

MF11S-K

MEMORY DIAGNOSTIC
MD-11-DZMML-B

EP-DZMML-B-DL-A
COPYRIGHT © 73-77
FICHE 1 OF 1

OCT 1977
digital
MADE IN USA

This microfiche card contains a grid of frames, each displaying technical data. The frames are organized into rows and columns. The data within the frames includes:

- Hexadecimal memory addresses (e.g., 0000, 0001, 0002, etc.).
- Corresponding hexadecimal values (e.g., 00, 01, 02, etc.).
- ASCII character representations (e.g., NUL, SOH, STX, etc.).
- Binary data representations (e.g., 00000000, 00000001, etc.).
- Other diagnostic information such as parity checks and error flags.

The frames are arranged in a grid that is approximately 10 frames wide and 20 frames high. The data is presented in a clear, structured format for easy reading and analysis.

B01

EOF1DZDJBESQ

00010000

771026

PDP10 411

E H0R1DZMMLBSEQ

00010000

771026

POP10 411 SEQ 0002
DZMMLB.P11

DZMML

MACY11 30(1046)

05-AUG-77 12:59

PAGE 1

.REPT 0

IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DZMML-B-D
PRODUCT NAME: MF11S-K MEMORY DIAGNOSTIC
DATE CREATED: MAY 1, 1977
DATE MODIFIED: AUG 1, 1977
MAINTAINER: DIAGNOSTIC GROUP
AUTHOR: JAMES P. RYAN

COPYRIGHT (C) 1977
DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.

THIS SOFTWARE IS FURNISHED UNDER A LICENSE FOR USE ONLY ON A SINGLE COMPUTER SYSTEM AND MAY BE COPIED ONLY WITH THE INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE, OR ANY OTHER COPIES THEREOF, MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON EXCEPT FOR USE ON SUCH SYSTEM AND TO ONE WHO AGREES TO THESE LICENSE TERMS. TITLE TO AND OWNERSHIP OF THE SOFTWARE SHALL AT ALL TIMES REMAIN IN DEC.

THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION.

DEC ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DEC.

CONTENTS

1.0	ABSTRACT
1.1	GETTING STARTED
2.0	REQUIREMENTS
2.1	EQUIPMENT
2.2	STORAGE
3.0	LOADING PROCEDURE
4.0	STARTING PROCEDURE
4.1	SWITCH SETTINGS
4.2	CONTROL-C FUNCTION
4.3	STARTING ADDRESS =200
	RESTART ADDRESS =250
4.4	PROGRAM AND/OR OPERATOR ACTION
5.0	PROGRAM HALTS (NORMAL + ERROR)
6.0	ERRORS
6.1	ERROR MESSAGE FORMAT.
6.2	ERROR DICTIONARY
6.3	ERROR HISTORY
6.4	ERROR RECOVERY
6.5	MOS CHIP CROSS REFERENCE TABLE
7.0	RESTRICTIONS
8.0	MISCELLANEOUS
8.1	ADDRESS/BANK RANGES IN OCTAL AND DECIMAL
8.2	EXECUTION TIME
8.3	PASS COUNT AND TEST NO. LOCATIONS
8.4	STACK POINTER
8.6	POWER FAIL
9.0	PROGRAM DESCRIPTION
9.1	NARRATIVE FLOW CHART
9.2	TEST TITLES AND TEST TIMES
10.0	RXDP & ACT11 & APT OPERATION
11.01	ABSTRACT

THIS DIAGNOSTIC WILL TEST 0 - 124K OF MF11S-K MEMORY ON ANY PDP-11 FAMILY COMPUTER THAT DOES NOT PERFORM A DATIP CYCLE BEFORE A DATO CYCLE FOR ALL INST-RUCTIONS.

THIS PROGRAM CAN BE RUN UNDER XXDP, APT AND ACT MONITORS. ON PROCESSORS WITH NO HARDWARE SWITCH REGISTER, SOFTWARE SWITCH REGISTER = LOCATION 176 AND SOFTWARE DISPLAY

REGISTER = LOCATION 174.

[1.1] GETTING STARTED

IF NO HARDWARE SWITCH REGISTER SET LOCATION 176 TO OBTAIN SWITCH OPTIONS.

TO START:

- A. SET SWITCH REGISTER = 00000
- B. START AT 200.
- C. THE ADDRESS OF THE CONTROL AND STATUS REGISTER (CSR) AND THE MEMORY LIMITS IT CONTROLS WILL BE PRINTED. IF TWO CONTROLLERS ARE PRESENT, BOTH CSR ADDRESSES WILL BE TYPED AS WELL AS BOTH MEMORY LIMITS.
- D. SEE SECTION 4.4 FOR REST OF PRINTOUTS EXPECTED.
- E. "END PASS #01" WILL BE TYPED. THIS IS A QUICK VERIFY PASS AND THE TEST IS NOT COMPLETE UNTIL "END PASS #02" IS TYPED.
- F. TO HALT THE TEST, TYPE CONTROL-C. THIS WILL INSURE THE PROGRAM IS RELOCATED BACK TO LOWER MEMORY AND THAT THE MEMORY HAS BEEN PURGED OF ANY DOUBLE ERRORS FORCED DURING TESTING. BE PATIENT, THE CONTROL-C IS ONLY RECOGNIZED AT THE END OF THE CURRENT SUBTEST.
- G. IF AN UNEXPECTED HALT OCCURS SEE SECTION 6.0. IF AN ERROR # IS TYPED SEE SECTION 6.2.

!CAUTION! BEFORE "DIGGING" INTO THE LISTING READ SECTION 9.

SWITCH SETTING SUMMARY (SEE SECTION 4.1 FOR DETAILS)

BIT15(100000)	HALT ON ERROR
BIT14(040000)	LOOP IN SUBTEST DEFINED BY BITS <4:0>
BIT13(020000)	INHIBIT ERROR PRINTOUTS
BIT12(010000)	DISABLE TESTING ABOVE 28K (MEMORY MANAGEMENT)
BIT11(004000)	ENABLE PRINTOUTS OF SINGLE ARRAY ERRORS
BIT10(002000)	HALT AFTER EACH SUBTEST
BIT09(001000)	INHIBIT PROGRAM RELOCATION
BIT08(000400)	TYPE FIRST FAILING BIT ERROR PER 4K.
BIT07(000200)	ENABLE XOR PRINTOUT FOR ARRAY TESTS
BIT06(000100)	INHIBIT MEMORY SIZING
BIT05(000040)	INHIBIT "END PASS #XX" AND "RELOC" PRINTOUTS
BIT04	TO CLEAR MEMORY OF DBE'S(NO TESTS RUN)
BIT03-BIT00	BEGINNING TEST NUMBER.

[2.0] REQUIREMENTS

[2.1] EQUIPMENT

STANDARD 11 FAMILY COMPUTER WITH A CONSOLE OUTPUT DEVICE

AND FROM 32K TO 124K OF MS11K MEMORY (WITH EXCEPTION IN 1.0) .

[2.2] STORAGE

PROGRAM STORAGE - 0000 - 14200. PROGRAM EXPANDS FOR ERROR HISTORY AND TO SAVE ABSOLUTE LOADER OR XXDP CHAIN MONITOR. (SEE SECTION 9. FOR DETAILS)

[3.0] LOADING PROCEDURE

USE STANDARD PROCEDURE FOR PDP-11 ABSOLUTE BINARY FORMATTED TAPES.

[4.0] STARTING PROCEDURE

[4.1] SWITCH SETTINGS

SOFTWARE SWITCH REGISTER = LOCATION 176

BIT15(100000) HALT ON ERROR (SEE 5.0)

BIT14(040000) LOOP ON TEST DEFINED BY SWITCH REGISTER BITS <4:0>
BIT13(020000) INHIBIT ERROR PRINTOUTS AND ERROR HISTORY PRINTOUTS.
ERROR HISTORY CAN STILL BE OBTAINED BY TYPING ↑C.
BIT12(010000) DISABLE TESTING ABOVE 28K. (MEMORY MANAGEMENT)

BIT11(004000) ENABLE PRINTOUTS OF SINGLE ERRORS ON ARRAY TESTS
(DISABLES ECC FOR ARRAY TESTS)

BIT10(002000) HALT AFTER EACH SUBTEST
!PRESS CONTINUE TO DO NEXT SUBTEST
BIT09(001000) INHIBIT PROGRAM RELOCATION
!IF SET LOCATIONS 430-7776 WILL NOT BE
!TESTED.

BIT08(000400) TYPE FIRST UNCORRECTABLE ERROR IN EACH 4K BANK ONLY.
!THE TOTAL ERROR COUNT (UP TO 377) WILL
!BE SAVED IN THE ERROR HISTORY.

BIT07(000200) ENABLE XOR PRINTOUT FOR ARRAY TESTS
BIT06(000100) INHIBIT MEMORY SIZING.
!THE MEMORY LIMITS MUST BE SETUP IN THE FOLLOWING LOCATIONS:
(VALUES TO TEST 0-28K ARE SHOWN)
(LOWTWO=LOCATION 332)

LOWTWO: 0 ;STORE BITS 17:16 OF LOW TEST ADDRESS
LOWADD: 0 ;STORE REST OF LOW TEST ADDRESS
HIGHTWO: 0 ;STORE BITS 17:16 OF HIGH TEST ADDRESS
HIGHADD: 157776 ;STORE REST OF HIGH TEST ADDRESS
NOTE: HIGHADD MUST BE SET TO A 4K BOUNDARY. (E.G. 37776)
AND LOCATION LDPLG MUST = 1 IF THE PROGRAM
IS RESIDING IN THE MS11K MEMORY.

BIT05(000040) INHIBIT "END PASS #XX" AND "RELOC" PRINTOUTS
 BIT04 SCANS MEMORY WITH ECCDIS. NO TESTS RUN.
 TERMINATE AFTER PASS#1 WITH ↑C.
 BIT03-BIT00 NUMBER OF TEST (0-17) TO RUN FIRST.
 !NORMALLY USED WITH BIT14 (LOOP ON TEST)

[4.2] CONTROL-C FUNCTION

CONTROL C (↑C) AFTER COMPLETION OF THE CURRENT TEST.
 THE ERROR HISTORY (SEE SEC. 6.3) WILL BE
 TYPED. THE PROGRAM WILL LOOP IN LOWER MEMORY.
 HALT AND RESTART AT 250 IF DESIRED.

IN ORDER TO COMPLETELY TEST THE ERROR CORRECTING
 LOGIC, DOUBLE ERRORS ARE FORCED INTO MEMORY ON
 CERTAIN TESTS. TO EXIT FROM THE PROGRAM AND HALT
 EXECUTION, ↑C MUST BE TYPED AND THE TEST
 PERMITTED TO CLEAN UP THE MEMORY UNDER TEST. IF
 NOT, THERE EXISTS THE POSSIBILITY OF UNCORRECT-
 ABLE ERRORS REMAINING IN THE MEMORY.

[4.3] STARTING ADDRESS = 200
RESTART ADDRESS = 250 OR 200

RESTART AT 200 CLEARS PASS COUNT (\$PASS) AND PRINTS "DZMML-A" TITLE.

[4.4] PROGRAM AND/OR OPERATOR ACTION

- 1) LOAD PROGRAM INTO MEMORY USING ABSOLUTE LOADER.
- 2) SET OPTIONS (SEE SEC. 4.1)
- 3) START THE PROGRAM AT 200
- 4) THE FOLLOWING IS AN EXAMPLE WITH EXPLANATIONS
OF THE PRINTOUTS EXPECTED.

"XXXXX-YYYYY" ;ADDRESSES OF TEST BOUNDARIES.

"NO RELOC-SBE IN LOC 0-500" ;RELOCATION IS INHIBITED BECAUSE
 ;SINGLE ERRORS IN THIS AREA WILL CORRUPT
 ;PROGRAM WHEN RELOCATED.

"RELOC" ;THE DIAGNOSTIC RELOCATES TO HIGHEST
 ;BANK UNDER TEST. AND RUNS TST0-TST13 AGAIN.

"END PASS #XX" ;WHERE "XX" IS THE PASS NO.

"TST7-NO ERROR FREE LOC IN SLICE AT XXXXXX" THE LAST PART OF
 TEST 7 CHECKS FOR PROPER 1K ADDRESSES IN

GO1

DZMML MACY11 30(1046) 05-AUG-77 12:59 PAGE 1-5
DZMMLB.P11 05-AUG-77 12:57

SEQ 0007

THE CSR AFTER A DOUBLE ERROR. THIS MESSAGE INDICATES THAT THIS SLICE COULD NOT BE TESTED AND IT IS PROCEEDING TO THE NEXT 1K SLICE.

ADDITIONAL PRINTOUTS

"NO MNG"

;PRINTED IF TESTING ABOVE 28K IS ENABLED
;(BIT 12 OF SWR) AND MEMORY MANAGEMENT IS
;NOT AVAILABLE.

[5.0] PROGRAM HALTS (NORMAL+ ERROR)

THIS IS A LIST OF EXPECTED HALTS. IF THE TEST HALTS IN A LOCATION NOT IN THIS LIST AND ITS LESS THAN 776, IT MAY BE DUE TO A DEVICE INTERRUPTING.

NOTE THE HALT AT END OF SUBTEST AND HALT ON ERROR HALT LOCATIONS MAY BE RELOCATED. THE ACTUAL LOCATIONS THEY ARE IN CAN BE FOUND BY SUBTRACTING 500 FROM 2362(SWHALT) AND ADDING THIS DIFFERENCE TO THE CONTENTS OF SAVR6 (LOC. 352).

PC ---	REASON -----	RECOVERY -----
112	TRAP TO LOC. 4	EXAMINE R6. IT CONTAINS THE POINTER TO THE PC WHERE THE TRAP OCCURRED. (SEE NOTE) THE PROGRAM WILL LOOP AT LOC BUSER UNTIL MANUALLY HALTED. CHECK THE LOCATION POINTED TO BY MINMEM(324) TO INSURE THAT NO DBE'S REMAIN IN THE TESTED AREA.
100	POWER FAIL	POWER UP WILL RECOVER IF IN CORE MEMORY.
2364	HALT AT END OF TEST SWITCH SET.	PRESS CONTINUE TO GO TO NEXT SUBTEST.
13624	HALT ON ERROR	RESTART AT 250
13712	CONTROL-C TYPED OR FATAL ERROR OCCURRED	RESTART AT 250

NOTE: ANY OF THE ABOVE HALT CONDITIONS THAT PERTAIN TO

APT ACTUALLY RESULT IN A TIGHT LOOP SO THAT THE
ERROR INFORMATION CAN BE RECOVERED. CONSULT LISTING
FOR MORE DETAIL, IF NECESSARY.

[6.0] ERRORS

ANY REFERENCE TO 'ERROR' IN THIS DOCUMENT INDICATES AN ERROR
AS DEFINED BY THE PROGRAM AT EXECUTION TIME. NORMALLY, IT IS
AN UNCORRECTABLE ERROR UNLESS ERROR CORRECTING IS DISABLED
THEN IT MEANS ALL ERRORS.

[6.1] ERROR MESSAGE FORMAT

THE ERROR PRINTOUT CONSISTS OF 6 OCTAL WORDS IN THE FOLLOWING
FORMAT:

"ADDR GOOD BAD PC ERR # PASFLG XOR "

"ADR ERR" WILL BE PRINTED PRIOR IF AN ADDRESSING ERROR IS SUSPECTED.

WHERE:

ADDR	=	FAILING MEMORY LOCATION
GOOD	=	GOOD DATA [DATA THAT WAS EXPECTED]
BAD	=	BAD DATA [DATA THAT WAS FOUND]
PC	=	PROGRAM COUNTER AT ERROR CALL.
ERR #	=	FAILING ERROR NO. (SEE SEC 6.2 - ERROR DICTIONARY)
PASFLG	=	CONTENTS OF LOCATION PASFLG. THIS MAY NOT BE RELEVANT. (SEE SEC. 6.2-ERROR DICTIONARY)
XOR	=	EXCLUSIVE 'ORING' OF GOOD VS. BAD (BITS IN ERROR). AVAILABLE ONLY IF SINGLE ERRORS ARE TO BE PRINTED OUT VIA SWREG BIT 07 HIGH.

!THE TEST WILL CONTINUE AFTER THE ERROR PRINTOUT.
!"NO MNG" WILL BE TYPED IF TESTING ABOVE 28K SELECTED AND NO MEMORY
!MANAGEMENT IS FOUND.

(FATAL ERRORS)

"ERROR #XXXXXX" WILL BE TYPED WHERE "XXXXXX" IS
THE ERROR NUMBER. THE DIAGNOSTIC WILL LOOP AT FATHLT UNTIL MANUALLY
HALTED. SEE SEC. 6.2 -ERROR DICTIONARY - FOR DESCRIPTIONS
OF THE ERROR.

