2(SP)        ;;ASCIZ CHAR. & PUT IT ON THE STACK
15866 065570  104416                           RESREG                  ;;RESTORE ALL REGISTERS
15867 065572  000207                           RETURN                  ;;RETURN TO USER
15868 065574  006203                   2$:     ASR     R3              ;;POSITION THE MASK FOR THE LAST DIGIT
15869 065576  006001                   3$:     ROR     R1              ;;POSITION THE BINARY NUMBER FOR
15870 065600  006000                           ROR     R0              ;;      THE NEXT OCTAL DIGIT
15871 065602  006001                           ROR     R1
15872 065604  006000                           ROR     R0
15873 065606  006001                           ROR     R1
15874 065610  006000                           ROR     R0
15875 065612  040302                           BIC     R3,R2           ;;MASK OUT ALL JUNK
15876 065614  062702  000060                   ADD     #'0,R2          ;;MAKE THIS CHAR. ASCII
15877 065620  000753                           BR      1$              ;;GO PUT IT IN THE DATA TABLE
15878 065622  000016                   $OCTVL: .REPT   14.             ;;RESERVE DATA TABLE
15881         065626                   $OCT8=$OCTVL+4                  ;POINTER TO 11 DIGIT NUMBER
```

```
                                        .SBTTL  TABLES

                                        .SBTTL  APT MAILBOX-ETABLE
                                $MAIL:
                                $MSGTY: .WORD   0       ;;MESSAGE TYPE CODE
                                $FATAL: .WORD   0       ;;FATAL ERROR NUMBER (ERROR PC)
                                $TESTN: .WORD   0       ;;TEST PATTERN NUMBER
                                $PASS:  .WORD   0       ;;PASS COUNT
                                $DEVCT: .WORD   0       ;;DEVICE COUNT
                                $UNIT:  .WORD   0       ;;I/O UNIT NUMBER
                                $MSGAD: .WORD   0       ;;MESSAGE ADDRESS
                                $MSGLG: .WORD   0       ;;MESSAGE LENGTH
                                $ETABLE:                ;;APT ENVIRONMENT TABLE
                                $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.
                                $ENVM:  .BYTE   0       ;ENVIRONMENT MODE
                                                        ;BIT7(200)=USE APT SIZE INFO ;BIT5(40)=NO CONSOLE
                                $SWREG: .WORD   101     ;;APT SWITCH REGISTER
                                $USWR:  .WORD   0       ;USED TO LIMIT THE NUMBER OF PASSES
                                $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
                                $MAMS1: .BYTE   1       ;;HIGH ADDRESS,M.S. BYTE ;DEFAULT = 64K
                                $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
                                $MADR1: .WORD   177776  ;;HIGH ADDRESS,BLK#1
                                ;*                        MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF 'TYPE'' ABOVE
                                $MAMS2: .BYTE   0       ;;HIGH ADDRESS,M.S. BYTE
                                $MTYP2: .BYTE   0       ;;MEM.TYPE,BLK#2
                                $MADR2: .WORD   0       ;;MEM.LAST ADDRESS,BLK#2
                                $MAMS3: .BYTE   0       ;;HIGH ADDRESS,M.S.BYTE
                                $MTYP3: .BYTE   0       ;;MEM.TYPE,BLK#3
                                $MADR3: .WORD   0       ;;MEM.LAST ADDRESS,BLK#3
                                $MAMS4: .BYTE   0       ;;HIGH ADDRESS,M.S.BYTE
                                $MTYP4: .BYTE   0       ;;MEM.TYPE,BLK#4
                                $MADR4: .WORD   0       ;;MEM.LAST ADDRESS,BLK#4
                                $VECT1: .WORD   0       ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
                                $VECT2: .WORD   0       ;;INTERRUPT VECTOR#2BUS PRIORITY#2
                                $BASE:  .WORD   0       ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
                                $DEVM:  .WORD   0       ;;DEVICE MAP

                                $CDW1:  .WORD   0
                                $CDW2:  .WORD   0
```

APT MAILBOX-ETABLE

```
 15937                                  ;THE FOLLOWING LOCATIONS SPECIFY WHICH PATTERNS
 15938                                  ;ARE TO BE RUN FOR PARTICULAR MEMORIES
 15939                                  ;
 15940                                  ;REFERENCE THE TABLE LISTED BELOW TO RELATE BITS TO PATTERNS.
 15941                                  ;BIT0 SET WILL RUN THE FIRST ENTRY IN THE TABLE, BIT0 SET
 15942                                  ;IN THE SECOND WORD WILL RUN THE 17TH ENTRY IN THE TABLE ...
 15943                                  ;
 15944                                  ;NOTE** NULL TESTS DO NOT TAKE ANY TIME
 15945                                  ;                                        ;FIELD SERVICE VALUE
 15946 065724  177777                   $DDW0:  .WORD   177777  ;ECC CSR TESTS          177777          TABLE = MKCSRT:
 15947 065726  177777                   $DDW1:  .WORD   177777  ;ECC CSR TESTS          177777          TABLE = MKCSRT:
 15948 065730  177777                   $DDW2:  .WORD   177777  ;ECC PATTERNS           103777          TABLE = MKPAT:
 15949 065732  177777                   $DDW3:  .WORD   177777  ;ECC PATTERNS           177777          TABLE = MKPAT:
 15950 065734  177777                   $DDW4:  .WORD   177777  ;PARITY PATTERNS        003777          TABLE = MJPAT:
 15951 065736  177777                   $DDW5:  .WORD   177777  ;PARITY PATTERNS        177774          TABLE = MJPAT:
 15955 065740                           $ETEND:
 15956                                  ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
 15957                                  ;INTERFACE SPEC.
 15958
 15959 065740                           $APTHD:
 15960 065740  000000                   $HIBTS: .WORD   0               ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
 15961 065742  065640                   $MBADR: .WORD   $MAIL           ;;ADDRESS OF APT MAILBOX (BITS 0-15)
 15962 065744  000043                   $TSTM:  .WORD   35.             ;;RUN TIM OF LONGEST TEST
 15963 065746  001274                   $PASTM: .WORD   700.            ;;RUN TIME IN SECS. OF 1ST PASS ON 128K (QUICK VERIFY)
 15964 065750  000000                   $UNITM: .WORD   0.              ;;EXTRA RUN TIME OF A PASS FOR EACH ADDITIONAL 128K (QV)
 15965 065752  000040                           .WORD   $ETEND-$MAIL/2  ;;LENGTH MAILBOX-ETABLE(WORDS)
```

```
                                        .SBTTL  ROUTINE TRAP DECODER

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

065754  010046                  $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
065756  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
065762  005740                          TST     -(R0)           ;;BACKUP BY 2
065764  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
065766  006300                          ASL     R0              ;;POSITION FOR INDEXING
065770  016000  066016                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
065774  000200                          RTS     R0              ;;GO TO ROUTINE


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

065776  011646                  $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
066000  016666  000004  000002          MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
066006  000002                          RTI                     ;;RESTORE THE PSW

066010                          $NOTRAP:TYPE    MSG006          ;UNDEFINED TRAP INSTRUCTION
066014  000000                  $HALT2: HALT
```

```
                                        .SBTTL  TRAP TABLE

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

                                ;       ROUTINE
                                ;       -------
 16001 066016  065776   $TRPAD: .WORD   $TRAP2
 16002 066020  056152           $TYPE   ;CALL=TYPEIT    TRAP+1(104401)  TTY TYPEOUT ROUTINE
 16003 066022  063110           $TYPOC  ;CALL=TYPOC     TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
 16004 066024  063064           $TYPOS  ;CALL=TYPOS     TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
 16005 066026  066010           $NOTRAP;$TYPON  ;CALL=TYPON     TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
 16006 066030  063312           $TYPDS  ;CALL=TYPDS     TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
 16007 066032  066010           $NOTRAP;$TYPBN  ;CALL=TYPBN     TRAP+6(104406)  TYPE BINARY (ASCII) NUMBER
 16008
 16009 066034  063712           $GTSWR  ;CALL=GTSWR     TRAP+7(104407)  GET SOFT-SWR SETTING
 16010 066036  063536           $CKSWR  ;CALL=CKSWR     TRAP+10(104410) TEST FOR CHANGE IN SOFT-SWR
 16011
 16012 066040  064256           $RDCHR  ;CALL=RDCHR     TRAP+11(104411) TTY TYPEIN CHARACTER ROUTINE
 16013 066042  054406           $RDLIN  ;CALL=RDLIN     TRAP+12(104412) TTY TYPEIN STRING ROUTINE
 16014 066044  064772           $RDOCT  ;CALL=RDOCT     TRAP+13(104413) READ AN OCTAL NUMBER FROM TTY
 16015 066046  065142           $RDDEC  ;CALL=RDDEC     TRAP+14(104414) READ A DECIMAL NUMBER FROM TTY
 16016
 16017 066050  065330           $SAVREG ;CALL=SAVREG    TRAP+15(104415) SAVE R0-R5 ROUTINE
 16018 066052  065366           $RESREG ;CALL=RESREG    TRAP+16(104406) RESTORE R0-R5 ROUTINE
 16019
 16020 066054  042644           $KERNEL  ;CALL=KERNEL   TRAP+17(104417) ENTER KERNEL MODE
 16021 066056  042654           $ENERGIZE;CALL=ENERGIZETRAP+20(104420) TURN ON MEMORY MANAGEMENT & TRAPS
 16022 066060  042664           $DEENERGI;CALL=DEENERGITRAP+21(104421) TURN OFF MEMORY MENAGEMENT & TRAPS
 16023
 16024 066062  045100           $KMAP    ;CALL=KMAP     TRAP+22(104422) MAP KERNEL 1 TO 1
 16025
 16026 066064  042674           $CACHN  ;CALL=CACHON    TRAP+23(104423) TURN CACHE ON
 16027 066066  042720           $CACHF  ;CALL=CACHOFF   TRAP+24(104424) TURN CACHE OFF
 16028
 16029 066070  042736           $LOADC  ;CALL=LOADCSR   TRAP+25(104425) LOAD CORRECT CSR
 16030 066072  043032           $READC  ;CALL=READCSR   TRAP+26(104426) READ CORRECT CSR
 16031
 16032 066074  056506           $PER01  ;CALL=PERR01    TRAP+27(104427) PROGRAM DETECTED ERROR
 16033 066076  056534           $PER02  ;CALL=PERR02    TRAP+30(104430) PROGRAM DETECTED ERROR
 16034 066100  056562           $PER03  ;CALL=PERR03    TRAP+31(104431) PROGRAM DETECTED ERROR
 16035 066102  056612           $PER04  ;CALL=PERR04    TRAP+32(104432) PROGRAM DETECTED ERROR
 16036 066104  056674           $PER07  ;CALL=PERR07    TRAP+33(104433) PROGRAM DETECTED ERROR
 16037 066106  056716           $PER10  ;CALL=PERR10    TRAP+34(104434) PROGRAM DETECTED ERROR
 16038 066110  056746           $PER11  ;CALL=PERR11    TRAP+35(104435) PROGRAM DETECTED ERROR
 16039 066112  056766           $PER12  ;CALL=PERR12    TRAP+36(104436) PROGRAM DETECTED ERROR
 16040 066114  057010           $PER13  ;CALL=PERR13    TRAP+37(104437) PROGRAM DETECTED ERROR
 16041 066116  057030           $PER14  ;CALL=PERR14    TRAP+40(104440) PROGRAM DETECTED ERROR
 16042 066120  057052           $PER15  ;CALL=PERR15    TRAP+41(104441) PROGRAM DETECTED ERROR
 16043 066122  057074           $PER16  ;CALL=PERR16    TRAP+42(104442) PROGRAM DETECTED ERROR
 16044 066124  057114           $PER17  ;CALL=PERR17    TRAP+43(104443) PROGRAM DETECTED ERROR
 16045 066126  057132           $PER20  ;CALL=PERR20    TRAP+44(104444) PROGRAM DETECTED ERROR
 16046 066130  057150           $PER21  ;CALL=PERR21    TRAP+45(104445) PROGRAM DETECTED ERROR
 16047 066132  057170           $PER22  ;CALL=PERR22    TRAP+46(104446) PROGRAM DETECTED ERROR
 16048 066134  057206           $PER23  ;CALL=PERR23    TRAP+47(104447) PROGRAM DETECTED ERROR
 16049 066136  057224           $PER24  ;CALL=PERR24    TRAP+50(104450) PROGRAM DETECTED ERROR
 16050 066140  053762           $PER25  ;CALL=PERR25    TRAP+51(104451) PROGRAM DETECTED ERROR
```

TRAP TABLE

```
 16051 066142  057414          $PER26  ;CALL=PERR26     TRAP+52(104452) PROGRAM DETECTED ERROR
 16052 066144  057434          $PER27  ;CALL=PERR27     TRAP+53(104453) PROGRAM DETECTED ERROR
 16053 066146  054210          $PER30  ;CALL=PERR30     TRAP+54(104454) PROGRAM DETECTED ERROR
 16054 066150  057624          $PER31  ;CALL=PERR31     TRAP+55(104455) PROGRAM DETECTED ERROR
 16055 066152  057722          $PER32  ;CALL=PERR32     TRAP+56(104456) PROGRAM DETECTED ERROR
 16056 066154  057770          $PER33  ;CALL=PERR33     TRAP+57(104457) PROGRAM DETECTED ERROR
 16057 066156  060050          $PER34  ;CALL=PERR34     TRAP+60(104460) PROGRAM DETECTED ERROR
 16058 066160  060102          $PER35  ;CALL=PERR35     TRAP+61(104461) PROGRAM DETECTED ERROR
 16059 066162  060136          $PER36  ;CALL=PERR36     TRAP+62(104462) PROGRAM DETECTED ERROR
 16060 066164  066010          $NOTRAP ;CALL=PERR37     TRAP+63(104463) PROGRAM DETECTED ERROR
 16061 066166  066010          $NOTRAP ;CALL=PERR40     TRAP+64(104464) PROGRAM DETECTED ERROR
 16062 066170  066010          $NOTRAP ;CALL=PERR41     TRAP+65(104465) PROGRAM DETECTED ERROR
 16063 066172  066010          $NOTRAP ;CALL=PERR42     TRAP+66(104466) PROGRAM DETECTED ERROR
 16064 066174  066010          $NOTRAP ;CALL=PERR43     TRAP+67(104467) PROGRAM DETECTED ERROR
 16065
 16066 066176  043256          $ECCDIS ;CALL=ECCDIS     TRAP+70(104470) DISABLE ECC ON ALL CSR'S
 16067 066200  043272          $ECC1DIS;CALL=ECC1DIS    TRAP+71(104471) DISABLE ECC ON 1 SELECTED CSR
 16068 066202  043304          $ECCINIT;CALL=ECCINIT    TRAP+72(104472) INITIALIZE ALL MK11 CSR'S
 16069 066204  043320          $ECC1INIT;CALL=ECC1INIT  TRAP+73(104473) INITIALIZE 1 SELECTED MK11 CSR
 16070 066206  043360          $CBCSR  ;CALL=CBCSR      TRAP+74(104474) WRITE GENERATED CHECKBITS IN ALL CSR'S
 16071 066210  043402          $CB1CSR ;CALL=CB1CSR     TRAP+75(104475) WRITE GENERATED CHECKBITS IN 1 SELECTED CSR
 16072 066212  043422          $WASSBE ;CALL=WASSBE     TRAP+76(104476) WAS THERE A SBE ON ANY CSR?
 16073 066214  043536          $WAS1SBE;CALL=WAS1SBE    TRAP+77(104477) WAS THERE A SBE ON 1 SELECTED CSR?
 16074 066216  043566          $WASDBE ;CALL=WASDBE     TRAP+100(104500)WAS THERE A DBE ON ANY CSR?
 16075 066220  043702          $WAS1DBE;CALL=WAS1DBE    TRAP+101(104501)WAS THERE A DBE ON 1 SELECTED CSR?
 16076 066222  043732          $CLRCSR ;CALL=CLRCSR     TRAP+102(104502)CLEAR ALL CSR'S
 16077 066224  043744          $CLR1CSR;CALL=CLR1CSR    TRAP+103(104503)CLEAR 1 SELECTED CSR
 16078 066226  043754          $CHKDIS ;CALL=CHKDIS     TRAP+104(104504)DISABLE ECC & WRITE CKBITS FROM ALL CSR'S
 16079 066230  043770          $CHK1DIS;CALL=CHK1DIS    TRAP+105(104505)DISABLE ECC & WRITE CKBITS FROM 1 CSR
 16080 066232  043332          $ENASBE ;CALL=ENASBE     TRAP+106(104506)ENABLE TRAPS ON SBE'S FROM ALL CSR'S
 16081 066234  043346          $ENA1SBE;CALL=ENA1SBE    TRAP+107(104507)ENABLE TRAPS ON SBE'S FROM 1 SELECTED CSR
 16082 066236  043052          $TSTRD  ;CALL=TSTREAD    TRAP+110(104510)TEST LOC (R1) & TST FOR SBE (WITHOUT FETCHES
 16083 066240  044050          $INVALID;CALL=INVALID    TRAP+111(104511)INVALIDATE BACKGROUND PATTERN ON BANK
 16084 066242  044100          $ERRGEN ;CALL=ERRGEN     TRAP+112(104512)TEST ERROR ADDRESS
 16085 066244  044350          $CBREG  ;CALL=CBREG      TRAP+112(104513)ENABLE CHECK/SYNDROME BIT REGISTER
 16086 066246  066010          $NOTRAP
 16087 066250  066010          $NOTRAP
 16088 066252  066010          $NOTRAP
 16089 066254  066010          $NOTRAP
 16090 066256  066010          $NOTRAP
 16091 066260  066010          $NOTRAP
```

TRAP TABLE

```
16094           177776                  ST      =       177776          ;STATUS REGISTER
```

```
16097                                           .SBTTL  TABLE   ERROR POINTER
16098
16099                                   ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
16100                                   ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
16101                                   ;*LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
16102                                   ;*NOTE1:        IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS (ERRPC).
16103                                   ;*NOTE2:        EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:
16104
16105                                   ;*      EM              ;;POINTS TO THE ERROR MESSAGE
16106                                   ;*      DH              ;;POINTS TO THE DATA HEADER
16107                                   ;*      DT              ;;POINTS TO THE DATA
16108                                   ;*      DF              ;;POINTS TO THE DATA FORMAT
16109
16110
16111 066262                           $ERRTB: ;ERROR  1
16112 066262  070750                           EM24
16113 066264  073105                           DH13
16114 066266  067306                           DT13
16115 066270  067667                           DF11
16116                                          ;ERROR  2
16117 066272  067735                           EM2
16118 066274  072414                           DH1
16119 066276  067132                           DT1
16120 066300  067545                           DF2
16121                                          ;ERROR  3
16122 066302  067773                           EM3
16123 066304  072474                           DH3
16124 066306  067150                           DT3
16125 066310  067662                           DF9
16126                                          ;ERROR  4
16127 066312  070025                           EM4
16128 066314  072474                           DH3
16129 066316  067160                           DT4
16130 066320  067662                           DF9
16131                                          ;ERROR  5
16132 066322  070073                           EM5
16133 066324  072530                           DH5
16134 066326  067170                           DT5
16135 066330  067545                           DF2
16136                                          ;ERROR  6
16137 066332  070150                           EM6
16138 066334  072530                           DH5
16139 066336  067170                           DT5
16140 066340  067545                           DF2
16141                                          ;ERROR  7
16142 066342  070175                           EM7
16143 066344  072530                           DH5
16144 066346  067170                           DT5
16145 066350  067545                           DF2
16146                                          ;ERROR  10
16147 066352  072161                           EM53
16148 066354  073627                           DH25
16149 066356  067464                           DT25
16150 066360  067545                           DF2
```

TABLE ERROR POINTER

```
16153                                   ;ERROR  11
16154 066362  070235                    EM11
16155 066364  072654                    DH7
16156 066366  067222                    DT7
16157 066370  067571                    DF3
16158                                   ;ERROR  12
16159 066372  070235                    EM11
16160 066374  072654                    DH7
16161 066376  067222                    DT7
16162 066400  067604                    DF4
16163                                   ;ERROR  13
16164 066402  070257                    EM12
16165 066404  072764                    DH10
16166 066406  067252                    DT10
16167 066410  067545                    DF2
16168                                   ;ERROR  14
16169 066412  070235                    EM11
16170 066414  072654                    DH7
16171 066416  067222                    DT7
16172 066420  067617                    DF5
16173                                   ;ERROR  15
16174 066422  070235                    EM11
16175 066424  072654                    DH7
16176 066426  067222                    DT7
16177 066430  067632                    DF6
16178                                   ;ERROR  16
16179 066432  070303                    EM13
16180 066434  073105                    DH13
16181 066436  067306                    DT13
16182 066440  067667                    DF11
16183                                   ;ERROR  17
16184 066442  070335                    EM14
16185 066444  073105                    DH13
16186 066446  067306                    DT13
16187 066450  067667                    DF11
16188                                   ;ERROR  20
16189 066452  070401                    EM15
16190 066454  073105                    DH13
16191 066456  067306                    DT13
16192 066460  067667                    DF11
16193                                   ;ERROR  21
16194 066462  072210                    EM55
16195 066464  073665                    DH26
16196 066466  067476                    DT26
16197 066470  067545                    DF2
16198                                   ;ERROR  22
16199 066472  070447                    EM17
16200 066474  072654                    DH7
16201 066476  067222                    DT7
16202 066500  067617                    DF5
16203                                   ;ERROR  23
16204 066502  072030                    EM50
16205 066504  073501                    DH23
16206 066506  067422                    DT23
16207 066510  067700                    DF13
```

TABLE ERROR POINTER

```
16210                                          ;ERROR  24
16211 066512  070507                           EM19
16212 066514  073105                           DH13
16213 066516  067306                           DT13
16214 066520  067667                           DF11
16215                                          ;ERROR  25
16216 066522  070561                           EM20
16217 066524  073105                           DH13
16218 066526  067306                           DT13
16219 066530  067667                           DF11
16220                                          ;ERROR  26
16221 066532  000000                           0               ;NO MESSAGE
16222 066534  073100                           DH12
16223 066536  067302                           DT12
16224 066540  067545                           DF2
16225                                          ;ERROR  27
16226 066542  070640                           EM21
16227 066544  073062                           DH11
16228 066546  067274                           DT11
16229 066550  067545                           DF2
16230                                          ;ERROR  30
16231 066552  070674                           EM22
16232 066554  073105                           DH13
16233 066556  067306                           DT13
16234 066560  067667                           DF11
16235                                          ;ERROR  31
16236 066562  000000                           0               ;NO MESSAGE
16237 066564  073202                           DH14
16238 066566  067330                           DT14
16239 066570  067545                           DF2
16240                                          ;ERROR  32
16241 066572  070721                           EM23
16242 066574  072530                           DH5
16243 066576  067170                           DT5
16244 066600  067545                           DF2
16245                                          ;ERROR  33
16246 066602  071027                           EM25
16247 066604  073261                           DH15
16248 066606  067346                           DT16
16249 066610  067645                           DF7
16250                                          ;ERROR  34
16251 066612  071054                           EM26
16252 066614  073400                           DH16
16253 066616  067376                           DT17
16254 066620  067571                           DF3
```

```
  16257                                          ;ERROR  35
  16258 066622  072134                           EM52
  16259 066624  073627                           DH25
  16260 066626  067464                           DT25
  16261 066630  067545                           DF2
  16262                                          ;ERROR  36
  16263 066632  071125                           EM27
  16264 066634  073400                           DH16
  16265 066636  067376                           DT17
  16266 066640  067660                           DF8
  16267                                          ;ERROR  37
  16268 066642  071622                           EM35
  16269 066644  072654                           DH7
  16270 066646  067222                           DT7
  16271 066650  067571                           DF3
  16272                                          ;ERROR  40
  16273 066652  071215                           EM29
  16274 066654  072654                           DH7
  16275 066656  067222                           DT7
  16276 066660  067571                           DF3
  16277                                          ;ERROR  41
  16278 066662  071277                           EM30
  16279 066664  072654                           DH7
  16280 066666  067222                           DT7
  16281 066670  067617                           DF5
  16282                                          ;ERROR  42
  16283 066672  072331                           EM60
  16284 066674  073422                           DH20
  16285 066676  067422                           DT23
  16286 066700  067700                           DF13
  16287                                          ;ERROR  43
  16288 066702  071407                           EM32
  16289 066704  072654                           DH7
  16290 066706  067222                           DT7
  16291 066710  067571                           DF3
  16292                                          ;ERROR  44
  16293 066712  071514                           EM33
  16294 066714  072654                           DH7
  16295 066716  067222                           DT7
  16296 066720  067571                           DF3
  16297                                          ;ERROR  45
  16298 066722  072064                           EM51
  16299 066724  073560                           DH24
  16300 066726  067444                           DT24
  16301 066730  067710                           DF14
  16302                                          ;ERROR  46
  16303 066732  071707                           EM36
  16304 066734  072607                           DH6
  16305 066736  067206                           DT6
  16306 066740  067545                           DF2
```

TABLE ERROR POINTER

```
 16309                                          ;ERROR  47
 16310 066742  071756                           EM40
 16311 066744  072451                           DH2
 16312 066746  067404                           DT20
 16313 066750  067545                           DF2
 16314                                          ;ERROR  50
 16315 066752  072231                           EM56
 16316 066754  073703                           DH27
 16317 066756  067504                           DT27
 16318 066760  067544                           DF1
 16319                                          ;ERROR  51
 16320 066762  072373                           EM61
 16321 066764  073560                           DH24
 16322 066766  067444                           DT24
 16323 066770  067710                           DF14
 16324                                          ;ERROR  52
 16325 066772  071215                           EM29
 16326 066774  073105                           DH13
 16327 066776  067306                           DT13
 16328 067000  067667                           DF11
 16329                                          ;ERROR  53
 16330 067002  071027                           EM25
 16331 067004  073757                           DH30
 16332 067006  067524                           DT30
 16333 067010  067726                           DF16
 16334                                          ;ERROR  54
 16335 067012  072263                           EM57
 16336 067014  073757                           DH30
 16337 067016  067524                           DT30
 16338 067020  067726                           DF16
 16339                                          ;ERROR  55
 16340 067022  070561                           EM20
 16341 067024  073560                           DH24
 16342 067026  067444                           DT24
 16343 067030  067710                           DF14
 16344                                          ;ERROR  56
 16345 067032  070561                           EM20
 16346 067034  073757                           DH30
 16347 067036  067524                           DT30
 16348 067040  067726                           DF16
 16349                                          ;ERROR  57
 16350 067042  070507                           EM19
 16351 067044  073560                           DH24
 16352 067046  067444                           DT24
 16353 067050  067710                           DF14
 16354                                          ;ERROR  60
 16355 067052  070303                           EM13
 16356 067054  073422                           DH20
 16357 067056  067422                           DT23
 16358 067060  067700                           DF13
 16359                                          ;ERROR  61
 16360 067062  071277                           EM30
 16361 067064  073422                           DH20
 16362 067066  067422                           DT23
 16363 067070  067700                           DF13
 16364                                          ;ERROR  62
 16365 067072  070561                           EM20
```

TABLE   ERROR POINTER

```
  16366 067074  073560                  DH24
  16367 067076  067444                  DT24
  16368 067100  067710                  DF14
  16369                                 ;ERROR  63
  16370 067102  070674                  EM22
  16371 067104  073560                  DH24
  16372 067106  067444                  DT24
  16373 067110  067710                  DF14
  16374                                 ;ERROR  64
  16375 067112  100300                  EM62
  16376 067114  072530                  DH5
  16377 067116  067170                  DT5
  16378 067120  067545                  DF2
  16379                                 ;ERROR  65
  16380 067122  070674                  EM22
  16381 067124  073422                  DH20
  16382 067126  067422                  DT23
  16383 067130  067700                  DF13
```

ERROR DATA TAGS (DT)