(APT MODE ERRORS)

ALL ERRORS ARE TREATED AS FATAL UNDER APT. WHEN AN
ERROR OCCURS UNDER APT A "1" IS STORED IN LOCATION
\$MSGTY AND THE PROGRAM LOOPS AT FATHLT.

\$FATAL CONTAINS THE ERROR NO. IN THE LOW BYTE AND
THE FAILING BANK NO. UNDER TEST IN THE HIGH BYTE.

[6.2] ERROR DICTIONARY

THIS IS A LIST OF ERROR NUMBERS PRINTED AND POSSIBLE
CAUSES FOR THE ERROR. IF THE ERROR IS SUCH THAT THE TESTING
CAN BE CONTINUED AFTER THE ERROR PRINTOUT, IT WILL DO SO.
THE ROUTINE NAME WHERE THE ERROR CALL ORIGINATED IS GIVEN IN
BRACKETS.
NOTE- "BAKPAT" REFERS TO THE BACKGROUND PATTERN WRITTEN INTO MEMORY
FOR VARIOUS TESTS. BAKPAT HAS THE VALUE = 377, AND
SWAPAT HAS THE VALUE = 177400.

.ENDR

```
;ERROR # 0      ;[BUSER] BUS ERROR TRAP TO LOC. 4 OCCURRED
                  ;      THIS ERROR IS NOT PRINTED AND IS FOR "APT" USE.

;ERROR # 1      ;[BEGIN]FATAL LOAD ERROR
                  ;DOUBLE ERROR EXISTS IN AREA
                  ;OF MEMORY THAT PROGRAM RESIDES

;ERROR # 2      ;[TSTTRP]FATAL DATA ERROR
                  ;LOCATIONS 0000-430 FAILED 1'S + 0'S TEST.
                  ;RD = GOOD DATA
                  ;R1 = ADDRESS OF FAILING LOCATION.

;ERROR # 3      ;[APTSIZ] APT FATAL ERROR
                  ;APT MEMORY TABLES NOT SETUP CORRECTLY.
                  ;CHECK LOCATIONS $MAMS1 [430] TO $MADR4[446]
                  ;, FOR CORRECT MEMORY SIZE DATA.

;ERROR # 4      ;[TSTSIZ] OPERATOR FATAL ERROR
                  ;SELECTED MEMORY SIZE GREATER THAN 28K, BUT
                  ;SR BIT12 (10000) SET.
                  ;CLEAR BIT12 AND RESTART AT 200.

;ERROR # 5      ;[TSTSIZ] OPERATOR FATAL ERROR
                  ;LOWEST SELECTED TEST LIMIT IS HIGHER THAN
                  ;HIGHEST TEST LIMIT. SET LOCATIONS "LOWTWO"[332]
                  ;TO "HIGHADD" [340] CORRECTLY AND RESTART
                  ;AT 200.

;ERROR # 6      ;[CLRMEM] TEST SELECTION FATAL ERROR
                  ;TEST SELECTED FOR LOOPING IS HIGHER
                  ;THAN HIGHEST EXISTING TEST NUMBER.

;ERROR # 7      ;[TST0] TEST SEQUENCE ERROR
                  ;TST0 HAS BEEN ENTERED OUT OF SEQUENCE
                  ;TESTN SHOULD = 00
```

J01

DZMML MACY11 30(1046) 05-AUG-77 12:59 PAGE 1-8
DZMMLB.P11 05-AUG-77 12:57

SEQ 0010

```
;THE DIAGNOSTIC HAS BEEN CORRUPTED.

;ERROR # 10      ;[TST0] DUAL ADDRESSING ERROR
                  ;FOR THIS ERROR THE GOOD DATA PRINTED IS AN
                  ;ADDRESS. THIS IS THE ADDRESS SELECTED WHEN
                  ;THE SAME DATA WAS WRITTEN INTO THE FAILING
                  ;LOCATION. CHECK BANK SELECT CIRCUITRY

;ERROR # 11      ;[TST0] ADDRESS AND DATA ERROR
                  ;IDENTICAL TO PREVIOUS ERROR EXCEPT THE DATA
                  ;WRITTEN INTO THE FAILING LOCATION WAS IN
                  ;ERROR ALSO.

;ERROR # 12      ;[TST0] DATA ERROR
                  ;IF BAD DATA = 0000 COULD BE AN ADDRESSING
                  ;ERROR , ELSE COMPARE GOOD AND BAD DATA FOR FAILING BITS.

;ERROR # 13      ;[TST1] TEST SEQUENCE ERROR
                  ;$TEST [404] SHOULD = 01
                  ;THE DIAGNOSTIC HAS BEEN CORRUPTED.

;ERROR # 14      ;[TST1] DATA ERROR LOWER WORD (SHIFTING 1)
                  ;COMPARE GOOD AND BAD PRINTED DATA, FAILING
                  ;DATA BITS MAY SHORTED OR SWAPPED.

;ERROR # 15      ;[TST1] DATA ERROR UPPER WORD
                  ;COMPARE GOOD AND BAD PRINTED DATA, FAILING
                  ;DATA BITS MAY BE SHORTED OR SWAPPED

;ERROR # 16      ;[TST1] DATA ERROR LOWER WORD (SHIFTING 0)

;ERROR # 17      ;[TST1] DATA ERROR UPPER WORD (SHIFTING 0)

;ERROR # 20      ;[TST2] TEST SEQUENCE ERROR
                  ;$TESTN [404] SHOULD = 02
                  ;THE DIAGNOSTIC HAS BEEN CORRUPTED.

;ERROR # 21      ;[TST2] ADDRESS BIT TEST
                  ;COMPARE GOOD VS. BAD DATA
                  ;SUSPECT BYTE ADDRESSING PROBLEM

;ERROR # 22      ;[TST2] ADDRESS BIT TEST
                  ;COMPLEMENTING BYTE DATA
                  ;SUSPECT BYTE ADDRESSING

;ERROR # 23      ;[TST2] ADDRESS BIT TEST
                  ;ADDRESS BIT DID NOT GET ASSERTED
                  ;ADDRESSING PROBLEM

;ERROR # 24      ;[TST2] ADDRESS BIT TEST
                  ;DATA DID NOT GET COMPLEMENTED
                  ;ADDRESSING PROBLEM ?

;ERROR # 25      ;[TST3] TEST SEQUENCE ERROR
                  ;$TESTN [404] SHOULD = 03
                  ;THE DIAGNOSTIC HAS BEEN CORRUPTED.
```


K01

DZMML MACY11 30(1046) 05-AUG-77 12:59 PAGE 1-9
DZMMLB.P11 05-AUG-77 12:57

SEQ 0011

;ERROR # 26 :[TST3] BYTE ADDRESSING TEST
:COMPARE GOOD VS. BAD
:ADDRESSING PROBLEM

;ERROR # 27 :[TST3] BYTE ADDRESSING TEST
:DATA DID NOT GET COMPLEMENTED
:COMPARE GOOD VS. BAD

;ERROR # 30 :[TST4] TEST SEQUENCE ERROR
:STESTN [404] SHOULD = 04.
: THE DIAGNOSTIC HAS BEEN CORRUPTED.

;ERROR # 31 :[TST4] CSR ACCESS ERROR
:TRAP TO 4 OCCURRED
:CSR DID NOT RESPOND

;ERROR # 32 :[TST4] CSR BIT ERROR
:BIT FAILURE IN CSR
:COMPARE GOOD VS. BAD
:IF BAD DATA HAS SBE OR DBE
:SET, SUSPECT OTHER THAN CSR

;ERROR # 33 :[TST5] TEST SEQUENCE ERROR
:STESTN [404] SHOULD = 05
: THE DIAGNOSTIC HAS BEEN CORRUPTED.

;ERROR # 34 :[TST5] ECC INH ERROR
:ECC WAS DISABLED BUT LOW WORD
:OF DATA WAS CORRECTED

;ERROR # 35 :[TST5] ECC INH ERROR
:ECC WAS DISABLED BUT HIGH WORD
:OF DATA WAS CORRECTED

;ERROR # 36 :[TST5] ECC ERROR
:ECC WAS ENABLED, BUT...
:LOW WORD WAS NOT CORRECTED

;ERROR # 37 :[TST5] ECC ERROR
:SINGLE ERROR BIT WAS NOT SET
:AS RESULT OF PREVIOUS TEST

;ERROR # 40 :[TST5] ECC ERROR
:ECC WAS ENABLED, BUT...
:HIGH WORD WAS NOT CORRECTED

;ERROR # 41 :[TST5] ECC ERROR
:SINGLE ERROR BIT WAS NOT SET
:AS RESULT OF PREVIOUS TEST

;ERROR # 42 :[TST6] TESTS SEQUENCE ERROR
:STESTN [404] SHOULD = 06
: DIAGNOSTIC HAS BEEN CORRUPTED

;ERROR # 43 :TST6] WRITE BYTE TO CLEAR SBE

L01

DZMML MACY11 30(1046) 05-AUG-77 12:59 PAGE 1-10
DZMMLB.P11 05-AUG-77 12:57

SEQ 0012

```

;WRITING INTO BYTE WITH SINGLE ERROR
;DID NOT RESULT IN CLEARING THE SBE.

;ERROR # 44      ;[TST6] WRITE BYTE TO CLEAR SBE
                  ;DATA COMPARE ERROR

;ERROR # 45      ;[TST6] WRITE BYTE TO CLEAR SBE
                  ;WRITING INTO BYTE WITH NO SINGLE ERROR
                  ;Cleared error in the byte with the error.

;ERROR # 46      ;[TST7] TEST SEQUENCE ERROR
                  ;STESTN SHOULD = 07
                  ;DIAGNOSTIC HAS BEEN CORRUPTED

;ERROR # 47      ;[TST7] DOUBLE ERROR STATUS
                  ;DOUBLE ERROR BIT (BIT 15)
                  ;NOT SET AFTER DBE

;ERROR # 50      ;[TST7] DOUBLE ERROR STATUS
                  ;TRAP THROUGH LOCATION 114
                  ;DID NOT OCCUR AFTER DBE
                  ;DISREGARD GOOD AND BAD DATA

;ERROR # 51      ;[TST7] DOUBLE ERROR STATUS
                  ;CSR DID NOT CONTAIN CORRECT
                  ;1K BANK POINTING TO ERROR
                  ;ADDR = ADDR WITH DBLE ERROR
                  ;GOOD DATA = 1K BANK CONTAINING DBE
                  ;BAD DATA = BANK POINTED TO BY CSR BITS 11-5
                  ;POSSIBLE MARGINAL SINGLE ERROR CAUSED
                  ;DOUBLE ERROR NOT TO BE SET.

;ERROR # 52      ;[TST10] TESTS SEQUENCE ERROR
                  ;STESTN SHOULD = 10
                  ;DIAGNOSTIC HAS BEEN CORRUPTED

;ERROR # 53      ;[TST10] DATIP WRITE INHIBIT ON DBE
                  ;'INC' INSTRUCTION DID NOT WRITE
                  ;INHIBIT ON LOWER WORD WITH DBE

;ERROR # 54      ;[TST10] DATIP WRITE INHIBIT ON DBE
                  ;'INC' INSTRUCTION DID NOT WRITE
                  ;INHIBIT ON UPPER WORD WITH DBE

;ERROR # 55      ;[TST11] TEST SEQUENCE ERROR
                  ;STESTN SHOULD = 11
                  ;DIAGNOSTIC HAS BEEN CORRUPTED

;ERROR # 56      ;[TST11] INHIBIT WRITE BYTE ON DBE
                  ;WRITE BYTE DID NOT GET INHIBITED
                  ;ON LOWER WORD WITH DBE

;ERROR # 57      ;[TST11] INHIBIT WRITE BYTE ON DBE
                  ;WRITE BYTE DID NOT GET INHIBITED
                  ;ON UPPER WORD WITH DBE

```


MO1

DZMML MACY11 30(1046) 05-AUG-77 12:59 PAGE 1-11
DZMMLB.P11 05-AUG-77 12:57

SEQ 0013

```
;ERROR # 60      ;[TST12] TEST SEQUENCE ERROR
                  ;STESTN SHOULD = 12
                  ;DIAGNOSTIC HAS BEEN CORRUPTED

;ERROR # 61      ;[TST12] WORD WRITE INHIBIT ON DBE
                  ;WORD WRITE DID NOT GET INHIBITED
                  ;ON LOWER WORD WITH DBE

;ERROR # 62      ;[TST12] WORD WRITE INHIBIT ON DBE
                  ;WORD WRITE DID NOT GET INHIBITED
                  ;ON UPPER WORD WITH DBE

;ERROR # 63      ;[TST13] TEST SEQUENCE ERROR
                  ;STESTN SHOULD = 13
                  ;DIAGNOSTIC HAS BEEN CORRUPTED

;ERROR # 64      ;[TST13] DUAL ADDRESSING ERROR
                  ;IF PASFLG = 0 THEN FAILING LOCATION
                  ;AND FAILING DATA ARE DUAL ADDRESS

;ERROR # 65      ;[TST14] TEST SEQUENCE ERROR
                  ;STESTN SHOULD = 14
                  ;DIAGNOSTIC HAS BEEN CORRUPTED

;ERROR # 66      ;[TST14] ADDRESS OR DATA ERROR
                  ;IF "ADD ERR" NOT PRINTED THEN THE
                  ;BYTE SELECT CIRCUITRY PROBABLY FAILED

;ERROR # 67      ;[TST15] TEST SEQUENCE ERROR
                  ;STESTN SHOULD = 15
                  ;DIAGNOSTIC HAS BEEN CORRUPTED

;ERROR # 70      ;[TST15] CHECK BIT WRITE ERROR
                  ;CHECK BITS SHOULD BE 4000
                  ;FROM WRITE OPERATION

;ERROR # 71      ;[TST15] CHECK BIT DATA ERROR
                  ;MARCHING MIN TO MAX
                  ;CHECK BITS SHOULD BE 3740

;ERROR # 72      ;[TST15] CHECKBIT DATA ERROR
                  ;MARCHING MIN TO MAX
                  ;CHECKBITS SHOULD BE 4000

;ERROR # 73      ;[TST15] CHECK BIT DATA ERROR
                  ;MARCHING MAX TO MIN
                  ;CHECKBITS SHOULD BE 3740

;ERROR # 74      ;[TST15] CHECKBIT DATA ERROR
                  ;MARCHING MIN TO MAX
                  ;CHECKBITS SHOULD BE 4000

;ERROR # 75      ;[TST16] TEST SEQUENCE ERROR
                  ;STESTN SHOULD = 16
                  ;DIAGNOSTIC HAS BEEN CORRUPTED
```

DZMML MACY11 30(1046) 05-AUG-77 12:59 PAGE 1-12
 DZMMLB.P11 05-AUG-77 12:57

SEQ 0014

```

;ERROR # 76 :[TST16] DATA ERROR
              ;DATA WRITE OR READ ERROR.

;ERROR # 77 :[TST16] MARCHING 1'S AND 0'S DATA ERROR
              ;IF PASFLG=0 FAILED MARCHING 1'S + 0'S IN
              ;              MAX TO MIN DIRECTION.
              ;IF PASFLG=1 FAILED MARCHING 1'S + 0'S IN
              ;              MIN TO MAX DIRECTION
              ;IF PASFLG=3 FAILED MARCHING 0'S + 1'S IN
              ;              MAX TO MIN DIRECTION.

;ERROR # 100 :[TST16] MARCHING 1'S AND 0'S DATA ERROR
              ;IDENTICAL TO PREVIOUS ERROR EXCEPT THE DATA IS
              ;CHECKED IMMEDIATELY AFTER BEING WRITTEN.

;ERROR # 101 :[TST17] TEST SEQUENCE ERROR
              ;STESTN SHOULD = 17
              ;THE DIAGNOSTIC HAS BEEN CORRUPTED.

;ERROR # 102 :[TST17] VOLATILITY/REFRESH TEST ERROR
              ;IF PASFLG=0 BAKPAT WRITE OR READ ERROR.
              ;IF PASFLG=1 THE FAILING LOCATION CHANGED WHILE
              ;              ANOTHER LOCATIONS WAS WRITTEN FOR
              ;              2 MS. THE OTHER LOCATION IS SAVED
              ;              IN SAVLOC (352)
              ;IF PASFLG=2 SWAPPED BAKPAT (77400 OR 77000)
              ;              WRITE OR READ ERROR.
              ;IF PASFLG=3 SAME AS IF PASFLG=2 EXCEPT
              ;              THE DATA IS SWAPPED BAKPAT.

;ERROR # 103 :[RELOC] FATAL RELOCATION ERROR
              ;A DOUBLE ERROR EXISTS IN THE AREA
              ;IN WHICH THE PROGRAM WANTS TO RELOCATE
              ;RELOCATION ABORTED.
  