```
                                                        .SBTTL  ERROR DATA TAGS (DT)
16385
16386 067132  002016  002032  002042  DT1:    .WORD   ERRPC,ADDRESS,GOOD,BAD,0
      067140  002050  000000
16387 067144  002016  000000          DT2:    .WORD   ERRPC,0
16388 067150  002016  002034  002070  DT3:    .WORD   ERRPC,PADDRESS,PARCNT,0
      067156  000000
16389 067160  002016  002032  002066  DT4:    .WORD   ERRPC,ADDRESS,NEMCNT,0
      067166  000000
16390 067170  002016  177572  177574  DT5:    .WORD   ERRPC,MMR0,MMR1,MMR2,MMR3,CPUERR,0
      067176  177576  172516  177766
      067204  000000
16391 067206  002016  002416  002374  DT6:    .WORD   ERRPC,APTPAR,LSIZE,APTECC,MSIZE,0
      067214  002420  002376  000000
16392 067222  002016  002174  002032  DT7:    .WORD   ERRPC,DUMMY,ADDRESS,DUMMY,GOOD,BAD,BADXOR
      067230  002174  002042  002050
      067236  002056
16393 067240  002174  002174  002174          .WORD   DUMMY,DUMMY,DUMMY,DUMMY,0
      067246  002174  000000
16394 067252  002176  002200  002202  DT10:   .WORD   DETR0,DETR1,DETR2,DETR3,DETR4,DETR5,DETSP,DETPSW,0
      067260  002204  002206  002210
      067266  002212  002214  000000
16395 067274  002016  002146  000000  DT11:   .WORD   ERRPC,CSR,0
16396 067302  002146  000000          DT12:   .WORD   CSR,0
16397 067306  002016  002174  002032  DT13:   .WORD   ERRPC,DUMMY,ADDRESS,DUMMY,TSTDAT,TSTDAT+2,CHECK,CSR,0
      067314  002174  002244  002246
      067322  002310  002146  000000
16398 067330  177746  177572  177574  DT14:   .WORD   CONTRL,MMR0,MMR1,MMR2,MMR3,CPUERR,0
      067336  177576  172516  177766
      067344  000000
16399 067346  002016  002174  002174  DT16:   .WORD   ERRPC,DUMMY,DUMMY,GOOD,GOOD2,GOOD3
      067354  002042  002044  002046
16400 067362  002050  002052  002054          .WORD   BAD,BAD2,BAD3,DUMMY,DUMMY,0
      067370  002174  002174  000000
16401 067376  002016  002174  000000  DT17:   .WORD   ERRPC,DUMMY,0
16402 067404  002016  002042  002050  DT20:   .WORD   ERRPC,GOOD,BAD,0
      067412  000000
16403 067414  002016  002174  000000  DT22:   .WORD   ERRPC,DUMMY,0
16404 067422  002016  002174  002042  DT23:   .WORD   ERRPC,DUMMY,GOOD,BAD,DUMMY,DUMMY,DUMMY,DUMMY,0
      067430  002050  002174  002174
      067436  002174  002174  000000
16405 067444  002016  002174  002146  DT24:   .WORD   ERRPC,DUMMY,CSR,DUMMY,DUMMY,DUMMY,DUMMY,0
      067452  002174  002174  002174
      067460  002174  000000
16406 067464  002016  002042  002146  DT25:   .WORD   ERRPC,GOOD,CSR,CSRNO,0
      067472  002150  000000
16407 067476  002016  002050  000000  DT26:   .WORD   ERRPC,BAD,0
16408 067504  002016  002174  002032  DT27:   .WORD   ERRPC,DUMMY,ADDRESS,DUMMY,DUMMY,DUMMY,DUMMY,0
      067512  002174  002174  002174
      067520  002174  000000
16409 067524  002016  002174  002174  DT30:   .WORD   ERRPC,DUMMY,DUMMY,GOOD,BAD,CSR,DUMMY,0
      067532  002042  002050  002146
      067540  002174  000000
```

ERROR DATA FORMATS (DF)

```
                                                        .SBTTL  ERROR DATA FORMATS (DF)
16412
16413 067544    000                             DF1:    .BYTE   0
16414 067545    000     000     000     DF2:    .BYTE   0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
      067550    000     000     000
      067553    000     000     000
      067556    000     000     000
      067561    000     000     000
      067564    000     000     000
      067567    000     000
16415 067571    000     005     000     DF3:    .BYTE   0,5,0,8.,0,0,0,3,6,2,4
      067574    010     000     000
      067577    000     003     006
      067602    002     004
16416 067604    000     005     000     DF4:    .BYTE   0,5,0,8.,0,8.,8.,3,6,2,4
      067607    010     000     010
      067612    010     003     006
      067615    002     004
16417 067617    000     005     000     DF5:    .BYTE   0,5,0,8.,9.,9.,9.,3,6,2,4
      067622    010     011     011
      067625    011     003     006
      067630    002     004
16418 067632    000     005     000     DF6:    .BYTE   0,5,0,8.,9.,8.,8.,3,6,2,4
      067635    010     011     010
      067640    010     003     006
      067643    002     004
16419 067645    000     005     010     DF7:    .BYTE   0,5,8.,0,0,9.,0,0,9.,2,4
      067650    000     000     011
      067653    000     000     011
      067656    002     004
16420 067660    000     005             DF8:    .BYTE   0,5
16421 067662    000     001     001     DF9:    .BYTE   0,1,1,1,1
      067665    001     001
16422 067667    000     005     000     DF11:   .BYTE   0,5,0,8.,0,0,0,0,0
      067672    010     000     000
      067675    000     000     000
16423 067700    000     005     000     DF13:   .BYTE   0,5,0,0,3,6,2,4
      067703    000     003     006
      067706    002     004
16424 067710    000     005     000     DF14:   .BYTE   0,5,0,3,6,2,4
      067713    003     006     002
      067716    004
16425 067717    000     005     000     DF15:   .BYTE   0,5,0,8.,3,6,4
      067722    010     003     006
      067725    004
16426 067726    000     005     010     DF16:   .BYTE   0,5,8.,0,0,3,4
      067731    000     000     003
      067734    004
```

```
16429                                           .SBTTL  ERROR MESSAGES (EM)
16435 067735    103     101     116  EM2:       .ASCIZ  /CAN'T SET 22 BIT MODE IN MMR3/
16436 067773    120     101     122  EM3:       .ASCIZ  /PARITY ERROP(S) IN BANK 0/
16437 070025    116     117     116  EM4:       .ASCIZ  /NON-EXISTANT MEMORY (HOLES) IN BANK 0/
16438 070073    111     114     114  EM5:       .ASCIZ  /ILLEGAL OR RESERVED INSTRUCTION (TRAP TO 10)/
16439 070150    125     116     105  EM6:       .ASCIZ  /UNEXPECTED TRAP TO 4/
16440 070175    115     105     115  EM7:       .ASCIZ  /MEMORY MANAGEMENT (TRAP TO 250)/
16441 070235    115     105     115  EM11:      .ASCIZ  /MEMORY DATA ERROR/
16442 070257    104     105     124  EM12:      .ASCIZ  /DETAILED ERROR DUMP/
16443 070303    115     111     123  EM13:      .ASCIZ  /MISSING EXPECTED SBE FLAG/
16444 070335    127     122     111  EM14:      .ASCIZ  /WRITE BYTE FAILED TO CLEAR SBE FLAG/
16445 070401    106     101     111  EM15:      .ASCIZ  /FAILED TO GET INTERRUPT WITH DBE FLAG/
16446 070447    115     105     115  EM17:      .ASCIZ  /MEMORY DATA ERROR IN CHECK BITS/
16447 070507    123     102     105  EM19:      .ASCIZ  /SBE-DBE CAUSED PARITY TRAP WHEN INHIBITED/
16448 070561    123     102     105  EM20:      .ASCIZ  /SBE-DBE DID NOT CAUSE PARITY TRAP WHEN ENABLED/
16449 070640    123     102     105  EM21:      .ASCIZ  /SBE-DBE ON MASTER TEST WORD/
16450 070674    115     111     123  EM22:      .ASCIZ  /MISSING EXPECTED DBE/
16451 070721    125     116     105  EM23:      .ASCIZ  /UNEXPECTED PARITY TRAP/
16452 070750    122     105     103  EM24:      .ASCIZ  /RECEIVED DBE FLAG WHEN EXPECTING ONLY SBE FLAG/
16453 071027    103     110     105  EM25:      .ASCIZ  /CHECK BIT DATA ERROR/
16454 071054    101     104     104  EM26:      .ASCIZ  /ADDRESS PARITY ERROR DID NOT CAUSE ABORT/
16455 071125    105     103     103  EM27:      .ASCIZ  /ECC INHIBIT MODE POINTER FAILURE - DID NOT PROTECT BANK/
16456 071215    103     117     122  EM29:      .ASCIZ  /CORRECTION FAILURE WITH ECC ENABLED ON FORCED SBE/
16457 071277    127     122     111  EM30:      .ASCII  /WRITE BYTE WITH ECC ENABLED FAILED TO CLEAR DATA AT/<CRLF>
16458 071363    106     117     122             .ASCIZ  /FORCED SBE LOCATION/
16459 071407    115     117     126  EM32:      .ASCIZ  /MOVB #360,(R2)+ WITH ECC ENABLED CHANGED DATA AT FORCED DBE LOCATION/
16460 071514    115     117     126  EM33:      .ASCIZ  /MOV #177400,(R1) WITH ECC ENABLED CHANGED DATA AT FORCED DBE LOCATION/
16461 071622    125     116     105  EM35:      .ASCIZ  /UNEXPECTED CORRECTION WITH ECC DISABLE ON FORCED SBE/
16462 071707    101     120     124  EM36:      .ASCIZ  /APT SIZE DISAGREES WITH PROGRAM SIZING/
16463 071756    102     122     101  EM40:      .ASCIZ  /BRANCH GOBBLE FAILED CONDITION CODES TEST/
16464 072030    102     101     104  EM50:      .ASCIZ  /BAD ERROR ADDRESS GENERATED/
16465 072064    106     114     101  EM51:      .ASCIZ  /FLAGS NOT SET ON FORCED UNCORRECTED SBE/
16466 072134    102     111     124  EM52:      .ASCIZ  /BIT SET ERROR IN CSR/
16467 072161    102     111     124  EM53:      .ASCIZ  /BIT CLEAR ERROR IN CSR/
16468 072210    111     114     114  EM55:      .ASCIZ  /ILLEGAL CSR TYPE/
16469 072231    102     101     104  EM56:      .ASCIZ  /BAD PARITY TRAP GENERATED/
16470 072263    127     122     117  EM57:      .ASCIZ  /WRONG CHECK BIT READ BACK FROM MEMORY/
16471 072331    127     122     117  EM60:      .ASCIZ  /WRONG SYNDROME BITS READ INTO CSR/
16472 072373    103     123     122  EM61:      .ASCIZ  /CSR UPDATE ERROR/
```

```
                                                .SBTTL  ERROR DATA HEADERS (DH)
16475
16476 072414    040     040     120     DH1:    .ASCIZ  /  PC    DEV ADD    GOOD    BAD/
16477 072451    040     040     120     DH2:    .ASCIZ  /  PC    GD-CC BD-CC/
16478 072474    040     040     120     DH3:    .ASCIZ  /  PC    1ST ADD  # OF ERRORS/
16479 072530    040     040     120     DH5:    .ASCIZ  /  PC      MMR0     MMR1     MMR2     MMR3   CPUERR/
16480 072607    040     040     120     DH6:    .ASCIZ  /  PC    APTPAR   LSIZE  APTECC  MSIZE/
16481 072654    040     040     120     DH7:    .ASCII  /  PC     BANK  VADD      PADD      GOOD/
16482 072720    040     040     040             .ASCIZ  /      BAD      XOR  CSR  MTYP INT PAT/
16483 072764    040     040     122     DH10:   .ASCIZ  /  R0       R1       R2       R3       R4       R5       SP       PSW/
16484 073062    040     040     120     DH11:   .ASCIZ  /  PC       CSR/
16485 073100    040     103     123     DH12:   .ASCIZ  / CSR/
16486 073105    040     040     120     DH13:   .ASCII  /  PC    BANK  VADD      PADD     WROTE1  WROTE2/
16487 073162    040     103     110             .ASCIZ  / CHKBITS    CSR/
16488 073202    103     117     116     DH14:   .ASCIZ  /CONTRL    MMR0     MMR1     MMR2     MMR3   CPUERR/
16489 073261    040     040     120     DH15:   .ASCII  /  PC    BANK    PADD   GD-WD1  GD-WD2  GD-CHK/
16490 073336    040     102     101             .ASCIZ  / BAD-WD1 BAD-WD2  BAD-CHK INT PAT/
16491 073400    040     040     120     DH16:   .ASCIZ  /  PC    BANK/
16492 073415    040     040     120     DH19:   .ASCIZ  /  PC/
16493 073422    040     040     120     DH20:   .ASCIZ  /  PC    BANK GD-CSR  (CSR)  CSR  MTYP INT  PAT/
16494 073501    040     040     120     DH23:   .ASCIZ  /  PC    BANK GD-ERR BAD-ERR CSR  MTYP INT  PAT/
16495 073560    040     040     120     DH24:   .ASCIZ  /  PC    BANK  (CSR) CSR  MTYP INT  PAT/
16496 073627    040     040     120     DH25:   .ASCIZ  /  PC    GD-DAT  (CSR)   CSRNO/
16497 073665    040     040     120     DH26:   .ASCIZ  /  PC  BADCODE/
16498 073703    040     040     120     DH27:   .ASCIZ  /  PC    BANK  VADD     PADD   CSR  MTYP PAT/
16499 073757    040     040     120     DH30:   .ASCIZ  /  PC    BANK   PADD    WROTE    READ  CSR  PAT/
```

MESSAGES

```
16502                                          .SBTTL  MESSAGES
16503 074036     200     040     040  MSG001: .ASCIZ  <CRLF>/                                MEMORY CONFIGURATION MAP/
16504 074120     200     040     040  MSG002: .ASCIZ  <CRLF>/                                    16K WORD BANKS/
16505 074175     200     040     040  MSG003: .ASCII  <CRLF>/                        1       2       3/
16506 074237     040     040     040          .ASCIZ  /       4       5       6       7       /
16507 074302     200     040     040  MSG004: .ASCII  <CRLF>/       01234567012345670123456701/
16508 074343     060     061     062          .ASCIZ  /01234567012345670123456701234567012345670123/
16509 074410     200     105     122  MSG005: .ASCIZ  <CRLF>/ERRORS  /
16510 074422     200     125     116  MSG006: .ASCIZ  <CRLF>/UNDEFINED TRAP INSTRUCTION/<32>
16511 074457     200     111     116  MSG007: .ASCIZ  <CRLF>/INTRLV  /                       ;INTERLEAVED CSR #
16512 074471     200     115     105  MSG009: .ASCIZ  <CRLF>/MEMTYPE /                       ;MEMORY TYPE
16513 074503     200     120     122  MSG010: .ASCIZ  <CRLF>/PROTECT /                       ;MEMORY PROTECTED
16514 074515     040     040     040  MSG011: .ASCIZ  /       0       1       2       3       4       5       6/
16515 074603     064     065     066  MSG012: .ASCIZ  /4567012345670123456701234567012345670123456701234567012345670123456701234567/
16516 074700     130     000          MSG013: .ASCIZ  /X/
16517 074702     040     000          MSG014: .ASCIZ  / /                     ;SPACE
16518 074704     000     000          MSG015: .BYTE   0,0                     ;FOR SINGLE ASCII CHARACTERS & TERMINATOR
16519 074706     200     103     123  MSG016: .ASCIZ  <CRLF>/CSR     /
16520 074720     040     040     040  MSG017: .ASCIZ  /       /                       ;8 SPACES
16521 074731     040     040     000  MSG018: .ASCIZ  /  /                            ;2 SPACES
16522 074734     040     040     040  MSG019: .ASCIZ  /   /                           ;3 SPACES
16523 074740     200     106     123  MSG020: .ASCIZ  <CRLF>/FS COMMAND MODE/
16524 074761     200     103     117  MSG021: .ASCII  <CRLF>/COMMANDS AVAILABLE:/
16525 075005     200     060     040          .ASCII  <CRLF>/0 = EXIT/
16526 075016     200     061     040          .ASCII  <CRLF>/1 = READ CSR/
16527 075033     200     062     040          .ASCII  <CRLF>/2 = LOAD CSR/
16528 075050     200     063     040          .ASCII  <CRLF>/3 = EXAMINE MEMORY/
16529 075073     200     064     040          .ASCII  <CRLF>/4 = MODIFY MEMORY/
16530 075115     200     065     040          .ASCII  <CRLF>/5 = SELECT BANK & TEST/
16531 075144     200     066     040          .ASCII  <CRLF>/6 = TYPE CONFIG MAP/
16532 075170     200     067     040          .ASCII  <CRLF>/7 = SOB-A-LONG TEST/
16533 075214     200     070     040          .ASCII  <CRLF>/8 = ERROR SUMMARY/
16534 075236     200     071     075          .ASCII  <CRLF>/9=  REFRESH TEST/
16535 075257     200     061     060          .ASCII  <CRLF>/10= SET FILL COUNT/
16536 075302     200     061     061          .ASCII  <CRLF>/11= ENTER KAMIKAZE MODE/
16537 075332     200     061     062          .ASCII  <CRLF>/12= EXIT KAMIKAZE MODE/
16538 075361     200     061     063          .ASCII  <CRLF>/13= TURN CACHE OFF/
16539 075404     200     061     064          .ASCII  <CRLF>/14= TURN CACHE ON/
16540 075426     200     061     065          .ASCII  <CRLF>/15= TEST SELECTED BANKS/
16541 075456     200     061     066          .ASCII  <CRLF>/16= TEST ALL BANKS/
16542 075501     200     061     067          .ASCII  <CRLF>/17= ENABLE TRACE/
16543 075522     200     061     070          .ASCII  <CRLF>/18= DISABLE TRACE/
16544 075544     015     012     000          .BYTE   15,12,0
16545 075547     200     127     110  MSG022: .ASCIZ  <CRLF>/WHICH CSR(0-F)? /
16546 075571     200     103     123  MSG023: .ASCIZ  <CRLF>/CSR WORD? /
16547 075605     200     103     123  MSG025: .ASCIZ  <CRLF>/CSR DOES NOT EXIST/
16548 075631     200     103     117  MSG026: .ASCIZ  <CRLF>/COMMAND:/
16549 075643     200     117     114  MSG027: .ASCIZ  <CRLF>/OLD CSR WAS/
16550 075660     200     103     123  MSG028: .ASCIZ  <CRLF>/CSR IS NOW/
16551 075674     200     105     130  MSG029: .ASCIZ  <CRLF>/EXAMINE MEMORY/
16552 075714     200     102     101  MSG030: .ASCIZ  <CRLF>/BANK(0-167)? /
16553 075733     200     120     110  MSG031: .ASCIZ  <CRLF>/PHYSICAL ADDRESS(0-17757776)? /
16554 075773     200     120     101  MSG032: .ASCIZ  <CRLF>/PARITY ABORT/<32>
16555 076012     200     124     111  MSG033: .ASCIZ  <CRLF>/TIMEOUT TRAP/<32>
16556 076031     200     102     131  MSGA34: .ASCIZ  <CRLF>/BYPASSING ECC TESTS ON BANK /
16557 076067     040     104     125  MSGB34: .ASCIZ  / DUE TO SBE LOCATIONS/
16558 076115     121     126     000  MSG035: .ASCIZ  /QV/
```

MESSAGES

```
16559 076120  200  115  117  MSG036: .ASCIZ  <CRLF>/MODIFY MEMORY/
16560 076137  200  117  114  MSG037: .ASCIZ  <CRLF>/OLD DATA WAS /
16561 076156  200  104  101  MSG038: .ASCIZ  <CRLF>/DATA IS NOW /
16562 076174  200  111  116  MSG039: .ASCIZ  <CRLF>/INPUT NEW DATA? /
16563 076216  200  123  105  MSG040: .ASCIZ  <CRLF>/SELECT BANK & TEST/
16564 076242  200  102  101  MSG041: .ASCIZ  <CRLF>/BANK NOT ACCESSABLE/
16565 076267  200  124  105  MSG042: .ASCIZ  <CRLF>/TEST(0-47)? /
16566 076305  200  124  105  MSG043: .ASCIZ  <CRLF>/TEST 0 DATA IS? /
16567 076327  200  124  117  MSG046: .ASCIZ  <CRLF>/TO ESCAPE TYPE ANY KEY/<CRLF><12><12>
16568 076362  200  124  105  MSG047: .ASCIZ  <CRLF>/TEST COMPLETE/
16569 076401  040  116  117  MSG048: .ASCIZ  / NOT AVAILABLE NOW - TRY LATER!/
16570 076441  200  102  101  MSG049: .ASCIZ  <CRLF>/BANK REQUIRES RELOCATION/
16571                          .EVEN
16572 076474  120  117  127  MSG051: .ASCIZ  /POWER RECOVERY/
16573 076513  200  123  117  MSG055: .ASCIZ  <CRLF>/SOB-A-LONG TEST/
16574 076534  200  102  105  MSG056: .ASCIZ  <CRLF>/BELL = EACH PASS COMPLETE/
16575 076567  200  040  040  MSG058: .ASCIZ  <CRLF>/  CSR     CSR .../
16576 076611  077  077  077  MSG061: .ASCIZ  /??????/
16577 076620  111  116  120  MSG062: .ASCIZ  /INPUT MUST BE A/
16578 076640  116  040  117  MSG063: .ASCIZ  /N OCTAL /
16579 076651  116  125  115  MSG064: .ASCIZ  /NUMBER/<CRLF>
16580 076661  040  104  105  MSG065: .ASCIZ  / DECIMAL /
16581 076673  200  105  122  MSG066: .ASCIZ  <CRLF>/ERRORS > 20 - ABORTING FOR XXDP CHAIN/
16582 076742  106  101  124  MSG067: .ASCIZ  /FATAL /
16583 076751  113  040  127  MSG070: .ASCIZ  /K WORDS OF MEMORY TOTAL/<CRLF>
16584 077002  200  122  105  MSG073: .ASCIZ  <CRLF>/REFRESH TEST/
16585 077020  200  122  105  MSG075: .ASCIZ  <CRLF>/RELOCATION NOT POSSIBLE/<32>
16586 077052  200  040  040  MSG076: .ASCIZ  <CRLF>/  BANK  ERRORS/<CRLF>
16587 077073  200  105  116  MSG077: .ASCIZ  <CRLF>/END PASS #/
16588 077107  040  105  122  MSG079: .ASCIZ  / ERROR(S) DETECTED/<CRLF>
16589 077133  200  106  111  MSG085: .ASCIZ  <CRLF>/FILL COUNT(OCTAL)? /
16590 077160  200  113  105  MSG088: .ASCIZ  <CRLF>/KERNEL STACK/
16591 077176  200  123  125  MSG089: .ASCIZ  <CRLF>/SUPERVISOR STACK/
16592 077220  200  125  123  MSG090: .ASCIZ  <CRLF>/USER STACK/
16593 077234  040  111  123  MSG091: .ASCIZ  / IS EMPTY/
16594 077246  122  105  114  MSG092: .ASCIZ  /RELOCATED  /
16595 077262  102  101  116  MSG093: .ASCIZ  /BANK=/
16596 077270  040  040  124  MSG095: .ASCIZ  /  TEST=/
16597 077300  200  105  116  MSG101: .ASCIZ  <CRLF>/ENTERING KAMIKAZE MODE/
16598 077330  200  114  105  MSG102: .ASCIZ  <CRLF>/LEAVING KAMIKAZE MODE/
16599 077357  200  114  105  MSG103: .ASCIZ  <CRLF>/LEAVING FS MODE/<CRLF>
16600 077401  032  000       MSG104: .BYTE   32,0                          ;CONTROL Z
16601 077403  200  105  116  MSG105: .ASCIZ  <CRLF>/ENTER BANKS - USE NUMBER 200 TO TERMINATE/
16602 077456  200  103  101  MSG106: .ASCIZ  <CRLF>/CACHE IS OFF/
16603 077474  200  103  101  MSG107: .ASCIZ  <CRLF>/CACHE IS ON (EXCEPT DURING ACTUAL PATTERNS)/
16604 077551  200  117  116  MSG110: .ASCIZ  <CRLF>/ONLY SELECTED BANKS WILL BE TESTED/
16605 077615  200  101  114  MSG111: .ASCIZ  <CRLF>/ALL BANKS WILL BE TESTED/
16606 077647  113  040  117  MSG112: .ASCIZ  /K OF MS11-L/<CRLF>
16607 077664  113  040  117  MSG113: .ASCIZ  /K OF MS11-M/<CRLF>
16608 077701  113  040  117  MSG114: .ASCIZ  /K OF MS11-P/<CRLF>
16609 077716  200  040  040  MSG117: .ASCIZ  <CRLF>"   11/44"
16610 077730  200  040  040  MSG119: .ASCIZ  <CRLF>/   NO/
16611 077737  040  103  101  MSG120: .ASCIZ  / CACHE AVAILABLE/
16612 077760  040  103  101  MSG121: .ASCIZ  / CACHE BYPASSED/
16613 100000  200  103  123  MSG122: .ASCII  <CRLF>/CSR NUMBER /
16614 100014  000              MSGA122:.BYTE   0
16615 100015  040  103  117          .ASCIZ  / CONTROLS TOO MANY BANKS/
```

MESSAGES

```
 16616 100046      040      120      101  MSG125: .ASCIZ  / PASSES COMPLETED/
 16617 100070      200      120      122  MSG126: .ASCIZ  <CRLF>/PROGRAM CSR COULD NOT BE DETERMINED/
 16618 100135      200      124      122  MSG127: .ASCIZ  <CRLF>/TRACE ENABLED/
 16619 100154      200      124      122  MSG128: .ASCIZ  <CRLF>/TRACE DISABLED/
 16620 100174      200      200      040  MSG008: .ASCIZ  <CRLF><CRLF>/                    CSR MAP/<CRLF>
 16621 100226      200      040      103  MSG000: .ASCIZ  <CRLF>" CZMSPA  MS11-L/M/P MEMORY DIAGNOSTIC"
 16622 100275      200      200      000  MSG129: .ASCIZ  <CRLF><CRLF>
 16623 100300      120      122      117  EM62:   .ASCIZ  /PROCESSOR NOT SUPPORTED BY THIS DIAGNOSTIC/
 16624                                              .EVEN
 16630 100354                                       $$END
 16631 100354                                END:
 16632 100354  001272                                 .PRINT  60000-SUPLIMIT  ;SUPERVISOR ADDRESSES LEFT
 16636         000200                                 .END START3
```

SYMBOL TABLE

ABORTF	002142	B0	004614	B63	044014	CPUERR=	177766	DH14	073202
ACFLAG	002114	B1	005550	B64	045234	CR =	000015	DH15	073261
ACTFLA	002344	B10	013462	B65	045242	CRLF =	000200	DH16	073400
ADDRES	002032	B100	055070	B66	045372	CSR	002146	DH19	073415
ANA2	010002	B101	055076	B67	045406	CSRADD=	172100	DH2	072451
APTDOW	047746	B102	062530	B7	013420	CSRCAS	020174	DH20	073422
APTECC	002420	B103	062534	B70	052536	CSRFBA	002230	DH23	073501
APTFLA	002346	B104	062632	B71	052760	CSRFIR	002224	DH24	073560
APTHAN	015316	B105	062724	B72	053234	CSRHOL	002522	DH25	073627
APTHLT	050014	B106	063010	B73	053530	CSRINC	002324	DH26	073665
APTPAR	002416	B11	014214	B74	053530	CSRINF	002456	DH27	073703
APTSIZ	002440	B12	014302	B75	053626	CSRINT	002234	DH3	072474
BACK	060536	B13	014344	B76	054566	CSRLAS	002226	DH30	073757
BACKGN	041224	B14	014570	B77	054572	CSRLBA	002232	DH5	072530
BAD	002050	B15	015362	CACHKF	002544	CSRLOO	002326	DH6	072607
BADPC	002020	B16	017000	CACHKN	002540	CSRMAP	005774	DH7	072654
BADPSW	002030	B17	017250	CACHOF=	104424	CSRNO	002150	DIAGFL	002002
BADSP	002024	B2	006010	CACHON=	104423	CSROUT	044002	DISPLA	002624
BADSTA	042614	B20	017256	CACHVE=	000114	CSRSTU=	000021	DISPRE=	000174
BADXOR	002056	B21	017274	CBCSR =	104474	CSR1S	002316	DISPTB	014770
BAD2	002052	B22	017336	CBITS	002312	CTEST	006656	DOBACK	015352
BAD3	002054	B23	017410	CBREG =	104513	CTLKVE	002144	DONE	006642
BAFPAF	015452	B24	024170	CB1CSR=	104475	DATARG=	177754	DSWR =	177570
BAFPAR	015560	B25	024174	CHECK	002310	DATBUF	002240	DT1	067132
BAKPAT	002616	B26	024374	CHKDIS=	104504	DBEMSK	002254	DT10	067252
BANK	002100	B27	024402	CHKGEN	044366	DDISP =	177570	DT11	067274
BANKIN	002102	B3	006060	CHKTAB	044474	DEENER=	104421	DT12	067302
BANKMO	046410	B30	024700	CHKTRP	040614	DETAIL	062274	DT13	067306
BANKOK	047446	B31	032410	CHK1DI=	104505	DETFLA	002216	DT14	067330
BAWPAF	015666	B32	034126	CKEND	064050	DETPSW	002214	DT16	067346
BAWPAR	016016	B33	034164	CKSWR =	104410	DETR0	002176	DT17	067376
BGTEST	036244	B34	034606	CLRCSR=	104502	DETR1	002200	DT2	067144
BITNO	002320	B35	034644	CLREX	007752	DETR2	002202	DT20	067404
BIT0 =	000001	B36	034660	CLRMEM	007642	DETR3	002204	DT22	067414
BIT1 =	000002	B37	035000	CLR1CS=	104503	DETR4	002206	DT23	067422
BIT10 =	002000	B4	006366	CMD16A	053614	DETR5	002210	DT24	067444
BIT11 =	004000	B40	035160	CMD16L=	000073	DETSP	002212	DT25	067464
BIT12 =	010000	B41	035202	CMD5B	052000	DET1	062764	DT26	067476
BIT13 =	020000	B42	036376	CMD5C	052274	DF1	067544	DT27	067504
BIT14 =	040000	B43	036536	CMD7B	052532	DF11	067667	DT3	067150
BIT15 =	100000	B44	036574	CMD7C	052606	DF13	067700	DT30	067524
BIT2 =	000004	B45	037066	CMD9B	053230	DF14	067710	DT4	067160
BIT3 =	000010	B46	037254	CMD9C	053304	DF15	067717	DT5	067170
BIT4 =	000020	B47	037452	CONFGE	002444	DF16	067726	DT6	067206
BIT5 =	000040	B5	006422	CONFIE	003654	DF2	067545	DT7	067222
BIT6 =	000100	B50	037646	CONFIG	002350	DF3	067571	DUMMY	002174
BIT7 =	000200	B51	037656	CONTFL	002220	DF4	067604	DUMPCS=	000102
BIT8 =	000400	B52	040166	CONTRL=	177746	DF5	067617	ECCDIS=	104470
BIT9 =	001000	B53	040336	CONTS	064220	DF6	067632	ECCINI=	104472
BLOCK1	050016	B54	040352	CONTS1	063576	DF7	067645	ECCTYP	005714
BLOCK2	050036	B55	040416	CONTS2	064222	DF8	067660	ECC1DI=	104471
BLOCK3	050052	B56	043434	CONTS3	056370	DF9	067662	ECC1IN=	104473
BMFLAG	002126	B57	043440	CONTT	064144	DH1	072414	EMTVEC=	000030
BOOT	047524	B6	012574	COUNT	002364	DH10	072764	EM11	070235
BOOT1	047570	B60	043600	CPERRF	061322	DH11	073062	EM12	070257
BRGOBB	036246	B61	043604	CPSAVE	061320	DH12	073100	EM13	070303
BSIZE	002370	B62	044010	CPUBIT	002104	DH13	073105	EM14	070335

SYMBOL TABLE