```

.REPT 0

[6.3] ERROR HISTORY

LOCATIONS IN MEMORY ARE SET ASIDE TO COLLECT A HISTORY OF THE FAILING BITS IN A PARTICULAR MEMORY BANK. THIS DATA IS COLLECTED FOR EVERY ERROR REGARDLESS OF SWITCH SETTINGS.

NORMALLY THE DATA IS OUTPUT AT THE END OF TESTING, BUT IF CONTROL-C IS TYPED IT IS OUTPUT AT THE END OF THE CURRENT TEST.

THE ERROR HISTORY IS INTENDED TO HIGHLIGHT IF THE ERRORS ARE DUE TO 1 BIT FAILING OR ONLY ADDRESS ERRORS.

ERROR HISTORY FORMAT:

ERROR BANK COUNT

WHERE:

ERROR = BIT THAT FAILED (NUMBER OF THE FAILING BIT IN DECIMAL I.E.
0-15 WILL BE TYPED OUT OR THE WORDS "ADR ERR" WILL
BE TYPED OUT IF ADDRESS ERROR WAS SEEN
IN THE SPECIFIC BANK OF MEMORY
BANK = 4K MEMORY BANK IN WHICH THIS FAILURE WAS SEEN
A 0 FOR 0 TO 4K, A 1 FOR 4 TO 8K AND SO ON
COUNT = NUMBER OF TIMES THIS MEMORY BANK FAILED.
(377 IS MAXIMUM FAILURE COUNT RECORDED.)

[6.4] ERROR RECOVERY

IF THE PROGRAM IS HALTED AFTER REPORTING AN ERROR IT CAN EITHER
BE CONTINUED OR RESTARTED AT 200 OR 250 (SEE SEC 4.2). HOWEVER FOR
CPU'S THAT DESTROY CONTENTS OF REGISTERS AFTER COMING TO A HALT
THE PROGRAM SHOULD ONLY BE RESTARTED.

[6.5] MOS CHIP CROSS REFERENCE

IN THE FOLLOWING MATRIX, THERE ARE TWO "E" NUMBERS UNDER THE
APPROPRIATE DATA BIT FOR EACH 8K WORTH OF ADDRESSING. THESE
ADDRESSES ARE RELATIVE TO THE BASE ADDRESS OF THE SPECIFIC
ARRAY BOARD BEING REFERENCED.

THE UPPER "E" NUMBER SHOULD BE USED IF THE LEAST SIGNIFICANT
OCTAL DIGIT OF THE ADDRESS IS EITHER A 2 OR A 6 (BIT 1 = 1).
THE LOWER "E" NUMBER SHOULD BE USED IF THE LEAST SIG OCTAL DIGIT
IS A 0 OR A 4 (BIT 1 = 0). FOR EXAMPLE, IF THE BAD ADDRESS IS
100404 AND THE XOR OF GOOD VS BAD IS 100, THE SUSPECT MOS CHIP
IS "E 167". THE BINARY BITS IN THE OCTAL XOR PRINTOUT
REFERENCE DATA BITS 15-0 ON THE CHART HEADER.

	DATA BITS 15-0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
140000-	88	77	67	56	46	35	24	12	184	172	160	148	135	122	110	97
177776	84	73	63	52	42	31	20	8	180	168	156	144	131	118	106	93
100000-	87	76	66	55	45	34	23	11	183	171	159	147	134	121	109	96
137776	83	72	62	51	41	30	19	7	179	167	155	143	130	117	105	92

40000-	86	75	65	54	44	33	22	10	182	170	158	146	133	120	108	95
77776	82	71	61	50	40	29	18	6	178	166	154	142	129	116	104	91
00000-	85	74	64	53	43	32	21	9	181	169	157	145	132	119	107	94
37776	81	70	60	49	39	28	17	5	177	165	153	141	128	115	103	90

[7.0] RESTRICTIONS

MEMORY UNDER TEST SHOULD BE CONTIGUOUS. FOR SYSTEMS HAVING NON-CONTIGUOUS MEMORY THE MEMORY BOUNDARIES SHOULD BE DEFINED BY THE OPERATOR. (CONTIGUOUS MEMORY IS DEFINED AS A MEMORY THAT CAN BE BOTH READ AND WRITTEN IN CONSECUTIVE LOCATIONS.)

IT IS POSSIBLE TO CONFIGURE A SYSTEM USING TWO (2) MF11S-K CONTROLLERS EACH WITH ITS OWN CSR. THE FIRST CSR WILL BE AT 172136 AND THE SECOND AT 172134. IN THE EVENT OF TWO CSR'S BOTH SYSTEMS WILL BE TESTED BEFORE A COMPLETE PASS IS ACKNOWLEDGED.

[8.0] MISCELLANEOUS

[8.1] ADDRESS/BANK RANGES IN OCTAL AND DECIMAL

THIS REFERENCE TABLE CROSS REFERENCES THE MEMORY BANK NO.5, THE RANGE AND THE PAR USED WHEN MEMORY MANAGEMENT IS ENABLED. IT IS ALSO USEFUL TO SHOW STARTING ADDRESSES IN A PARTICULAR 4K BANK.

BANK NO.	DECIMAL RANGE	OCTAL RANGE	[PAGE ADDRESS REGISTER] USED/CONTENT	UNIBUS ADDRESS
0	0 - 4K	000000-017776	0 0000	772340
1	4K - 8K	020000-037776	1 0200	772342
2	8K-12K	040000-057776	NOT USED	
3	12K-16K	060000-077776	NOT USED	
4	16K-20K	100000-117776	NOT USED	
5	20K-24K	120000-137776	NOT USED	
6	24K-28K	140000-157776	NOT USED	
7	28K-32K	160000-177776	2 1600	772344
8	32K-36K	200000-217776	3 2000	772346
9	36K-40K	220000-237776	4 2200	772350

10	40K-44K	240000-257776	5	2400	772352
11	44K-48K	260000-277776	6	2600	772354
12	48K-52K	300000-317776	2	3000	
13	52K-56K	320000-337776	3	3200	
14	56K-60K	340000-357776	4	3400	
15	60K-64K	360000-377776	5	3600	
16	64K-68K	400000-417776	6	4000	
17	68K-72K	420000-437776	2	4200	
18	72K-76K	440000-457776	3	4400	
19	76K-80K	460000-477776	4	4600	
20	80K-84K	500000-517776	5	5000	
21	84K-88K	520000-537776	6	5200	
22	88K-92K	540000-557776	2	5400	
23	92K-96K	560000-577776	3	5600	
24	96K-100K	600000-617776	4	6000	
25	100K-104K	620000-637776	5	6200	
26	104K-108K	640000-657776	6	6400	
27	108K-112K	660000-677776	2	6600	
28	112K-116K	700000-717776	3	7000	
29	116K-120K	720000-737776	4	7200	
30	120K-124K	740000-757776	5	7400	
31	124K-128K	760000-777776	7	7600	772356

NOTES:

1. THE PAR (PAGE ADDRESS REGISTER) CONTENTS ARE SHOWN IN A TEST THAT SELF SIZES. IF THE LIMITS OF TESTING ARE SET BY THE OPERATOR AND IF THE BANK IS ABOVE 28K PAR NO. 1 WILL BE SET TO THE BEGINNING PAGE. FOR EXAMPLE IF THE TESTING WAS TO BEGIN WITH BANK 8 PAR NO. 3 WOULD EQUAL 2000, PAR 4 WOULD EQUAL 2200 ETC.

[B.2] EXECUTION TIME

HERE ARE SOME TYPICAL EXECUTION TIMES.

32K (PASS 1 + PASS 2) = 3 MIN 40 SEC.
64K (PASS 1 + PASS 2) = 5 MIN 10 SEC.

[B.3] PASS COUNT AND TEST NO. LOCATIONS

SPASS [406] = PASS COUNT - CLEARED BY START AT 200.

STESTN [404] = CURRENT TEST NO.

DURING EXECUTION, THE CURRENT TEST # WILL BE AVAILABLE IN THE HARDWARE DISPLAY REG (177570) OR THE SOFTWARE DISPLAY REG (174).

[8.4] STACK POINTER

THE STACK STARTS AT 500 WHEN THE PROGRAM IS NOT RELOCATED. SAVR6[346] CONTAINS THE STACK STARTING VALUE WHEN THE DIAGNOSTIC IS RELOCATED.

SAVR6 ALSO CONTAINS THE STARTING ADDRESS OF THE PROGRAM WHEN IT IS RELOCATED.

[8.5] POWER FAIL

THE DIAGNOSTIC CAN BE POWER FAILED WITH NO ERRORS. TO USE, START THE TEST AS USUAL AND POWER DOWN THEN UP AT ANY TIME. THE PROGRAM SHOULD TYPE "P" AND CONTINUE TO RUN FROM TEST 0 IN THE SAME STATE (I.E. STATE OF RELOCATION) AS IT WAS BEFORE THE POWER WAS INTERRUPTED. HOWEVER IF THE DIAGNOSTIC WAS IN A MEMORY THAT CAN NOT HOLD DATA WITH THE POWER DOWN THEN THE PROGRAM WILL NOT RECOVER FROM POWER FAIL.

[9.0] PROGRAM DESCRIPTION

[9.1] NARRATIVE FLOW CHART

THE TEST IS LOADED INTO LOCATIONS 0000 - 14216 BUT EXPANDS DEPENDING ON HOW MUCH MEMORY IS UNDER TEST. SEE STEP 6. BELOW FOR A DETAILED EXPLANATION.

THE FOLLOWING NARRATIVE FLOW CHART DESCRIBES MAJOR PROGRAM OPERATION. FOR THE PERSON WHO NEEDS DETAIL THE TAG ASSOCIATED WITH THE OPERATION IS GIVEN IN BRACKETS.

FOR THIS DISCUSSION SWITCH SETTINGS ARE IGNORED AND EVERYTHING IS ASSUMED ENABLED.

1. [START] PRINT "DZMML-A" TITLE
2. [TSTRP] SAVE DATA FROM LOCATIONS 0-376
3. [TSTRP] TEST LOCATIONS 0-376 BY WRITING AND READING 1'S AND 0'S. NOTE THIS IS THE ONLY EXPLICIT TESTING OF THESE LOCATIONS.
4. [SLFSIZ] SIZE MEMORY BY SETTING THE DIAGNOSTIC CHECK BIT IN THE MS11K CONTROL AND STATUS REGISTER (172136) AND LOOKING FOR MEMORY THAT RESPONDS BY LOADING CHECK BITS INTO THE CSR (BITS 11-5).
5. [TYPsiz] TYPE MEMORY TEST LIMITS AND THE ADDRESS OF THE CSR CONTROLLING THE MEMORY UNDER TEST.
6. [SETSTK] SPACE IS SAVED AT THE END OF THE TEST FOR AN ERROR HISTORY. FOR EACH 4K BANK 18 BYTES ARE SAVED IN THE FOLLOWING FORMAT:


```

:ADR ERR:PAR ERR:
:BIT15 :ERR CNT:
:BIT13 :BIT14 :
:BIT11 :BIT12 :
:BIT09 :BIT10 :
:BIT07 :BIT07 :
:BIT05 :BIT06 :
:BIT03 :BIT04 :
:BIT01 :BIT02 :
:UNUSED :BIT00 :

```

THE ABSOLUTE LOADER (300 ADDRESSES) IS APPENDED AND IF IN XXDP CHAIN MODE, 5674 ADDRESSES ARE APPENDED TO THE TEST. THIS SAVES THE XXDP MONITOR AND ALLOWS THE LOCATIONS OCCUPIED BY XXDP TO BE TESTED.

7. [CLRMEM] INITIALIZE BAKPAT AND SWAPAT
8. [CONT] CLEAR CHUNK OF MEMORY UNDER TEST AND DISPATCH CONTROL TO THE APPROPRIATE TEST.
9. [TST0] EXECUTE TEST 0. SEE SECTION 10 FOR TEST DESCRIPTIONS.
10. [TSTSCP] COMES HERE AFTER EACH TEST AND IF CNTRL-C TYPED THEN GO TO ERROR HISTORY PRINTOUT.
IF SR=2000 THEN HALT
IF SR=40000 THEN LOOP ON TEST DEFINED BY <3:0>
ELSE CONTINUE TO NEXT TEST.
11. [TST1-TST17] EXECUTE TST1-TST17 EACH TIME GOING TO STEP 9. *** SEE NOTE BELOW
12. [RELOC] THE PROGRAM RELOCATES TO 2ND 16K (IF NECESSARY AND IF NO SINGLE ERRORS EXIST IN LOCATIONS 0-500) TO TEST THE LOCATIONS IT OCCUPIES. (430-ENDPRG).
WHERE "ENDPRG" IS THE CONTENTS OF ENDSTK(314).
I.E THE LAST PROGRAM ADDRESS. NOTE "RELOC" IS PRINTED JUST PRIOR TO THE ACTUAL RELOCATION.
13. TESTS 0-17 ARE RUN AS DESCRIBED ABOVE EXCEPT ONLY BANK 0 LOCATIONS 430-ENDPRG ARE TESTED.
*** SEE NOTE BELOW
14. [RELOER] RELOCATE THE PROGRAM BACK TO LOWER MEMORY.
15. [LOWER] IF CONTROL-C TYPED GO PRINT ERROR HISTORY.
16. [TSTMM] IF MEMORY MANAGEMENT SELECTED AND AVAILABLE, RUN TESTS 0-17 ON THE FIRST 20K SLICE ABOVE 28K.
17. [CONTMM] CALL "UPMM" TO UPDATE MEMORY MANAGEMENT PAR REGISTERS TO POINT TO THE NEXT 20K SLICE OF

UPPER MEMORY.

18. [MAXADR] REPEAT STEPS 18 + 19 UNTIL ALL MEMORY ABOVE 28K IS TESTED.
19. [ENDPAS] PRINT ERROR HISTORY OF FAILING BITS
20. [SEOP]
PRINT "END PASS #XX"

*** NOTE: THERE EXISTS TWO DIFFERENT EXECUTION MODES.
THEY ARE PASS #1 AND PASS #2 AND ALL SUCCESSIVE
PASSES. THEY ARE DESCRIBED BELOW.

PASS 1 - QUICK VERIFY - ONE ITERATION ONLY OF ECC
TESTS (TEST5-12) AND NO EXECUTION OF TEST20
AND TEST 21.

PASS 2 AND ABOVE - COMPLETE EXECUTION OF ALL TESTS (TEST0-21)
THE FIRST COMPLETE PASS IS ACTUALLY PASS 2.

[9.2] TEST TITLES AND TEST TIMES PER 16K

SEE THE TEST HEADINGS IN THE LISTING FOR DETAILS ON EACH TEST.

TEST 0: TEST FOR PROPER BANK SELECTION	<1 SEC
TEST 1: CHECK DATI/DATO LINES	<1 SEC
TEST 2: TEST ADDRESS LINES FOR UNIQUENESS	<1 SEC
TEST 3: TEST BYTE ADDRESSING	<1 SEC
TEST 4: MS11K STATUS REGISTER TESTS	<1 SEC
TEST 5: SINGLE ERROR TESTS	15 SEC
TEST 6: TEST THAT WRITE BYTE CLEARS SBE	7 SEC
TEST 7: DOUBLE BIT ERROR- STATUS & TRAP	15 SEC
TEST 10: DBE- WRITE INH ON DATIP CYCLE	1 SEC
TEST 11: DBE- WRITE INH ON WRITE BYTE	15 SEC
TEST 12: DBE- WRITE INH ON WORD WRITE	15 SEC
TEST 13: DUAL ADDRESS TEST	2 SEC
TEST 14: TEST MEMORY FOR HOLDING 1'S AND 0'S	2 SEC
TEST 15: MARCHING 1'S AND 0'S IN CHECKBIT CHIPS	3 SEC
TEST 16: MARCHING 1'S AND 0'S IN DATA CHIPS	2 SEC
TEST 17: CELLS VOLATILITY TEST (REFRESH)	4 SEC

[10.0] RXDP & ACT11 & APT OPERATION

RXDP CHAIN MODE

OPERATION IS IDENTICAL TO STAND ALONE EXCEPT:

1. NO "DZMML-A" TITLE IS PRINTED.
2. THE PROGRAM ALWAYS HALTS ON ERROR.

H02

DZMML MACY11 30(1046) 05-AUG-77 12:59 PAGE 1-19
DZMMLB.P11 05-AUG-77 12:57

SEQ 0021

3. AT THE END OF TEST (\$ENDAD) CONTROL IS RETURNED TO THE RXDP CHAIN MONITOR VIA LOCATION 42.

ACT11

OPERATION IS IDENTICAL TO STAND ALONE EXCEPT:

1. NO PRINTOUTS EXCEPT ERROR PRINTOUTS.
2. THE PROGRAM ALWAYS HALTS ON ERROR.
3. AT THE END OF TEST (\$ENDAD) CONTROL IS RETURNED TO THE ACT11 MONITOR VIA LOCATION 42.