```
EM15    070401  E100    055132  E66     045606  HT    = 000011  LOOP    014742
EM17    070447  E101    055132  E67     045520  I       002446  LOWMAP  046344
EM19    070507  E102    062576  E7      013562  IBSAVE  061316  LSIZE   002374
EM2     067735  E103    062576  E70     052600  IIII  = 177777  LWDBE = 000060
EM20    070561  E104    062662  E71     053062  ILLCSR  014124  LWSBE = 000056
EM21    070640  E105    062756  E72     053276  IMPTES  013064  L0      004624
EM22    070674  E106    063042  E73     053570  INCBNK  047510  L1      004666
EM23    070721  E11     014274  E74     053570  INCPAT  047464  L10     005164
EM24    070750  E12     014342  E75     053644  INCRPT  047464  L100    013054
EM25    071027  E13     014372  E76     054626  INHBAN  002534  L101    013056
EM26    071054  E14     014710  E77     054626  INHECC  002532  L102    013324
EM27    071125  E15     015450  FASTCI= 177640  INTFLA  002134  L103    013324
EM29    071215  E16     017020  FATAL$  002062  INT64K  002136  L104    013224
EM3     067773  E17     017604  FCMD10  053370  INVALI= 104511  L105    013232
EM30    071277  E2      006054  FCMD11  053416  IOTVEC= 000020  L106    013324
EM32    071407  E20     017564  FCMD12  053440  JMPRL1  045740  L107    013276
EM33    071514  E21     017514  FCMD13  053460  KAMIKA  002004  L11     005172
EM35    071622  E22     017514  FCMD14  053502  KAMITE  026774  L110    013304
EM36    071707  E23     017464  FCMD15  053520  KDIAG = 000010  L111    013330
EM4     070025  E24     024352  FCMD16  053604  KDPAR0= 172360  L112    013546
EM40    071756  E25     024336  FCMD17  053646  KDPAR6= 172374  L113    013546
EM5     070073  E26     024560  FCMD18  053662  KDPAR7= 172376  L114    013536
EM50    072030  E27     024544  FIELDS  050102  KERNEL= 104417  L115    013534
EM51    072064  E3      006176  FINDBA= 000066  KERSTK= 002000  L116    013540
EM52    072134  E30     024756  FINT    007174  KFLAG   002524  L117    014264
EM53    072161  E31     032474  FIRST = 060000  KIPAR0= 172340  L12     005242
EM55    072210  E32     034402  FLIPLO  002602  KIPAR4= 172350  L120    014260
EM56    072231  E33     034370  FLIPWA  041074  KIPAR5= 172352  L121    014252
EM57    072263  E34     035152  FLUSH   015062  KIPAR6= 172354  L122    014256
EM6     070150  E35     035136  FSCMD0  050300  KIPDR0= 172300  L123    014264
EM60    072331  E36     034752  FSCMD1  050402  KMAP  = 104422  L124    014360
EM61    072373  E37     035122  FSCMD2  050512  KPFLAG  002112  L125    014430
EM62    100300  E4      006564  FSCMD3  050660  KSIZE   002372  L126    014542
EM7     070175  E40     035176  FSCMD4  051134  KSTACK  002560  L127    014742
ENASBE= 104506  E41     035222  FSCMD5  051454  LAST  = 157776  L13     005236
ENA1SB= 104507  E42     036520  FSCMD6  052372  LASTBA  002552  L130    014700
END     100354  E43     036724  FSCMD7  052400  LASTBL  002554  L131    014616
ENERGI= 104420  E44     036712  FSCMD8  052672  LASTER  002014  L132    014620
ENEXBK  047436  E45     037172  FSCMD9  053076  LBLS0 = 000607  L133    014660
ERRADD  002454  E46     037424  FSINFL  002436  LBLS1 = 000106  L134    014674
ERRGEN= 104512  E47     037622  FSPAT   052154  LBLS2 = 000601  L135    014740
ERRMAX  002550  E5      006554  FSSTAC  002302  LBLS3 = 000574  L136    014742
ERROR = 104000  E50     040132  FS1     050166  LBLS4 = 000446  L137    015056
ERRPC   002016  E51     040116  FS7FLA  002442  LBLS5 = 000450  L14     005242
ERRPSW  002026  E52     040322  FULLRE  002536  LBLS6 = 000023  L140    015166
ERRSP   002022  E53     040606  GBLENG= 000076  LCSROU= 000062  L141    015176
ERRVEC= 000004  E54     040576  GETCSR  041324  LCSRRE= 000100  L142    015346
EUFLAG  002130  E55     040564  GETDAT  054256  LCSRSA= 000076  L143    015346
EVEN    002360  E56     043506  GETDA1  054354  LEGALC= 000010  L144    015434
EXBANK  047020  E57     043506  GETDIS  060450  LF    = 000012  L145    015546
EXCMD3  051066  E6      013012  GOOD    002042  LINK1   002516  L146    015654
EXCMD4  051406  E60     043652  GOOD2   002044  LINK2   002520  L147    016004
EXIT    047632  E61     043652  GOOD3   002046  LKS   = 177546  L15     005502
EXIT2   047636  E62     044044  GTSWR = 104407  LOADBA  002426  L150    016134
E0      004642  E63     044044  HEADER  002576  LOADCS= 104425  L151    016264
E1      005666  E64     045372  HIPAT   047500  LOADER= 000064  L152    016436
E10     013546  E65     045352  HOLDLO= 000016  LOADHO  002562  L153    016566
```

SYMBOL TABLE

L154	016740	L237	023266	L324	034714	L420	042160	L502	052450
L155	017020	L24	005762	L325	034732	L421	042204	L503	052452
L156	017056	L240	023536	L326	034740	L422	042306	L504	052564
L157	017066	L241	023546	L327	034764	L423	043220	L505	053062
L16	005502	L242	023546	L33	006142	L424	043226	L506	053046
L160	017514	L243	023556	L330	035122	L425	043464	L507	053016
L161	017476	L244	024322	L331	035022	L426	043464	L51	006544
L162	017546	L245	024270	L332	035034	L427	043526	L510	053146
L163	020016	L246	024322	L333	035066	L43	006422	L511	053150
L164	020126	L247	024366	L334	035052	L430	043534	L512	053262
L165	020264	L25	005772	L335	035064	L431	043630	L513	053552
L166	020312	L250	024530	L336	035110	L432	043630	L514	053554
L167	020316	L251	024476	L337	035076	L433	043672	L515	054020
L17	005614	L252	024530	L34	006152	L434	043700	L516	054034
L170	020320	L253	024742	L340	035110	L435	044022	L517	054040
L171	020364	L254	024742	L341	035222	L436	045206	L520	054056
L172	020420	L255	025050	L342	036570	L437	045222	L521	054204
L173	020424	L256	025022	L343	036574	L44	006452	L522	054206
L174	020426	L257	025100	L344	036660	L440	045234	L523	054252
L175	020574	L260	025270	L345	036676	L441	045234	L524	054476
L176	021250	L261	025622	L346	036702	L442	045336	L525	054604
L177	022116	L262	026112	L351	037020	L443	045366	L526	055110
L2	004774	L263	026150	L352	037156	L444	045510	L527	056426
L20	005642	L264	026250	L354	037330	L445	045510	L530	056430
L200	022126	L265	026274	L355	037400	L446	045510	L531	057060
L201	022126	L266	026346	L357	037534	L447	045470	L532	057272
L202	022136	L267	026416	L36	006250	L45	006524	L533	057310
L203	022174	L27	006144	L360	037604	L450	045510	L534	057312
L204	022204	L270	026460	L362	037760	L451	045552	L535	057332
L205	022204	L271	026524	L363	040036	L452	045552	L536	057344
L206	022214	L272	026564	L364	040054	L453	045602	L537	057362
L207	022244	L273	026624	L365	040102	L454	045722	L54	006600
L21	005660	L274	026664	L37	006254	L455	047156	L540	057364
L210	022250	L275	026724	L370	040232	L456	047364	L541	057400
L211	022300	L276	027016	L371	040274	L457	047546	L542	057602
L212	022310	L277	027024	L373	040412	L46	006514	L543	057610
L213	022310	L3	005016	L374	040416	L460	047652	L544	057636
L214	022320	L30	006124	L375	040470	L461	047656	L545	057650
L215	022364	L300	027030	L376	040472	L462	047664	L546	057666
L216	022374	L301	030646	L377	040530	L463	047700	L547	057700
L217	022374	L302	030644	L4	005034	L464	047712	L55	006642
L220	022404	L303	030646	L40	006320	L465	050124	L550	057712
L221	022454	L304	032344	L400	040550	L466	050134	L551	057734
L222	022464	L305	032402	L401	040554	L467	050270	L552	060002
L223	022464	L306	032462	L405	040646	L47	006516	L553	060062
L224	022474	L31	006116	L406	040644	L470	050330	L554	060076
L225	022526	L310	034160	L407	040666	L471	050334	L555	060100
L226	022542	L311	034164	L41	006362	L472	050364	L556	060204
L227	022552	L312	034260	L410	040666	L473	050376	L557	060274
L23	005764	L313	034336	L411	041004	L474	051564	L56	011562
L230	022552	L314	034354	L412	041052	L475	051624	L560	060304
L231	022632	L315	034360	L413	041464	L476	051664	L561	060612
L232	022642	L32	006124	L414	042050	L477	052072	L562	060612
L233	022642	L320	034630	L415	042152	L5	005112	L563	061052
L234	022660	L321	034640	L416	042142	L50	006540	L564	061006
L235	023170	L322	034752	L417	042150	L500	052106	L565	061032
L236	023222	L323	034702	L42	006416	L501	052120	L566	061052

SYMBOL TABLE

```
L567    061072          MMR2  = 177576          MSG055  076513          MTPA25  035664          MT0006  021750
L57     011566          MMR3  = 172516          MSG056  076534          MTPA26  036014          MT0007  022004
L570    061264          MMTRAP  042570          MSG058  076567          MTPB03  027550          MT0010  022046
L571    061134          MMVEC = 000250          MSG061  076611          MTPB04  027702          MT0011  022102
L572    061262          MONFLG  002272          MSG062  076620          MTPB21  034460          MT0012  022160
L573    061212          MSEEDH  002572          MSG063  076640          MTPB24  035314          MT0013  022264
L574    061224          MSEEDL  002574          MSG064  076651          MTPB25  035706          MT0014  022350
L575    061262          MSGA12  100014          MSG065  076661          MTPB26  036030          MT0015  022440
L576    061272          MSGA34  076031          MSG066  076673          MTPC03  027610          MT0016  022516
L577    061626          MSGB34  076067          MSG067  076742          MTPC21  034514          MT0017  022574
L6      005172          MSG000  100226          MSG070  076751          MTPC24  035330          MT0020  022616
L60     011570          MSG001  074036          MSG073  077002          MTPC25  035746          MT0021  022706
L600    062554          MSG002  074120          MSG075  077020          MTPC26  036064          MT0022  023160
L601    062760          MSG003  074175          MSG076  077052          MTPD03  027626          MT0023  023212
L602    062764          MSG004  074302          MSG077  077073          MTPD21  034550          MT0024  023256
L603    063044          MSG005  074410          MSG079  077107          MTPD25  035612          MT0025  023522
L604    063050          MSG006  074422          MSG085  077133          MTPD26  036104          MT0026  023600
L605    064164          MSG007  074457          MSG088  077160          MTPE25  035634          MT0027  024102
L606    064254          MSG008  100174          MSG089  077176          MTP000  027400          MT0030  024566
L607    064256          MSG009  074471          MSG090  077220          MTP001  027424          MT0031  025070
L61     012402          MSG010  074503          MSG091  077234          MTP002  027456          MT0032  025260
L62     012416          MSG011  074515          MSG092  077246          MTP005  027722          MT0033  025612
L63     012542          MSG012  074603          MSG093  077262          MTP006  027756          MT0034  026000
L64     012520          MSG013  074700          MSG095  077270          MTP007  030156          MT0035  026152
L65     012532          MSG014  074702          MSG101  077300          MTP010  030256          MT0036  026264
L66     012546          MSG015  074704          MSG102  077330          MTP011  030364          MT0037  026336
L67     012776          MSG016  074706          MSG103  077357          MTP012  031162          MT0040  026404
L7      005172          MSG017  074720          MSG104  077401          MTP013  031550          MT0041  026406
L70     012776          MSG018  074731          MSG105  077403          MTP014  032264          MT0042  026450
L71     012776          MSG019  074734          MSG106  077456          MTP015  032510          MT0043  026514
L72     012654          MSG020  074740          MSG107  077474          MTP016  033254          MT0044  026554
L73     012660          MSG021  074761          MSG110  077551          MTP017  034036          MT0045  026614
L74     012776          MSG022  075547          MSG111  077615          MTP020  034114          MT0046  026654
L75     012730          MSG023  075571          MSG112  077647          MTP022  034600          MT0047  026714
L76     012776          MSG025  075605          MSG113  077664          MTP025  035346          MT0999  026760
L77     013050          MSG026  075631          MSG114  077701          MTP030  036122          MT1     017026
MAINT = 177750          MSG027  075643          MSG117  077716          MTP031  036132          MT2     017032
MAPH0 = 170202          MSG028  075660          MSG119  077730          MTP032  036210          MUT     002106
MAPKER  046706          MSG029  075674          MSG120  077737          MTP033  036242          NC      056430
MAPL0 = 170200          MSG030  075714          MSG121  077760          MTP034  036340          NEMCNT  002066
MAPL1 = 170204          MSG031  075733          MSG122  100000          MTP035  036364          NEWBAN  002304
MAPPER  044534          MSG032  075773          MSG125  100046          MTP036  036526          NEWKER  046640
MASK    002314          MSG033  076012          MSG126  100070          MTP037  036752          NEWLOA  046742
MBERR   014002          MSG035  076115          MSG127  100135          MTP041  037024          NOCH    063560
MEMDON  015010          MSG036  076120          MSG128  100154          MTP042  037176          NOERRO  002424
MFPT  = 000007          MSG037  076137          MSG129  100275          MTP043  037432          NOFSMO  002422
MJPAT   020620          MSG038  076156          MSIZE   002376          MTP044  037626          NONEM   002076
MJTEST  020514          MSG039  076174          MTA030  024600          MTP045  040142          NONEXI  042512
MKCNT   017666          MSG040  076216          MTEST   016752          MTP046  040330          NOOJ    041464
MKCONT  017046          MSG041  076242          MTLA11  030412          MTP047  040670          NOPAR   002074
MKCSRT  020204          MSG042  076267          MTLB11  030424          MTST3   012422          NORES   003664
MKFLAG  002116          MSG043  076305          MTLC11  030436          MT0000  020700          NOSCOP  002434
MKLOOP  017230          MSG046  076327          MTLD11  030532          MT0001  020760          NOSUPE  002452
MKPAT   020434          MSG047  076362          MTPA03  027510          MT0002  021100          NOTAB   002366
MKTEST  020274          MSG048  076401          MTPA04  027646          MT0003  021240          NOTRCE  060220
MMR0  = 177572          MSG049  076441          MTPA21  034430          MT0004  021472          NO22BI  002450
MMR1  = 177574          MSG051  076474          MTPA24  035254          MT0005  021614          NULLFL  002340
```

SYMBOL TABLE

```
NXTCSR  005642        PERR32= 104456        SBENT   020146        SUPDR4  002166        TSTRD1  043230
OLDCAC  002276        PERR33= 104457        SBESYN  034410        SUPDR5  002170        TSTREA= 104510
OLDCSR  002154        PERR34= 104460        SBETES  017670        SUPDR6  002172        TST1    005522
ONES    002600        PERR35= 104461        SCOPE = 000004        SUPLIM  056506        TST2    011404
PADDRE  002034        PERR36= 104462        SDPAR0= 172260        SUPSTK= 000740        TST3    011570
PAFBAF  016146        PERR37= 104463        SDPAR5= 172272        SWAPAT  002620        TST4    012556
PAFBAW  016276        PERR40= 104464        SDPAR6= 172274        SWR     002622        TST5    014742
PARBAF  016450        PERR41= 104465        SDPAR7= 172276        SWREG = 000176        TST6    015014
PARBAW  016600        PERR42= 104466        SEEDHI  002566        SW0   = 000001        TYPDS = 104405
PARCNT  002070        PERR43= 104467        SEEDLO  002570        SW1   = 000002        TYPEIT= 104401
PARITY  042406        PERXOR  057452        SELONL  002000        SW10  = 002000        TYPOC = 104402
PARTHE  002300        PFECDF  061740        SETPAT  047500        SW11  = 004000        TYPOS = 104403
PARVEC= 000114        PFECDH  061700        SHADL1  012452        SW12  = 010000        TYPS0 = 000000
PASFLG  002262        PFECDT  061730        SHUTUP  047664        SW13  = 020000        TYPS1 = 000002
PASSNO  002264        PFECEM  061644        SIPAR0= 172240        SW14  = 040000        TYPS2 = 000000
PATERR  002072        PFECWS  061634        SIPAR3= 172246        SW15  = 100000        TYPS3 = 000000
PATPLU  004606        PFLAG   002120        SIPAR5= 172252        SW2   = 000004        TYPS4 = 000000
PATTER  002110        PGMCSR  002526        SIPAR6= 172254        SW3   = 000010        TYPS5 = 000000
PCBUMP  002322        PHEBE   014004        SIPDR0= 172200        SW4   = 000020        TYPS6 = 000002
PCONFI  041352        PHYADD  002036        SIZE  = 040000        SW5   = 000040        T12A    033254
PCONFS  041652        PMEMFL  002140        SKIPKA  002006        SW6   = 000100        T12B    033276
PCONF1  041562        PROTYP  003752        SKIPMK  002336        SW7   = 000200        UDPAR0= 177660
PCONF2  041620        PSIZE   002400        SKJ     060246        SW8   = 000400        UDPAR7= 177676
PDP110  042602        PSW   = 177776        SKPERR  002064        SW9   = 001000        UIPAR0= 177640
PD1     054476        PTABLE  036732        SKUB    045602        SYSSIZ  003754        UIPAR1= 177642
PERA05  056644        PWRVEC= 000024        SKUJ    014006        TAG2$   012042        UIPAR2= 177644
PERBNK  057476        QUICK   002432        SOBK    002556        TAG3$   012076        UIPAR3= 177646
PERECC  057556        QVFLAG  002342        SOBLEN= 000056        TAG4$   027124        UIPAR4= 177650
PERRAB  057314        RANODD  036044        SOFTPA  002604        TAG70$  061744        UIPAR5= 177652
PERRAW  057242        RDCHR = 104411        SOURCE  002306        TAG71$  061754        UIPAR6= 177654
PERRA3  054044        RDDEC = 104414        SPLTCS  002236        TAG72$  061764        UIPDR0= 177600
PERRA7  057366        RDLIN = 104412        SSP   =%000006        TAG73$  062034        UNITOP  002412
PERR01= 104427        RDOCT = 104413        ST    = 177776        TAG74$  062074        UNMAP   046774
PERR02= 104430        READCS= 104426        STACK = 002000        TAG75$  062106        UNRELO  046056
PERR03= 104431        READON  002404        START   003654        TAG76$  062120        UPPFLG  002263
PERR04= 104432        REALPA  002274        START1  000300        TAG77$  062164        USERMA  046556
PERR05  056640        REFRES  035154        START2  000310        TAG78$  062172        USESTK= 000700
PERR06  056666        REFSUB  035224        START3  000200        TAG79$  062252        USP   =%000006
PERR07= 104433        REGCOP  041064        START27 024162        TAG9$   011670        WARN1   011756
PERR10= 104434        RELENT  045612        STOPOK  002414        TBG4$   027302        WARN2   027522
PERR11= 104435        RELOCA  045172        STRIPE  002362        TCFIG1  041726        WARN3   027536
PERR12= 104436        RELOC1  045626        SUBAAA  004644        TCFIG2  042066        WARN4   027562
PERR13= 104437        RESREG= 104416        SUBAAB  004774        TCFIG3  042222        WARN5   027576
PERR14= 104440        RESTAR  002612        SUBAAI  012446        TCONFI  041654        WARN6   041306
PERR15= 104441        RESVEC= 000010        SUBAAP  014166        TEMP    002430        WARN6A  041246
PERR16= 104442        RES0    050376        SUBAAR  013372        TESTAD  002406        WARN6B  041300
PERR17= 104443        RES1    050456        SUBAAS  011400        TESTMO  002546        WARN7   024140
PERR20= 104444        RES2    050624        SUCCES  002330        TIME    002334        WASDBE= 104500
PERR21= 104445        RLFLAG  002124        SUPDOA  002260        TIMEOU  042556        WASSBE= 104476
PERR22= 104446        RRFLAG  002122        SUPDO1  027030        TKVEC = 000060        WAS1DB= 104501
PERR23= 104447        RTNVAL=%000000        SUPDO2  027044        TMFLAG  002132        WAS1SB= 104477
PERR24= 104450        RWCSR   006206        SUPDO3  027206        TOOMAN  002402        WHICHC  053674
PERR25= 104451        SAVCSR  002152        SUPDO4  027222        TOTCSR  002222        WOOPEN  055704
PERR26= 104452        SAVMON  002270        SUPDR0  002156        TRACE   006204        WOOPS   055336
PERR27= 104453        SAVPAR  002266        SUPDR1  002160        TRAPVE= 000034        WOOPSA  055734
PERR30= 104454        SAVREG= 104415        SUPDR2  002162        TSTBAN  012310        WOOPUP  055522
PERR31= 104455        SBEMSK  002250        SUPDR3  002164        TSTDAT  002244        WORST   002564
```

CZMSPA0 MS11-L/M/P MEMORY DIAG. MACRO M1113 26-APR-82 09:41 PAGE 517-8
SYMBOL TABLE

```
XOCHAR  056276          $DOAGN  015242          $LF     002645          $PER10  056716          $SWREG  065662
XXDPCH  002350          $DOWN   054726          $LL     = 000105        $PER11  056746          $T      = 000610
ZEROS   002332          $DTBL   063516          $LOADC  042736          $PER12  056766          $TESTN  065644
$APTHD  065740          $ECCDI  043256          $LPADR  002606          $PER13  057010          $TKB    002630
$AUTO   002060          $ECCIN  043304          $LPERR  002610          $PER14  057030          $TKS    002626
$BANK   002011          $ECC1D  043272          $LS     = 000000        $PER15  057052          $TN     = 000007
$BASE   065714          $ECC1I  043320          $MADR1  065672          $PER16  057074          $TPB    002634
$BELL   002637          $ENASB  043332          $MADR2  065676          $PER17  057114          $TPFLG  002354
$CACHF  042720          $ENA1S  043346          $MADR3  065702          $PER20  057132          $TPS    002632
$CACHN  042674          $ENDAD  015232          $MADR4  065706          $PER21  057150          $TRAP   065754
$CBCSR  043360          $ENERG  042654          $MAIL   065640          $PER22  057170          $TRAP2  065776
$CBREG  044350          $ENV    065660          $MAMS1  065670          $PER23  057206          $TRPAD  066016
$CB1CS  043402          $ENVM   065661          $MAMS2  065674          $PER24  057224          $TSTM   065744
$CDW1   065720          $EOP    015066          $MAMS3  065700          $PER25  053762          $TSTRD  043052
$CDW2   065722          $ERFLG  002012          $MAMS4  065704          $PER26  057414          $TTYIN  064704
$CHARC  056464          $ERRGE  044100          $MBADR  065742          $PER27  057434          $TYPDS  063312
$CHKDI  043754          $ERROR  060522          $MNEW   064760          $PER30  054210          $TYPE   056152
$CHK1D  043770          $ERRTB  066262          $MSGAD  065654          $PER31  057624          $TYPEC  056300
$CKSWR  063536          $ERRTY  061324          $MSGLG  065656          $PER32  057722          $TYPEX  056466
$CLRCS  043732          $ERTTL  002614          $MSGTY  065640          $PER33  057770          $TYPOC  063110
$CLR1C  043744          $ESCAP  002356          $MSWR   064747          $PER34  060050          $TYPON  063124
$CMTAG  002000          $ETABL  065660          $MTYP1  065671          $PER35  060102          $TYPOS  063064
$CMTGE  002540          $ETEND  065740          $MTYP2  065675          $PER36  060136          $T1     = 000000
$CNTLC  064730          $EXHAL  047656          $MTYP3  065701          $PWRDN  054356          $T2     = 000607
$CNTLG  064742          $ES     = 000001        $MTYP4  065705          $PWRUP  054732          $UNIT   065652
$CNTLK  064136          $FATAL  065642          $NOTRA  066010          $QUES   002643          $UNITM  065750
$CNTLU  064735          $FILLC  002636          $NULL   002352          $R      = 177777        $USWR   065664
$CPUOP  065666          $FILLS  002353          $NWTST= 000001          $RAND   065424          $VECT1  065710
$CRLF   002644          $FS     = 000000        $OCNT   063306          $RDCHR  064256          $VECT2  065712
$DBLK   063526          $GTSWR  063712          $OCTVL  065622          $RDDEC  065142          $WASDB  043566
$DB20   065520          $HALT   061106          $OCT8 = 065626          $RDLIN  064406          $WASSB  043422
$DDW0   065724          $HALT2  066014          $OMODE  063310          $RDOCT  064772          $WAS1D  043702
$DDW1   065726          $HIBTS  065740          $OVER   060436          $READC  043032          $WAS1S  043536
$DDW2   065730          $HIOCT  065140          $OS     = 000000        $RESRE  065366          $XTSTR  060314
$DDW3   065732          $ILLUP  055330          $PASS   065646          $SAVRE  065330          $YS     = 000000
$DDW4   065734          $INVAL  044050          $PASTM  065746          $SAVR6  055334          $ZAP42  015212
$DDW5   065736          $ITEMB  002013          $PATMA  002010          $SCOPE  060166          $ZS     = 000000
$DEENE  042664          $IS     = 000001        $PER01  056506          $STN    = 000001        $$S     = 000000
$DEVCT  065650          $KERNE  042644          $PER02  056534          $SVLAD  060422          $$T     = 000573
$DEVM   065716          $KMAP   045100          $PER03  056562          $SVS    = 00000C        $$TT    = 000601
$DIDDO= 000000          $KS     = 000102        $PER04  056612          $SWR    = 163000        $OFILL  063307
$DOAGA  015346          $L      = 000107        $PER07  056674