APT

OPERATION IS SIMILAR TO STAND ALONE EXCEPT:

1. THE SOFTWARE SWITCH REGISTER BECOMES LOCATION 422 (\$SWREG).
2. AUTO SIZING CAN BE INHIBITED BY SETTING BIT 7 OF BYTE LOCATION 421 (\$ENVM).
3. ALL PRINTOUTS CAN BE INHIBITED BY SETTING BIT 5 OF BYTE LOCATION 421 (\$ENVM).
4. ALL ERRORS CAUSE LOCATION 400 (\$MSGTY) TO BE SET = 0001 AND THE PROGRAM HALTS AT LOCATION 6214 (\$FATHLT). LOCATION 402 (\$FATAL) CONTAINS THE ERROR NO. IN THE LOW BYTE AND THE FAILING MEMORY BANK NO. IN THE HIGH BYTE.

APT MANAGER INFORMATION

THE FOLLOWING IS AN EXAMPLE SCRIPT TO TEST A 4K MEMORY. IT IS RECOMMENDED THAT DIFFERENT SCRIPTS BE USED FOR DIFFERENT MEMORY SIZES TO SAVE AUTO SIZING TIME.

THE EXAMPLE ASSUMES YOU ARE LOGGED INTO THE APT MONITOR.

READY

RUN APPLU
APT 11 PAPER TAPE PROGRAM LOAD UTILITY

THE FOLLOWING COMMANDS ARE VALID

ED	EDIT A PROGRAM
LI	LIST A PROGRAM

COMMAND: ED
PROGRAM NAME TO EDIT: EXAMPL
DO YOU WANT TO LOAD A NEW REV OF THE PROGRAM(Y/N)? N
FIRST PASS RUN TIME IN SECONDS <110>:
LONGEST TEST TIME IN SECONDS <10>:
ADDITIONAL RUN TIME IN SECONDS <0>:
WHICH ETABLE DO YOU WISH TO EDIT? A
SOFTWARE ENVIRONMENT<000>: 1

DZMML MACY11 30(1046) 05-AUG-77 12:59 PAGE 1-20
 DZMMLB.P11 05-AUG-77 12:57

SEQ 0022

ENVIRONMENTAL MODE<000>: 240
 SWITCH 1 <000000>:
 SWITCH 2 <000000>:
 CPU OPTIONS<0000>:
 MEMORY TYPE 1 <000>:
 MAXIMUM ADDRESS<00000000>:
 MEMORY TYPE 2 <000>:
 MAXIMUM ADDRESS<00000000>:
 MEMORY TYPE 3 <000>:
 MAXIMUM ADDRESS<00000000>:
 MEMORY TYPE 4 <000>: 1
 MAXIMUM ADDRESS<00000000>: 17776
 WHICH ETABLE DO YOU WISH TO EDIT?
 COMMAND: OFF

.ENDR
 .ABS
 .NLIST MD,MC,CND

1164
 1169
 (1)
 (1)
 (1)
 (1)
 (1)
 (1)
 (1)
 (1)
 (1)
 (1)
 (1)
 1170
 1171
 1172
 1173
 1205
 1206
 1214
 1215
 1222
 1223
 1224
 1225
 1226
 1227
 1228
 (1)
 (2)
 (1)
 (1)
 (1)
 (1)
 (1)
 (1)
 (1)
 1229

160000

000240
 000042 000042
 000042 000000
 000046 000156
 000052 000052
 000052 040000
 000044

.LIST ME,BIN,SEQ,LOC
 .TITLE DZMML
 ;*COPYRIGHT (C) FEBRUARY 1977
 ;*DIGITAL EQUIPMENT CORP.
 ;*MAYNARD, MASS. 01754
 ;*
 ;*PROGRAM BY JIM RYAN
 ;*
 ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
 ;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
 ;*
 \$SWR=160000 ;;HALT ON ERROR, LOOP ON TEST, INHIBIT ERROR TYP0UT

;;TRAP CATCHER OF .+2 AND HALT FOR 0-776 LOCATIONS

SCOPE =NOP

.=42
 .WORD 0 ;FOR ACT/XXDP

.SBTTL ACT11 HOOKS

 ;HOOKS REQUIRED BY ACT11

\$SVPC=. ;SAVE PC

.=46

SENDAD

.=52

.WORD 40000
 .=SVPC

;;1)SET LOC.46 TO ADDRESS OF SENDAD IN .SEOP
 ;;2)SET LOC.52 TO 40000
 ;; RESTORE PC

J02

DZMML MACY11 30(1046) 05-AUG-77 12:59 PAGE 1-21
DZMMLB.P11 05-AUG-77 12:57 ACT11 HOOKS

SEQ 0023

1230	000044	016777	002640	002656	PWRDN:	MOV	ECCDIS, JCSRADR	; DISABLE ECC
1231	000052	005077	000242			CLR	JMINMEM	; CLEAR FIRST ADDRESS
1232	000056	005077	000240			CLR	JMINMEM+2	; SECOND WORD
1233	000062	042701	000003			BIC	#3, R1	
1234	000066	005011				CLR	(R1)	; CLR A LOC
1235	000070	005061	000002			CLR	2(R1)	; NEXT
1236	000074	012737	000136	000024		MOV	#PWRUP, J#24	
1237	000102	000000				HALT		
1238								

K02

DZMML MACY11 30(1046) 05-AUG-77 12:59 PAGE 1-22
 DZMMLB.P11 05-AUG-77 12:57 ACT11 HOOKS

SEQ 0024

```

1240
1241
1242
1243      000120      ;114 AND 116 ARE RESERVED FOR PARITY TRAP VECTORS.USED IN TST7.
1244      .=120
1245
1246
1247
1248
1249
1250      ;* WRITE MEMORY BACKGROUND
1251      ;* -----
1252      ;*
1253      ;* THIS ROUTINE IS USED TO WRITE THE MEMORY BACKGROUND TO
1254      ;* THE VALUE STORED AT LOCATION BAKPAT. THE ROUTINE ASSUMES
1255      ;* THAT R4 IS POINTING TO THE LOWEST LOCATION AND R5 TO THE
1256      ;* HIGHEST LOCATION TO BE WRITTEN. THE PROGRAM LEAVES THE
1257      ;* SUBROUTINE WITH R0 CONTAINING THE CONTENTS OF BAKPAT.
1258      ;*
1259
1260      000120      010401      000312      WRTMEM: MOV      R4,R1      ;SET R1 TO LOWEST LOCATION UNDER TEST
1261      000122      013700      MOV      @#BAKPAT,R0      ;LOAD R0 WITH THE CONTENTS OF LOCATION BAKPAT
1262      000126      010021      2$:      MOV      R0,(R1)+      ;STARTING FROM THE LOWEST LOCATION WRITE THE
1263      000130      020105      CMP      R1,R5      ;MEMORY TO BACK GROUND PATTERN
1264      000132      103775      BLO      2$
1265      000134      000207      RTS      PC      ;RETURN FROM THE SUBROUTINE
1266
1267      000136      013706      000346      PWRUP: MOV      @#SAVR6,SP      ;RESTORE STACK POINTER
1268      000142      012700      MOV      @#PNTMES-BEGIN,R0      ;GET THE INDIRECT ADDRESS OF LOCATION
1269      000146      060600      ADD      SP,R0      ;TPCRLF RELATIVE TO LOCATION OF THE
1270
1271
1272      000150      004710      JSR      PC,(R0)      ;DIAGNOSTIC IN MEMORY AND GO TO THE
1273      000152      000120      .ASCIZ  /P/      ;;TYPE ROUTINE AND TYPE CR, LF & A "P".
1274
1275      000154      000411      .EVEN
1276
1277
1278
1279      000156      004710      BR      START
1280
1281
1282
1283
1284
1285
1286
1287
1288      000174      000000      ;* SERVICE XXDP/ACT11
1289      000176      000000      $ENDAD: JSR      PC,(R0)      ;RETURN TO ACT11/XXDP MONITOR
1290
1291
1292
1293
1294
1295
1296      000160      000240      NOP
1297      000162      000240      NOP
1298      000164      000240      NOP
1299      000166      000430      BR      RESTRT      ;IF QUICK VERIFY=CLR #-1 ELSE INC #0
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
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
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
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
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2220
2221
2222
2223
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
2278
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
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2390
2391
2392
2393
2394
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429
2430
2431
2432
2433
2434
2435
2436
2437
2438
2439
2440
2441
2442
2443
2444
2445
2446
2447
2448
2449
2450
2451
2452
2453
2454
2455
2456
2457
2458
2459
2460
2461
2462
2463
2464
2465
2466
2467
2468
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2479
2480
2481
2482
2483
2484
2485
2486
2487
2488
2489
2490
2491
2492
2493
2494
2495
2496
2497
2498
2499
2500
2501
2502
2503
2504
2505
2506
2507
2508
2509
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2549
2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
2560
2561
2562
2563
2564
2565
2566
2567
2568
2569
2570
2571
2572
2573
2574
2575
2576
2577
2578
2579
2580
2581
2582
2583
2584
2585
2586
2587
2588
2589
2590
2591
2592
2593
2594
2595
2596
2597
2598
2599
2600
2601
2602
2603
2604
2605
2606
2607
2608
2609
2610
2611
2612
2613
2614
2615
2616
2617
2618
2619
2620
2621
2622
2623
2624
2625
2626
2627
2628
2629
2630
2631
2632
2633
2634
2635
2636
2637
2638
2639
2640
2641
2642
2643
2644
2645
2646
2647
2648
2649
2650
2651
2652
2653
2654
2655
2656
2657
2658
2659
2660
2661
2662
2663
2664
2665
2666
2667
2668
2669
2670
2671
2672
2673
2674
2675
2676
2677
2678
2679
2680
2681
2682
2683
2684
2685
2686
2687
2688
2689
2690
2691
2692
2693
2694
2695
2696
2697
2698
2699
2700
2701
2702
2703
2704
2705
2706
2707
2708
2709
2710
2711
2712
2713
2714
2715
2716
2717
2718
2719
2720
2721
2722
2723
2724
2725
2726
2727
2728
2729
2730
2731
2732
2733
2734
2735
2736
2737
2738
2739
2740
2741
2742
2743
2744
2745
2746
2747
2748
2749
2750
2751
2752
2753
2754
2755
2756
2757
2758
2759
2760
2761
2762
2763
2764
2765
2766
2767
2768
2769
2770
2771
2772
2773
2774
2775
2776
2777
2778
2779
2780
2781
2782
2783
2784
2785
2786
2787
2788
2789
2790
2791
2792
2793
2794
2795
2796
2797
2798
2799
2800
2801
2802
2803
2804
2805
2806
2807
2808
2809
2810
2811
2812
2813
2814
2815
2816
2817
2818
2819
2820
2821
2822
2823
2824
2825
2826
2827
2828
2829
2830
2831
2832
2833
2834
2835
2836
2837
2838
2839
2840
2841
2842
2843
2844
2845
2846
2847
2848
2849
2850
2851
2852
2853
2854
2855
2856
2857
2858
2859
2860
2861
2862
2863
2864
2865
2866
2867
2868
2869
2870
2871
2872
2873
2874
2875
2876
2877
2878
2879
2880
2881
2882
2883
2884
2885
2886
2887
2888
2889
2890
2891
2892
2893
2894
2895
2896
2897
2898
2899
2900
2901
2902
2903
2904
2905
2906
2907
2908
2909
2910
2911
2912
2913
2914
2915
2916
2917
2918
2919
2920
2921
2922
2923
2924
2925
2926
2927
2928
2929
2930
2931
2932
2933
2934
2935
2936
2937
2938
2939
2940
2941
2942
2943
2944
2945
2946
2947
2948
2949
2950
2951
2952
2953
2954
2955
2956
2957
2958
2959
2960
2961
2962
2963
2964
2965
2966
2967
2968
2969
2970
2971
2972
2973
2974
2975
2976
2977
2978
2979
2980
2981
2982
2983
2984
2985
2986
2987
2988
2989
2990
2991
2992
2993
2994
2995
2996
2997
2998
2999
3000
3001
3002
3003
3004
3005
3006
3007
3008
3009
3010
3011
3012
3013
3014
3015
3016
3017
3018
3019
3020
3021
3022
3023
3024
3025
3026
3027
3028
3029
3030
3031
3032
3033
3034
3035
3036
3037
3038
3039
3040
3041
3042
3043
3044
3045
3046
3047
3048
3049
3050
3051
3052
3053
3054
3055
3056
3057
3058
3059
3060
3061
3062
3063
3064
3065
3066
3067
3068
3069
3070
3071
3072
3073
3074
3075
3076
3077
3078
3079
3080
3081
3082
3083
3084
3085
3086
3087
3088
3089
3090
3091
3092
3093
3094
3095
3096
3097
3098
3099
3100
3101
3102
3103
3104
3105
3106
3107
3108
3109
3110
3111
3112
3113
3114
3115
3116
3117
3118
3119
3120
3121
3122
3123
3124
3125
3126
3127
3128
3129
3130
3131
3132
3133
3134
3135
3136
3137
3138
3139
3140
3141
3142
3143
3144
3145
3146
3147
3148
3149
3150
3151
3152
3153
3154
3155
3156
3157
3158
3159
3160
3161
3162
3163
3164
3165
3166
3167
3168
3169
3170
3171
3172
3173
3174
3175
3176
3177
3178
3179
3180
3181
3182
3183
3184
3185
3186
3187
3188
3189
3190
3191
3192
3193
3194
3195
3196
3197
3198
3199
3200
3201
3202
3203
3204
3205
3206
3207
3208
3209
3210
3211
3212
3213
3214
3215
3216
3217
3218
3219
3220
3221
3222
3223
3224
3225
3226
3227
3228
3229
3230
3231
3232
3233
3234
3235
3236
3237
3238
3239
3240
3241
3242
3243
3244
3245
3246
3247
3248
3249
3250
3251
3252
3253
3254
3255
3256
3257
3258
3259
3260
3261
3262
3263
3264
3265
3266
3267
3268
3269
3270
3271
3272
3273
3274
3275
3276
3277
3278
3279
3280
3281
3282
3283
3284
3285
3286
3287
3288
3289
3290
3291
3292
3293
3294
3295
3296
3297
3298
3299
3300
3301
3302
3303
3304
3305
3306
3307
3308
3309
3310
3311
3312
3313
3314
3315
3316
3317
3318
3319
3320
3321
3322
3323
3324
3325
3326
3327
3328
3329
3330
3331
3332
3333
3334
3335
3336
3337
3338
3339
3340
3341
3342
3343
3344
3345
3346
3347
3348
3349
3350
3351
3352
3353
3354
3355
3356
3357
3358
3359
3360
3361
3362
3363
3364
3365
3366
3367
3368
3369
3370
3371
3372
3373
3374
3375
3376
3377
3378
3379
3380
3381
3382
3383
3384
3385
3386
3387
3388
3389
3390
3391
3392
3393
3394
3395
3396
3397
3398
3399
3400
3401
3402
3403
3404
3405
3406
3407
3408
3409
3410
3411
3412
3413
3414
3415
3416
3417
3418
3419
3420
3421
3422
3423
3424
3425
3426
3427
3428
3429
3430
3431
3432
3433
3434
3435
3436
3437
3438
3439
3440
3441
3442
3443
3444
3445
3446
3447
3448
3449
3450
3451
3452
3453
3454
3455
3456
3457
3458
3459
3460
3461
3462
3463
3464
3465
3466
3467
3468
3469
3470
3471
3472
3473
3474
3475
3476
3477
3478
3479
3480
3481
3482
3483
3484
3485
3486
3487
3488
3489
3490
3491
3492
3493
3494
3495
3496
3497
3498
3499
3500
3501
3502
3503
3504
3505
3506
3507
3508
3509
3510
3511
3512
3513
3514
3515
3516
3517
3518
3519
3520
3521
3522
3523
3524
3525
3526
3527

```


L02

DZMML MACY11 30(1046) 05-AUG-77 12:59 PAGE 1-23
 DZMMLB.P11 05-AUG-77 12:57 START AND RESTART ROUTINES

SEQ 0025

```

1299 000214 005043      15: CLR      -(R3)      ;CLEAR A MAILBOX LOCATION
1300 000216 022703 000400    CMP      $MAIL,R3    ;DONE?
1301 000222 001374      BNE      15          ;BRANCH IF NO
1302 000224 105737 000042    TSTB     $42         ;ACT11 MODE?
1303 000230 001007      BNE      RESTR       ;BRANCH IF YES
1304 000232 105737 000170    TSTB     $REL        ;ARE WE RELOCATED?
1305 000236 100404      BMI      RESTR       ;BR IF YES- SINCE TPCRLF IS RELOCATED ALSO-
1306 000240 004767 015472    JSR      PC,PRITL    ;PRINT "DZMML-A" TITLE
1307 000244 000240      NOP                     ;SO WE FALL THRU TO RESTART
1308 000246 000240      NOP                     ;DITTO
1309
1310 000250 012703 000344    RESTR: MOV     $SAVR5,R3    ;POINT R3 TO THE LOC SAVR5
1311 000254 012305      MOV     (R3)+,R5    ;RESTORE R5
1312 000256 012306      MOV     (R3)+,SP    ;AND RESTORE R6 JUST IN CASE IT IS A RESTART
1313 000260 010600      MOV     SP,R0       ;PLACE THE STARTING ADDRESS OF THE TEST IN R0
1314 000262 012746 000340    MOV     $340,-(SP) ;SET HIGH PRIORITY FOR RTI
1315 000266 010046      MOV     R0,-(SP)
1316 000270 000002      RTI
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326

```

;GO TO "START"-MAY BE RELOCATED.
 ;IF RELOCATED SEE LOCATION SAVR6 FOR START.

.SBTTL APT PARAMETER BLOCK

```

(1)
(2)
(1)
(2)
(1) 000272
(1) 000024 000024
(1) 000024 000200
(1) 000044 000044
(1) 000044 000272
(1) 000272
(2)
(1)
(1)
(1)
(1) 000272
(1) 000272 000000
(1) 000274 000400
(1) 000276 000031
(1) 000300 000156
(1) 000302 000000
(1) 000304 000024
1327
1328
1329 000272 000272
1330 000272
1331
1332
1333 000273

```

;*****
 ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
 ;*****
 .SX= . ;SAVE CURRENT LOCATION
 .=24 . ;SET POWER FAIL TO POINT TO START OF PROGRAM
 200 . ;FOR APT START UP
 .=44 . ;POINT TO APT INDIRECT ADDRESS PNTR.
 \$APTHDR . ;POINT TO APT HEADER BLOCK
 .=.SX . ;RESET LOCATION COUNTER
 ;*****
 ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
 ;INTERFACE SPEC.
 \$APTHD: .WORD 0 ;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
 \$HIBTS: .WORD \$MAIL ;ADDRESS OF APT MAILBOX (BITS 0-15)
 \$MBADR: .WORD 25. ;RUN TIM OF LONGEST TEST
 \$TSTM: .WORD 110. ;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
 \$PASTM: .WORD ;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
 \$UNITM: .WORD SETEND-\$MAIL/2 ;LENGTH MAILBOX-ETABLE(WORDS)
 ;
 MAVA: .=\$APTHD ;THIS BYTE IS USED TO DETERMINE IF MEMORY
 ;MANAGEMENT IS AVAILABLE OR NOT
 ;
 .=\$MAVA+1

M02

DZMML MACY11 30(1046) 05-AUG-77 12:59 PAGE 1-24
 DZMMLB.P11 05-AUG-77 12:57 APT PARAMETER BLOCK

SEQ 0026

1334	000273	TYPENB:		; THIS BYTE IS USED TO DETERMINE IF THE
1335				; TYPE OUT OF ERROR HAS BEEN ENABLED OR NOT
1336				
1337	000274	SPRERR: . = TYPENB+1		; THIS BYTE DETERMINES IF THE PROGRAM HAS FOUND
1338				; A PARITY ERROR
1339				
1340				
1341	000275	SADERR: . = SPRERR+1		; THIS BYTE IS USED TO DETERMINE IF THE
1342				; PROGRAM HAS ENCOUNTERED ADDRESS ERROR
1343				
1344				
1345	000276	STRTDI: . = SADERR+1		
1346				
1347	000300	LOWBNK: . = STRTDI+2		
1348				
1349	000302	PASFLG: . = LOWBNK+2		; LOWER BYTE OF THIS WORD GIVES THE PASS NUMBER FOR
1350				; THE SPECIFIC TEST WHEREAS THE UPPER BYTE
1351				; HAS BEEN USED BY DIFFERENT TEST FOR DIFFERENT PURPOSES
1352				
1353				
1354	000304	ENDSTK: . = PASFLG+2		
1355				
1356	000306	PBNK: . = ENDSTK+2		; HOLDS BANK UNDER TEST FOR "TST BNK XX" PRINTOUT.
1357		DECWRD:		
1358	000310	TYPCNT: . = DECWRD+2		
1359				; THIS BYTE DETERMINES THE NUMBER OF WORDS
1360	000310	. BYTE 0		; TO BE TYPED
1361				; THIS LOCATION IS USED TO SAVE THE CHARACTER
1362	000311	SAVKBB: . BYTE 0		; HIT BY THE OPERATOR
1363				; ALSO IS USED AS TEMP IN ROUTINE \$GTSIZ.
1364				
1365		. EVEN		
1366				
1367				
1368	177560	TKS= 177560		
1369	177562	\$KBB= 177562		
1370	177564	\$TPS= 177564		
1371	177566	\$TPB= 177566		
1372	177572	SRO= 177572		
1373	000312	BAKPAT: . WORD 377		; BACKGROUND PATTERN WRITTEN TO MEMORY.
1374				
1375	000314	SWAPAT: . WORD		
1376	000316	RELBOT: BEGIN-50		; HOLDS LOWEST TEST ADDRESS WHEN RELOCATED.
1377	000320	MINMEM: . WORD 100000		; FIRST ADDR FOR ECC IN CASE OF APT
1378				
1379				
1380		;; *****		
1381	000322	LOCATIONS TO BE MODIFIED IF LIMITS SET BY OPERATOR		
1382	000324	LOWTWO: 0		; HOLDS BITS 17:16 OF LOW TEST ADDRESS
1383		LOWADD: 0		; HOLDS BITS 15:0 OF LOW TEST ADDRESS
1384	000326	HIGHTWO: 0		; HOLDS BITS 17:16 OF HIGH TEST ADDRESS
1385	000330	HIGHADD: 37776		; HOLDS BITS 15:0 OF HIGH TEST ADDRESS
1386		;; *****		
1387				
1388	000332	\$HIMAX: 0		; HOLDS BITS 17:16 OF MAXIMUM AVAILABLE MEMORY
1389	000334	\$MAXM: 17776		; HOLDS BITS 15:0 OF MAXIMUM AVAILABLE MEMORY

N02

DZMML MACY11 30(1046) 05-AUG-77 12:59 PAGE 1-25
 DZMMLB.P11 05-AUG-77 12:57 APT PARAMETER BLOCK

SEQ 0027

```

1390
1391 000336 000000          MAXMEM: .WORD          ;MAXIMUM CURRENT VIRTUAL MEMORY UNDER TEST
1392
1393 000340 000000          SAVMAX: .WORD
1394 000342 000000          SAVR4:  .WORD
1395 000344 000000          SAVR5:  .WORD
1396
1397          ;* SAVR6 POINTS TO WHERE THE PROGRAM STARTS EVEN WHEN RELOCATED.
1398 000346 000500          SAVR6:  .WORD BEGIN      ;CONTAINS START ADDRESS WHEN RELOCATED ALSO.
1399 000350 000000          SAVLOC: 0              ;TEST 17 STORES ERROR INFO HERE
1400
1401          ;GET HERE IF AN ILLEGAL TRAP TO LOC. 4 OCCURRED
1402
1403 000352 012737 000001 000400 BUSER: MOV    #1,2#MSGTY ;TELL APT FATAL ERROR #000
1404 000360 000774          BR      BUSER             ;LOOP SO APT CAN GET INFO
1411
1412
1413          ;*364-400 IS USED AS A STACK AREA BY ERRCHK ROUTINE FOR ERROR HISTORY PRINTOUT
  
```

DZMPL MACY11 30(1046) 05-AUG-77 12:59 PAGE 1-26
 DZMPLB.P11 05-AUG-77 12:57 APT PARAMETER BLOCK

SEQ 0028

```

1415      000400      .=400
1416      .SBTTL  APT MAILBOX-ETABLE
(1)      ;*****
(1)      .EVEN
(1)      $MAIL:      APT MAILBOX
(1)      000400      000000      $MSGTY: .WORD  AMSGTY  ;MESSAGE TYPE CODE
(1)      000402      000000      $FATAL: .WORD  AFATAL  ;FATAL ERROR NUMBER
(1)      000404      000000      $TESTN: .WORD  ATESTN  ;TEST NUMBER
(1)      000406      000000      $PASS:  .WORD  APASS   ;PASS COUNT
(1)      000410      000000      $DEVCT: .WORD  ADEVCT  ;DEVICE COUNT
(1)      000412      000000      $UNIT:  .WORD  AUNIT   ;I/O UNIT NUMBER
(1)      000414      000000      $MSGAD: .WORD  AMSCAD  ;MESSAGE ADDRESS
(1)      000416      000000      $MSGLG: .WORD  AMSGLG  ;MESSAGE LENGTH
(1)      000420      000000      $ETABLE: ;APT ENVIRONMENT TABLE
(1)      000420      000      $ENV:   .BYTE  AENV    ;ENVIRONMENT BYTE
(1)      000421      000      $ENVH: .BYTE  AENVH   ;ENVIRONMENT MODE BITS
(1)      000422      000000      $SWREG: .WORD  ASWREG  ;APT SWITCH REGISTER
(1)      000424      000000      $USWR:  .WORD  AUSWR   ;USER SWITCHES
(1)      000426      000000      $CPUOP: .WORD  ACPUOP  ;CPU TYPE, OPTIONS
(1)      ;BITS 15-11=CPU TYPE
(1)      ;11/04=01, 11/05=02, 11/20=03, 11/40=04, 11/45=05
(1)      ;11/70=06, PDQ=07, Q=10
(1)      ;BIT 10=REAL TIME CLOCK
(1)      ;BIT 9=FLOATING POINT PROCESSOR
(1)      ;BIT 8=MEMORY MANAGEMENT
(1)      000430      000      $MAMS1: .BYTE  AMAMS1  ;HIGH ADDRESS, M.S. BYTE
(1)      000431      000      $MTYP1: .BYTE  AMTYP1  ;MEM. TYPE, BLK#1
(1)      ;MEM. TYPE BYTE -- (HIGH BYTE)
(1)      ;900 NSEC CORE=001
(1)      ;300 NSEC BIPOLAR=002
(1)      ;500 NSEC MOS=003
(1)      000432      000000      $MADR1: .WORD  AMADR1  ;HIGH ADDRESS, BLK#1
(1)      ;MEM. LAST ADDR.=3 BYTES, THIS WORD AND LOW OF "TYPE" ABOVE
(1)      000434      000      $MAMS2: .BYTE  AMAMS2  ;HIGH ADDRESS, M.S. BYTE
(1)      000435      000      $MTYP2: .BYTE  AMTYP2  ;MEM. TYPE, BLK#2
(1)      000436      000000      $MADR2: .WORD  AMADR2  ;MEM. LAST ADDRESS, BLK#2
(1)      000440      000      $MAMS3: .BYTE  AMAMS3  ;HIGH ADDRESS, M.S. BYTE
(1)      000441      000      $MTYP3: .BYTE  AMTYP3  ;MEM. TYPE, BLK#3
(1)      000442      000000      $MADR3: .WORD  AMADR3  ;MEM. LAST ADDRESS, BLK#3
(1)      000444      000      $MAMS4: .BYTE  AMAMS4  ;HIGH ADDRESS, M.S. BYTE
(1)      000445      000      $MTYP4: .BYTE  AMTYP4  ;MEM. TYPE, BLK#4
(1)      000446      000000      $MADR4: .WORD  AMADR4  ;MEM. LAST ADDRESS, BLK#4
(1)      000450      .SETEND:
(1)      .MEXIT
1417      ;*****
1418      .SBTTL  BEGIN OF AREA TESTED (+20) WHEN PROGRAM RELOCATES.
1419
1420
1421      000500      000500      .=500
1422      000500      010706      BEGIN:  MOV      PC, SP      ;SET UP STACK POINTER
1423      000502      005746      TST      -(SP)      ;TO EQUAL BEGIN ADDRESS
1424      000504      012704      016132      MOV      #ENDPROG, R4 ;PUT END OF PROG ADDRESS INTO R4
1425      000510      010637      000346      MOV      SP, @SAVR6 ;SAVE SP FOR FUTURE USE
1426      000514      012737      000044      MOV      #PWRDN, @24 ;PREPARE FOR FUTURE POWER DOWN
1427      000522      004767      014560      JSR      PC, CLRM   ;CLEAR MEM MGMT.
  
```


DZMPL MACY11 30(1046) 05-AUG-77 12:59 PAGE 1-27
 DZMPLB.P11 05-AUG-77 12:57

BEGIN OF AREA TESTED (+20) WHEN PROGRAM RELOCATES.

SEQ 0029

1428	000526	005037	000310		CLR	2#TYP CNT	
1429	000532	105737	000170		TSTB	2#REL	:RELOCATED ?
1430	000536	100002			BPL	4\$	:BR IF NOT
1431	000540	000167	001344		JMP	TSTSIZ	
1432	000544	005067	002146	4\$:	CLR	LDFLG	
1433	000550	012737	000566	000004	MOV	87\$ 2#4	:PREPARE FOR NON-EXISTANT
1434	000556	012737	000014	177746	MOV	814 2#177746	:11/60 CACHE CONTROL REG
1435	000564	000402			BR	6\$	:BR IF ALL OK
1436	000566	062706	000004	7\$:	ADD	84 SP	:RECOVER FROM TRAP
1437	000572	012701	100000	6\$:	MOV	8100000,R1	:2ND 16K
1438	000576	005021			CLR	(R1)+	:CLEAR FIRST LOC
1439	000600	005011			CLR	(R1)	:CLEAR SECOND LOC
1440	000602	012777	020004	002120	MOV	820004,2CSRADR	:DIAG BIT
1441	000610	005741			TST	-(R1)	:READ THE LOC TO GET CHECKBITS IF ANY
1442	000612	032777	007740	002110	BIT	87740,2CSRADR	:IF SET WE'RE IN MS11K
1443	000620	001007			BNE	5\$	:BR IF IN MEM UNDER TEST
1444	000622	042767	020000	002060	BIC	820000,ECCDIS	:JUST IN CASE WE ARE TESTING
1445	000630	042767	020000	002054	BIC	820000,DIAGA	:THE SECOND CSR (172134)
1446	000636	000533			BR	SETSWR	:BR IF NOT IN MEM UNDER TEST
1447	000640	005077	002064	5\$:	CLR	2CSRADR	:CLEAR CSR
1448	000644	012767	000001	002044	MOV	81,LDFLG	:INDICATE PROG IN MEM UNDER TEST
1449	000652	052767	020000	002030	BIS	820000,ECCDIS	:DISABLE ECCINH IN LOW 16K
1450	000660	052767	020000	002024	BIS	820000,DIAGA	:SAME FOR DIAG CHECK
1451	000666	005001			CLR	R1	
1452	000670	005000			CLR	RO	:IT IS, CHECK FOR ERRORS
1453	000672	005067	002030		CLR	INHREL	
1454	000676	005077	002026	1\$:	CLR	2CSRADR	:DON'T WANT TO TRAP
1455	000702	005711			TST	(R1)	:READ THE LOC
1456	000704	032777	100000	002016	BIT	8100000,2CSRADR	:CHECK FOR DOUBLE ERROR
1457	000712	001403			BEQ	8\$	
1458	000714	004767	013274		JSR	PC,FATERR	:*ERROR* REPORT ERROR MESSAGE AND HALT AT FATHLT
(1)	000720	000001			1		:*****ERROR NUMBER 1*****
(1)							
1459	000722	005767	002000	8\$:	TST	INHREL	:ANY ERRORS FOUND YET ?
1460	000726	001012			BNE	2\$	:YES, DON'T LOOK ANY FURTHER
1461	000730	020127	000500		CMP	R1,8BEGIN	
1462	000734	101007			BHI	2\$	
1463	000736	032777	000020	001764	BIT	820,2CSRADR	:LOOKING FOR SBE'S IN 0-500
1464	000744	001403			BEQ	2\$	:BR IF NO ERRORS
1465	000746	012767	000001	001752	MOV	81,INHREL	:INHIBIT RELOCATION
1466	000754	022701	016132	2\$:	CMP	8ENDPROG,R1	:END OF PROGRAM YET?
1467	000760	003403			BLE	3\$	:BR IF YES
1468	000762	062701	000004		ADD	84,R1	:NO POINT TO NEXT LOCATION
1469	000766	000743			BR	1\$	:KEY READING
1470	000770	005767	001732	3\$:	TST	INHREL	:RELOCATION ALLOWED ?
1471	000774	001415			BEQ	ONEPAS	:BR IF YES
1472	000776	004767	013430		JSR	PC,TPCRLF	:PRINT ROUTINE
1473	001002	047516	051040	046105	.ASCIZ	/NO RELOC-SBE IN 0-500/	
	001010	041517	051455	042502			
	001016	044440	020116	026460			
	001024	030065	000060				
1474					.EVEN		
1475	001030	005000		ONEPAS:	CLR	RO	:INITIALIZE POINTER
1476	001032	005737	000404		TST	2#TESTN	:IS THIS THE FIRST PASS
1477	001036	001402			BEQ	TSTRP	:BR IF YES (TEST TRAP CATCHERS)
1478	001040	000167	000062		JMP	SETSWR	:SET UP SWREG

DZMPL MACY11 30(1046) 05-AUG-77 12:59 PAGE 1-28
DZMPLB.P11 05-AUG-77 12:57

SEG 0030

BEGIN OF AREA TESTED (+20) WHEN PROGRAM RELOCATES.