. ABS.  100354      000
        000000      001
ERRORS DETECTED:  0

VIRTUAL MEMORY USED:  26094 WORDS  ( 102 PAGES)
DYNAMIC MEMORY:  21558 WORDS  ( 82 PAGES)
ELAPSED TIME:  00:29:08
CZMSPA.BIN,CZMSPA/-SP/CR=CZMSPA/ML,CZMSPA.P11
```

CZMSPA CREATED BY MACRO ON 26-APR-82 AT 09:54 PAGE 1

SYMBOL CROSS REFERENCE CREF

SYMBOL	VALUE	REFERENCES								
ABORTF	002142	#139-5600	*337-11069	421-13214	423-13260	*423-13276	454-14585	454-14588	454-14597	454-14600
ACFLAG	002114	#139-5589	170-6863	172-6903	182-7045	191-7343	193-7359	195-7389	197-7419	199-7456
		201-7497	203-7535	205-7580	207-7618	211-7712	238-8445	238-8474	240-8517	363-11619
		*371-11906	*371-11918	371-11920	393-12463	397-12533				
ACTFLA	002344	#139-5665	*153-5981	167-6724	173-6939	217-7908	217-7923	226-8178	226-8187	226-8202
		227-8213	229-8227	229-8237	231-8252	234-8372	253-8814	363-11613	377-12024	466-14898
ADDRES	002032	#139-5565	*160-6196	*167-6718	*273-9213	*273-9219	*273-9240	*273-9253	*275-9329	*275-9343
		*277-9417	*282-9476	*283-9560	*283-9566	*285-9658	*285-9665	*305-10119	*305-10131	*315-10328
		*315-10343	*316-10380	*317-10418	*323-10594	*323-10610	*339-11095	*421-13213	*421-13213	*421-13239
		*423-13259	*423-13259	423-13262	423-13272	*451-14478	*451-14479	*451-14484	*451-14485	*451-14490
		*451-14491	*451-14496	*451-14497	*451-14504	*451-14512	*451-14517	*451-14517	*451-14522	*451-14527
		*453-14533	*453-14538	*453-14543	*453-14548	*453-14553	*453-14558	*453-14563	*453-14568	*453-14573
		*453-14578	458-14649	*460-14677	*460-14686	473-15088	509-16386	509-16389	509-16392	509-16397
		509-16408								
ANA2	010002	#166-6449								
APTDOW	047746	153-5975	#377-12058	377-12061	*377-12063					
APTECC	002420	#139-5689	*186-7232	186-7237	186-7237	509-16391				
APTFLA	002346	#139-5666	*153-5974	186-7208	189-7325	226-8178	226-8187	226-8202	227-8213	229-8227
		229-8237	231-8252	234-8372	253-8814	363-11613	377-12024	466-14898		
APTHAN	015316	#189-7327	189-7333							
APTHLT	050014	#377-12065								
APTPAR	002416	#139-5688	*186-7229	186-7237	509-16391					
APTSIZ	002440	#139-5697	*153-5971	186-7208						
BACK	060536	#465-14825	466-14915							
BACKGN	041224	222-8115	222-8133	224-8161	232-8268	234-8335	246-8710	246-8715	#328-10835	
BAD	002050	#139-5571	*157-6102	*160-6244	*282-9490	*282-9504	*291-9784	*291-9796	*317-10417	*319-10454
		*321-10494	*321-10503	*322-10541	*322-10551	*323-10596	*323-10609	*324-10660	*324-10668	*325-10722
		*326-10779	*326-10789	*421-13216	*421-13242	*423-13262	*423-13274	*451-14480	*451-14486	*451-14493
		*451-14498	*451-14503	*451-14513	*451-14518	*451-14523	*451-14528	*453-14534	*453-14539	*453-14544
		*453-14549	*453-14554	*453-14559	*453-14564	*453-14569	*453-14574	*453-14579	456-14626	*458-14648
		*460-14678	*460-14687	*460-14688	*460-14707	*460-14714	509-16386	509-16392	509-16400	509-16402
		509-16404	509-16407	509-16409						
BADPC	002020	#139-5560	*339-11122	421-13227	454-14586	454-14598	456-14609	458-14661	460-14676	460-14685
		460-14696	465-14861	465-14862	*465-14866					
BADPSW	002030	#139-5564	*339-11123	460-14707	465-14865					
BADSP	002024	#139-5562	*339-11120	*339-11121	465-14864					
BADSTA	042614	337-11070	337-11078	339-11103	339-11107	339-11110	#339-11119	421-13227	451-14506	454-14586
		454-14598	456-14609	458-14661	460-14676	460-14685	460-14696	460-14706		
BADXOR	002056	#139-5574	*456-14626	*456-14627	509-16392					
BAD2	002052	#139-5572	*421-13243	509-16400						
BAD3	002054	#139-5573	*421-13245	*421-13246	509-16400					
BAFPAF	015452	187-7248	#193-7355	193-7380						
BAFPAR	015560	187-7249	#195-7385	195-7410						
BAKPAT	002616	#141-5742	*147-5871	232-8267	234-8334					
BANK	002100	#139-5583	*169-6739	*169-6751	169-6752	169-6754	169-6761	*169-6780	*170-6860	*170-6866
		*170-6869	*170-6873	170-6875	*172-6899	*173-6936	173-6936	*173-6938	174-6948	*182-7041
		*182-7071	182-7071	*191-7341	*191-7349	191-7349	*193-7356	*195-7386	*197-7416	*199-7453
		*201-7492	*203-7530	*205-7575	*207-7613	211-7681	211-7708	211-7745	*211-7761	213-7817
		213-7842	232-8278	232-8292	234-8345	236-8395	236-8408	*238-8443	238-8447	*238-8465
		238-8465	*238-8472	238-8475	*238-8494	238-8494	*240-8514	*240-8523	240-8523	*240-8533
		242-8551	244-8580	246-8644	255-8822	257-8855	319-10430	321-10473	321-10479	326-10763
		354-11366	361-11576	*363-11617	363-11620	*363-11630	363-11630	365-11723	*365-11724	*365-11728

SYMBOL	VALUE	REFERENCES								
		*365-11732	369-11857	371-11909	*373-11988	373-11989	386-12209	*386-12214	*386-12218	386-12221
		386-12230	*386-12245	388-12251	*388-12256	*388-12260	388-12263	388-12269	*388-12297	390-12303
		*390-12308	390-12309	390-12311	390-12330	390-12339	390-12354	390-12363	*390-12427	390-12428
		393-12440	*393-12461	*393-12467	393-12467	*393-12474	393-12475	395-12481	*395-12490	395-12491
		395-12498	*395-12504	395-12504	*395-12506	397-12511	*397-12531	*397-12537	397-12537	*397-12544
		397-12545	430-13473	*430-13474	430-13475	*430-13495	432-13514	*432-13515	432-13516	*432-13529
		458-14635	463-14792	472-15032	472-15046	472-15063	473-15070	473-15090	485-15538	
BANKIN	002102	#139-5584	*169-6757	169-6813	170-6862	172-6902	182-7043	211-7687	226-8192	246-8674
		246-8700	329-10859	355-11375	361-11554	363-11625	365-11726	365-11730	*371-11912	
BANKMO	046410	189-7324	363-11623	365-11721	#367-11774	377-12048				
BANKOK	047446	201-7495	203-7533	205-7578	207-7616	#373-11963				
BAWPAF	015666	187-7250	#197-7415	197-7439	197-7447					
BAWPAR	016016	187-7251	#199-7452	199-7476	199-7484					
BGTEST	036244	#313-10264	313-10285							
BITNO	002320	#139-5655	*291-9763	*291-9770	291-9806	*316-10362	*316-10370	316-10393	*325-10697	*325-10705
		325-10734								
BIT0	= 000001	#111-4739	158-6126	159-6159	159-6166	160-6197	160-6235	160-6261	162-6298	162-631;
		166-6521	166-6541	166-6579	166-6600	166-6610	166-6644	166-6652	169-6820	169-6839
		315-10323	315-10338	333-10935	335-10988	341-11136	341-11140	341-11147	344-11208	344-11210
		346-11231	346-11235	346-11239	346-11243	348-11278	348-11280	348-11286	348-11289	350-11313
		350-11315	350-11323	350-11325	363-11641	363-11694	363-11703	365-11742	365-11745	371-11941
		386-12224	388-12265	458-14638	462-14743	465-14852	465-14854			
BIT1	= 000002	#111-4738	151-5944	158-6133	158-6136	159-6160	159-6166	162-6298	162-6311	162-6319
		163-6363	163-6393	166-6521	166-6541	166-6548	166-6654	182-7084	182-7107	182-7119
		325-10709	335-10954	335-10989	335-11013	346-11223	346-11227	346-11239	346-11243	346-11248
		346-11253	352-11339	352-11344	371-11916					
BIT10	= 002000	#111-4729	315-10331							
BIT11	= 004000	#111-4728	158-6128	166-6581	182-7060	279-9454	283-9601	287-9703	371-11953	
BIT12	= 010000	#111-4727	155-6038	155-6039	155-6045	166-6577	182-7060	182-7090	182-7121	226-8193
		335-10952	363-11705	371-11950	390-12361	478-15221	478-15222	478-15237		
BIT13	= 020000	#111-4726	155-6038	155-6045	157-6095	157-6096	174-6950	174-6966	189-7286	246-8680
		246-8694	273-9282	279-9451	282-9487	283-9598	287-9700	321-10492	321-10501	323-10592
		326-10777	326-10787	343-11172	344-11198	354-11369	363-11627	363-11674	365-11731	365-11736
		365-11754	365-11767	375-12016	478-15221	478-15237				
BIT14	= 040000	#111-4725	160-6236	166-6629	166-6631	255-8844	297-9965	297-9966	321-10476	344-11217
		355-11401	355-11403	363-11680	365-11739	367-11798	367-11814	371-11947	400-12611	400-12627
		426-13326	426-13332	429-13429	429-13433	430-13476	432-13517			
BIT15	= 100000	#111-4724	157-6115	163-6405	166-6594	166-6620	213-7811	213-7829	213-7832	213-7836
		273-9224	324-10658	325-10721	326-10766	344-11207	344-11217	350-11302	363-11626	363-11642
		363-11680	363-11689	363-11710	365-11727	365-11739	367-11798	367-11814	406-12646	426-13326
		429-13433	460-14697	463-14797	473-15093					
BIT2	= 000004	#111-4737	154-6018	154-6019	154-6020	160-6236	162-6300	166-6523	166-6567	291-9774
		291-9792	315-10323	316-10374	325-10709	337-11072	346-11248	346-11253	352-11339	352-11344
		355-11429								
BIT3	= 000010	#111-4736	154-6018	154-6019	154-6022	157-6093	160-6211	160-6221	160-6251	160-6269
BIT4	= 000020	#111-4735	137-5526	145-582[illegible]	145-5829	157-6093	166-6594	166-6620	189-7320	213-7811
		213-7836	273-9224	321-1048?	344-11207	348-11267	348-11287	460-14697		
BIT5	= 000040	#111-4734	153-5967	166-6514	166-6585	166-6615	189-7320	363-11685	363-11693	365-11749
		365-11752	371-11956	375-12007	377-12052					
BIT6	= 000100	#111-4733	172-6919	331-10875	331-10912	335-11037	363-11645	371-11913	390-12350	390-12423
		393-12456	393-12472	397-12527	397-12542					
BIT7	= 000200	#111-4732	153-5970	154-5989	154-5990	170-6871	273-9273	279-9440	283-9587	287-9689

SYMBOL	VALUE	REFERENCES
		331-10874 344-11193 363-11641 363-11645 371-11935 390-12349 393-12455 397-12526
BIT8	= 000400	#111-4731 184-7157 363-11644 371-11929 486-15608
BIT9	= 001000	#111-4730 174-6976 174-6991 184-7158 371-11932
BLOCK1	050016	169-6748 220-8022 220-8044 220-8070 222-8094 222-8097 222-8123 222-8143 232-8296
		232-8299 232-8303 232-8306 234-8355 236-8412 236-8420 238-8438 240-8507 244-8609
		246-8678 328-10848 344-11203 #379-12075
BLOCK2	050036	#379-12080
BLOCK3	050052	222-8098 222-8099 222-8124 222-8144 234-8356 234-8357 236-8413 236-8415 242-8553
		246-8647 363-11681 365-11740 367-11799 367-11815 #379-12084 430-13479 430-13482
BMFLAG	002126	#139-5594 197-7421 199-7458 203-7537 207-7620 363-11619 *371-11907 *371-11927 *371-11943
		*371-11946
BOOT	047524	#375-11993 483-15485 486-15612
BOOT1	047570	375-12001 375-12006 #375-12008
BRGOBB	036246	#313-10265 313-10269 313-10303
BSIZE	002370	#139-5678 *184-7171 *184-7172 *184-7173 *184-7174 184-7175
CACHKF	002544	#141-5719 *145-5821 341-11154 382-12159
CACHKN	002540	#141-5718 *145-5823 154-6026 *154-6026 *154-6027 341-11144 341-11146 341-11151 380-12106
		382-12157 382-12159 382-12162 399-12575 *399-12575 *399-12576 399-12581 *399-12581 425-13296
		429-13453 434-13591 434-13651
CACHOF	= 104424	#110-4615 154-6025 162-627° 166-6510 167-6708 169-6770 169-6787 169-6804 169-6830
		172-6897 211-7724 213-7795 255-8846 257-8883 282-9473 291-9756 296-9927 316-10356
		317-10409 321-10472 322-10525 323-10567 324-10646 325-10687 326-10762 380-12110 399-12574
		434-13595
CACHON	= 104423	#110-4614 147-5890 163-6413 166-6686 167-6692 167-6711 169-6778 169-6791 169-6812
		169-6835 173-6944 211-7736 213-7847 213-7855 255-8848 257-8885 282-9512 291-9810
		296-9905 316-10397 317-10421 321-10514 322-10557 323-10637 324-10677 325-10739 326-10797
		399-12582 425-13299
CACHVE	= 000114	#111-4754 111-4755
CBCSR	= 104474	#110-4658
CBITS	002312	#139-5652 *166-6550 *166-6553 166-6555
CBREG	= 104513	#110-4673 174-6977 174-6992 282-9477 291-9759 291-9787 316-10358 321-10483 321-10511
		322-10526 322-10544 324-10647 324-10663 325-10692 326-10767
CB1CSR	= 104475	#110-4659 166-6531 273-9204 275-9314 277-9409 283-9549 285-9648 305-10108 317-10412
		317-10414 319-10437 319-10443 319-10459 323-10570
CHECK	002310	#139-5651 *166-6481 *273-9279 *273-9280 *279-9445 *279-9447 *279-9449 *283-9592 *283-9594
		*283-9596 *287-9694 *287-9696 *287-9698 *316-10381 *317-10411 *317-10413 *319-10436 *319-10442
		346-11247 346-11252 *357-11460 509-16397
CHKDIS	= 104504	#110-4666
CHKGEN	044366	273-9192 275-9305 277-9398 283-9536 285-9636 303-10048 #357-11434
CHKTAB	044474	357-11444 #359-11465
CHKTRP	040614	325-10719 #325-10743
CHK1DI	= 104505	#110-4667 174-6955 174-6967 174-6982 174-6997 421-13230
CKEND	064050	483-15466 483-15468 483-15470 #483-15499
CKSWR	= 104410	#110-4598 462-14735 465-14824 466-14887
CLRCSR	= 104502	#110-4664 162-6328 163-6406 #217-7911 238-8433 240-8526 390-12370 #393-12447 #397-12518
CLREX	007752	164-6426 164-6440 #164-6444
CLRMEM	007642	162-6333 163-6414 #164-6426
CLR1CS	= 104503	#110-4665 166-6530 166-6539 166-6676 174-6959 174-6972 174-6987 174-7001 174-7015
		213-7803 213-7827 213-7851 273-9170 273-9231 273-9244 273-9290 275-9297 275-9317
		275-9372 277-9378 277-9392 277-9411 279-9462 282-9510 283-9530 283-9551 283-9610
		285-9652 287-9712 291-9778 291-9809 303-10083 315-10336 316-10396 319-10445 319-10461
		321-10489 321-10513 322-10535 323-10635 324-10655 324-10676 325-10726 325-10738 326-10774

CZMSPA CREATED BY MACRO ON 26-APR-82 AT 09:54 PAGE 4
SYMBOL CROSS REFERENCE CREF

SYMBOL	VALUE	REFERENCES
		326-10794 421-13236
CMD16A	053614	400-12600 #400-12623
CMD5B	052000	#390-12354 390-12379
CMD5C	052274	390-12335 390-12346 #390-12422
CMD7B	052532	#393-12461 393-12469
CMD7C	052606	393-12452 #393-12471
CMD9B	053230	#397-12531 397-12539
CMD9C	053304	397-12523 #397-12541
CONFGE	002444	#139-5699 *166-6602 *166-6646 *169-6821 *169-6840 *182-7067 *182-7146
CONFIE	003654	#143-5785 151-5940
CONFIG	002650	#143-5782 151-5938 *154-5989 *154-5990 166-6514 *166-6570 *166-6574 *166-6577 *166-6581
		*166-6585 166-6586 *166-6586 166-6587 *166-6587 *166-6600 *166-6601 166-6603 *166-6603
		166-6604 *166-6604 166-6613 *166-6613 166-6614 *166-6614 *166-6615 *166-6644 *166-6645
		*167-6720 169-6762 *169-6820 *169-6822 *169-6839 *169-6843 *170-6871 172-6912 *172-6919
		173-6924 174-6976 174-6991 *174-7018 *174-7019 182-7046 182-7054 *182-7060 *182-7061
		182-7083 182-7090 182-7107 182-7119 182-7121 184-7156 184-7157 184-7158 184-7178
		189-7285 211-7688 211-7699 226-8193 246-8680 *246-8694 246-8701 315-10331 329-10860
		333-10935 335-10952 335-10954 335-10960 335-10981 335-10983 335-11012 335-11033 335-11037
		*354-11369 *363-11626 *363-11627 363-11641 363-11642 363-11643 363-11644 363-11645 363-11665
		*363-11674 363-11699 363-11705 363-11707 *365-11727 *365-11731 *365-11736 365-11753 371-11913
		371-11917 371-11922 371-11929 371-11932 371-11935 371-11941 371-11947 371-11950 371-11953
		371-11956 390-12314 390-12341 390-12361 395-12493 395-12499 *400-12611 *400-12627 *458-14638
		*458-14639 *458-14641 458-14642
CONTFL	002220	#139-5625 *187-7243 *193-7370 *195-7400 *197-7432 *199-7469 *201-7503 *203-7543 *205-7586
		*207-7626 209-7662 *211-7759
CONTRL	= 177746	#111-4760 145-5817 154-6005 *154-6018 *154-6019 154-6020 154-6022 *341-11146 *341-11147
		*341-11154 380-12108 *382-12164 425-13298 *429-13455 434-13593 *434-13653 509-16398
CONTS	064220	483-15459 #485-15547
CONTS1	063576	483-15444 #483-15449 485-15554
CONTS2	064222	#485-15549 485-15550 485-15556
CONTS3	056370	#434-13629 434-13630 434-13636
CONTT	064144	462-14738 478-15187 483-15454 #485-15522
COUNT	002364	#139-5676 *296-9906 296-9909 *296-9909 *296-9910 296-9911 *296-9918 *296-9925 297-9930
		*297-9930 *297-9931 297-9932 *297-9951
CPERRF	061322	*145-5857 *145-5859 462-14740 465-14849 #466-14919
CPSAVE	061320	*462-14742 462-14743 *465-14851 465-14852 #466-14918 470-15011
CPUBIT	002104	#139-5585 *151-5944 167-6720 169-6822 169-6843 184-7156 184-7178 335-10981 363-11643
		371-11917 390-12314
CPUERR	= 177766	#111-4766 *145-5860 *151-5956 *164-6446 *167-6721 *169-6781 *382-12176 *382-12180 *384-12192
		*384-12204 *386-12246 *388-12298 *462-14773 *466-14897 *476-15179 509-16390 509-16398
CR	= 000015	#111-4694 434-13642
CRLF	= 000200	#111-4695 434-13571 513-16457 517-16503 517-16504 517-16505 517-16507 517-16509 517-16510
		517-16511 517-16512 517-16513 517-16519 517-16523 517-16524 517-16525 517-16526 517-16527
		517-16528 517-16529 517-16530 517-16531 517-16532 517-16533 517-16534 517-16535 517-16536
		517-16537 517-16538 517-16539 517-16540 517-16541 517-16542 517-16543 517-16545 517-16546
		517-16547 517-16548 517-16549 517-16550 517-16551 517-16552 517-16553 517-16554 517-16555
		517-16556 517-16559 517-16560 517-16561 517-16562 517-16563 517-16564 517-16565 517-16566
		517-16567 517-16567 517-16568 517-16570 517-16573 517-16574 517-16575 517-16579 517-16581
		517-16583 517-16584 517-16585 517-16586 517-16586 517-16587 517-16588 517-16589 517-16590
		517-16591 517-16592 517-16597 517-16598 517-16599 517-16599 517-16601 517-16602 517-16603
		517-16604 517-16605 517-16606 517-16607 517-16608 517-16609 517-16610 517-16613 517-16617
		517-16618 517-16619 517-16620 517-16620 517-16620 517-16621 517-16622 517-16622

SYMBOL CROSS REFERENCE CREF

SYMBOL	VALUE	REFERENCES
CSR	002146	#139-5602 *160-6209 *160-6219 *160-6245 *160-6267 166-6536 166-6620 *174-6950 *174-6966
		*174-6978 *174-6980 *174-6993 *174-6995 213-7811 213-7836 273-9224 273-9224 *282-9478
		*282-9487 282-9488 282-9490 *291-9775 *291-9781 291-9782 291-9784 *291-9789 291-9793
		291-9796 *315-10323 *315-10338 *316-10375 319-10448 *321-10486 *321-10492 321-10493 321-10494
		*321-10498 *321-10501 321-10502 321-10503 *322-10532 *322-10538 322-10539 322-10541 *322-10547
		322-10548 322-10551 *323-10580 *323-10588 323-10591 *323-10604 323-10607 323-10609 *324-10650
		324-10658 324-10660 *324-10666 324-10667 324-10668 *324-10674 *325-10710 325-10721 325-10722
		*326-10768 *326-10771 *326-10777 326-10778 326-10779 *326-10787 326-10788 326-10789 *343-11172
		343-11173 *343-11181 344-11207 344-11207 *344-11218 *346-11223 *346-11227 *346-11231 *346-11235
		*346-11239 *346-11243 *346-11247 *346-11248 *346-11252 *346-11253 348-11267 348-11287 350-11302
		350-11321 *352-11330 *352-11334 *352-11339 *352-11344 355-11392 *355-11428 *355-11429 *384-12195
		*421-13229 421-13235 426-13343 *428-13410 460-14697 478-15190 478-15199 *478-15205 509-16395
		509-16396 509-16397 509-16405 509-16406 509-16409
CSRADD	= 172100	#113-4971 157-6085 *162-6300 *162-6303 *162-6305 *162-6308 162-6310 *163-6353 163-6368
		*163-6385 163-6398 *166-6523 *166-6526 *166-6624 166-6626 *166-6629 166-6630 *166-6631
		*343-11173 343-11181 344-11187 *355-11401 355-11402 *355-11403
CSRCAS	020174	211-7733 #215-7861
CSRFBA	002230	#139-5630 *211-7682 211-7707 *211-7752
CSRFIR	002224	#139-5628 *211-7707 211-7713 211-7715 *211-7749 211-7749
CSRHOL	002522	#139-5708 *211-7692 *211-7702 211-7704
CSRINC	002324	#139-5657
CSRINF	002456	#139-5704 *157-6092 *157-6093 157-6101 157-6102 *158-6126 *158-6133 *158-6136 159-6158
		159-6159 159-6160 159-6166 160-6197 *160-6211 *160-6221 160-6235 *160-6251 160-6261
		*160-6269 162-6298 162-6311 162-6319 163-6363 163-6393 166-6521 166-6541 166-6548
		166-6571 166-6652 166-6654 182-7038 *182-7049 182-7077 182-7079
CSRINT	002234	#139-5632 *211-7684 *211-7698 211-7751
CSRLAS	002226	#139-5629 *211-7713 *211-7714 211-7740
CSRLBA	002232	#139-5631 *211-7683 *211-7696 211-7749
CSRLOO	002326	#139-5658 *211-7686 *211-7697 211-7755
CSRMAP	005774	157-6117 #159-6140
CSRNO	002150	#139-5603 *160-6195 *162-6286 *162-6292 *163-6347 *163-6360 *163-6379 *163-6390 *166-6502
		*166-6509 166-6623 *172-6915 172-6916 172-6917 *173-6927 *211-7704 *211-7705 *246-8705
		246-8706 *329-10864 343-11162 343-11168 343-11170 343-11180 344-11188 344-11194 344-11196
		*348-11263 *348-11273 348-11273 *350-11298 *350-11308 350-11308 *354-11353 *354-11359 354-11359
		355-11400 380-12109 *382-12156 390-12338 *390-12345 390-12369 *390-12371 *390-12430 *406-12648
		*406-12662 425-13302 *426-13339 *426-13346 426-13346 *428-13407 *428-13414 428-13414 *429-13449
		478-15190 *478-15195 *478-15203 478-15203 *478-15205 509-16406
CSROUT	044002	346-11224 346-11232 346-11240 346-11249 352-11331 352-11340 #354-11349
CSR1S	002316	#139-5654 *160-6202 *160-6203 160-6213 160-6217 160-6218
CTEST	006656	157-6118 #162-6279
CTLKVE	002144	#139-5601 *211-7731 *211-7732 *217-7920 *217-7921 *219-7972 *219-7973 483-15463
DATARG	= 177754	#111-4763 154-6009
DATBUF	002240	#139-5634 *267-9037 *267-9038 267-9039 267-9040 267-9042 267-9047 267-9051 *267-9053
		*267-9053 *267-9056 *267-9057 267-9058 267-9059 267-9061 267-9066 267-9070 *267-9072
		*267-9072 *273-9177 *273-9178 273-9183 273-9184 273-9265 *273-9267 *273-9267 *275-9298
		*275-9299 275-9302 275-9303 275-9362 *275-9364 *275-9364 *277-9380 *277-9381 277-9386
		277-9387 *283-9518 *283-9519 283-9524 283-9525 *285-9619 *285-9620 285-9625 285-9626
		451-14514 451-14519
DBEMSK	002254	#139-5637 *277-9384 *277-9385 277-9393 277-9395 277-9403 277-9405 279-9425 *279-9427
		*279-9427 *279-9444 279-9448 *279-9450 279-9451 *279-9456 *279-9457 *283-9522 *283-9523
		283-9531 283-9533 283-9541 283-9543 283-9573 *283-9575 *283-9575 *283-9591 283-9595
		*283-9597 283-9598 *283-9603 *283-9604 *285-9623 *285-9624 285-9631 285-9633 285-9641

SYMBOL	VALUE	REFERENCES
		285-9643 287-9675 *287-9677 *287-9677 *287-9693 287-9697 *287-9699 287-9700 *287-9705 *287-9706
DDISP	= 177570	#111-4688 141-5747 151-5951
DEENER	= 104421	#110-4611 155-6033 375-12004 377-12049
DETAIL	062274	470-15002 #476-15150 476-15182
DETFLA	002216	#139-5624 466-14894 470-15001 *476-15151 476-15152 476-15154 *478-15252
DETPSW	002214	#139-5623 509-16394
DETR0	002176	#139-5616 *476-15159 476-15168 509-16394
DETR1	002200	#139-5617 476-15160 509-16394
DETR2	002202	#139-5618 509-16394
DETR3	002204	#139-5619 509-16394
DETR4	002206	#139-5620 509-16394
DETR5	002210	#139-5621 509-16394
DETSP	002212	#139-5622 509-16394
DET1	062764	478-15220 #478-15237
DF1	067544	508-16318 #511-16413
DF11	067667	500-16115 502-16182 502-16187 502-16192 504-16214 504-16219 504-16234 508-16328 #511-16422
DF13	067700	502-16207 506-16286 508-16358 508-16363 508-16383 #511-16423
DF14	067710	506-16301 508-16323 508-16343 508-16353 508-16368 508-16373 #511-16424
DF15	067717	#511-16425
DF16	067726	508-16333 508-16338 508-16348 #511-16426
DF2	067545	500-16120 500-16135 500-16140 500-16145 500-16150 502-16167 502-16197 504-16224 504-16229 504-16239 504-16244 506-16261 506-16306 508-16313 508-16378 #511-16414
DF3	067571	502-16157 504-16254 506-16271 506-16276 506-16291 506-16296 #511-16415
DF4	067604	502-16162 #511-16416
DF5	067617	502-16172 502-16202 506-16281 #511-16417
DF6	067632	502-16177 #511-16418
DF7	067645	504-16249 #511-16419
DF8	067660	506-16266 #511-16420
DF9	067662	500-16125 500-16130 #511-16421
DH1	072414	500-16118 #515-16476
DH10	072764	502-16165 #515-16483
DH11	073062	504-16227 #515-16484
DH12	073100	504-16222 #515-16485
DH13	073105	500-16113 502-16180 502-16185 502-16190 504-16212 504-16217 504-16232 508-16326 #515-16486
DH14	073202	504-16237 #515-16488
DH15	073261	504-16247 #515-16489
DH16	073400	504-16252 506-16264 #515-16491
DH19	073415	#515-16492
DH2	072451	508-16311 #515-16477
DH2C	073422	506-16284 508-16356 508-16361 508-16381 #515-16493
DH23	073501	502-16205 #515-16494
DH24	073560	506-16299 508-16321 508-16341 508-16351 508-16366 508-16371 #515-16495
DH25	073627	500-16148 506-16259 #515-16496
DH26	073665	502-16195 #515-16497
DH27	073703	508-16316 #515-16498
DH3	072474	500-16123 500-16128 #515-16478
DH30	073757	508-16331 508-16336 508-16346 #515-16499
DH5	072530	500-16133 500-16138 500-16143 504-16242 508-16376 #515-16479
DH6	072607	506-16304 #515-16480
DH7	072654	502-16155 502-16160 502-16170 502-16175 502-16200 506-16269 506-16274 506-16279 506-16289 506-16294 #515-16481

SYMBOL	VALUE	REFERENCES
DIAGFL	002002	#139-5550 *233-8324 *233-8326 296-9923
DISPLA	002624	#141-5747 *151-5951 *151-5958 429-13447 463-14802
DISPRE	= 000174	#113-4967 151-5958 *429-13448 *463-14803
DISPTB	014770	187-7246 #187-7248
DOBACK	015352	187-7257 187-7268 #191-7339
DONE	006642	160-6261 #160-6271
DSWR	= 177570	#111-4687 141-5746 151-5950
DT1	067132	500-16119 #509-16386
DT10	067252	502-16166 #509-16394
DT11	067274	504-16228 #509-16395
DT12	067302	504-16223 #509-16396
DT13	067306	500-16114 502-16181 502-16186 502-16191 504-16213 504-16218 504-16233 508-16327 #509-16397
DT14	067330	*145-5824 *145-5836 *145-5842 504-16238 #509-16398
DT16	067346	504-16248 #509-16399
DT17	067376	504-16253 506-16265 #509-16401
DT2	067144	#509-16387
DT20	067404	508-16312 #509-16402
DT22	067414	#509-16403
DT23	067422	502-16206 506-16285 508-16357 508-16362 508-16382 #509-16404
DT24	067444	506-16300 508-16322 508-16342 508-16352 508-16367 508-16372 #509-16405
DT25	067464	500-16149 506-16260 #509-16406
DT26	067476	502-16196 #509-16407
DT27	067504	508-16317 #509-16408
DT3	067150	500-16124 #509-16388
DT30	067524	508-16332 508-16337 508-16347 #509-16409
DT4	067160	500-16129 #509-16389
DT5	067170	*145-5835 *145-5841 500-16134 500-16139 500-16144 504-16243 508-16377 #509-16390
DT6	067206	506-16305 #509-16391
DT7	067222	502-16156 502-16161 502-16171 502-16176 502-16201 506-16270 506-16275 506-16280 506-16290 506-16295 #509-16392
DUMMY	002174	#139-5614 509-16392 509-16392 509-16393 509-16393 509-16393 509-16393 509-16397 509-16397 509-16399 509-16399 509-16400 509-16400 509-16401 509-16403 509-16404 509-16404 509-16404 509-16404 509-16404 509-16405 509-16405 509-16405 509-16405 509-16405 509-16408 509-16408 509-16408 509-16408 509-16409 509-16409 509-16409
ECCDIS	= 104470	#110-4654 217-7909 240-8512 390-12367 393-12445 397-12516
ECCINI	= 104472	#110-4656 167-6693 #167-6727 #173-6942 211-7758 #217-7926 240-8529 240-8536 240-8540 375-12000 377-12041 #421-13253
ECCTYP	005714	157-6097 #158-6125
ECC1DI	= 104471	#110-4655 174-7011 213-7796 213-7819 273-9202 273-9209 273-9286 275-9312 275-9367 277-9407 279-9459 283-9546 283-9606 285-9646 287-9708 303-10088 305-10096 305-10106 325-10714
ECC1IN	= 104473	#110-4657 213-7846 213-7854 282-9493 282-9496 303-10053 305-10099
EMTVEC	= 000030	#111-4749 *147-5876 *147-5877
EM11	070235	502-16154 502-16159 502-16169 502-16174 #513-16441
EM12	070257	502-16164 #513-16442
EM13	070303	502-16179 508-16355 #513-16443
EM14	070335	502-16184 #513-16444
EM15	070401	502-16189 #513-16445
EM17	070447	502-16199 #513-16446
EM19	070507	504-16211 508-16350 #513-16447
EM2	067735	500-16117 #513-16435
EM20	070561	504-16216 508-16340 508-16345 508-16365 #513-16448

SYMBOL CROSS REFERENCE CREF

SYMBOL		VALUE	REFERENCES
EM21		070640	504-16226 #513-16449
EM22		070674	504-16231 508-16370 508-16380 #513-16450
EM23		070721	504-16241 #513-16451
EM24		070750	500-16112 #513-16452
EM25		071027	504-16246 508-16330 #513-16453
EM26		071054	504-16251 #513-16454
EM27		071125	506-16263 #513-16455
EM29		071215	506-16273 508-16325 #513-16456
EM3		067773	500-16122 #513-16436
EM30		071277	506-16278 508-16360 #513-16457
EM32		071407	506-16288 #513-16459
EM33		071514	506-16293 #513-16460
EM35		071622	506-16268 #513-16461
EM36		071707	506-16303 #513-16462
EM4		070025	500-16127 #513-16437
EM40		071756	508-16310 #513-16463
EM5		070073	500-16132 #513-16438
EM50		072030	502-16204 #513-16464
EM51		072064	506-16298 #513-16465
EM52		072134	506-16258 #513-16466
EM53		072161	500-16147 #513-16467
EM55		072210	502-16194 #513-16468
EM56		072231	508-16315 #513-16469
EM57		072263	508-16335 #513-16470
EM6		070150	500-16137 #513-16439
EM60		072331	506-16283 #513-16471
EM61		072373	508-16320 #513-16472
EM62		100300	508-16375 #517-16623
EM7		070175	500-16142 #513-16440
ENASBE	=	104506	#110-4668 167-6725 173-6940 217-7924 421-13251
ENA1SB	=	104507	#110-4669 303-10057 303-10090 305-10101 #325-10716
END		100354	#517-16631
ENERGI	=	104420	#110-4610 155-6060 189-7321
ENEXBK		047436	371-11928 371-11951 371-11957 #371-11959
ERRADD		002454	#139-5703 213-7814 213-7839 *355-11422
ERRGEN	=	104512	#110-4672 213-7813 213-7838 273-9230 273-9256 279-9422 315-10334
ERRMAX		002550	#141-5721 458-14642
ERROR	=	104000	#111-4682 154-5994 154-6013 154-6016 157-6103 160-6210 160-6220 160-6247 #160-6249
			160-6268 167-6715 167-6719 174-7020 186-7238 273-9225 275-9330 277-9418 282-9484
			282-9491 282-9505 291-9785 291-9797 303-10062 305-10120 305-10132 315-10329 315-10344
			316-10384 317-10419 321-10496 321-10506 322-10542 322-10552 323-10598 323-10612 324-10661
			324-10670 325-10724 325-10746 325-10751 326-10782 326-10792 337-11079 339-11104 339-11108
			339-11111 382-12175 384-12191 384-12200 421-13248 454-14589 #454-14591 454-14601 #454-14603
			456-14612 458-14659 458-14664 458-14667 458-14670 460-14681 460-14692 460-14698 #460-14700
			460-14709 460-14716 462-14745 476-15170 476-15176
ERRPC		002016	#139-5559 *465-14839 *465-14840 465-14843 *465-14862 *465-14863 465-14868 468-14936 470-15011
			509-16386 509-16387 509-16388 509-16389 509-16390 509-16391 509-16392 509-16395 509-16397
			509-16399 509-16401 509-16402 509-16403 509-16404 509-16405 509-16406 509-16407 509-16408
			509-16409
ERRPSW		002026	#139-5563 *465-14842 *465-14865 476-15167
ERRSP		002022	#139-5561 *465-14841 *465-14864 476-15166
ERRVEC	=	000004	#111-4742 *147-5886 *147-5887 462-14765 *462-14766 *462-14768 *462-14774

SYMBOL VALUE REFERENCES

```
EUFLAG    002130  #139-5595   222-8076   355-11382  355-11397
EVEN      002360  #139-5674  *296-9894   296-9895  *297-9957   297-9957
EXBANK    047020   170-6861   172-6901   182-7042   191-7342   193-7358   195-7388   197-7418   199-7455   201-7494
                   203-7532   205-7577   207-7615   238-8444   238-8473   240-8515   240-8534   363-11618  365-11725
                   365-11729 #371-11883  390-12332  390-12429  393-12462  393-12476  397-12532  397-12546
EXCMD3    051066   386-12235   386-12238  386-12242 #386-12244
EXCMD4    051406   388-12278   388-12282 #388-12296
EXIT      047632  #377-12022   462-14756  466-14900   486-15611
EXIT2     047636  #377-12024
E1        005666  #157-6111
E2        006054  #159-6155
E3        006176  #159-6176
E31       032474  #282-9508
E32       034402  #291-9808
E33       034370  #291-9806
E4        006564  #160-6260
E43       036724  #316-10395
E44       036712  #316-10393
E45       037172  #319-10460
E46       037424  #321-10512
E47       037622  #322-10556
E5        006554  #160-6259
E50       040132  #323-10634
E51       040116  #323-10632
E52       040322  #324-10675
E53       040606  #325-10737
E54       040576  #325-10736
E55       040564  #325-10734
FASTCI  = 177640  #111-4796   169-6776   169-6810   255-8847   344-11204  379-12076
FATALS    002062  #139-5576  *167-6715  *167-6719  *303-10062 *337-11079 *339-11104 *339-11108 *339-11111  466-14898
                   468-14953
FCMD10    053370   380-12130 #399-12551
FCMD11    053416   380-12131 #399-12561
FCMD12    053440   380-12132 #399-12566
FCMD13    053460   380-12133 #399-12572
FCMD14    053502   380-12134 #399-12579
FCMD15    053520   380-12139 #400-12598
FCMD16    053604   380-12140 #400-12618
FCMD17    053646   380-12141 #402-12632
FCMD18    053662   380-12142 #404-12638
FIELDS    050102  #380-12097   483-15451
FINT      007174   162-6332  #163-6339
FIRST   = 060000  #113-4974   169-6765   169-6798   169-6828   172-6900   211-7682   219-7965   219-7966   220-8014
                   220-8027   220-8051   222-8080   222-8116   222-8134   224-8162   232-8273   234-8339   236-8389
                   238-8448   238-8477   240-8519   242-8553   242-8558   242-8559   244-8585   244-8615   244-8622
                   246-8647   246-8650   246-8651   246-8652   246-8653   246-8670   249-8751   282-9476   289-9723
                   296-9907   296-9926   297-9962   297-9964   297-9965   315-10322  319-10434  323-10594  323-10610
                   328-10838 *377-12061 *377-12062 *377-12063  386-12223  388-12264  390-12359  390-12360  430-13477
                   430-13478  430-13479 *430-13480 *430-13481  430-13482  430-13484 *432-13518 *432-13519  432-13524
                   473-15089
FLIPLO    002602  #141-5734  *147-5865   222-8106  *327-10810 *327-10811  327-10812  327-10814  327-10816
FLIPWA    041074   222-8079  #327-10808
```