1479	001044	012704	016132		TSTRP:	MOV	#ENDPROG,R4	: ADDRESS OF END OF PROGRAM
1480	001050	012700	000377			MOV	#377,R0	
1481	001054	005001				CLR	R1	: POINT TO ADDRESS 0
1482	001056	012124			1\$:	MOV	(R1)+,(R4)+	: SAVE 0000 TO BEGIN-
1483	001060	020127	000400			CMP	R1,#\$MAIL	: BEGINNING OF ETABLE
1484	001064	103774				BLO	1\$	: BR IF NOT DONE
1485	001066	005741			3\$:	TST	-(R1)	: ADJUST POINTER TO TRAP VECTORS
1486	001070	010011			4\$:	MOV	R0,(R1)	: WRITE 1'S AND 0'S
1487	001072	020011				CMP	R0,(R1)	: NOW COMPARE
1488	001074	001403				BEQ	5\$	: BR IF OK
1489	001076	004767	013112			JSR	PC,FATERR	: *ERROR* REPORT ERROR MESSAGE AND HALT AT FATHLT
(1)	001102	000002				2		: *****ERROR NUMBER 2*****
(1)								
1490								
1491	001104	000300			5\$:	SWAB	R0	
1492	001106	001370				BNE	4\$	: BR IF BOTH BYTES NOT TESTED
1493	001110	005701				TST	R1	: HAVE WE REACHED THE BOTTOM YET
1494	001112	001365				BNE	3\$	: BRANCH IF NO
1495	001114	012701	000400			MOV	#\$MAIL,R1	: APT, RESTORE TO \$MAIL
1496	001120	014441			6\$:	MOV	-(R4),-(R1)	: HERE!
1497	001122	005701				TST	R1	: FINISHED?
1498	001124	001375				BNE	6\$	: BR IF NOT
1499	001126	005077	001576		SETSWR:	CLR	\$CSRADR	: CLEAR CSR
1500	001132	012700	000006			MOV	#6,R0	: ADDRESS OF PSW FOR TIMEOUT TRAP
1501	001136	012710	000340			MOV	#340,(R0)	: SET UP TIME OUT TRAP PSW
1502	001142	012740	001170			MOV	#2\$,-(R0)	: AND RETURN ADDRESS
1503	001146	105737	000420			TSTB	\$SENV	: RUNNING UNDER APT
1504	001152	001403				BEQ	1\$	: IF NOT, SET UP ADDRESS FOR SWR
1505	001154	012737	000422	002732		MOV	#SSWREG,\$SWR	: OTHERWISE, PREPARE TO USE APT SWR
1506	001162	005777	001544		1\$:	TST	\$SWR	: DOES SWITCH REG POINTED TO BY SWR EXIST
1507	001166	000410				BR	APTSIZ	: YES, GO TO APTSIZ
1508	001170	062706	000004		2\$:	ADD	#4,SP	: RESTORE STACK POINTER
1509	001174	012737	000176	002732		MOV	#SWREG,\$SWR	: PLACE ADDRESS OF SWITCH REG DESIGNED
1510	001202	012737	000174	002734		MOV	#DSPREG,\$DISPLAY	: FOR COMPUTERS NOT HAVING HARDWARE
1511								: SWITCH REG AND RUNNING STAND ALONE.
1512								: R4=# END PROG ON EXIT
1513								
1514								
1515								
1516								
1517								
1518								
1519								
1520								
1521	001210	012703	000336		APTSIZ:	MOV	#MAXMEM,R3	: POINT R3 TO MAXMEM
1522	001214	013737	000326	000332		MOV	\$HIGHTWO,\$SHIMAX	: IN CASE NO SELF SIZING DONE
1523	001222	013737	000330	000334		MOV	\$HIGHADD,\$\$MAXM	
1524	001230	105737	000421			TSTB	\$SENV	: DOES APT ALLOW SELF SIZING
1525	001234	100021				BPL	TRYR	: BR IF YES
1526	001236	012701	000451			MOV	#SMTYP4+4,R1	: POINT R1 TO BLOCK TYPE 4+4
1527	001242	162701	000004		1\$:	SUB	#4,R1	: POINT TO NEXT BLOCK TYPE
1528	001246	105711				TSTB	(R1)	: IS IT NON-ZERO
1529	001250	001006				BNE	2\$	: BR IF YES (MEMORY EXISTS)
1530	001252	020127	000431			CMP	R1,\$SMTYP1	: ALL APT BLOCK TYPES BEEN CHECKED
1531	001256	101371				BHI	1\$	: BR IF NO
1532	001260	004767	012730			JSR	PC,FATERR	: *ERROR* REPORT ERROR MESSAGE AND HALT AT FATHLT

: APTSIZ-THIS ROUTINE WILL SEARCH THE APT MEMORY ETABLE
: AND WHEN A NON ZERO TYPE IS FOUND WILL SETUP TO TEST TO
: GIVEN HIGH ADDRESS. IF APT DEFINES SIZE LO ADDRESS MUST -00000

E03

DZMPL MACY11 30(1046) 05-AUG-77 12:59 PAGE 1-29
 DZMPLB.P11 05-AUG-77 12:57

BEGIN OF AREA TESTED (+20) WHEN PROGRAM RELOCATES.

SEQ 0031

```

(1) 001264 000003          3          ;*****ERROR NUMBER 3*****
(1)
1533
1534 001266 004767 013720      2S:   JSR   PC,GETADR      ;SET MAX APT ADDR INTO $MAXM +$HIMAX
1535 001272 004767 013714      JSR   PC,GETADR      ;ALSO INTO HIGHADD + HIGHTWO
1536 001276 000564              BRTPSZ: BR   TYPsiz      ;TYPE SIZE OF MEM UNDER TEST
1537
1538 001300 032777 000100 001424 TRYSR: BIT   $100,JSWR    ;USER DEFINED BOUNDARIES
1539 001306 001160              BNE   TYPsiz      ;BR IF YES (DON'T SIZE MEM)
1540
1541 001310 005067 001400      SLFSiz: CLR   MSYES        ;INITIALIZE MSYES
1542 001314 012703 000324      MOV   $LOWADD,R3      ;GET READY FOR LOW TEST ADDRESS
1543 001320 005001              CLR   R1              ;FIRST TEST LOC (0)
1544 001322 012710 001514      MOV   $4S,(R0)        ;SET UP RETURN FROM TIMEOUT TRAP
1545 001326 005077 001376      1S:   CLR   @CSRADR      ;
1546 001332 011146              MOV   (R1),-(SP)        ;SAVE THE CONTENTS OF TEST LOC.
1547 001334 016146 000002      MOV   2(R1),-(SP)      ;
1548 001340 005011              CLR   (R1)            ;TO SET CHECK BITS OF MS11K
1549 001342 005061 000002      CLR   2(R1)            ;
1550 001346 016777 001340 001354 MOV   DIAGA,@CSRADR  ;DIAGNOSTIC BIT
1551 001354 005711              TST   (R1)            ;READ TO GET CHECK BITS
1552 001356 017702 001346      MOV   @CSRADR,R2        ;GET CSR CONTENTS
1553 001362 005077 001342      CLR   @CSRADR      ;
1554 001366 012661 000002      MOV   (SP)+,2(R1)      ;RESTORE THE HI WORD
1555 001372 012611              MOV   (SP)+,(R1)      ;RESTORE THE LO WORD
1556 001374 042702 170037      BIC   $170037,R2      ;ONLY CARE ABOUT 11-5
1557 001400 005702              TST   R2              ;
1558 001402 001440              BEQ   3S              ;BR IF NO MS11K
1559 001404 005767 001304      TST   MSYES          ;MS11K-FIRST SUCCESSFUL TEST
1560 001410 001032              BNE   22S          ;NO, BRANCH
1561 001412 020127 100000      CMP   R1,$100000      ;IF THIS IS FIRST READABLE LOC
1562 001416 001015              BNE   11S          ;THEN FIRST ADDRESS IS 0000, ELSE BR
1563 001420 105737 000272      TSTB   @MMAVA        ;MEM MGMT AVAIL ?
1564 001424 001012              BNE   11S          ;YES, GO PUT ADDR AWAY
1565 001426 005767 001264      TST   LDFLG        ;RUNNING IN MEM UNDER TEST
1566 001432 001407              BEQ   11S          ;BR IF NOT
1567 001434 005037 000324      CLR   @LOWADD      ;CLEAR THESE LOCATIONS
1568 001440 005037 000322      CLR   @LOWTWO      ;INDICATE SOME MS11K FOUND
1569 001444 005267 001244      INC   MSYES        ;CONTINUE TO FIND HIGH LIMIT
1570 001450 000404              BR    2S          ;PLACE ADDR AWAY (START ADDR)
1571 001452 004767 013444      11S:  JSR   PC,PUTADR    ;INDICATE SOME MS11 FOUND
1572 001456 005267 001232      INC   MSYES        ;FIRST TEST ADDRESS
1573 001462 010137 000320      2S:   MOV   R1,@MINMEM    ;SAVE FOR RECOVERY FROM RELOC
1574 001466 010167 001230      MOV   R1,$AVMIN    ;POINT TO $MAXMEM
1575 001472 062703 000012      ADD   $12,R3        ;NEXT 4K BOUNDARY
1576 001476 062701 020000      22S:  ADD   $20000,R1    ;GO TEST IT
1577 001502 000711              BR    1S          ;FOUND ANY MS11K YET
1578 001504 005767 001204      3S:   TST   MSYES        ;BR IF NO
1579 001510 001772              BEQ   22S          ;
1580 001512 000430              BR    7S          ;
1581 001514 062706 000004      4S:   ADD   $4,SP      ;RESTORE STACK POINTER AFTER TRAP
1582 001520 004767 013212      JSR   PC,MEMMNG    ;SERVICE MM IF AVAILABLE
1583 001524 105737 000272      TSTB   @MMAVA        ;SEE IF MM HAS TO BE TESTED
1584 001530 001421              BEQ   7S          ;BR IF NO MM
1585 001532 012710 001544      5S:   MOV   $6S,(R0)    ;SET UP RETURN ADDRESS FROM TRAP
1586 001536 012701 040000      MOV   $40000,R1    ;BEGIN CHECKING ABOVE 28K

```

F03

DZMML MACY11 30(1046) 05-AUG-77 12:59 PAGE 1-30
 DZMMLB.P11 05-AUG-77 12:57

BEGIN OF AREA TESTED (+20) WHEN PROGRAM RELOCATES.

SEQ 0032

1587	001542	000671			BR	1\$	
1588	001544	062706	000004		ADD	#4, SP	: RESTORE STACK POINTER
1589	001550	022701	160000		CMP	#160000, R1	: IF R1 DID NOT READ THE HIGHEST
1590							: LOCATION POINTED TO BY PAR6,
1591							: IT HAS REACHED MAX MS11K.
1592	001554	001007			BNE	7\$	: GO TO 7\$ IF DONE
1593	001556	022737	007400	172352	CMP	#7400, #172352	: CHECK PAR5 FOR MAX
1594	001564	001403			BEQ	7\$	: BR IF DONE
1595	001566	004767	013322		JSR	PC, UPMM	: NOT DONE UPDATE MEM MGMT.
1596	001572	000757			BR	5\$	
1597							
1598							
1599	001574	024341			CMP	-(R3), -(R1)	: CAUSE R3 TO POINT TO \$MAXM AND
1600							: R1 TO MAX AVAILABLE MEMORY
1601	001576	020127	037776		CMP	R1, #37776	: MEANS WE MOVED INTO PAR2
1602	001602	001014			BNE	8\$	: IF WE DID WE MUST ADJUST ADDRESS
1603	001604	005721			TST	(R1)+	: RESTORE R1
1604	001606	004767	013310		JSR	PC, PUTADR	: PUT AWAY ORIGINAL ADDRESS
1605	001612	162713	000002		SUB	#2, (R3)	: ADJUST BITS 15-0
1606	001616	162703	000004		SUB	#4, R3	: POINT TO HIGHADD
1607	001622	004767	013274		JSR	PC, PUTADR	: PUT IT AWAY
1608	001626	162713	000002		SUB	#2, (R3)	: ADJUST
1609	001632	000406			BR	TYPSIZ	: AND CONTINUE TO TYPE THE SIZE
1610	001634	004767	013262		JSR	PC, PUTADR	: PLACE ADDR IN R1 AT LOCATIONS
1611							: \$MAXM AND \$HIMAX
1612	001640	162703	000004		SUB	#4, R3	: POINT TO HIGHADD
1613	001644	004767	013252		JSR	PC, PUTADR	: PLACE ADDR IN R1 AT HIGHADD & HIGHTWO
1614							
1615							
1616	001650	012710	000352		MOV	#BUSER, (R0)	: SET UP VECTOR FOR FUTURE TRAP
1617	001654	005737	000406		TST	#SPASS	: ONLY CARE ABOUT FIRST PASS
1618	001660	001035			BNE	SETSTK	: BR IF NOT
1619	001662	004767	012544		JSR	PC, TPCRLF	: PRINT ROUTINE
1620	001666	051503	020122	042101	.ASCIZ	/CSR ADDR = /	
1621	001674	051104	036440	000040			
1622					.EVEN		
1623	001702	012703	002664		MOV	#DATABUF, R3	: NEED A LOCATION TO PRINT FROM
1624	001706	005023			CLR	(R3)+	: CLEAR HIGH ORDER BITS
1625	001710	016713	001014		MOV	CSRADR, (R3)	: ADDRESS TO BE TYPED
1626	001714	105237	000310		INCB	#TYPCNT	: ONE WORD TO BE TYPED
1627	001720	004767	012656		JSR	PC, TYPOCT	: TYPE THE WORD
1628	001724	010403			MOV	R4, R3	: POINT R3 TO LOWEST AVAIL MEM
1629	001726	012701	000322		MOV	#LOWTWO, R1	
1630	001732	004767	012462		JSR	PC, PCRLF	: TYPE <CR><LF>
1631	001736	004767	012630		JSR	PC, OCTTYP	: TYPE LOW TEST ADDRESS
1632							: (LOWTWO + LOWADD)
1633	001742	004767	012364		JSR	PC, \$TYPE	
1634	001746	000055			.ASCIZ	1-1	: TYPE "--"
1635					.EVEN		
1636	001750	004767	012616		JSR	PC, OCTTYP	: TYPE HIGHEST TEST ADDRESS
1637							: (HIGHTWO + HIGHADD)
1638	001754	012703	000326		MOV	#HIGHTWO, R3	: POINT R3 TO HIGH ORDER BITS OF HIGH MEM
1639	001760	004767	013242		JSR	PC, \$GTSIZ	: GET THE BITS 13-17 ON TOP ADDRESS
1640							: PLACED IN BITS 0-4 OF R2
1641	001764	010401			MOV	R4, R1	: #ENDPROG
1642	001766	062704	000022		ADD	#18., R4	: APPEND THE ERROR STACK FOR MEMORY

H03

DZMML MACY11 30(1046) 05-AUG-77 12:59 PAGE 1-32
DZMMLB.P11 05-AUG-77 12:57

SEQ 0034

BEGIN OF AREA TESTED (+20) WHEN PROGRAM RELOCATES.

```

1696 002200 101001      BHI      9$      ;IF 50, GO TO 9$
1697 002202 011305      MOV      (R3),R5 ;MODIFY R5
1698 002204 020405      9$:      CMP      R4,R5 ;IS LOW LIMIT LOWER THAN HIGH LIMIT
1699 002206 103403      BLO      10$
1700 002210 004767 012000 JSR      PC,FATERR ;*ERROR* REPORT ERROR MESSAGE AND HALT AT FATHLT
(1) 002214 000005      5 ;*****ERROR NUMBER 5*****
(1)
1701
1702
1703 002216 012703 000340 10$:      MOV      #SAVMAX,R3
1704 002222 011343      MOV      (R3),-(R3) ;RESTORE CONTENTS OF MAXMEM
1705 002224 062713 000002 MEMTST: ADD      #2,(R3) ;MAKE THE CONTENTS OF MAXMEM=
1706 ;MAX AVAIL MEM+2
1707 002230 005725      TST      (R5)+ ;SET R5=MAXMEM +2
1708
1709
1710 ;INIT BAKPAT AND SWAPAT
1711 ;AND CONTINUE TO CLEAR ACTIVE
1712 ;CHUNK OF MEMORY UNDER TEST
1713
1714 002232 012702 000312 CLRMEM: MOV      #BAKPAT,R2
1715 002236 012212      MOV      (R2)+,(R2) ;COPY IT INTO SWAPAT
1716 002240 000312      SWAB      (R2) ;SWAP THE BYTES
1717 002242 017702 000464      MOV      #SWR,R2 ;GET BITS IN SW REG
1718 002246 042702 177740      BIC      #177740,R2 ;ONLY WANT THE TEST NUMBER FOR LOOPING
1719 002252 022702 000021      CMP      #21,R2 ;CAN'T BE HIGHER THAN 20
1720 002256 101003      BHI      1$ ;BR IF OK
1721 002260 004767 011730      JSR      PC,FATERR ;*ERROR* REPORT ERROR MESSAGE AND HALT AT FATHLT
(1) 002264 000006      6 ;*****ERROR NUMBER 6*****
(1)
1722
1723 002266 013701 000320 1$:      MOV      #MINMEM,R1 ;GET FIRST TEST LOC IN SLICE
1724 002272 004767 013200 2$:      JSR      PC,TSTADD ;CHECK FOR ERROR FREE TEST ADDRESS
1725 002276 005703      TST      R3 ;R3 = 0 MEANS ERROR FREE LOC FOUND
1726 002300 001446      BEQ      4$ ;BR AND SAVE IT
1727 002302 016777 000402 000420 3$:      MOV      ECCDIS,#CSRADR ;ECC DISABLE
1728 002310 062701 000004      ADD      #4,R1 ;NEXT WORD
1729 002314 020105      CMP      R1,R5 ;END OF SLICE YET ?
1730 002316 103765      BLO      2$ ;BR IF NOT
1731 002320 004767 012106      JSR      PC,TPCRLF ;PRINT ROUTINE
1732 002324 047516 042440 051122 .ASCII /NO ERROR FREE LOC FOR ECC TESTS/
002332 051117 043040 042522
002340 020105 047514 020103
002346 047506 020122 041505
002354 020103 042524 052123
002362 123
1733 002363 015 041012 043505 .ASCIIZ <15><12>/BEGINNING AT TST13/
002370 047111 044516 043516
002376 040440 020124 051524
002404 030524 000063
1734
1735 002410 012702 000013 .EVEN
1736 002414 000402      MOV      #13,R2 ;START AT TST13
1737 002416 010137 000320 4$:      BR      CONT
1738 002422 016777 000262 000300 CONT:      MOV      R1,#MINMEM ;SET UP TEST LOC
1739 002430 010500      MOV      ECCDIS,#CSRADR ;INHIBIT ECC
MOV      R5,R0 ;

```


DZMML MACY11 30(1046) 05-AUG-77 12:59 PAGE 1-33
 DZMMLB.P11 05-AUG-77 12:57

BEGIN OF AREA TESTED (+20) WHEN PROGRAM RELOCATES.

SEQ 0035

```

1740 002432 005040          SS:  CLR  -(R0)      ;BEGIN CLEARING FROM TOP
1741 002434 020004          CMP   R0,R4        ;TO BOTTOM
1742 002436 101375          BHI   SS           ;
1743 002440 005077 000264   CLR   @CSRADR      ;RESTORE CSR
1744
1745                          ;ENTER HERE FROM TSTSCP ROUTINE AT END OF EACH TEST
1746
1747 002444 005037 000302     CLR   @PASFLG      ;INIT SOFTEST PASS FLAG
1748 002450 110237 000404     MOVB  R2,@$TESTN   ;SET UP $TESTN WITH THE TEST
1749 002454 110277 000254     MOVB  R2,@DISPLAY  ;NUMBER TO BE EXECUTED & DISPLAYED
1750
1751 002460 005077 000244     LOOP: CLR   @CSRADR ;
1752 002464 032777 004000 000240  BIT   #4000,@SWR ;ENABLE PRINTOUT OF SBE'S(INH ECC)
1753 002472 001403          BEQ   IS           ;BR IF NO
1754 002474 016777 000210 000226  MOV   ECCDIS,@CSRADR ;YES, INHIBIT ECC
1755 002502 010401          MOV   R4,R1        ;LOWEST ADDRESS UNDER TEST
1756 002504 010246          MOV   R2,-(SP)     ;SAVE R2
1757 002506 012703 000376     MOV   #376,R3    ;POINT R3 TO SCRATCH STACK
1758 002512 004767 012404     JSR   PC,PUTADR   ;GEN 18 BIT ADDRESS FROM R1
1759
1760 002516 005743          TST   -(R3)        ;AND STORE IT IN (R3) AND (R3-2)
1761 002520 004767 012502     JSR   PC,$GTSIZ  ;POINT R3 TO HIGH ORDER BITS
1762
1763 002524 010400          MOV   R4,R0        ;PLACE BITS 13-17 OF ADDRESS BITS
1764
1765 002526 010401          MOV   R4,R1        ;INTO 0-4 OF R2.
1766 002530 010403          MOV   R4,R3        ;PLACE ADDRESS OF LOWEST LOC
1767 002532 012602          MOV   (SP)+,R2     ;UNDER TEST IN R0
1768 002534 006302          ASL   R2           ;AND INTO R1
1769 002536 060702          ADD   PC,R2        ;AND INTO R3
1770 002540 066207 000004     ADD   TBL-,(R2),PC ;RESTORE R2
1771
1772          TBL:  TST0-TBL ;DOUBLE IT
1773          TST1-TBL ;GO TO TEST # STORED IN BITS
1774          TST2-TBL ;3-0 OF SWITCH REG.
1775          TST3-TBL ;RELATIVE ADDRESS OF TEST 0
1776          TST4-TBL ;RELATIVE ADDRESS OF TEST 1
1777          TST5-TBL ;RELATIVE ADDRESS OF TEST 2
1778          TST6-TBL ;RELATIVE ADDRESS OF TEST 3
1779          TST7-TBL ;RELATIVE ADDRESS OF TEST 4
1780          TST10-TBL ;RELATIVE ADDRESS OF TEST 5
1781          TST11-TBL ;RELATIVE ADDRESS OF TEST 6
1782          TST12-TBL ;RELATIVE ADDRESS OF TEST 7
1783          TST13-TBL ;RELATIVE ADDRESS OF TEST 10
1784          TST14-TBL ;RELATIVE ADDRESS OF TEST 11
1785          TST15-TBL ;RELATIVE ADDRESS OF TEST 12
1786          TST16-TBL ;RELATIVE ADDRESS OF TEST 13
1787          TST17-TBL ;RELATIVE ADDRESS OF TEST 14
1788          RELOC-TBL ;RELATIVE ADDRESS OF TEST 15
1789
1790                          ;SCOPE ROUTINE
1791                          ;
1792                          ;PROG COMES HERE AFTER EACH TEST AND
1793                          ;IF CTRL C GO TO ERROR HISTORY TYPEOUT.
1794                          ;IF SR=2000 (BIT 10) THEN HALT
1795                          ;IF SR=40000 (BIT 14) THEN LOOP ON TEST
                          ;DEFINED BY SR BITS <3:0>, ELSE GO ON

```

J03

DZMML MACY11 30(1046) 05-AUG-77 12:59 PAGE 1-34
DZMMLB.P11 05-AUG-77 12:57

SEQ 0036

BEGIN OF AREA TESTED (+20) WHEN PROGRAM RELOCATES.

TO NEXT TEST.

```

1796
1797
1798 002606 005077 000116 TSTSCP: CLR ;CSRADR ;
1799 002612 105737 000420 TSTB ;SENV ;RUNNING UNDER APT?
1800 002616 001002 BNE CNTSCP ;IF SO GO TO CNTSCP
1801 002620 004767 013230 JSR PC,CHECKC ;TEST FOR CTRL-C AND IF TYPED
1802 ;GO TO ERROR HISTORY AND HALT
1803 002624 113702 000404 CNTSCP: MOVB ;$TESTN,R2 ;PUT TEST NUMBER IN R2 LO BYTE
1804 002630 005237 000410 INC ;$DEVCT ;TELL APT EVERYTHING'S OK
1805 002634 032777 002000 000070 BIT ;2000,$SWR ;HALT AFTER EACH TEST?
1806 002642 001401 BEQ TSTGO ;BRANCH IF NOT HALTING
1807 002644 000000 SWHALT: HALT
1808 002646 032777 040000 000056 TSTGO: BIT ;40000,$SWR ;LOOP ON TEST DESIRED?
1809 002654 001301 BNE LOOP ;YES, GO START SAME TEST
1810 002656 105202 INCB R2
1811 002660 000167 177536 JMP CONT ;GO CONTINUE EXECUTING NEXT TEST
1812
1813 002664 000000 000000 DATBUF: .WORD 0,0
1814 002670 000000 000000 TSTDAT: .WORD 0,0
1815 002674 000000 000000 SBEMSK: .WORD 0,0
1816 002700 000000 000000 DBEMSK: .WORD 0,0
1817 002704 000000 000000 DATSAV: .WORD 0,0
1818 002710 000002 ECCDIS: .WORD 2
1819 002712 000004 DIAGA: .WORD 4
1820 002714 000000 MSYES: .WORD 0
1821 002716 000000 LDFLG: .WORD 0
1822 002720 000000 BASE: .WORD 0 ;OFFSET FOR RELOC
1823 002722 100000 SAVMIN: .WORD 100000 ;TO SAVE MINMEM FROM RELOC
1824 002724 000000 ERRFLG: .WORD 0 ;FLAG TO INDICATE FIRST ERROR FOR HEADER
1825 002726 000000 INHREL: .WORD 0 ;FLAG TO INDICATE NO RELOCATION ALLOWED
1826 002730 172136 CSRADR: 172136 ;DEFAULT CSR ADDRESS
1827
1828
1829
1830
1831
1832
1833
1834
1835

```

```

:*****
SWR: 177570 ;CHANGES TO SWREG IF NO HARDWARE SWITCH REGISTER
DISPLAY: 177570 ;CHANGES TO DSPREG IF NO HARDWARE DISPLAY REG

```

```

;*(1) THIS TEST ASSUMES THAT THE MEMORY IS IN A STATE
;* OF ALL 0'S AND R0 HAS THE ADDRESS OF THE LOWEST
;* LOCATION UNDER TEST
;*(2) IT CHECKS FOR PROPER BANK SELECTION BY WRITING
;* 1'S IN A LOCATION AND CHECKING FOR 0'S IN THE SAME
;* LOCATIONS OF OTHER 4K BANKS OF THE MEMORY
;* [I.E. LOCATIONS LIKE 7766 AND 27766 ETC.]

```

```

1847 002736 122737 000000 000404 TSTO: CMPB ;0,$TESTN ;CHECK FOR PROPER TEST SEQUENCE
1848 002744 001403 BEQ +10
1849 002746 004767 011242 JSR PC,SEQERR ;*ERROR* REPORT ERROR MESSAGE AND HALT AT FATHLT
(1) 002752 000007 7 ;*****ERROR NUMBER 7*****
(1)

```


K03

 DZMML MACY11 30(1046) 05-AUG-77 12:59 PAGE 1-35
 DZMMLB.P11 05-AUG-77 12:57

SEQ 0037

BEGIN OF AREA TESTED (+20) WHEN PROGRAM RELOCATES.

1850	002754	012703	177777		MOV	#177777,R3	
1851	002760	010401		1\$:	MOV	R4,R1	;R1 = ADDRESS OF LOWEST LOCATION OF MEMORY UNDER TEST
1852	002762	010310			MOV	R3,(R0)	;SET ALL THE BITS AT (R0)
1853	002764	020001		2\$:	CMP	R0,R1	;IS R0 POINTING TO THE SAME MEMORY LOCATION AS R1
1854	002766	001417			BEQ	4\$	;IN WHICH CASE CHECK FOR ALL 1'S AT (R1)
1855	002770	005711			TST	(R1)	;OTHERWISE CHECK (R1) FOR ALL 0'S
1856	002772	001430			BEQ	5\$	
1857	002774	020311			CMP	R3,(R1)	;IF R1 IS NOT EQUAL TO R0 AND (R1)
1858							;DOES NOT CONTAIN ALL 0'S THEN
1859							;CHECK TO SEE IF (R0) = (R1)
1860	002776	001004			BNE	3\$	
1861	003000	012767	000010	000042	MOV	#10,12\$	;*****ERROR NUMBER 10*****
(1)							
1862	003006	000403			BR	10\$	
1863	003010						
(1)	003010	012767	000011	000032	3\$:	MOV	#11,12\$
(1)							;*****ERROR NUMBER 11*****
1864	003016	010046			10\$:	MOV	R0,-(SP)
1865	003020	105237	000275			INCB	2\$ADERR
1866	003024	000407				BR	11\$
1867	003026	020311			4\$:	CMP	R3,(R1)
1868	003030	001411				BEQ	5\$
1869	003032	012767	000012	000010		MOV	#12,12\$
(1)							;*****ERROR NUMBER 12*****
1870	003040	010046				MOV	R0,-(SP)
1871	003042	010300				MOV	R3,R0
1872	003044	004767	010470		11\$:	JSR	PC,ERROR
1873	003050	000000			12\$:	.WORD	
1874	003052	012600				MOV	(SP)+,R0
1875							;RESTORE RO ON STACK
1876	003054	013706	000346		5\$:	MOV	2\$SAVR6,SP
1877	003060	062701	020000			ADD	#20000,R1
1878							;CAUSE R1 TO POINT TO THE SAME CHIP
1879							;LOCATION IN THE NEXT 4K BANK OF MEMORY
1880	003064	020105				CMP	R1,R5
1881							;BY ADDING 1 TO THE 14TH BIT OF ADDRESS IN R1
1882	003066	103736				BLO	2\$
1883							;COMPARE R1 WITH THE HIGHEST MEMORY
1884	003070	000646					;LOCATION WHICH IS STORED IN R5
1885							;IF R1 LESS THAN R5 THEN REPEAT THE TEST FROM 2\$
1886					END0:	BR	TSTSCP
1887							;INITIAL DATA TEST
1888							THIS TEST CHECKS THE DI/DO LINES BY
1889	003072	122737	000001	000404	TST1:	CMPB	#1,2\$TESTN
1890	003100	001403				BEQ	+10
1891	003102	004767	011106			JSR	PC,SEQERR
(1)	003106	000013				13	;*****ERROR NUMBER 13*****
(1)							
1892	003110	013701	000320			MOV	2\$MINMEM,R1
1893	003114	012767	000001	177542	1\$:	MOV	#1,DATBUF
1894	003122	005067	177540			CLR	DATBUF+2
1895	003126	016711	177532		2\$:	MOV	DATBUF,(R1)
1896	003132	016761	177530	000002		MOV	DATBUF+2,2(R1)
1897	003140	026711	177520			CMP	DATBUF,(R1)
1898	003144	001405				BEQ	4\$
1899	003146	016700	177512			MOV	DATBUF,R0
							;GET TEST ADDRESS
							;SET THE FIRST TEST BIT
							;CLEAR 2ND WORD
							;WRITE TEST WORD 1
							;AND TEST WORD 2
							;NOW READ THEM
							;BR IF FIRST 16 OK
							;GET GOOD DATA FOR ERROR MSG

L03

DZMML MACY11 30(1046) 05-AUG-77 12:59 PAGE 1-36
 DZMMLB.P11 05-AUG-77 12:57 BEGIN OF AREA TESTED (+20) WHEN PROGRAM RELOCATES.