```
SYMBOL  VALUE           REFERENCES
FLUSH     015062        #187-7273
FSCMD0    050300        #380-12120 #382-12147
FSCMD1    050402         380-12121 #382-12169
FSCMD2    050512         380-12122 #384-12184
FSCMD3    050660         380-12123 #386-12208
FSCMD4    051134         380-12124 #388-12250
FSCMD5    051454         380-12125 #390-12302
FSCMD6    052372         380-12126 #391-12433
FSCMD7    052400         380-12127 #393-12439
FSCMD8    052672         380-12128 #395-12480
FSCMD9    053076         380-12129 #397-12510
FSINFL    002436        #139-5696  *189-7284
FSPAT     052154         390-12377 #390-12381
FSSTAC    002302        #139-5648  *382-12171  382-12179 *384-12186  384-12203 *390-12304  390-12422 *393-12441  393-12471
                        *397-12512  397-12541
FS1       050166        #380-12112  380-12118  380-12144
FS7FLA    002442        #139-5698   238-8467
FULLRE    002536        #139-5713  *240-8502  *240-8530  *240-8541   363-11657
GBLENG  = 000076         246-8647   246-8650  #313-10304
GETCSR    041324         227-8220   231-8259   247-8722   248-8737   249-8749   250-8760  #329-10854
GETDAT    054256        #423-13267  454-14585  454-14597
GETDA1    054354        *423-13269  423-13277 #423-13280
GETDIS    060450         255-8823   257-8856   462-14787 #463-14791  465-14828
GOOD      002042        #139-5568  *160-6208  *160-6218  *160-6266  *282-9480   282-9488  *282-9503  *291-9783  *291-9795
                        *317-10410 *319-10455 *321-10495 *321-10504 *322-10540 *322-10550 *323-10595 *323-10608 *324-10649
                         324-10667 *324-10673 *326-10780 *326-10790 *421-13220 *421-13222 *421-13240 *423-13258 *451-14481
                        *451-14487 *451-14492 *451-14499 *451-14502 *451-14509 *451-14514 *451-14519 *451-14524 *451-14529
                        *453-14535 *453-14540 *453-14545 *453-14550 *453-14555 *453-14560 *453-14565 *453-14570 *453-14575
                        *453-14580  456-14625 *458-14650 *458-14652 *460-14679 *460-14689 *460-14708 *460-14713  509-16386
                         509-16392  509-16399  509-16402  509-16404  509-16406  509-16409
GOOD2     002044        #139-5569  *456-14615 *456-14615  509-16399
GOOD3     002046        #139-5570  *456-14616 *456-14620  509-16399
GTSWR   = 104407        #110-4597   155-6053
HEADER    002576        #141-5732  *147-5866  *187-7263  *187-7265  *191-7344  *191-7347  *209-7656  *209-7671  *211-7727
                        *273-9239  *273-9242  *273-9252  *273-9255  *275-9328  *275-9331  *275-9342  *275-9345  *277-9416
                        *277-9419  *282-9489  *282-9502  *291-9794  *305-10121 *305-10133 *321-10505 *322-10549 *323-10597
                        *323-10611 *324-10659 *324-10669 *325-10723 *325-10745 *325-10750 *326-10781 *326-10791 *390-12353
                        *393-12459 *397-12529 *458-14655 *458-14672 *460-14680 *460-14682 *460-14691 *460-14693 *460-14715
                        *460-14717 *466-14910  468-14951  468-14963 *470-14999  476-15156 *476-15157  476-15172 *476-15173
                        *476-15180
HIPAT     047500         205-7573   207-7611  #373-11982
HT      = 000011        #111-4692   434-13568
I         002446        #139-5700  *149-5923  *149-5929   149-5929  *166-6517  *166-6519   166-6567   166-6575   166-6579
                         166-6606   166-6610   166-6656   166-6661   166-6663   166-6667  *184-7169  *184-7175   184-7186
                        *184-7186   184-7200  *211-7726   211-7729  *211-7735   211-7735  *238-8442   238-8452   238-8458
                        *238-8466   238-8466  *238-8471   238-8481   238-8487  *238-8495   238-8495
IBSAVE    061316        *465-14822  465-14847 *465-14855 *465-14858  466-14913 #466-14917
IIII    = 177777         157-6111   157-6111   157-6111  #157-6111   159-6155   159-6155   159-6155  #159-6155   159-6176
                         159-6176   159-6176  #159-6176   160-6259   160-6259   160-6259  #160-6259   160-6260   160-6260
                         160-6260  #160-6260   282-9508   282-9508   282-9508  #282-9508   291-9806   291-9806   291-9806
                        #291-9806   291-9808   291-9808   291-9808  #291-9808   316-10393  316-10393  316-10393 #316-10393
                         316-10395  316-10395  316-10395 #316-10395  319-10460  319-10460  319-10460 #319-10460  321-10512
```

SYMBOL	VALUE	REFERENCES
		321-10512 321-10512 #321-10512 322-10556 322-10556 322-10556 #322-10556 323-10632 323-10632
		323-10632 #323-10632 323-10634 323-10634 323-10634 #323-10634 324-10675 324-10675 324-10675
		#324-10675 325-10734 325-10734 325-10734 #325-10734 325-10736 325-10736 325-10736 #325-10736
		325-10737 325-10737 325-10737 #325-10737
ILLCSR	014124	182-7081 182-7098 #182-7138
IMPTES	013064	172-6921 173-6929 #174-6947
INCBNK	047510	193-7371 195-7401 197-7433 199-7470 201-7504 203-7544 205-7587 207-7627 #373-11986
INCPAT	047464	193-7367 197-7429 #373-11974
INCRPT	047464	201-7507 203-7547 #373-11973
INHBAN	002534	#139-5712 *363-11651 363-11659
INHECC	002532	#139-5711 211-7680 343-11163 363-11649 *363-11650 363-11658 *363-11665 *363-11673 *365-11733
INTFLA	002134	#139-5597 170-6863 172-6906 173-6923 182-7053 211-7693 355-11411 *371-11908 *371-11952
INT64K	002136	#139-5598 170-6864 355-11414 361-11555 *371-11908 *371-11955
INVALI	= 104511	#110-4671 211-7709 217-7918 219-7970 238-8446 388-12270 390-12331 393-12464 397-12534
IOTVEC	= 000020	#111-4747 *147-5874 *147-5875
JMPRL1	045740	363-11684 #363-11694
KAMIKA	002004	#139-5551 253-8814 380-12109 *380-12111 *382-12154 *399-12563 *399-12568
KAMITE	026774	233-8312 233-8320 234-8330 242-8546 244-8575 246-8639 #253-8813
KDIAG	= 000010	#296-9893 296-9909 297-9930 297-9956
KDPAR0	= 172360	#111-4887 222-8098 222-8100 222-8124 222-8125 222-8144 222-8145 234-8357 234-8360
		236-8413 236-8416 236-8421
KDPAR6	= 172374	#111-4893 *222-8101 *234-8361
KDPAR7	= 172376	#111-4894 *222-8126 *222-8146 *236-8414
KERNEL	= 104417	#110-4608 169-6777 169-6790 169-6811 169-6834 174-7024 213-7845 213-7853 242-8554
		244-8610 246-8648 255-8849 257-8886 363-11682 365-11741 367-11800 367-11816 377-12064
		386-12233 386-12244 388-12274 388-12292 388-12296 421-13217 421-13233 421-13244 423-13263
		423-13275 426-13328 426-13334 429-13431 429-13435 430-13494 432-13528
KERSTK	= 002000	#111-4679
KFLAG	002524	#139-5709 273-9222 275-9322
KIPAR0	= 172340	#111-4877 361-11515 361-11591 369-11824 369-11844 430-13483 432-13501
KIPAR4	= 172350	#111-4881 *162-6281 *163-6342 *163-6357 *163-6407 *163-6411 *163-6415 *164-6429 *164-6438
		164-6439 *166-6497 *166-6505 166-6591 166-6639 *166-6678 166-6679 166-6681 369-11859
		*369-11860 369-11868 *369-11877
KIPAR5	= 172352	#111-4882 *166-6498 *166-6506 166-6642 *166-6659 *166-6665 *166-6671 *166-6679 *321-10477
		*369-11862 *369-11879
KIPAR6	= 172354	#111-4883
KIPDR0	= 172300	#111-4857 361-11593 369-11826 425-13317 428-13393 428-13417 432-13502
KMAP	= 104422	#110-4612 155-6058
KPFLAG	002112	#139-5588 *371-11905 *371-11915
KSIZE	002372	#139-5679
KSTACK	002560	#141-5725 144-5794 166-6507 189-7317 478-15210
LAST	= 157776	#113-4975 211-7683 220-8049 232-8271 234-8340 242-8560 244-8586 244-8614 246-8654
		289-9725 296-9908 297-9929 297-9966 315-10346 386-12226 388-12266
LASTBA	002552	#141-5722 *144-5799 *145-5840 166-6490 166-6492 169-6752 170-6875 173-6936 182-7071
		*182-7113 184-7152 191-7349 234-8351 238-8465 238-8472 240-8523 331-10880 331-10889
		361-11572 361-11579 363-11630 363-11637 373-11989 386-12221 388-12263 390-12309 393-12467
		395-12504 397-12537 400-12623
LASTBL	002554	#141-5723 *166-6488 *166-6495 166-6681
LASTER	002014	#139-5558 *189-7290
LBLS0	= 000404	157-6111 159-6155 159-6176 160-6260 282-9508 291-9808 316-10395 319-10460 321-10512
		322-10556 323-10634 324-10675 325-10737
LBLS1	= 000403	160-6259 291-9806 316-10393 323-10632 325-10736

SYMBOL CROSS REFERENCE CREF

SYMBOL	VALUE	REFERENCES
LBLS2	= 000402	325-10734
LF	= 000012	#111-4693 434-13646
LINK1	002516	#139-5706 *169-6745 *169-6749 169-6764 169-6803 *238-8436 *238-8440 238-8456 238-8485 *240-8508 *240-8510 240-8520 *242-8560 *242-8568 *244-8584 *244-8597 *246-8654 *246-8662 309-10205 311-10252 313-10295
LINK2	002520	#139-5707 *169-6746 *169-6750 169-6789 *244-8587 *244-8598 244-8616 244-8623
LKS	= 177546	#111-4689
LOADBA	002426	#139-5692 *363-11621 365-11719 365-11724
LOADCS	= 104425	#110-4617 282-9479 291-9776 315-10324 315-10339 316-10376 321-10487 321-10499 322-10533 323-10581 323-10589 324-10653 325-10711 326-10769 326-10772 346-11228 346-11236 346-11244 346-11254 352-11335 352-11345 354-11356 355-11430 384-12196 428-13411
LOADHO	002562	#141-5726 189-7322 363-11622 365-11720 365-11728 377-12047
LOOP	014742	#187-7242 189-7336
LOWMAP	046344	363-11692 365-11751 #365-11761 428-13390
LSIZE	002374	#139-5680 *184-7151 *184-7164 184-7188 184-7190 186-7237 509-16391
LST$$	= ******	157-6111 157-6111 157-6114 157-6114 159-6155 159-6155 159-6176 159-6176 160-6259 160-6259 160-6260 160-6260 282-9508 282-9508 291-9806 291-9806 291-9808 291-9808 316-10393 316-10393 316-10395 316-10395 319-10460 319-10460 321-10512 321-10512 322-10556 322-10556 323-10632 323-10632 323-10634 323-10634 324-10675 324-10675 325-10734 325-10734 325-10736 325-10736 325-10737 325-10737
MAINT	= 177750	#111-4761 145-5819 154-6007
MAPH0	= 170202	#113-4902 *363-11691 *365-11750 426-13361 *428-13389
MAPKER	046706	227-8218 231-8257 247-8726 248-8736 250-8759 251-8768 251-8776 251-8784 251-8792 251-8800 #369-11855
MAPL0	= 170200	#113-4901 *363-11689 *365-11750 365-11763 426-13361 *428-13389
MAPL1	= 170204	#113-4903 365-11764
MAPPER	044534	155-6059 169-6761 173-6937 174-6948 211-7708 242-8551 244-8580 246-8644 255-8822 257-8855 #361-11514 377-12059 386-12230 388-12269 390-12330 390-12363 390-12428 393-12475 397-12545 430-13475 432-13516
MASK	002314	#139-5653
MBERR	014002	*182-7076 *182-7092 182-7096 *182-7100 *182-7105 #182-7115
MEMDON	015010	187-7247 #187-7257
MFPT	= 000007	#111-4696 145-5845
MJPAT	020620	149-5914 219-7969 219-7969 219-7974 #219-7982
MJTEST	020514	209-7669 #219-7962
MKCNT	017666	*211-7706 *211-7751 211-7751 #211-7763
MKCONT	017046	209-7664 #211-7675
MKCSRT	020204	149-5900 #215-7867
MKFLAG	002116	#139-5590 172-6904 182-7050 209-7659 240-8516 *371-11905 *371-11931 373-11969
MKLOOP	017230	#211-7704 211-7757
MKPAT	020434	149-5907 217-7917 217-7917 217-7922 #217-7934
MKTEST	020274	191-7345 209-7667 #217-7907
MMR0	= 177572	#111-4770 *341-11136 *341-11140 *363-11694 *363-11703 *365-11742 *365-11745 426-13349 *428-13402 430-13493 *432-13512 509-16390 509-16398
MMR1	= 177574	#111-4771 426-13349 *428-13402 430-13492 *432-13511 509-16390 509-16398
MMR2	= 177576	#111-4772 426-13349 *428-13402 430-13491 *432-13510 509-16390 509-16398
MMR3	= 172516	#111-4773 145-5826 *145-5827 *145-5828 145-5829 *189-7320 *363-11685 *363-11693 *365-11749 *365-11752 *375-12007 *377-12052 426-13352 *428-13401 430-13490 *432-13509 509-16390 509-16398
MMTRAP	042570	147-5888 #339-11107
MMVEC	= 000250	#111-4757 *147-5888 *147-5889
MONFLG	002272	#139-5644 *144-5792 *154-5993 *154-6012 *154-6015 465-14878
MPT	= ******	187-7269 311-10216 339-11113 361-11601 377-12038 377-12053 380-12135 399-12585 406-12664

CREF

SYMBOL	CROSS REFERENCE VALUE	REFERENCES
		425-13288 428-13375 430-13462 434-13596 434-13639 435-13658 462-14746 463-14799 470-14987
		473-15107 479-15256 483-15435 485-15525 485-15541 493-15952
MSEEDH	002572	#141-5730 *147-5867 147-5869
MSEEDL	002574	#141-5731 *147-5868 147-5870
MSGA12	100014	*182-7144 #517-16614
MSGA34	076031	211-7744 #517-16556
MSGB34	076067	211-7746 #517-16557
MSG000	100226	154-5996 #517-16621
MSG001	074036	331-10877 #517-16503
MSG002	074120	331-10878 #517-16504
MSG003	074175	331-10879 #517-16505
MSG004	074302	331-10887 331-10907 #517-16507
MSG005	074410	333-10934 #517-16509
MSG006	074422	494-15990 #517-16510
MSG007	074457	335-10949 #517-16511
MSG008	100174	159-6142 #517-16620
MSG009	074471	159-6156 335-10980 #517-16512
MSG010	074503	335-11032 #517-16513
MSG011	074515	331-10893 331-10906 #517-16514
MSG012	074603	331-10896 331-10908 #517-16515
MSG013	074700	333-10937 #517-16516
MSG014	074702	159-6153 159-6173 331-10905 333-10939 335-11042 465-14833 472-15035 472-15039 473-15105 #517-16517
MSG015	074704	*159-6151 159-6152 *159-6161 *159-6163 *159-6167 *159-6170 159-6172 *335-10956 *335-10958 *335-10968 335-10969 *335-10990 *335-10992 *335-10995 *335-10998 335-10999 *335-11011 *335-11021 335-11022 *335-11035 *335-11039 335-11040 #517-16518
MSG016	074706	159-6143 335-11010 #517-16519
MSG017	074720	331-10892 331-10895 #517-16520
MSG018	074731	470-14994 473-15103 478-15200 478-15215 478-15230 478-15245 #517-16521
MSG019	074734	473-15073 #517-16522
MSG020	074740	380-12099 #517-16523
MSG021	074761	380-12117 #517-16524
MSG022	075547	406-12651 #517-16545
MSG023	075571	384-12193 #517-16546
MSG025	075605	382-12178 384-12202 #517-16547
MSG026	075631	380-12112 #517-16548
MSG027	075643	384-12189 #517-16549
MSG028	075660	384-12198 #517-16550
MSG029	075674	386-12211 #517-16551
MSG030	075714	390-12306 400-12603 #517-16552
MSG031	075733	386-12212 388-12254 #517-16553
MSG032	075773	386-12237 388-12277 #517-16554
MSG033	076012	386-12241 388-12281 #517-16555
MSG035	076115	189-7295 #517-16558
MSG036	076120	388-12253 #517-16559
MSG037	076137	388-12284 #517-16560
MSG038	076156	388-12293 #517-16561
MSG039	076174	388-12286 #517-16562
MSG040	076216	390-12305 #517-16563
MSG041	076242	390-12315 #517-16564
MSG042	076267	390-12319 #517-16565
MSG043	076305	390-12324 #517-16566

SYMBOL CROSS REFERENCE CREF

SYMBOL	VALUE	REFERENCES
MSG046	076327	390-12337 393-12451 397-12522 #517-16567
MSG047	076362	#517-16568
MSG048	076401	380-12102 #517-16569
MSG049	076441	390-12334 #517-16570
MSG051	076474	429-13451 #517-16572
MSG055	076513	393-12442 #517-16573
MSG056	076534	393-12449 397-12520 #517-16574
MSG058	076567	478-15191 #517-16575
MSG061	076611	473-15082 #517-16576
MSG062	076620	487-15711 487-15770 #517-16577
MSG063	076640	487-15712 #517-16578
MSG064	076651	487-15713 487-15772 #517-16579
MSG065	076661	487-15771 #517-16580
MSG066	076673	466-14903 #517-16581
MSG067	076742	468-14955 #517-16582
MSG070	076751	184-7201 #517-16583
MSG073	077002	397-12513 #517-16584
MSG075	077020	363-11632 363-11669 #517-16585
MSG076	077052	395-12495 #517-16586
MSG077	077073	189-7293 #517-16587
MSG079	077107	395-12487 #517-16588
MSG085	077133	399-12553 #517-16589
MSG088	077160	478-15209 #517-16590
MSG089	077176	478-15225 #517-16591
MSG090	077220	478-15240 #517-16592
MSG091	077234	478-15234 478-15249 #517-16593
MSG092	077246	485-15535 #517-16594
MSG093	077262	485-15537 #517-16595
MSG095	077270	485-15539 #517-16596
MSG101	077300	399-12562 #517-16597
MSG102	077330	399-12567 #517-16598
MSG103	077357	382-12148 #517-16599
MSG104	077401	470-15004 #517-16600
MSG105	077403	400-12599 #517-16601
MSG106	077456	399-12573 #517-16602
MSG107	077474	399-12580 #517-16603
MSG110	077551	400-12614 #517-16604
MSG111	077615	400-12619 #517-16605
MSG112	077647	184-7191 #517-16606
MSG113	077664	184-7195 #517-16607
MSG114	077701	184-7199 #517-16608
MSG117	077716	154-6010 #517-16609
MSG119	077730	154-6029 #517-16610
MSG120	077737	154-6030 #517-16611
MSG121	077760	154-6024 #517-16612
MSG122	100000	182-7145 #517-16613
MSG125	100046	395-12485 #517-16616
MSG126	100070	163-6416 #517-16617
MSG127	100135	402-12633 #517-16618
MSG128	100154	404-12639 #517-16619
MSG129	100275	159-6177 #517-16622
MSIZE	002376	#139-5681 *184-7151 *184-7161 184-7192 184-7194 186-7237 509-16391

SYMBOL CROSS REFERENCE CREF

SYMBOL	VALUE	REFERENCES								
MTA030	024600	#240-8503	240-8535							
MTEST	016752	193-7365	195-7395	197-7427	199-7464	201-7501	203-7541	205-7584	207-7624	#209-7655
MTLA11	030412	273-9172	#273-9177	273-9272						
MTLB11	030424	#273-9180	273-9268							
MTLC11	030436	#273-9183	273-9262							
MTLD11	030532	#273-9200	273-9284							
MTPA03	027510	222-8088	222-8094	#261-8939						
MTPA04	027646	222-8120	222-8123	222-8141	#265-8998	265-9018				
MTPA21	034430	232-8282	232-8296	#292-9818						
MTPA24	035254	234-8355	#299-9982	301-10041						
MTPA25	035664	303-10052	303-10087	305-10095	#305-10106					
MTPA26	036014	236-8399	236-8400	236-8412	#307-10138	307-10181				
MTPB03	027550	222-8091	222-8097	#261-8962	263-8995					
MTPB04	027702	222-8124	*222-8139	*222-8141	222-8144	265-9002	#265-9011	265-9024		
MTPB21	034460	232-8284	232-8299	#292-9835						
MTPB24	035314	234-8349	234-8356	299-10023	#301-10026					
MTPB25	035706	303-10089	305-10097	#305-10113						
MTPB26	036030	236-8403	236-8404	236-8420	#307-10145					
MTPC03	027610	222-8098	261-8964	#263-8980	263-8994					
MTPC21	034514	232-8287	232-8303	#292-9852						
MTPC24	035330	234-8357	301-10032	#301-10034						
MTPC25	035746	303-10091	305-10100	305-10102	#305-10125					
MTPC26	036064	236-8413	307-10139	307-10148	#307-10163					
MTPD03	027626	222-8099	263-8986	#263-8988						
MTPD21	034550	232-8289	232-8306	#294-9870						
MTPD25	035612	303-10064	303-10066	303-10069	303-10071	#303-10087				
MTPD26	036104	*236-8399	*236-8403	236-8415	#307-10176					
MTPE25	035634	303-10075	303-10077	303-10080	303-10082	#305-10095				
MTP000	027400	220-8019	220-8022	#259-8902	259-8904	*328-10842	328-10843	*328-10845	328-10848	
MTP001	027424	220-8041	220-8044	#259-8913						
MTP002	027456	220-8067	220-8070	#259-8925						
MTP005	027722	222-8138	222-8139	222-8143	#265-9020					
MTP006	027756	224-8155	#267-9034							
MTP007	030156	224-8165	#269-9077							
MTP010	030256	224-8172	#271-9117							
MTP011	030364	226-8183	#273-9153							
MTP012	031162	226-8198	#275-9294							
MTP013	031550	226-8207	#277-9376							
MTP014	032264	227-8221	#282-9467							
MTP015	032510	229-8232	#283-9515							
MTP016	033254	229-8241	#285-9614							
MTP017	034036	229-8246	#289-9716							
MTP020	034114	231-8260	#291-9750							
MTP022	034600	233-8315	233-8323	#296-9888						
MTP025	035346	234-8377	#303-10044							
MTP030	036122	240-8507	240-8511	#309-10184						
MTP031	036132	242-8553	#309-10190	309-10212						
MTP032	036210	244-8609	244-8628	#311-10241						
MTP033	036242	246-8647	#313-10261	313-10304						
MTP034	036340	238-8438	238-8455	238-8461	238-8484	238-8490	246-8678	246-8684	246-8690	#314-10307
		314-10309								
MTP035	036364	246-8711	#315-10319							

SYMBOL CROSS REFERENCE CREF

SYMBOL	VALUE	REFERENCES							
MTP036	036526	247-8727	#316-10351						
MTP037	036752	248-8738	#317-10404						
MTP041	037024	249-8750	#319-10425						
MTP042	037176	250-8761	#321-10466						
MTP043	037432	251-8770	#322-10519						
MTP044	037626	251-8778	#323-10560						
MTP045	040142	251-8786	#324-10640						
MTP046	040330	251-8794	#325-10680						
MTP047	040670	251-8803	#326-10756						
MTST3	012422	169-6745	169-6746	169-6748	169-6774	169-6808	#169-6846	169-6847	
MT0000	020700	#220-8012	390-12381						
MT0001	020760	217-7938	219-7987	#220-8025	390-12382				
MT0002	021100	217-7939	219-7988	#220-8047	390-12383				
MT0003	021240	219-7989	#222-8075	390-12384					
MT0004	021472	217-7940	219-7990	#222-8111	390-12385				
MT0005	021614	217-7941	219-7991	#222-8129	390-12386				
MT0006	021750	215-7867	219-7984	#224-8151	390-12387				
MT0007	022004	217-7937	219-7986	#224-8158	390-12388				
MT0010	022046	215-7884	#224-8168	390-12389					
MT0011	022102	215-7890	#226-8177	390-12390					
MT0012	022160	215-7891	#226-8186	390-12391					
MT0013	022264	215-7892	#226-8201	390-12392					
MT0014	022350	215-7873	#227-8212	390-12393					
MT0015	022440	215-7893	#229-8226	390-12394					
MT0016	022516	215-7894	#229-8235	390-12395					
MT0017	022574	217-7936	219-7985	#229-8244	390-12396				
MT0020	022616	215-7876	#231-8251	390-12397					
MT0021	022706	217-7942	219-7992	#232-6264	390-12398				
MT0022	023160	217-7943	219-7994	#233-8311	390-12399	397-12535			
MT0023	023212	219-7995	#233-8319	390-12400					
MT0024	023256	217-7945	219-7997	#234-8329	390-12401				
MT0025	023522	215-7889	#234-8371	390-12402					
MT0026	023600	217-7944	219-7996	#236-8382	390-12403				
MT0027	024102	187-7264	#238-8429	390-12404					
MT0030	024566	187-7274	#240-8500	375-12003	377-12044	390-12405			
MT0031	025070	217-7946	219-7998	#242-8545	390-12406	393-12465			
MT0032	025260	217-7947	219-7999	#244-8574	390-12407				
MT0033	025612	217-7948	219-8000	#246-8638	390-12408				
MT0034	026000	217-7917	217-7935	217-7949	219-7969	219-7983	219-8001	#246-8668	390-12409
MT0035	026152	219-7993	#246-8698	390-12410					
MT0036	026264	215-7875	#247-8719	390-12411					
MT0037	026336	215-7877	#248-8731	390-12412					
MT0040	026404	#248-8743	390-12413						
MT0041	026406	215-7878	#249-8746	390-12414					
MT0042	026450	215-7879	#250-8755	390-12415					
MT0043	026514	215-7880	#251-8765	390-12416					
MT0044	026554	215-7872	#251-8773	390-12417					
MT0045	026614	215-7874	#251-8781	390-12418					
MT0046	026654	215-7881	#251-8789	390-12419					
MT0047	026714	215-7882	#251-8797	390-12420					
MT0999	026760	149-5926	215-7898	215-7899	215-7900	215-7901	215-7902	217-7917	217-7951	217-7952
		217-7953	217-7954	217-7955	217-7956	217-7957	217-7958	217-7959	219-7969	219-8003

SYMBOL	VALUE	REFERENCES								
		219-8004	219-8005	219-8006	219-8007	#253-8808				
MT1	017026	209-7660	#209-7669							
MT2	017032	209-7668	#209-7670							
MUT	002106	#139-5586	*187-7263	*187-7266	*191-7344	*191-7346	*209-7657	*209-7670	*390-12353	*393-12459
		*397-12529	470-14997	476-15156	*476-15158	476-15172	*476-15174	*476-15180		
NC	056430	434-13624	434-13628	434-13634	#434-13638					
NEMCNT	002066	#139-5578	*167-6703	167-6716	*169-6760	169-6784	169-6796	169-6818	169-6842	*339-11092
		339-11093	*339-11097	509-16389						
NEWBAN	002304	#139-5649	240-8533	*363-11677	363-11686	363-11696	*365-11737	369-11832	369-11845	
NEWKER	046640	363-11695	365-11743	#369-11842						
NEWLOA	046742	363-11624	365-11722	#369-11865						
NOCH	063560	483-15441	#483-15445							
NOERRO	002424	#139-5691	*382-12174	*384-12190	*384-12199	465-14823	465-14860	466-14883	*466-14912	468-14949
		468-14961	470-15001	*476-15169	*476-15175					
NOFSMO	002422	#139-5690	*238-8441	*238-8468	*238-8496	*240-8513	*240-8530	*240-8541	380-12101	
NONEM	002076	#139-5582	*167-6704	*169-6741	*169-6827	*169-6836	339-11090			
NONEXI	042512	167-6705	169-6742	#339-11090						
NOOJ	041464	331-10881	#331-10884							
NOPAR	002074	#139-5581	*144-5804	*144-5810	*157-6086	*164-6428	*167-6702	*169-6740	*169-6782	*211-7725
		*217-7913	*217-7928	*219-7963	*219-7975	*226-8208	*240-8504	*242-8550	*244-8579	*246-8643
		*282-9475	*303-10049	*303-10084	*315-10320	*315-10348	*325-10696	337-11063	337-11067	337-11074
		386-12209	*386-12210	*386-12245	388-12251	*388-12252	*388-12297	*390-12373	*390-12378	393-12440
		*393-12474	397-12511	*397-12544						
NORES	003664	144-5790	#144-5792							
NOSCOP	002434	#139-5695	*232-8265	*232-8308	*234-8332	*234-8369	*240-8513	*240-8530	*240-8541	*242-8548
		*242-8570	*244-8577	*244-8632	*246-8641	*246-8665	462-14758	466-14888		
NOSUPE	002452	#139-5702	*145-5834	*145-5853	155-6034	220-8029	220-8053	242-8561	244-8588	246-8655
		257-8878	355-11376	361-11518	361-11535	361-11560	361-11581	425-13309	426-13330	426-13350
		428-13399	429-13426	430-13488	432-13507	478-15219				
NOTAB	002366	#139-5677	335-10970	335-11000	335-11023	*472-15034	*472-15037	*472-15048	*472-15050	*473-15072
		*473-15075								
NOTRCE	060220	462-14737	#462-14739							
NO22BI	002450	#139-5701	*145-5837	*145-5839	154-5992	166-6486	189-7318	355-11395	361-11574	375-12005
		377-12050								
NULLFL	002340	#139-5663	*253-8810							
NXTCSR	005642	157-6087	#157-6105							
OLDCAC	002276	#139-5646								
OLDCSR	002154	#139-5605	*172-6898	172-6916	*172-6917					
ONES	002600	#141-5733	169-6802	174-6968	174-6970	213-7822	213-7824	222-8114	296-9897	296-9899
		327-10819	327-10824	453-14540						
PADDRE	002034	#139-5566	509-16388							
PAFBAF	016146	187-7252	#201-7489	201-7514	201-7522					
PAFBAW	016276	187-7253	#203-7527	203-7554	203-7559	203-7567				
PARBAF	016450	187-7254	#205-7572	205-7597	205-7605					
PARBAW	016600	187-7255	#207-7610	207-7637	207-7642	207-7650				
PARCNT	002070	#139-5579	*167-6701	167-6713	*169-6759	169-6794	169-6816	169-6838	*282-9474	282-9483
		*282-9498	282-9501	*282-9511	*325-10704	325-10744	325-10749	*337-11065	509-16388	
PARITY	042406	147-5882	#337-11063							
PARTHE	002300	#139-5647	*303-10051	*305-10113	*305-10125	*305-10128	*315-10325	*315-10340	337-11076	386-12209
		*386-12228	*386-12245	388-12251	*388-12267	*388-12297				
PARVEC	= 000114	#111-4755	*147-5882	*147-5883						
PASFLG	002262	#139-5639	*160-6223	*160-6226	160-6228	160-6235	160-6246	160-6253	160-6260	*209-7658

SYMBOL	VALUE	REFERENCES								
		*211-7728	*240-8501	240-8524	*240-8525	*247-8723	273-9185	273-9257	273-9269	*273-927
		*273-9273	277-9388	279-9423	279-9435	*279-9437	*279-9440	283-9526	283-9571	283-958
		*283-9585	*283-9587	285-9627	287-9673	287-9685	*287-9687	*287-9689	*291-9758	*291-9761
		291-9764	291-9800	291-9808	*316-10357	*316-10360	316-10363	316-10387	316-1039	*319-10435
		*319-10439	319-10460	*321-10478	*321-10485	321-10512	*322-10527	*322-10531	322-10556	*323-10568
		*323-10573	323-10615	323-10634	*325-10691	*325-10694	325-10698	325-10728	325-10736	*390-12358
PASSNO	002264	#139-5641	*282-9494	*282-9497	282-9508	*323-10576	*323-10578	323-10632	*324-10648	*324-10652
		324-10675	*325-10696	*325-10689	325-10713	325-10737	325-10743			
PATERR	002072	#139-5580	*169-6758	169-6792	169-6814	*169-6852				
PATPLU	004606	149-5902	149-5905	149-5909	149-5912	149-5916	149-5919	#149-5922		
PATTER	002110	#139-5587	*191-7340	*193-7363	*195-7397	*197-7425	*199-7466	*201-7490	*203-7528	*205-7590
		*207-7630	217-7915	219-7967	*373-11977	373-11978	*373-11983	390-12303	*390-12321	390-12322
		390-12323	390-12375	*390-12427	393-12440	*393-12474	397-12511	*397-12544		
PCBUMP	002322	#139-5656	*217-7914	*219-7964	*222-8078	*222-8113	*222-8131	*224-8153	*224-8170	*236-8384
		337-11071	390-12303	*390-12374	*390-12426					
PCONFI	041352	184-7203	#331-10868	391-12434						
PCONFS	041652	*331-10870	331-10911	#331-10917						
PCONF1	041562	331-10883	#331-10902							
PCONF2	041620	331-10871	331-10890	331-10900	#331-10911					
PDP110	042602	147-5884	#339-11110							
PD1	054476	425-13310	#425-13317							
PERA05	056644	#451-14503	451-14510							
PERBNK	057476	211-7747	315-10330	315-10345	421-13247	454-14584	454-14596	456-14611	#458-14632	460-14705
		465-14869								
PERECC	057556	#458-14648	458-14662							
PERRAB	057314	451-14525	451-14530	#454-14595						
PERRAW	057242	421-13224	423-13265	451-14482	451-14488	451-14494	451-14500	451-14515	451-14520	453-14536
		453-14541	453-14546	453-14551	453-14556	453-14561	453-14571	453-14576	453-14581	#454-14583
PERRA3	054044	#421-13226	456-14617	456-14621						
PERRA7	057366	453-14566	#456-14608							
PERR01	= 104427	#110-4620	259-8921	311-10256						
PERR02	= 104430	#110-4621	259-8909	259-8934	311-1025?	314-10314				
PERR03	= 104431	#110-4622	263-8983	263-8991						
PERR04	= 104432	#110-4623	265-9007	265-9029						
PERR05	056640	#451-14502								
PERR06	056666	#451-14509								
PERR07	= 104433	#110-4624	267-9044	267-9063						
PERR10	= 104434	#110-4625	267-9049	267-9068						
PERR11	= 104435	#110-4626	269-9085	271-9135						
PERR12	= 104436	#110-4627	269-9091	271-9142						
PERR13	= 104437	#110-4628	269-9099							
PERR14	= 104440	#110-4629	269-9105							
PERR15	= 104441	#110-4630								
PERR16	= 104442	#110-4631								
PERR17	= 104443	#110-4632	292-9823	292-9845	292-9856	294-9880	297-9935	297-9939		
PERR20	= 104444	#110-4633	292-9829	292-9839	292-9862	294-9874	297-9944	297-9948		
PERR21	= 104445	#110-4634								
PERR22	= 104446	#110-4635	289-9738							
PERR23	= 104447	#110-4636	299-10011	301-10040						
PERR24	= 104450	#110-4637	299-10016							
PERR25	= 104451	#110-4638	307-10151	307-10156						
PERR26	= 104452	#110-4639								

SYMBOL	VALUE	REFERENCES								
PERR27	= 104453	#110-4640								
PERR30	= 104454	#110-4641	309-10197							
PERR31	= 104455	#110-4642	273-9214	273-9220	283-9561	283-9567	285-9659	285-9666	319-10456	
PERR32	= 104456	#110-4643	273-9235	273-9248						
PERR33	= 104457	#110-4644	275-9336							
PERR34	= 104460	#110-4645	273-9241	273-9254	275-9344					
PERR35	= 104461	#110-4646	313-10273	313-10280						
PERR36	= 104462	#110-4647	355-11421							
PERR37	= 104463	#110-4648								
PERR40	= 104464	#110-4649								
PERR41	= 104465	#110-4650								
PERR42	= 104466	#110-4651								
PERR43	= 104467	#110-4652								
PERXOR	057452	454-14587	454-14599	456-14610	#456-14623	458-14654	460-14690			
PFECDF	061740	470-15007	#470-15012							
PFECDH	061700	470-15007	#470-15009							
PFECDT	061730	470-15007	#470-15011							
PFECEM	061644	470-15007	#470-15008							
PFECWS	061634	468-14940	#470-15007							
PFLAG	002120	#139-5591	172-6918	363-11619	*371-11907	*371-11937				
PGMCSR	002526	#139-5710	*162-6280	*162-6326	162-6331	*163-6374	*163-6404	*163-6405	*163-6417	246-8706
		343-11164	343-11166	343-11170	344-11190	344-11192	344-11196	363-11678	*363-11678	*363-11704
		*363-11711	365-11744	*365-11744						
PHEBE	014004	#182-7116	*182-7118	*182-7123	182-7126	182-7128	182-7130	182-7132	*182-7134	*182-7136
		273-9171	273-9228							
PHYADD	002036	#139-5567	*473-15088	*473-15089	*473-15090	*473-15091	*473-15093	473-15094		
PMEMFL	002140	#139-5599	226-8181	226-8190	226-8205	227-8216	229-8230	229-8236	231-8256	234-8375
		247-8720	248-8732	249-8747	250-8756	251-8766	251-8774	251-8782	251-8790	251-8798
		*371-11905	*371-11934							
PROTYP	003752	#145-5815	*145-5850	145-5851	145-5860	151-5956	164-6446	167-6721	169-6743	169-6772
		169-6781	169-6806	220-8017	220-8039	220-8065	222-8085	222-8118	222-8136	232-8274
		232-8276	234-8341	234-8343	236-8391	236-8393	238-8434	238-8453	238-8459	238-8482
		238-8488	240-8505	244-8606	244-8626	246-8676	246-8682	246-8688	328-10840	344-11199
		363-11683	365-11747	382-12176	382-12180	384-12192	384-12204	386-12246	388-12298	426-13359
		428-13387	462-14771	466-14895	476-15177					
PSIZE	002400	#139-5682	*184-7151	*184-7159	184-7196	184-7198	186-7237			
PSW	= 177776	#111-4684	*155-6038	*155-6039	*155-6045	*169-6771	*169-6788	*169-6805	*169-6831	*174-6951
		*213-7794	*242-8552	*244-8600	*246-8646	*255-8844	*257-8877	*331-10874	*333-10933	*335-10951
		*344-11215	*344-11217	*363-11680	*365-11739	*367-11798	*367-11814	*377-12060	*386-12231	*388-12271
		*388-12289	*390-12349	*393-12455	*397-12526	*421-13215	*421-13231	*421-13241	*423-13261	*423-13273
		*426-13326	*426-13332	*429-13429	*429-13433	*430-13476	*432-13517	*478-15221	*478-15222	*478-15237
PTABLE	036732	291-9772	316-10372	#316-10402	325-10707					
PWRVEC	= 000024	#111-4748	*147-5880	*147-5881	*153-5975	*425-13300	*425-13301	*426-13367	*428-13378	*429-13450
		430-13477	430-13478	*430-13480	*430-13481	*432-13518	*432-13519			
QUICK	002432	#139-5694	*153-5974	377-12043						
QVFLAG	002342	#139-5664	*153-5974	*153-5979	189-7294	*189-7296	246-8713	273-9264	275-9361	279-9430
		283-9578	287-9680							
RANODD	036044	*236-8402	*236-8419	#307-10153	*307-10157					
RDCHR	= 104411	#110-4600	486-15601							
RDDEC	= 104414	#110-4603	380-12113							
RDLIN	= 104412	#110-4601	406-12652	487-15682	487-15733					
RDOCT	= 104413	#110-4602	384-12194	386-12213	388-12255	388-12287	390-12307	390-12320	390-12325	399-12554

SYMBOL	VALUE	REFERENCES
		400-12604
READCS	= 104426	#110-4618 166-6534 273-9223 282-9486 291-9780 291-9788 303-10061 305-10118 305-10130 315-10333 319-10447 321-10491 321-10500 322-10537 322-10546 323-10590 323-10603 324-10657 324-10664 325-10720 326-10776 326-10786 348-11266 348-11285 350-11301 350-11320 382-12173 384-12188 384-12197 421-13234 426-13342 478-15198
READON	002404	#139-5684
REALPA	002274	#139-5645 *220-8013 *220-8026 *220-8048 *222-8077 *222-8112 *222-8130 *224-8152 *224-8159 *224-8169 *226-8182 *226-8191 *226-8206 *227-8217 *229-8231 *229-8240 *229-8245 *231-8255 *232-8266 *233-8314 *233-8322 *234-8333 *234-8376 *236-8383 *238-8432 *240-8503 *242-8549 *244-8578 *246-8642 *246-8669 *246-8699 *247-8721 *248-8733 *249-8748 *250-8757 *251-8767 *251-8775 *251-8783 *251-8791 *251-8799 *253-8809 363-11660 363-11667 458-14658 458-14663 458-14666 458-14669 463-14793 472-15057 485-15540
REFRES	035154	296-9923 #297-9960
REFSUB	035224	297-9963 297-9967 #297-9971
REGCOP	041064	220-8016 220-8038 #327-10802
RELENT	045612	363-11661 363-11662 #363-11674
RELOCA	045172	193-7377 195-7407 197-7444 199-7481 201-7519 203-7564 205-7602 207-7647 240-8527 #363-11611
RELOC1	045626	#363-11677
RESREG	= 104416	#110-4606 222-8090 222-8104 234-8365 236-8405 236-8422 328-10846 328-10851 367-11781 367-11810 380-12103 382-12166 470-15000 491-15866
RESTAR	002612	*137-5538 *137-5540 #141-5738 151-5937
RESVEC	= 000010	#111-4713 *147-5884 *147-5885
RES0	050376	382-12158 382-12163 #382-12166
RES1	050456	382-12172 #382-12178
RES2	050624	384-12187 #384-12202
RLFLAG	002124	#139-5593 193-7374 195-7404 197-7441 199-7478 201-7516 203-7561 205-7599 207-7644 *363-11712 *365-11746 371-11938 375-12002 377-12042 380-12101 425-13315 463-14795 485- 5534
RRFLAG	002122	#139-5592 191-7343 193-7361 195-7391 197-7423 199-7460 201-7499 203-7539 205-7582 207-7622 211-7712 238-8445 238-8474 240-8517 *371-11907 *371-11937 *371-11940 *371-11948 390-12333 393-12463 397-12533
RWCSR	006206	157-6100 #160-6191
SAVCSR	002152	#139-5604 *390-12338 390-12430
SAVMON	002270	#139-5643 *144-5793 465-14879
SAVPAR	002266	#139-5642 *369-11859 369-11877 *369-11878 369-11879
SAVREG	= 104415	#110-4605 222-8087 222-8095 234-8348 234-8354 236-8398 236-8411 328-10837 367-11779 367-11782 380-12098 468-14930 491-15851
SBEMSK	002250	#139-5636 *273-9180 *273-9181 273-9196 273-9198 273-9259 *273-9261 *273-9261 *275-9300 *275-9301 275-9306 275-9308 275-9319 275-9356 *275-9358 *275-9358 *277-9382 *277-9383 277-9393 277-9395 277-9399 277-9401 279-9431 *279-9433 *279-9433 *279-9443 279-9446 *279-9453 279-9454 279-9456 *283-9520 *283-9521 283-9531 283-9533 283-9537 283-9539 283-9579 *283-9581 *283-9581 *283-9590 283-9593 *283-9600 283-9601 283-9603 *285-9621 *285-9622 285-9631 285-9633 285-9637 285-9639 287-9681 *287-9683 *287-9683 *287-9692 287-9695 *287-9702 287-9703 287-9705
SBENT	020146	213-7799 213-7801 213-7806 213-7808 213-7817 213-7823 213-7825 213-7831 213-7833 213-7842 #213-7851
SBESYN	034410	291-9790 #291-9815
SBETES	017670	211-7722 #213-7766
SCOPE	= 000004	#111-4683 157-6063 167-6694 169-6731 172-6879 187-7242 187-7259 255-8850 257-8887
SDPAR0	= 172260	#111-4847 222-8099 222-8101 234-8356 234-8358 234-8359 236-8415
SDPAR5	= 172272	#111-4852 *222-8103 *234-8360
SDPAR6	= 172274	#111-4853 *236-8417

```
SYMBOL  VALUE          REFERENCES
SDPAR7  = 172276       #111-4854  *222-8102
SEEDHI    002566       #141-5728  *147-5869   236-8386  *236-8425   489-15824  489-15831 *489-15836
SEEDLO    002570       #141-5729  *147-5870   236-8385  *236-8424   489-15823  489-15829 *489-15835
SELONL    002000       #139-5549   187-7262   211-7679   371-11947 *400-12615 *400-12620
SETPAT    047500        195-7393   199-7462  #373-11981
SHADL1    012452       #170-6861   170-6876
SHUTUP    047664        189-7305   377-12023 #377-12031
SIPAR0  = 172240       #111-4837   361-11516
SIPAR3  = 172246       #111-4840   273-9173  *273-9174   355-11378  361-11534
SIPAR5  = 172252       #111-4842   220-8031   220-8055   242-8563   244-8590   246-8657   273-9174  *273-9175   361-11564
                        361-11585
SIPAR6  = 172254       #111-4843   220-8031   220-8055   242-8563   244-8590   246-8657  *361-11585
SIPDR0  = 172200       #111-4817   361-11517  425-13319  42[illegible]3419
SIZE    = 040000       #113-4976   167-6707   169-6767   169-6800   220-8015   220-8028   220-8050   222-8117  222-8135
                        236-8390   238-8450   238-8479   240-8518   244-8583   244-8620   246-8671   328-10839 363-11681
                        365-11740  367-11799  367-11815
SKIPKA    002006       #139-5552   382-12150 *382-12152 *399-12563 *399-12569
SKIPMK    002336       #139-5662   172-6905   209-7663  *371-11908 *371-11958
SKJ       060246        462-14741  462-14744 #462-14754
SKPERR    002064       #139-5577  *213-7812  *213-7837   355-11419 *355-11423
SKUB      045602        363-11668 #363-11671
SKUJ      014006        182-7114  #182-7117
SOBK      002556       #141-5724   242-8555
SOBLEN  = 000056        242-8553   242-8558 #309-10212
SOFTPA    002604       #141-5735   246-8672
SOURCE    002306       #139-5650  *273-9191  *275-9304  *277-9397  *283-9535  *285-9635  *303-10047  357-11445
SPLTCS    002236       #139-5633  *172-6908  *173-6930  *211-7685  *211-7695   211-7717  *211-7750  *211-7753  *211-7760
                        361-11542  361-11547 *390-12357 *390-12362
SSP     =%000006       #111-4701  *155-6042  *255-8845  *257-8882   426-13333 *429-13430  478-15223
ST      = 177776       #498-16094
STACK   = 002000       #111-4678   111-4679   137-5531   137-5542   141-5725   153-5978   189-7301
START     003654        137-5539   137-5541  #144-5788   377-12028
START1    000300        137-5535  #137-5538
START2    000310        137-5536  #137-5540
START3    000200       #137-5535   517-16636
STAR27    024162        238-8437  #238-8442
STOPOK    002414       #139-5687  *174-7027   462-14754 *462-14755 *486-15610
STRIPE    002362       #139-5675  *296-9902   296-9906   296-9925  *297-9956   297-9956
SUBAAA    004644        149-5920  #151-5933
SUBAAB    004774       #153-5963
SUBAAI    012446        169-6783  #170-6856
SUBAAP    014166        182-7137  #184-7150
SUBAAR    013372        173-6945  #174-7027
SUBAAS    011400        166-6508  #167-6692
SUCCES    002330       #139-5659  *211-7711  *211-7737   211-7743  *395-12489  395-12494 *395-12496
SUPDOA    002260       #139-5638  *220-8019  *220-8041  *220-8067  *222-8088  *222-8091  *222-8120  *222-8138  *224-8155
                       *224-8165  *224-8172  *226-8183  *226-8198  *226-8207  *229-8232  *229-8241  *229-8246  *232-8282
                       *232-8284  *232-8287  *232-8289  *233-8315  *233-8323  *234-8349  *234-8358  *234-8377  *236-8400
                       *236-8404  *238-8439  *238-8455  *238-8461  *238-8484  *238-8490  *240-8511  *242-8559  *244-8615
                       *244-8628  *246-8651  *246-8679  *246-8684  *246-8690  *246-8711  *249-8750   257-8884  *328-10843
SUPD01    027030        220-8023   220-8045   220-8071   222-8096   232-8297   236-8418   236-8423   238-8440   240-8508
                        246-8693  #255-8821   328-10850
```

SYMBOL	VALUE	REFERENCES								
SUPDO2	027044	222-8105	222-8127	222-8147	232-8300	232-8304	232-8307	244-8631	#255-8823	
SUPDO3	027206	220-8020	220-8042	220-8068	222-8089	224-8156	224-8166	224-8173	226-8184	226-8199
		226-8209	229-8233	229-8242	229-8247	232-8283	233-8316	233-8325	234-8378	236-8401
		236-8406	238-8436	238-8462	238-8491	240-8510	246-8685	246-8691	246-8712	249-8752
		#257-8855	328-10844							
SUPDO4	027222	222-8092	222-8121	222-8140	232-8285	232-8288	232-8290	234-8362	234-8368	242-8569
		244-8619	244-8629	246-8664	246-8716	#257-8856				
SUPDR0	002156	#139-5607	*255-8825	255-8833	*257-8858	257-8866				
SUPDR1	002160	#139-5608	255-8826	257-8859						
SUPDR2	002162	#139-5609								
SUPDR3	002164	#139-5610								
SUPDR4	002166	#139-5611								
SUPDR5	002170	#139-5612								
SUPDR6	002172	#139-5613	255-8836	257-8869						
SUPLIM	056506	#434-13656	517-16631	517-16632						
SUPSTK	= 000740	#111-4680	155-6041	255-8845	257-8882	478-15226	478-15232			
SWAPAT	002620	#141-5743	*147-5872							
SWR	002622	#141-5746	*151-5950	151-5952	*151-5957	*153-5976	155-6052	167-6724	173-6939	184-7202
		187-7244	189-7294	217-7908	217-7923	273-9263	275-9360	279-9429	283-9577	287-9679
		331-10883	363-11612	363-11631	363-11666	390-12366	393-12444	397-12515	421-13250	426-13363
		428-13385	462-14754	462-14761	462-14779	465-14830	465-14870	465-14873	466-14884	466-14898
		470-15001	483-15465	483-15473	483-15495	486-15608				
SWREG	= 000176	#113-4968	151-5957	155-6052	483-15465					
SW0	= 000001	#111-4721	167-6724	173-6939	217-7908	217-7923	390-12366	393-12444	397-12515	421-13250
SW1	= 000002	#111-4720								
SW10	= 002000	#111-4711	465-14830							
SW11	= 004000	#111-4710	189-7294	273-9263	275-9360	279-9429	283-9577	287-9679		
SW12	= 010000	#111-4709	363-11612							
SW13	= 020000	#111-4708	363-11631	363-11666	465-14870					
SW14	= 040000	#111-4707	462-14761							
SW15	= 100000	#111-4706								
SW2	= 000004	#111-4719								
SW3	= 000010	#111-4718								
SW4	= 000020	#111-4717	331-10883							
SW5	= 000040	#111-4716	465-14873							
SW6	= 000100	#111-4715	184-7202							
SW7	= 000200	#111-4714	470-15001							
SW8	= 000400	#111-4713	462-14754							
SW9	= 001000	#111-4712	462-14779	466-14888						
SYSSIZ	003754	145-5814	#145-5816							
TAG2$	012042	169-6753	#169-6780							
TAG3$	012076	169-6779	#169-6784							
TAG4$	027124	255-8834	#255-8836							
TAG70$	061744	470-14977	#472-15019							
TAG71$	061754	470-14978	#472-15025							
TAG72$	061764	470-14979	#472-15031							
TAG73$	062034	470-14980	#472-15045							
TAG74$	062074	470-14981	#472-15057							
TAG75$	062106	470-14982	#472-15063							
TAG76$	062120	470-14983	#473-15069							
TAG77$	062164	470-14984	#473-15082							
TAG78$	062172	470-14985	#473-15088							

SYMBOL CROSS REFERENCE CREF

SYMBOL	VALUE	REFERENCES
TAG79$	062252	470-14986 #473-15103
TAG9$	011670	169-6747 #169-6751 169-6786 169-6823 169-6845
TBG4$	027302	257-8867 #257-8869
TCFIG1	041726	#335-10951 335-10972 472-15036
TCFIG2	042066	#335-10981 335-11002 473-15074
TCFIG3	042222	#335-11011 335-11025 472-15049
TCONFI	041654	331-10888 331-10899 331-10909 #333-10933
TEMP	002430	#139-5693 *166-6538 166-6594 166-6596 *186-7209 186-7226 *186-7227 *395-12482 *395-12483 395-12484
TESTAD	002406	#139-5685 162-6282 *162-6283 *162-6284 166-6482 *166-6512 *166-6513 *166-6658 *166-6669 *166-6670 *211-7715 211-7716 *211-7716 *211-7719 *211-7721 *211-7740 211-7740 213-7792 213-7793 *219-7965 *219-7966 224-8154 224-8171 273-9200 273-9201 273-9287 273-9288 275-9310 275-9351 275-9368 277-9379 279-9460 283-9545 283-9552 283-9556 283-9607 285-9645 285-9654 287-9709 303-10050 390-12303 *390-12359 *390-12360 *390-12364 *390-12426 458-14649
TESTMO	002546	#141-5720 *145-5833 *145-5854 169-6771 169-6788 169-6805 169-6831 174-6951 213-7794 242-8552 244-8600 246-8646 257-8877 344-11215 377-12060 386-12231 388-12271 388-12289 421-13215 421-13231 421-13241 423-13261 423-13273
TIME	002334	#139-5661
TIMEOU	042556	145-5860 147-5886 151-5956 164-6446 167-6721 169-6781 #339-11103 382-12176 382-12180 384-12192 384-12204 386-12246 388-12298
TKVEC	= 000060	#111-4751 331-10869 331-10869 *331-10871 *331-10872 *331-10914 *331-10914 390-12303 390-12303 *390-12346 *390-12347 *390-12424 *390-12424 393-12440 393-12440 *393-12452 *393-12453 *393-12474 *393-12474 397-12511 397-12511 *397-12523 *397-12524 *397-12544 *397-12544
TMFLAG	002132	#139-5596 *201-7510 *203-7550 *205-7593 *207-7633 373-11967
TOOMAN	002402	#139-5683 *458-14644 465-14873 *466-14912
TOTCSR	002222	#139-5626 *157-6116 162-6287 163-6345 163-6377 166-6483 348-11260 350-11295 354-11351 406-12645 426-13337 428-13404 478-15193
TRACE	006204	#159-6179 *164-6427 *164-6433 164-6435 *164-6441 *164-6445 *402-12634 *404-12640 462-14736
TRAPVE	= 000034	#111-4750 *147-5878 *147-5879
TSTBAN	012310	169-6763 #169-6826
TSTDAT	002244	#139-5635 *273-9183 *273-9184 *273-9187 *273-9188 273-9189 273-9190 273-9191 *273-9197 *273-9199 273-9203 273-9205 273-9211 273-9217 *273-9276 *273-9277 *275-9302 *275-9303 275-9304 *275-9307 *275-9309 275-9313 275-9316 *277-9385 *277-9387 *277-9390 *277-9391 277-9397 *277-9400 *277-9402 *277-9404 *277-9406 277-9408 277-9410 *279-9441 *279-9442 *283-9524 *283-9525 *283-9526 *283-9529 283-9535 *283-9538 *283-9540 *283-9542 *283-9544 283-9547 283-9550 283-9558 283-9564 *283-9588 *283-9589 *285-9625 *285-9626 *285-9629 *285-9630 285-9635 *285-9638 *285-9640 *285-9642 *285-9644 285-9647 285-9649 285-9656 285-9663 *287-9690 *287-9691 *303-10045 *303-10046 303-10047 *303-10063 *303-10065 *303-10067 *303-10068 *303-10070 *303-10073 *303-10074 *303-10076 *303-10078 *303-10079 *303-10081 305-10107 305-10109 *316-10382 *316-10383 453-14545 453-14550 458-14650 458-14652 509-16397 509-16397
TSTRD1	043230	344-11201 344-11203 #344-11214
TSTRLA	= 104510	#110-4670 213-7810 213-7835 273-9237 273-9250 275-9326 275-9338
TST1	005_2	#157-6063
TST2	011404	#167-6694
TST3	011570	#169-6731
TST4	012556	#172-6879
TST5	014742	#187-7242
TST6	015014	#187-7259
TYPDS	= 104405	#110-4594 184-7190 184-7194 184-7198 184-7200 189-7298 395-12484 395-12486 395-12501 472-15025
TYPEIT	= 104401	#110-4590 154-5996 154-6010 154-6024 154-6029 154-6030 159-6142 159-6143 159-6152

SYMBOL	VALUE	REFERENCES								
		159-6153	159-6156	159-6172	159-6173	159-6177	163-6416	182-7145	184-7187	184-7191
		184-7195	184-7199	184-7201	189-7293	189-7295	211-7744	211-7746	331-10877	331-10878
		331-10879	331-10887	331-10891	331-10892	331-10893	331-10894	331-10895	331-10896	331-10905
		331-10906	331-10907	331-10908	333-10934	333-10937	333-10939	335-10949	335-10969	335-10980
		335-10999	335-11010	335-11022	335-11032	335-11040	335-11042	363-11632	363-11669	380-12099
		380-12102	380-12112	380-12117	382-12148	382-12178	384-12189	384-12193	384-12198	384-12202
		386-12211	386-12212	386-12237	386-12241	388-12253	388-12254	388-12277	388-12281	388-12284
		388-12286	388-12293	390-12305	390-12306	390-12315	390-12319	390-12324	390-12334	390-12337
		393-12442	393-12449	393-12451	393-12468	395-12485	395-12487	395-12495	395-12502	397-12513
		397-12520	397-12522	397-12538	399-12553	399-12562	399-12567	399-12573	399-12580	400-12599
		400-12603	400-12614	400-12619	402-12633	404-12639	406-12651	429-13451	434-13574	465-14832
		465-14833	466-14903	468-14931	468-14955	468-14956	468-14958	468-14965	468-14967	470-14994
		470-15004	472-15035	472-15039	473-15073	473-15082	473-15097	473-15103	473-15105	478-15191
		478-15192	478-15200	478-15209	478-15213	478-15215	478-15225	478-15228	478-15230	478-15234
		478-15240	478-15243	478-15245	478-15249	478-15251	481-15352	482-15417	483-15462	483-15471
		483-15472	483-15474	483-15483	483-15488	483-15497	483-15515	485-15524	485-15535	485-15537
		485-15539	486-15605	486-15618	486-15624	486-15629	486-15633	486-15638	486-15639	486-15641
		486-15644	486-15648	487-15709	487-15711	487-15712	487-15713	487-15768	487-15770	487-15771
		487-15772	494-15990							
TYPOC	= 104402	#110-4591	468-14936	472-15019	478-15199	478-15214	478-15216	478-15229	478-15231	478-15244
		478-15246	483-15473							
TYPOS	= 104403	#110-4592	211-7745	386-12234	388-12285	388-12294	395-12498	472-15057	472-15063	473-15104
		485-15538	485-15540							
TYPS0	= 000000	157-6111	159-6155	159-6176	160-6260	282-9508	291-9808	316-10395	319-10460	321-10512
		322-10556	323-10634	324-10675	325-10737					
TYPS1	= 000000	160-6259	291-9806	316-10393	323-10632	325-10736				
TYPS2	= 000000	325-10734								
T12A	033254	#285-9619	287-9688							
T12B	033276	#285-9623	287-9684							
UDPAR0	= 177660	#111-4807	234-8361							
UDPAR7	= 177676	#111-4814	*234-8359							
UIPAR0	= 177640	111-4796	#111-4797	160-6192	*160-6227	*160-6234	160-6259	*160-6272	361-11520	369-11823
UIPAR1	= 177642	#111-4798	*222-8100	*236-8416	*236-8421					
UIPAR2	= 177644	#111-4799	169-6749	222-8102	236-8417	*328-10849				
UIPAR3	= 177646	#111-4800	169-6750	238-8439	246-8679	355-11380	361-11537			
UIPAR4	= 177650	#111-4801	367-11785	367-11802						
UIPAR5	= 177652	#111-4802	220-8034	220-8058	*222-8125	*222-8145	242-8566	244-8593	246-8660	361-11562
		361-11583								
UIPAR6	= 177654	#111-4803	220-8034	220-8058	222-8126	222-8146	242-8566	244-8593	246-8660	*361-11583
UIPDR0	= 177600	#111-4776	361-11521	369-11825	425-13318	428-13418				
UNITOP	002412	#139-5686	*184-7179	*184-7182	*184-7183	*184-7184	*184-7185	184-7186	184-7186	
UNMAP	046774	227-8222	231-8261	247-8728	248-8739	250-8762	251-8771	251-8779	251-8787	251-8795
		251-8804	#369-11876							
UNRELO	046056	193-7381	195-7411	197-7448	199-7485	201-7523	203-7568	205-7606	207-7651	240-8537
		#365-11716	375-12002	377-12042						
UPPFLG	002263	#139-5640	*285-9650	287-9669	*287-9671					
USERMA	046556	363-11679	365-11738	367-11780	#369-11822					
USESTK	= 000700	#111-4681	155-6047	257-8880	478-15241	478-15247				
USP	=%000006	#111-4702	*155-6048	*257-8080	426-13327	*429-13434	478-15238			
WARN1	011756	#169-6764								
WARN2	027522	#261-8951	*327-10828							
WARN3	027536	#261-8956	*327-10829							

SYMBOL	VALUE	REFERENCES
WARN4	027562	#261-8968 *327-10830
WARN5	027573	#261-8973 *327-10831
WARN6	041306	#328-10849
WARN6A	041246	#328-10842
WARN6B	041300	328-10841 #328-10848
WARN7	024140	#238-8439
WASDBE	= 104500	#110-4662
WASSBE	= 104476	#110-4660
WAS1DB	= 104501	#110-4663 174-6961 174-6974 174-6989 174-7004 277-9414
WAS1SB	= 104477	#110-4661
WHICHC	053674	382-12170 384-12185 #406-12644
WOOPEN	055704	430-13479 430-13482 430-13484 432-13500 432-13523 #432-13531 432-13534
WOOPS	055336	425-13315 #430-13472
WOOPSA	055734	*430-13477 *430-13478 430-13479 432-13518 432-13519 432-13522 #432-13534
WOOPUP	055522	430-13479 430-13479 430-13480 430-13482 430-13482 430-13482 #432-13499 432-13523 432-13524 432-13534
WORST	002564	#141-5727 *147-5864 *197-7436 *199-7473 *203-7556 *207-7639 371-11944
XOCHAR	056276	#434-13588 *434-13622 *434-13625 *434-13626 434-13627 *434-13631 *434-13632 434-13633 483-15440 483-15442 *483-15443
XXDPCH	002350	#139-5667 *153-5983 466-14902
ZEROS	002332	#139-5660 145-5824 296-9896 296-9900
$APTHD	065740	137-5533 #493-15959
$AUTO	002040	#139-5575 *153-5974 *153-5979 155-6051 483-15469
$BANK	002011	#139-5555 *463-14792
$BASE	065714	#492-15931
$BELL	002637	#141-5753 393-12468 397-12538 465-14832
$CACHF	042720	#341-11151 496-16027
$CACHN	042674	#341-11144 496-16026
$CBCSR	043360	#346-11247 496-16070
$CBREG	044350	#355-11427 496-16085
$CB1CS	043402	#346-11252 496-16071
$CDW1	065720	#492-15934
$CDW2	065722	#492-15935
$CHARC	056464	*434-13576 *434-13586 *434-13644 #434-13649
$CHKDI	043754	#352-11339 496-16078
$CHK1D	043770	#352-11344 496-16079
$CKSWR	063536	#483-15434 496-16010
$CLRCS	043732	#352-11330 496-16076
$CLR1C	043744	#352-11334 496-16077
$CMTAG	002000	#139-5548 144-5795
$CMTGE	002540	#139-5714 144-5797
$CNTLC	064730	483-15483 486-15605 #486-15560
$CNTLG	064742	483-15471 #486-15662
$CNTLK	064136	483-15462 #483-15517
$CNTLU	064735	483-15488 486-15633 #486-15661
$CPUOP	065666	#492-15904
$CRLF	002644	#141-5755 184-7187 331-10891 331-10894 395-12502 434-13575 468-14931 468-14958 468-14967 478-15192 478-15213 478-15228 478-15243 478-15251 483-15497 485-15524 486-15638
$DBLK	063526	482-15387 482-15417 #482-15425
$DB20	065530	473-15095 #491-15851
$DDW0	065724	149-5898 #493-15946
$DDW1	065726	#493-15947

```
SYMBOL  VALUE           REFERENCES
$DDW2     065730        #493-15948
$DDW3     065732        #493-15949
$DDW4     065734        #493-15950
$DDW5     065736        #493-15951
$DEENE    042664        #341-11140  496-16022
$DEVCT    065650        *189-7329  *462-14730 *462-14732 #492-15891
$DEVM     065716        #492-15932
$DOAGA    015346         189-7300   189-7302  #189-7336
$DOAGN    015242        #189-7312
$DOWN     054726        #426-13368  426-13369
$DTBL     063516         482-15390 #482-15421
$ECCDI    043256        #346-11223  496-16066
$ECCIN    043304        #346-11231  496-16068
$ECC1D    043272        #346-11227  496-16067
$ECC1I    043320        #346-11235  496-16069
$ENASB    043332        #346-11239  496-16080
$ENA1S    043346        #346-11243  496-16081
$ENDAD    015232         137-5524   153-5980   154-5991  #189-7308
$ENERG    042654        #341-11136  496-16021
$ENV      065660         144-5789   153-5973  #492-15896
$ENVM     065661         153-5967   153-5970  #492-15900
$EOP      015066        #189-7284
$ERFLG    002012        #139-5556   462-14777 *462-14783 *465-14826
$ERRGE    044100        #355-11373  496-16084
$ERROR    060522         147-5876  #465-14822
$ERRTB    066262         468-14946 #500-16111
$ERRTY    061324         465-14877 #468-14930
$ERTTL    002614        #141-5739   189-7290   395-12486  395-12488 *465-14834 *465-14836  466-14902
$ESCAP    002356        #139-5673  *462-14786  466-14891  466-14893
$ETABL    065660        #492-15895
$ETEND    065740        #493-15955  493-15965
$EXHAL    047656        #377-12027
$ES     = 000001        #157-6114
$FATAL    065642        *465-14868 #492-15888
$FILLC    002636        #141-5752   434-13579
$FILLS    002353        #139-5670  *399-12557
$GTSWR    063712        #483-15472  496-16009
$HALT     061106        #466-14886
$HALT2    066014        #494-15991
$HIBTS    065740        #493-15960
$HIOCT    065140         386-12214  388-12256 *487-15704 #487-15715
$ILLUP    055330         425-13300  428-13378 #429-13457  429-13458
$INVAL    044050        #354-11364  496-16083
$ITEMB    002013        #139-5557  *465-14843  465-14845  465-14855 *465-14856  468-14933
$KERNE    042644        #341-11132  496-16020
$KMAP     045100        #361-11589  496-16024
$LF       002645        #141-5756   486-15648
$LOADC    042736        #343-11161  496-16029
$LPADR    002606        #141-5736   255-8824  *255-8834   255-8835  *255-8851   257-8857  *257-8867   257-8868  *257-8888
                        *462-14781 *462-14784  462-14788
$LPERR    002610        #141-5737   255-8824  *255-8835  *255-8851   257-8857  *257-8868  *257-8888   462-14781 *462-14785
                         466-14889
```

SYMBOL CROSS REFERENCE CREF

SYMBOL	VALUE	REFERENCES
L	= 000000	#157-6111 #159-6155 #159-6176 #160-6259 #160-6260 #282-9508 #291-9806 #291-9808 #316-10393 #316-10395 #319-10460 #321-10512 #322-10556 #323-10632 #323-10634 #324-10675 #325-10734 #325-10736 #325-10737
$MADR1	065672	#492-15918
$MADR2	065676	#492-15922
$MADR3	065702	#492-15925
$MADR4	065706	#492-15928
$MAIL	065640	#492-15886 493-15961 493-15965
$MAMS1	065670	186-7210 #492-15911
$MAMS2	065674	#492-15920
$MAMS3	065700	#492-15923
$MAMS4	065704	#492-15926
$MBADR	065742	#493-15961
$MNEW	064760	483-15474 #486-15664
$MSGAD	065654	#492-15893
$MSGLG	065656	#492-15894
$MSGTY	065640	*466-14899 #492-15887
$MSWR	064747	483-15472 #486-15663
$MTYP1	065671	#492-15912
$MTYP2	065675	#492-15921
$MTYP3	065701	#492-15924
$MTYP4	065705	#492-15927
$NOTRA	066010	#494-15990 496-16005 496-16007 496-16060 496-16061 496-16062 496-16063 496-16064 496-16086 496-16087 496-16088 496-16089 496-16090 496-16091
$NULL	002352	#139-5669 434-13581
$NWTST	= 000001	#157-6063 157-6063 #157-6063 #167-6694 167-6694 #167-6694 #169-6731 169-6731 #169-6731 #172-6879 172-6879 #172-6879 #187-7242 187-7242 #187-7242 #187-7259 187-7259 #187-7259
$OCNT	063306	*481-15324 *481-15353 #481-15366
$OCTVL	065622	491-15853 #491-15878 491-15381
$OCT8	= 065626	473-15097 #491-15881
$OMODE	063310	*481-15319 *481-15323 481-15328 *481-15331 *481-15342 #481-15368
$OVER	060436	462-14761 462-14782 #462-14787
$PASS	065646	*153-5966 *189-7291 *189-7292 189-7294 189-7298 189-7326 *189-7332 226-8179 226-8188 226-8203 227-8214 229-8228 229-8238 231-8253 234-8373 363-11614 395-12482 #492-15890
$PASTM	065746	#493-15963
$PATMA	002010	#139-5554 429-13447 429-13448 *463-14793 *463-14797 463-14802 463-14803 465-14829
$PER01	056506	#451-14478 496-16032
$PER02	056534	#451-14484 496-16033
$PER03	056562	#451-14490 496-16034
$PER04	056612	#451-14496 496-16035
$PER07	056674	#451-14512 496-16036
$PER10	056716	#451-14517 496-16037
$PER11	056746	#451-14522 496-16038
$PER12	056766	#451-14527 496-16039
$PER13	057010	#453-14533 496-16040
$PER14	057030	#453-14538 496-16041
$PER15	057052	#453-14543 496-16042
$PER16	057074	#453-14548 496-16043
$PER17	057114	#453-14553 496-16044
$PER20	057132	#453-14558 496-16045
$PER21	057150	#453-14563 496-16046
$PER22	057170	#453-14568 496-16047

SYMBOL	VALUE	REFERENCES
$PER23	057206	#453-14573 496-16048
$PER24	057224	#453-14578 496-16049
$PER25	053762	#421-13213 496-16050
$PER26	057414	#456-14615 496-16051
$PER27	057434	#456-14619 496-16052
$PER30	054210	#423-13258 496-16053
$PER31	057624	#458-14658 496-16054
$PER32	057722	#460-14676 496-16055
$PER33	057770	#460-14685 496-16056
$PER34	060050	#460-14696 496-16057
$PER35	060102	#460-14705 496-16058
$PER36	060136	#460-14713 496-16059
$PWRDN	054356	147-5880 #425-13287 429-13450
$PWRUP	054732	426-13367 #428-13374 432-13530
$QUES	002643	#141-5754 483-15515 486-15641
$RAND	065424	#489-15822
$RDCHR	064256	#486-15567 496-16012
$RDDEC	065142	#487-15730 496-16015
$RDLIN	064406	#486-15596 496-16013
$RDOCT	064772	#487-15679 496-16014
$READC	043032	#343-11179 496-16030
$RESRE	065366	#488-15803 496-16018
$SAVRE	065330	#488-15792 496-16017
$SAVR6	055334	*426-13365 428-13380 *428-13381 *428-13382 #429-13459 432-13513
$SCOPE	060166	147-5874 #462-14730
$STN	= 000001	#157-6063 #167-6694 #169-6731 #172-6879 #187-7242 #187-7259
$SVLAD	060422	462-14769 462-14778 #462-14784
$SWR	= 163000	#108-4554 157-6063 167-6694 169-6731 172-6879 187-7242 187-7259
$SWREG	065662	153-5976 #492-15902
$TESTN	065644	*465-14829 470-15011 #492-15889
$TKB	002630	#141-5749 331-10873 331-10913 390-12348 390-12425 393-12454 393-12473 397-12525 397-12543 434-13625 434-13631 483-15447 483-15479 485-15551 486-15571 486-15577
$TKS	002626	#141-5748 331-10875 331-10912 390-12350 390-12423 393-12456 393-12472 397-12527 397-12542 434-13623 434-13629 483-15445 483-15477 485-15549 486-15569 486-15575
$TN	= 000007	#108-4555 157-6063 157-6063 #157-6063 167-6694 167-6694 #167-6694 169-6731 169-6731 #169-6731 172-6879 172-6879 #172-6879 187-7242 187-7242 #187-7242 187-7259 187-7259 #187-7259
$TPB	002634	#141-5751 434-13638
$TPFLG	002354	#139-5671 *153-5968 434-13558
$TPS	002632	#141-5750 434-13620
$TRAP	065754	147-5878 #494-15975
$TRAP2	065776	#494-15986 496-16001
$TRPAD	066016	494-15980 #496-16001
$TSTM	065744	#493-15962
$TSTRD	043052	#344-11186 496-16082
$TTYIN	064704	486-15598 486-15599 486-15621 486-15639 486-15653 #486-15657
$TYPDS	063312	#482-15380 496-16006
$TYPE	056152	#434-13558 496-16002
$TYPEC	056300	434-13578 434-13585 #434-13589 483-15501
$TYPEX	056466	434-13645 434-13647 #434-13650
$TYPOC	063110	#481-15322 496-16003
$TYPON	063124	481-15321 #481-15324

SYMBOL CROSS REFERENCE CREF

SYMBOL	VALUE	REFERENCES
$TYPOS	063064	#481-15317 496-16004
$T2	= 000053	157-6111 159-6155 159-6176 160-6259 160-6260 282-9508 291-9806 291-9808 316-10393 316-10395 319-10460 321-10512 322-10556 323-10632 323-10634 324-10675 325-10734 325-10736 325-10737
$UNIT	065652	*189-7330 *462-14733 #492-15892
$UNITM	065750	#493-15964
$USWR	065664	189-7326 #492-15903
$VECT1	065710	#492-15929
$VECT2	065712	#492-15930
$WASDB	043566	#350-11294 496-16074
$WASSB	043422	#348-11259 496-16072
$WAS1D	043702	#350-11320 496-16075
$WAS1S	043536	#348-11285 496-16073
$XTSTR	060314	#462-14763
$ZAP42	015212	#189-7301 466-14906
$$T	= 000404	#157-6111 #157-6111 #159-6155 #159-6155 #159-6176 #159-6176 #160-6259 #160-6259 #160-6260 #160-6260 #282-9508 #282-9508 #291-9806 #291-9806 #291-9808 #291-9808 #316-10393 #316-10393 #316-10395 #316-10395 #319-10460 #319-10460 #321-10512 #321-10512 #322-10556 #322-10556 #323-10632 #323-10632 #323-10634 #323-10634 #324-10675 #324-10675 #325-10734 #325-10734 #325-10736 #325-10736 #325-10737 #325-10737
$OFILL	063307	*481-15318 *481-15322 481-15332 #481-15367

MACRO NAME	REFERENCES									
AND	#108-4544									
ANDB	#108-4547									
BEGIN	#108-4545	#182-7052	#209-7661	#211-7710	#348-11262	#350-11297	#354-11352	#363-11616	#363-11636	#400-12601
	#426-13338	#428-13406	#478-15194							
BMOV	#126-5283	169-6748	220-8022	220-8044	220-8070	222-8094	222-8097	222-8098	222-8099	222-8123
	222-8124	222-8143	222-8144	232-8296	232-8299	232-8303	232-8306	234-8355	234-8356	234-8357
	236-8412	236-8413	236-8415	236-8420	238-8438	240-8507	242-8553	244-8609	246-8647	246-8678
	328-10848	344-11203	363-11601	365-11740	367-11799	367-11815	430-13479	430-13482		
CASE	#108-4544	#215-7862	#380-12119							
CLEAR	#108-4542	#144-5792	#184-7151	#213-7797	#213-7804	#213-7852	#240-8530	#240-8541	#273-9289	#279-9461
	#283-9608	#327-10821	#331-10886	#331-10904	#352-11330	#352-11334	#363-11673	#365-11733	#365-11750	#371-11905
	#371-11907	#371-11908	#466-14912							
CLEARB	#108-4542									
DLEFT	#135-5485	267-9053	267-9072	273-9261	273-9267	275-9358	275-9364	279-9427	279-9433	283-9575
	283-9581	287-9677	287-9683							
DOWNTO	#108-4545									
ELSE	#108-4544	153-5977	153-5982	154-5995	158-6135	159-6162	159-6169	160-6199	160-6230	160-6248
	160-6255	167-6726	170-6867	170-6872	172-6909	173-6941	174-6979	174-6994	182-7064	184-7160
	184-7163	186-7217	217-7910	217-7925	226-8195	246-8686	253-8816	291-9766	291-9802	296-9898
	296-9914	297-9941	316-10365	316-10389	323-10618	325-10700	325-10715	325-10730	325-10748	335-10991
	335-10994	344-11209	348-11279	350-11314	377-12026	382-12153	382-12161	390-12368	393-12446	397-12517
	421-13221	421-13252	434-13635	454-14590	454-14602	458-14651	460-14699	466-14909	478-15233	478-15248
	485-15555									
END	#108-4545	149-5927	149-5929	151-5942	151-5959	153-5969	153-5972	153-5984	153-5985	153-5986
	154-5997	154-5998	155-6054	155-6055	157-6098	157-6104	157-6110	158-6134	158-6137	159-6164
	159-6165	159-6168	159-6171	160-6201	160-6212	160-6222	160-6232	160-6237	160-6250	160-6252
	160-6258	160-6270	167-6728	169-6841	169-6844	170-6870	170-6874	172-6911	172-6920	173-6931
	173-6932	173-6933	173-6934	173-6935	173-6936	173-6943	174-6981	174-6996	174-7005	174-7006
	174-7007	174-7010	#182-7063	#182-7066	182-7068	182-7069	182-7070	182-7071	184-7162	184-7165
	184-7166	184-7167	184-7176	184-7180	184-7181	184-7204	186-7219	186-7230	186-7233	186-7235
	186-7236	186-7239	186-7240	187-7267	189-7297	189-7334	189-7335	191-7348	191-7349	209-7665
	209-7666	211-7735	#211-7739	211-7740	211-7741	211-7742	211-7748	211-7749	211-7751	213-7818
	213-7843	215-7903	217-7912	217-7919	217-7927	219-7971	#226-8180	#226-8189	226-8197	#226-8204
	#227-8215	#229-8229	#229-8239	#231-8254	#234-8374	238-8457	238-8463	238-8464	238-8465	238-8466
	238-8470	238-8486	238-8492	238-8493	238-8494	238-8495	240-8521	240-8522	240-8523	240-8532
	240-8539	246-8695	253-8818	273-9226	273-9227	282-9485	282-9492	282-9506	291-9768	291-9786
	291-9798	291-9805	296-9901	296-9917	296-9920	297-9936	297-9940	297-9945	297-9949	297-9950
	297-9953	297-9956	297-9957	297-9964	297-9969	315-10346	316-10367	316-10385	316-10392	317-10420
	319-10457	321-10497	321-10507	322-10543	322-10553	323-10599	323-10613	323-10630	324-10662	324-10671
	325-10702	325-10717	325-10725	325-10733	325-10747	325-10752	325-10753	326-10783	326-10793	335-10993
	335-10996	344-11211	#348-11270	348-11271	348-11273	348-11274	348-11281	#350-11305	350-11306	350-11508
	350-11309	350-11316	354-11357	354-11359	354-11360	#363-11615	#363-11629	363-11630	363-11633	363-11635
	#363-11648	363-11652	363-11653	363-11654	363-11655	363-11656	363-11663	363-11664	363-11670	363-11672
	363-11690	371-11919	371-11949	377-12029	377-12045	380-12105	380-12143	382-12155	382-12165	390-12317
	390-12327	390-12336	390-12365	390-12372	393-12448	393-12466	393-12467	395-12497	395-12503	395-12504
	395-12505	397-12519	397-12536	397-12537	#400-12608	400-12612	400-12613	400-12628	421-13218	421-13223
	421-13254	423-13264	426-13344	426-13346	426-13347	428-13412	428-13414	428-13415	434-13637	454-14592
	454-14604	458-14653	458-14660	458-14665	458-14668	458-14671	460-14701	462-14734	462-14757	462-14760
	465-14837	465-14838	465-14867	465-14872	465-14875	465-14876	465-14881	466-14890	466-14901	466-14907
	466-14908	466-14911	470-15003	478-15201	478-15203	478-15204	478-15217	478-15232	478-15235	478-15247
	478-15250	485-15536	485-15557							
FATAL	#115-4987	167-6715	167-6719	303-10062	337-11079	339-11104	339-11108	339-11111		

MACRO NAME	REFERENCES									
FOR	#108-4545	149-5923	172-6899	182-7041	184-7155	184-7170	184-7177	186-7211	191-7341	211-7706
	211-7707	211-7715	211-7726	238-8442	238-8443	238-8471	238-8472	240-8514	296-9894	296-9902
	297-9962	315-10322	#348-11263	#350-11298	#354-11353	#363-11617	363-11640	393-12461	395-12490	397-12531
	400-12626	#426-13339	#428-13407	#478-15195	478-15212	478-15227	478-15242			
GOTO	#108-4546	390-12316	393-12469	397-12539	465-14874					
IF	#108-4544	151-5937	151-5952	#153-5978	153-5980	154-5991	154-5992	155-6051	155-6052	157-6096
	157-6101	158-6128	158-6132	159-6158	159-6159	159-6160	159-6166	160-6197	160-6207	160-6217
	160-6228	160-6235	160-6243	160-6246	160-6253	160-6261	160-6265	167-6724	169-6838	169-6842
	170-6863	170-6864	172-6903	172-6904	172-6905	172-6906	172-6916	172-6918	173-6923	173-6939
	174-6976	174-6991	182-7045	182-7050	#182-7053	182-7059	#184-7156	184-7157	184-7158	#184-7178
	184-7186	184-7202	186-7208	186-7215	186-7237	187-7262	189-7294	189-7325	189-7326	191-7343
	#209-7662	209-7663	211-7679	#211-7680	211-7712	211-7743	213-7811	213-7817	213-7836	213-7842
	217-7908	217-7917	217-7923	219-7969	222-8076	226-8178	226-8179	226-8181	226-8187	226-8188
	226-8190	226-8193	226-8202	226-8203	226-8205	227-8213	227-8214	227-8216	229-8227	229-8228
	229-8230	229-8236	#229-8237	229-8238	231-8252	231-8253	231-8256	234-8372	234-8373	234-8375
	238-8445	238-8452	238-8458	238-8467	238-8474	238-8481	238-8487	240-8516	240-8517	240-8524
	246-8680	246-8713	247-8720	248-8732	249-8747	250-8756	251-8766	251-8774	251-8782	251-8790
	251-8798	253-8814	273-9222	273-9224	273-9263	273-9264	275-9360	275-9361	279-9429	279-9430
	282-9483	282-9488	282-9501	283-9577	283-9578	287-9679	287-9680	291-9782	291-9793	296-9895
	#296-9909	#296-9910	#296-9911	296-9923	#297-9930	#297-9931	#297-9932	297-9934	297-9938	297-9943
	297-9947	316-10379	317-10416	319-10453	321-10493	321-10502	322-10539	322-10548	323-10593	323-10607
	324-10658	324-10667	325-10713	325-10721	325-10743	325-10744	#325-10749	326-10778	326-10788	331-10883
	335-10970	335-10988	335-10989	335-11000	335-11023	343-11163	344-11207	348-11267	348-11272	350-11302
	350-11307	354-11358	363-11612	#363-11613	363-11614	363-11619	363-11631	#363-11641	363-11642	363-11643
	363-11644	363-11645	363-11649	363-11657	363-11658	363-11666	371-11917	371-11947	375-12002	377-12024
	377-12042	#377-12043	380-12101	382-12150	382-12159	388-12263	388-12265	388-12266	390-12309	390-12314
	390-12322	390-12323	390-12333	390-12361	390-12366	393-12444	393-12463	395-12488	395-12494	397-12515
	397-12533	400-12606	421-13214	421-13219	421-13227	421-13250	423-13260	425-13315	426-13345	428-13413
	434-13633	454-14585	#454-14586	454-14588	454-14597	#454-14598	454-14600	456-14609	458-14649	458-14658
	458-14661	458-14663	458-14666	458-14669	460-14676	460-14685	460-14696	#460-14697	462-14731	462-14754
	462-14758	462-14761	465-14823	465-14835	465-14860	465-14861	465-14870	465-14873	465-14878	466-14883
	466-14888	466-14894	466-14898	466-14902	470-15001	478-15202	478-15226	478-15241	485-15534	485-15553
IFB	#108-4547	153-5967	153-5970	153-5973	186-7212	186-7228	186-7231	291-9764	291-9800	316-10363
	316-10387	323-10615	325-10698	325-10728	395-12493					
JUMPTO	#108-4546	157-6118								
LEAVE	#108-4546	#182-7062	#182-7065	#211-7738	#348-11269	#350-11304	#363-11628	#363-11647	#400-12607	
LET	#108-4546	#157-6095	#157-6116	#158-6127	#158-6130	#158-6131	#160-6193	#160-6195	#160-6196	#160-6198
	#160-6200	#160-6202	#160-6204	#160-6205	#160-6208	#160-6209	#160-6213	#160-6214	#160-6218	#160-6219
	#160-6223	#160-6227	#160-6229	#160-6231	#160-6238	#160-6239	#160-6241	#160-6244	#160-6245	#160-6262
	#160-6263	#160-6266	#160-6267	#160-6271	#184-7159	#184-7161	#184-7164	#184-7179	#211-7729	#227-8219
	#231-8258	#238-8447	#238-8475	#249-8750	#249-8751	#251-8769	#251-8777	#251-8785	#251-8793	#251-8801
	#251-8802	#282-9474	#282-9475	#282-9476	#282-9478	#282-9480	#282-9481	#282-9490	#282-9499	#282-9503
	#282-9504	#291-9761	#291-9762	#291-9763	#291-9765	#291-9767	#291-9771	#291-9772	#291-9775	#291-9777
	#291-9783	#291-9784	#291-9790	#291-9795	#291-9796	#296-9896	#296-9897	#296-9899	#296-9900	#296-9906
	#296-9907	#296-9912	#296-9913	#296-9915	#296-9916	#296-9918	#296-9919	#296-9925	#296-9926	#297-9933
	#297-9937	#297-9942	#297-9946	#297-9951	#297-9952	#297-9965	#297-9968	#316-10360	#316-10361	#316-10362
	#316-10364	#316-10366	#316-10371	#316-10372	#316-10375	#316-10377	#316-10380	#316-10381	#316-10382	#316-10383
	#317-10410	#317-10411	#317-10413	#317-10415	#317-10417	#317-10418	#319-10430	#319-10433	#319-10434	#319-10435
	#319-10440	#319-10441	#319-10442	#319-10444	#319-10448	#319-10450	#319-10454	#319-10455	#319-10458	#321-10473
	#321-10478	#321-10479	#321-10486	#321-10488	#321-10494	#321-10495	#321-10498	#321-10503	#321-10504	#321-10508
	#321-10509	#322-10528	#322-10529	#322-10532	#322-10534	#322-10536	#322-10540	#322-10541	#322-10550	#322-10551
	#322-10555	#323-10568	#323-10569	#323-10571	#323-10573	#323-10576	#323-10578	#323-10579	#323-10580	#323-10582

MACRO NAME	REFERENCES
	#323-10588 #323-10591 #323-10594 #323-10595 #323-10596 #323-10605 #323-10608 #323-10609 #323-10610 #323-10631
	#323-10633 #324-10648 #324-10649 #324-10650 #324-10654 #324-10660 #324-10668 #324-10673 #324-10674 #325-10689
	#325-10694 #325-10695 #325-10696 #325-10697 #325-10699 #325-10701 #325-10704 #325-10706 #325-10707 #325-10710
	#325-10712 #325-10722 #326-10763 #326-10768 #326-10770 #326-10771 #326-10773 #326-10779 #326-10780 #326-10789
	#326-10790 #363-11677 #365-11737 #421-13213 #421-13216 #421-13220 #421-13222 #421-13239 #423-13258 #423-13259
	#423-13262 #451-14517 #451-14518 #451-14519 #451-14522 #451-14523 #451-14524 #451-14527 #451-14528 #451-14529
	#453-14533 #453-14534 #453-14535 #453-14538 #453-14539 #453-14540 #453-14543 #453-14544 #453-14545 #453-14548
	#453-14549 #453-14550 #453-14553 #453-14554 #453-14555 #453-14558 #453-14559 #453-14560 #453-14563 #453-14564
	#453-14565 #453-14568 #453-14569 #453-14570 #453-14573 #453-14574 #453-14575 #453-14578 #453-14579 #453-14580
	#456-14615 #456-14616 #456-14620
MAP	#123-5346 #155-6059 #169-6761 #173-6937 #174-6948 #211-7708 #242-8551 #244-8580 #246-8644 #255-8822
	#257-8855 #377-12059 #386-12230 #388-12269 #390-12330 #390-12363 #390-12428 #393-12475 #397-12545 #430-13475
	#432-13516
NEWTST	#117-5037 #157-6063 #167-6694 #169-6731 #172-6879 #187-7242 #187-7259
ON.ERR	#108-4547 157-6108 174-7008 193-7378 195-7408 197-7445 199-7482 201-7520 203-7565 205-7603
	207-7648 233-8313 233-8321 234-8331 240-8528 242-8547 244-8576 246-8640 348-11265 348-11277
	350-11300 350-11312 354-11355 363-11688 426-13341 428-13409 478-15197
ON.NOE	#108-4547 149-5925 174-6965 174-6975 174-6990 211-7723
OR	#108-4544
ORB	#108-4547
POP	#108-4546 #155-6059 #160-6272 #166-6540 #166-6637 #169-6761 #169-6837 #173-6937 #174-6948 #174-7021
	#174-7023 #182-7104 #189-7306 #211-7708 #211-7734 #211-7761 #213-7848 #213-7856 #242-8551 #244-8580
	#246-8644 #255-8822 #255-8851 #257-8855 #257-8888 #327-10832 #331-10914 #343-11174 #343-11182 #344-11206
	#348-11276 #350-11311 #354-11361 #354-11370 #355-11408 #355-11424 #357-11461 #361-11586 #361-11607 #365-11732
	#365-11758 #365-11770 #369-11852 #369-11873 #371-11959 #377-12059 #379-12091 #380-12114 #382-12154 #382-12156
	#382-12164 #384-12195 #386-12215 #386-12230 #386-12245 #388-12257 #388-12269 #388-12288 #388-12297 #390-12308
	#390-12321 #390-12326 #390-12330 #390-12363 #390-12371 #390-12424 #390-12426 #390-12427 #390-12428 #393-12474
	#393-12475 #395-12506 #397-12544 #397-12545 #399-12555 #399-12558 #400-12605 #406-12653 #421-13249 #423-13278
	#428-13385 #428-13389 #428-13395 #428-13401 #428-13402 #428-13410 #429-13428 #429-13432 #429-13439 #429-13444
	#429-13449 #429-13455 #430-13475 #430-13495 #432-13516 #432-13529 #434-13650 #434-13653 #434-13654 #456-14628
	#458-14645 #463-14804 #472-15038 #472-15051 #473-15076 #476-15180 #478-15205 #478-15224 #478-15239 #478-15253
	#482-15416 #485-15544 #485-15548 #487-15705 #487-15763 #488-15808 #489-15837
PUSH	#108-4546 155-6041 155-6047 155-6059 160-6192 166-6520 166-6625 169-6761 169-6826 173-6937
	174-6948 174-6952 174-6954 182-7072 189-7304 #211-7681 211-7708 211-7730 213-7791 242-8551
	244-8580 246-8644 255-8822 255-8824 257-8855 257-8857 327-10809 331-10869 343-11161 343-11179
	344-11186 348-11259 350-11294 354-11350 354-11365 355-11374 355-11379 357-11442 361-11514 361-11589
	365-11718 365-11723 365-11762 369-11843 369-11867 371-11904 377-12059 379-12075 379-12080 379-12084
	380-12108 380-12109 386-12209 386-12230 388-12251 388-12269 390-12303 390-12330 390-12363 #390-12369
	390-12428 393-12440 393-12475 395-12481 397-12511 397-12545 399-12552 #421-13228 423-13268 425-13298
	425-13302 425-13306 425-13313 426-13329 426-13335 426-13343 426-13349 426-13352 426-13356 426-13361
	426-13363 430-13473 430-13475 432-13514 432-13516 434-13589 434-13593 434-13594 456-14624 458-14634
	463-14794 472-15031 472-15045 473-15069 476-15156 476-15172 478-15190 478-15208 482-15380 485-15523
	487-15681 487-15732 488-15793 489-15822
RCC	#108-4543
RCS	#108-4543
REPEAT	#108-4545 #157-6088 #159-6145 #159-6157 #160-6224 #160-6233 #282-9495 #291-9760 #291-9769 #316-10359
	#316-10368 #319-10438 #321-1048. #322-10530 #323-10572 #323-10577 #324-10651 #325-10688 #325-10693 #325-10703
	#400-12602
REQ	#108-4543
RES4	#133-5458 #145-5860 #151-5956 #164-6446 #167-6721 #169-6781 #382-12176 #382-12180 #384-12192 #384-12204
	#386-12246 #388-12298
RGE	#108-4543

MACRO NAME REFERENCES

RGT #108-4543
RHI #108-4543
RHIS #108-4543
RLE #108-4543
RLO #108-4543
RLOS #108-4543
RLT #108-4543
RMI #108-4543
RNE #108-4543 157-6114
RPL #108-4543
SET #108-4542 #137-5540 #147-5864 #147-5866 #153-5968 #153-5971 #153-5974 #153-5979 #153-5981 #153-5983
#154-5993 #154-6012 #154-6015 #166-6602 #166-6646 #169-6821 #169-6840 #174-7027 #182-7067 #182-7146
#187-7263 #187-7265 #191-7344 #191-7347 #209-7656 #209-7657 #209-7671 #211-7727 #211-7737 #211-7759
#213-7812 #213-7837 #232-8265 #233-8324 #234-8332 #238-8441 #240-8502 #240-8513 #240-8525 #242-8548
#244-8577 #246-8641 #253-8810 #273-9239 #273-9242 #273-9252 #273-9255 #275-9328 #275-9331 #275-9342
#275-9345 #277-9416 #277-9419 #282-9489 #282-9502 #291-9794 #305-10121 #305-10133 #321-10505 #322-10549
#323-10597 #323-10611 #324-10659 #324-10669 #325-10723 #325-10745 #325-10750 #326-10781 #326-10791 #337-11069
#348-11268 #350-11303 #363-11650 #363-11712 #371-11906 #371-11915 #371-11927 #371-11931 #371-11934 #371-11937
#371-11943 #371-11948 #371-11952 #371-11955 #371-11958 #380-12111 #382-12174 #384-12190 #384-12199 #390-12353
#393-12459 #395-12496 #397-12529 #399-12563 #399-12569 #400-12615 #458-14644 #458-14655 #458-14672 #460-14680
#460-14682 #460-14691 #460-14693 #460-14715 #460-14717 #466-14910 #472-15034 #472-15048 #473-15072 #476-15157
#476-15169 #476-15173 #476-15175
SETB #108-4542
SET4 #133-5446 145-5816 145-5818 145-5825 145-5844 145-5856 151-5949 154-6004 154-6006 154-6008
157-6087 163-6339 163-6355 163-6375 163-6408 164-6426 166-6480 167-6705 169-6742 375-12001
382-12172 384-12187 386-12229 388-12268
SMACIT #108-4541 108-4556
SUBTST #119-5102 144-5788 144-5800 145-5813 147-5863 147-5873 149-5893 149-5922 151-5933 151-5945
153-5963 154-5988 154-6000 155-6032 155-6050 155-6057 158-6120 159-6140 160-6181 164-6419
166-6449 167-6723 170-6856 182-7036 184-7150 186-7207 187-7273 191-7339 193-7355 195-7385
197-7415 199-7452 201-7489 203-7527 205-7572 207-7610 209-7655 211-7675 213-7766 215-7861
217-7907 219-7962 220-8012 220-8025 220-8047 222-8075 222-8111 222-8129 224-8151 224-8158
224-8168 226-8177 226-8186 226-8201 227-8212 229-8226 229-8235 229-8244 231-8251 232-8264
233-8311 233-8319 234-8329 234-8371 236-8382 238-8429 240-8500 242-8545 244-8574 246-8638
246-8668 246-8698 247-8719 248-8731 248-8743 249-8746 250-8755 251-8765 251-8773 251-8781
251-8789 251-8797 253-8808 253-8813 255-8821 259-8902 259-8913 259-8925 261-8939 261-8962
263-8980 263-8988 265-8998 265-9011 265-9020 267-9034 269-9077 271-9117 273-9153 275-9294
277-9376 282-9467 283-9515 285-9614 289-9716 291-9750 292-9818 296-9888 297-9960 299-9982
301-10026 301-10034 303-10044 307-10138 307-10145 307-10163 307-10176 309-10184 309-10190 311-10241
313-10261 314-10307 315-10319 316-10351 317-10404 319-10425 321-10466 322-10519 323-10560 324-10640
325-10680 326-10756 327-10802 327-10808 328-10835 329-10854 331-10868 333-10920 339-11119 354-11349
354-11364 355-11373 355-11427 357-11434 361-11503 363-11611 365-11716 365-11761 367-11774 369-11822
369-11842 369-11855 369-11865 369-11876 371-11883 373-11963 373-11974 373-11982 373-11986 375-11993
377-12022 377-12031 377-12058 379-12068 380-12097 382-12147 382-12169 384-12184 386-12208 388-12250
390-12302 391-12433 393-12439 395-12480 397-12510 399-12551 399-12561 399-12566 399-12572 399-12579
400-12598 400-12618 402-12632 404-12638 406-12644 421-13226 423-13267 430-13472 432-13499 454-14583
454-14639 456-14608 456-14623 458-14632 463-14791 476-15150 485-15522 485-15547
SUPERV #130-5375 255-8844 426-13332 429-13429 430-13476 432-13517
TESTAR #131-5418 #169-6771 #169-6788 #169-6805 #169-6831 #174-6951 #213-7794 #242-8552 #244-8600 #246-8646
#257-8877 #344-11215 #377-12060 #386-12231 #388-12271 #388-12289 #421-13215 #421-13231 #421-13241 #423-13261
#423-13273
THEN #108-4544

MACRO NAME REFERENCES

THRU #108-4545
TO #108-4545
TYPDEC #125-5247 #184-7190 #184-7194 #184-7198 #184-7200 #189-7298 #395-12484 #395-12486 #395-12501 #472-15025
TYPE #115-5001 #154-5996 154-6010 154-6024 154-6029 154-6030 159-6142 159-6143 159-6152 159-6153
159-6156 159-6172 159-6173 159-6177 163-6416 182-7145 #184-7187 184-7191 184-7195 184-7199
184-7201 189-7293 189-7295 211-7744 211-7746 331-10877 331-10878 331-10879 331-10887 331-10891
331-10892 331-10893 331-10894 331-10895 331-10896 331-10905 331-10906 331-10907 331-10908 333-10934
333-10937 333-10939 335-10949 335-10969 335-10980 335-10999 335-11010 335-11022 335-11032 335-11040
335-11042 363-11632 363-11669 380-12099 380-12102 380-12112 380-12117 382-12148 382-12178 384-12189
384-12193 384-12198 384-12202 386-12211 386-12212 386-12237 386-12241 388-12253 388-12254 388-12277
388-12281 388-12284 388-12286 388-12293 390-12305 390-12306 390-12315 390-12319 390-12324 390-12334
390-12337 393-12442 393-12449 393-12451 393-12468 395-12485 395-12487 395-12495 395-12502 397-12513
397-12520 397-12522 397-12538 399-12553 399-12562 399-12567 399-12573 399-12580 400-12599 400-12603
400-12614 400-12619 402-12633 404-12639 406-12651 429-13451 434-13574 465-14832 465-14833 466-14903
468-14931 468-14955 468-14956 468-14958 468-14965 468-14967 470-14994 470-15004 472-15035 472-15039
473-15073 473-15082 473-15097 473-15103 473-15105 478-15191 478-15192 478-15200 478-15209 #478-15213
478-15215 478-15225 #478-15228 478-15230 #478-15234 478-15240 #478-15243 478-15245 #478-15249 478-15251
481-15352 482-15417 483-15462 483-15471 483-15472 483-15474 483-15483 483-15488 483-15497 483-15515
485-15524 485-15535 485-15537 485-15539 486-15605 486-15618 486-15624 486-15629 486-15633 486-15638
486-15639 486-15641 486-15644 486-15648 487-15709 487-15711 487-15712 487-15713 487-15768 487-15770
487-15771 487-15772 494-15990
TYPOCS #123-5194 #211-7745 #386-12234 #388-12285 #388-12294 #395-12498 #472-15057 #472-15063 #473-15104 #485-15538
#485-15540
TYPOCT #121-5148 468-14936 472-15019 478-15199 478-15214 478-15216 478-15229 478-15231 478-15244 478-15246
483-15473
UNTIL #108-4545 157-6111 159-6155 159-6176 160-6259 160-6260 282-9508 291-9806 316-10393 324-10675
325-10734 325-10737
UNTILB #108-4547 291-9808 316-10395 319-10460 321-10512 322-10556 323-10632 323-10634 325-10736
USER #130-5396 363-11680 365-11739 367-11798 367-11814 426-13326 429-13433
WHILE #108-4544 296-9908 297-9929 297-9966
WHILEB #108-4547
$CALL #108-4548
$RETUR #108-4548 #213-7849 #213-7857 #253-8815 #253-8817 #363-11634 #363-11671 #363-11713
$SUBTS #119-5109 144-5788 144-5800 145-5813 147-5863 147-5873 149-5893 149-5922 151-5933 151-5945
153-5963 154-5988 154-6000 155-6032 155-6050 155-6057 158-6120 159-6140 160-6181 164-6419
166-6449 167-6723 170-6856 182-7036 184-7150 186-7207 187-7273 191-7339 193-7355 195-7385
197-7415 199-7452 201-7489 203-7527 205-7572 207-7610 209-7655 211-7675 213-7766 215-7861
217-7907 219-7962 220-8012 220-8025 220-8047 222-8075 222-8111 222-8129 224-8151 224-8158
224-8168 226-8177 226-8186 226-8201 227-8212 229-8226 229-8235 229-8244 231-8251 232-8264
233-8311 233-8319 234-8329 234-8371 236-8382 238-8429 240-8500 242-8545 244-8574 246-8638
246-8668 246-8698 247-8719 248-8731 248-8743 249-8746 250-8755 251-8765 251-8773 251-8781
251-8789 251-8797 253-8808 253-8813 255-8821 259-8902 259-8913 259-8925 261-8939 261-8962
263-8980 263-8988 265-8998 265-9011 265-9020 267-9034 269-9077 271-9117 273-9153 275-9294
277-9276 282-9467 283-9515 285-9614 289-9716 291-9750 292-9818 296-9888 297-9960 299-9982
301-10026 301-10034 303-10044 307-10138 307-10145 307-10163 307-10176 309-10184 309-10190 311-10241
313-10261 314-10307 315-10319 316-10351 317-10404 319-10425 321-10466 322-10519 323-10560 324-10640
325-10680 326-10756 327-10802 327-10808 328-10835 329-10854 331-10868 333-10920 339-11119 354-11349
354-11364 355-11373 355-11427 357-11434 361-11503 363-11611 365-11716 365-11761 367-11774 369-11824
369-11842 369-11855 369-11865 369-11876 371-11883 373-11963 373-11974 373-11982 373-11986 375-11993
377-12022 377-12031 377-12058 379-12068 380-12097 382-12147 382-12169 384-12184 386-12208 388-12250
390-12302 391-12433 393-12439 395-12480 397-12510 399-12551 399-12561 399-12566 399-12572 399-12579
400-12598 400-12618 402-12632 404-12638 406-12644 421-13226 423-13267 430-13472 432-13499 454-14583

```
MACRO NAME      REFERENCES
                 454-14595  456-14608  456-14623  458-14632  463-14791  476-15150  485-15522  485-15547
$$END           #108-4546  #517-16630
$$NEWT          #119-5069  #157-6063  #167-6694  #169-6731  #172-6879  #187-7242  #187-7259
.ARITH          #108-4547
.EMIT           #108-4541
.EMITL          #108-4541  #157-6111  #159-6155  #159-6176  #160-6259  #160-6260  #282-9508  #291-9806  #291-9808  #316-10393
                #316-10395 #319-10460 #321-10512 #322-10556 #323-10632 #323-10634 #324-10675 #325-10734 #325-10736 #325-10737
.EMITN          #108-4541   157-6114
.EMITR          #108-4541
.GENBR          #108-4542
.GOTO           #108-4544
.IFARI          #108-4544
.IFOPR          #108-4542
.IS             #108-4542
.LEAVE          #108-4544
.OPADD          #108-4542
.OPSUB          #108-4542
.OR             #108-4544
.SIMPL          #108-4547
..BRAN          #108-4541
..POP           #108-4541  #157-6111  #159-6155  #159-6176  #160-6259  #160-6260  #282-9508  #291-9806  #291-9808  #316-10393
                #316-10395 #319-10460 #321-10512 #322-10556 #323-10632 #323-10634 #324-10675 #325-10734 #325-10736 #325-10737
..PUSH          #108-4541
..TAG           #108-4541   157-6111   159-6155   159-6176   160-6259   160-6260   282-9508   291-9806   291-9808   316-10393
                 316-10395  319-10460  321-10512  322-10556  323-10632  323-10634  324-10675  325-10734  325-10736  325-10737
```