SEQ 0038

```

1900 003152 004767 010362      JSR    PC,ERROR      ;*ERROR* REPORT ERROR MESSAGE
      (1) 003156 000014      14      ;*****ERROR NUMBER 14*****
      (1)
1901
1902 003160 026761 177502 000002 4$:    CMP    DATBUF+2,2(R1) ;NOW READ SECOND WORD
1903 003166 001405      BEQ    5$      ;BR IF OK
1904 003170 016700 177472      MOV    DATBUF+2,R0      ;GOOD DATA FOR ERROR MSG
1905 003174 004767 010340      JSR    PC,ERROR      ;*ERROR* REPORT ERROR MESSAGE
      (1) 003200 000015      15      ;*****ERROR NUMBER 15*****
      (1)
1906
1907
1908 003202 005767 177460      5$:    TST    DATBUF+2      ;HAS LAST BIT BEEN TESTED ?
1909 003206 100404      BMI    6$      ;MINUS MEANS BIT 31
1910 003210 004567 010140      JSR    R5,DASHL      ;NO, SHIFT TEST BIT LEFT
1911 003214 002664      .WORD  DATBUF      ;IN DATBUF
1912 003216 000743      BR      2$      ;GO WRITE NEW TEST DATA
1913
1914      ;NOW GOING TO SHIFT A 0 IN DATA DIRECTION
1915 003220 012767 177776 177436 6$:    MOV    #177776,DATBUF ;PUT A 0 IN BIT 0
1916 003226 012767 177777 177432      MOV    #-1,DATBUF+2 ;AND 1'S IN ALL OTHERS
1917 003234 012702 002664      MOV    #DATBUF,R2      ;LOC OF WRITE DATA
1918 003240 066702 177454      ADD    BASE,R2      ;OFFSET IT IN CASE OF RELOC
1919 003244 016711 177414      66$:   MOV    DATBUF,(R1) ;WRITE THE DATA
1920 003250 016761 177412 000002      MOV    DATBUF+2,2(R1) ;2 WORDS WORTH
1921 003256 026711 177402      CMP    DATBUF,(R1) ;NOW READ FIRST WORD
1922 003262 001405      BEQ    7$      ;BR IF OK
1923 003264 016700 177374      MOV    DATBUF,R0      ;GOOD DATA
1924 003270 004767 010244      JSR    PC,ERROR      ;*ERROR* REPORT ERROR MESSAGE
      (1) 003274 000016      16      ;*****ERROR NUMBER 16*****
      (1)
1925
1926 003276 026761 177364 000002 7$:    CMP    DATBUF+2,2(R1) ;NOW, READ SECOND WORD
1927 003304 001405      BEQ    8$      ;BR IF OK
1928 003306 016700 177354      MOV    DATBUF+2,R0      ;GOOD DATA
1929 003312 004767 010222      JSR    PC,ERROR      ;*ERROR* REPORT ERROR MESSAGE
      (1) 003316 000017      17      ;*****ERROR NUMBER 17*****
      (1)
1930
1931 003320 022762 077777 000002 8$:    CMP    #77777,2(R2) ;TESTED BIT 31 YET?
1932 003326 001410      BEQ    10$     ;BR IF YES, WE'RE DONE
1933 003330 006162 000002      ROL    2(R2)      ;SHIFT DATBUF
1934 003334 006112      ROL    (R2)      ;AND DATBUF +2
1935 003336 103403      BCS    9$      ;
1936 003340 005362 000002      DEC    2(R2)      ;
1937 003344 000261      SEC      ;SET CARRY FOR NEXT ROL
1938 003346 000736      9$:    BR      66$     ;KEEP GOING
1939 003350 062701 020000      10$:   ADD    #20000,R1 ;NEXT 4K BANK
1940 003354 020105      CMP    R1,R5      ;COMPARE AGAINST HIGHEST VIRTUAL MEM
1941 003356 103656      BLO    1$      ;
1942
1943 003360 000167 177222      END1:   JMP    TSTSCP
1944
1945
1946      ;TEST2 THIS TEST CHECKS TO SEE THAT EACH ADDRESS
1947      ;      BIT IN EACH 4K BANK CAN BE ASSERTED UNIQUELY.

```


MO3

DZMML MACY11 30(1046) 05-AUG-77 12:59 PAGE 1-37
DZMMLB.P11 05-AUG-77 12:57

BEGIN OF AREA TESTED (+20) WHEN PROGRAM RELOCATES.

SEQ 0039

```

1948                                     ;
1949                                     ;
1950                                     ;
1951                                     ;
1952 003364 122737 000002 000404 TST2: CMPB #2,2#STESTN ;CHECK FOR PROPER TEST SEQUENCE
1953 003372 001403 BEQ 1$
1954 003374 004767 010614 JSR PC,SEQERR ;*ERROR* REPORT ERROR MESSAGE AND HALT AT FATHLT
(1) 003400 000020 20 ;*****ERROR NUMBER 20*****
(1)
1955 003402 012701 040000 1$: MOV #40000,R1 ;
1956 003406 005000 CLR R0 ;
1957 003410 010146 MOV R1, -(SP) ;SAVE START ADDRESS
1958 003412 005021 11$: CLR (R1)+ ;CLEAR A LOCATION
1959 003414 020105 CMP R1,R5 ;END OF 20K SLICE YET ?
1960 003416 103775 BLO 11$ ;BR IF NO
1961 003420 011601 MOV (SP),R1 ;RESTORE R1 NOW
1962 003422 012702 000001 2$: MOV #1,R2 ;SET FIRST BIT
1963 003426 050201 BIS R2,R1 ;POINT R1 TO FIRST BYTE
1964 003430 105711 TSTB (R1) ;READ AND COMPARE FOR ZEROS
1965 003432 001403 BEQ 3$ ;BR IF OK
1966 003434 004767 010100 JSR PC,ERROR ;*ERROR* REPORT ERROR MESSAGE
(1) 003440 000021 21 ;*****ERROR NUMBER 21*****
(1)
1967 003442 105100 3$: COMB R0 ;FOR ERROR MSG
1968 003444 105111 COMB (R1) ;COMPLEMENT THE BYTE
1969 003446 105711 TSTB (R1) ;READ FOR NON ZEROS
1970 003450 001003 BNE 4$ ;BR IF OK
1971 003452 004767 010062 JSR PC,ERROR ;*ERROR* REPORT ERROR MESSAGE
(1) 003456 000022 22 ;*****ERROR NUMBER 22*****
(1)
1972 003460 040201 4$: BIC R2,R1 ;MASK OFF THE ASSERTED BIT
1973 003462 006302 ASL R2 ;SHIFT R2 FOR NEXT BIT
1974 003464 050201 BIS R2,R1 ;SET THE NEW BIT INTO R1
1975 003466 005711 TST (R1) ;READ THE NEW ADDRESS
1976 003470 001404 BEQ 5$ ;READ FOR ZEROS
1977 003472 005000 CLR R0 ;
1978 003474 004767 010040 JSR PC,ERROR ;*ERROR* REPORT ERROR MESSAGE
(1) 003500 000023 23 ;*****ERROR NUMBER 23*****
(1)
1979 003502 005100 5$: COM R0 ;
1980 003504 005111 COM (R1) ;COMPL THE WORD
1981 003506 005711 TST (R1) ;READ IT AGAIN
1982 003510 001003 BNE 6$ ;
1983 003512 004767 010022 JSR PC,ERROR ;*ERROR* REPORT ERROR MESSAGE
(1) 003516 000024 24 ;*****ERROR NUMBER 24*****
(1)
1984 003520 032702 010000 6$: BIT #10000,R2 ;CHECK FOR MSB IN 4K BANK
1985 003524 001755 BEQ 4$ ;NOT LAST BIT, BRANCH
1986 003526 040201 BIC R2,R1 ;LAST BIT, CLR THE MASK
1987 003530 062701 020000 ADD #20000,R1 ;MOVE TO NEXT 4K
1988 003534 020105 CMP R1,R5 ;TOP + 2
1989 003536 103731 BLO 2$
1990 003540 000167 177042 END2: JMP TSTSCP
1991
1992 ;TEST 3 THIS TEST CHECKS FOR PROPER
1993 ; BYTE ADDRESSING.

```

NO3

DZMML MACY11 30(1046) 05-AUG-77 12:59 PAGE 1-38

SEQ 0040

DZMMLB.P11 05-AUG-77 12:57

BEGIN OF AREA TESTED (+20) WHEN PROGRAM RELOCATES.

1994											
1995	003544	122737	000003	000404	TST3:	CMPB	#3,2\$TESTN			;CHECK FOR PROPER SEQUENCE NUMBER	
1996	003552	001403				BEQ	1\$			;OK, BRANCH	
1997	003554	004767	010434			JSR	PC,SEQERR			;*ERROR* REPORT ERROR MESSAGE AND HALT AT FATHLT	
(1)	003560	000025				25				;*****ERROR NUMBER 25*****	
(1)											
1998											
1999	003562	010446			1\$:	MOV	R4,-(SP)			;SAVE R4	
2000	003564	013702	000320			MOV	2\$MINMEM,R2			;MINMEM IS LOWEST ADDRESS	
2001	003570	010203				MOV	R2,R3			;PUT IT IN R3 ALSO	
2002	003572	062702	000004			ADD	#4,R2			;POINT R2 TO LAST BYTE +1	
2003	003576	012713	177777			MOV	#-1,(R3)			;WRITE ALL ONES IN	
2004	003602	012763	177777	000002		MOV	#-1,2(R3)			;THE 4 TEST BYTES	
2005	003610	105013			2\$:	CLRB	(R3)			;CLEAR A BYTE	
2006	003612	013701	000320			MOV	2\$MINMEM,R1			;INITIALIZE R1 FOR EACH PASS	
2007	003616	020201			3\$:	CMP	R2,R1			;IF EQUAL, JUST READ LAST BYTE	
2008	003620	001456				BEQ	5\$			;BR IF EQUAL	
2009	003622	020301				CMP	R3,R1			;IS THIS THE BYTE OF ZEROS	
2010	003624	001022				BNE	4\$			;BR IF NOT	
2011	003626	122711	000000			CMPB	#0,(R1)			;IT IS, COMPARE FOR ZEROS	
2012	003632	001415				BEQ	6\$				
2013	003634	012700	000377			MOV	#377,RO			;SET UP LO BYTE	
2014	003640	032701	000001			BIT	#1,R1			;WHICH BYTE ARE WE TESTING ?	
2015	003644	001001				BNE	9\$			;LO BYTE RO IS OK	
2016	003646	000300				SWAB	RO			;HIGH BYTE, MAKE RO RIGHT	
2017	003650	010146			9\$:	MOV	R1,-(SP)			;SAVE R1	
2018	003652	042701	000001			BIC	#1,R1			;FOR PRINTOUT	
2019	003656	004767	007656			JSR	PC,ERROR			;*ERROR* REPORT ERROR MESSAGE	
(1)	003662	000026				26				;*****ERROR NUMBER 26*****	
(1)											
2020	003664	012601				MOV	(SP)+,R1			;RESTORE R1 FOR FURTHER TESTING	
2021	003666	005201			6\$:	INC	R1			;NEXT BYTE	
2022	003670	000752				BR	3\$			;RETURN	
2023	003672	122711	177777		4\$:	CMPB	#-1,(R1)			;ITS NOT THE BYTE OF 0'S, READ 1'S	
2024	003676	001425				BEQ	7\$				
2025	003700	010304				MOV	R3,R4			;GET ADDRESS	
2026	003702	042704	177775			BIC	#177775,R4			;TO FIND BYTE	
2027	003706	030104				BIT	R1,R4			;IS IT SET IN TEST ADDRESS ?	
2028	003710	001003				BNE	12\$			;NO, BRANCH CAUSE IT'S NOT IN SAME WORD	
2029	003712	012700	177777			MOV	#-1,RO			;DATA SHOULD BE ALL ONES	
2030	003716	000406				BR	10\$			;GO REPORT ERROR	
2031	003720	012700	000377		12\$:	MOV	#377,RO			;IT'S IN SAME WORD, BUT WHICH BYTE ?	
2032	003724	032701	000001			BIT	#1,R1			;CHECK FOR LO BYTE	
2033	003730	001401				BEQ	10\$			;NO, LEAVE RO ALONE	
2034	003732	000300				SWAB	RO			;SWAP BYTES	
2035	003734	010146			10\$:	MOV	R1,-(SP)			;SAVE R1	
2036	003736	042701	000001			BIC	#1,R1			;FOR PRINTOUT	
2037	003742	004767	007572			JSR	PC,ERROR			;*ERROR* REPORT ERROR MESSAGE	
(1)	003746	000027				27				;*****ERROR NUMBER 27*****	
(1)											
2038	003750	012601				MOV	(SP)+,R1			;RESTORE R1	
2039	003752	005201			7\$:	INC	R1			;MOVE TO NEXT BYTE	
2040	003754	000720				BR	3\$				
2041	003756	112713	177777		5\$:	MOVB	#-1,(R3)			;RESTORE 1'S TO BYTE JUST TESTED	
2042	003762	005203				INC	R3			;INC TO NEXT BYTE	
2043	003764	020302				CMP	R3,R2			;WAS THAT JUST THE LAST ONE?	

B04

DZMPL MACY11 30(1046) 05-AUG-77 12:59 PAGE 1-39
 DZMPLB.P11 05-AUG-77 12:57 BEGIN OF AREA TESTED (+20) WHEN PROGRAM RELOCATES.

SEQ 0041

```

2044 003766 001310      BNE      2$          ;BR IF NO
2045 003770 012604      MOV      (SP)+,R4      ;RESTORE R4
2046 003772 000167 176610  END3:  JMP      TSTSCP
2047
2048
2049 ;STATUS REGISTER RESPONSE
2050 ;THIS TEST INSURES THAT THE MS11K
2051 ;CSR RESPONDS AND THAT EACH BIT
2052 ;CAN BE WRITTEN UNIQUELY.
2053
2054 003776 122737 000004 000404  TST4:  CMPB      #4,2$STESTN      ;TEST FOR PROPER TEST SEQUENCE
2055 004004 001403      BEQ      1$
2056 004006 004767 010202      JSR      PC,SEQERR      ;*ERROR* REPORT ERROR MESSAGE AND HALT AT FATHLT
(1) 004012 000030      30      ;*****ERROR NUMBER 30*****
(1)
2057
2058 004014 012702 004042      1$:  MOV      #2$,R2
2059 004020 066702 176674      ADD      BASE,R2      ;OFFSET FOR RELOC
2060 004024 010237 000004      MOV      R2,2$4
2061 004030 012701 172136      MOV      #172136,R1      ;FOR ERROR PRINTOUT
2062 004034 005777 176670      TST      2$CSRADR      ;TRY ACCESSING CSR
2063 004040 000403      BR      3$      ;IF WE GOT HERE ALL IS WELL
2064 004042
(1) 004042 004767 010146      2$:  JSR      PC,FATERR      ;*ERROR* REPORT ERROR MESSAGE AND HALT AT FATHLT
(1) 004046 000031      31      ;*****ERROR NUMBER 31*****
(1)
2065
2066 004050 012700 000001      3$:  MOV      #1,R0      ;SET FIRST TEST BIT
2067 004054 012737 000352 000004  MOV      #BUSER,2$4      ;SET UP FOR FUTURE TRAPS
2068 004062 005767 176630      4$:  TST      LDFLG      ;RUNNING IN MEM UNDER TEST?
2069 004066 001004      BNE      5$      ;YES, BRANCH TO 5$
2070 004070 032700 050020      BIT      #50020,R0      ;NO, CHECK FOR UNUSED BITS
2071 004074 001017      BNE      7$
2072 004076 000403      BR      8$
2073 004100 032700 050026      5$:  BIT      #50026,R0
2074 004104 001013      BNE      7$
2075 004106 010077 176616      8$:  MOV      R0,2$CSRADR      ;WRITE THE TEST BIT
2076 004112 017702 176612      MOV      2$CSRADR,R2      ;GET CSR CONTENTS
2077 004116 042702 000020      BIC      #20,R2      ;SINGLE ERRORS WILL CONFUSE US
2078 004122 020002      CMP      R0,R2      ;READ AND COMPARE
2079 004124 001403      BEQ      7$
2080 004126 004767 007406      JSR      PC,ERROR      ;*ERROR* REPORT ERROR MESSAGE
(1) 004132 000032      32      ;*****ERROR NUMBER 32*****
(1)
2081
2082 004134 006300      7$:  ASL      R0      ;SHIFT TO NEXT BIT
2083 004136 103351      BCC      4$      ;CONTINUE TESTING THRU BIT 15
2084 004140 012737 000352 000004  MOV      #BUSER,2$4      ;FOR ANY FUTURE TRAPS
2085 004146 000167 176434  END4:  JMP      TSTSCP
2086
2087 ;SINGLE BIT ERROR TEST
2088 ;THIS TEST FIRST CHECKS FOR UNCORRECTED DATA
2089 ;WITH THE INH ECC BIT SET, CHECKS THAT THE SINGLE
2090 ;ERROR INDICATE OCCURRED AND FINALLY, WITH ECC
2091 ;ENABLED, CHECKS FOR CORRECTED DATA.
2092

```

DZMPL MACY11 30(1046) 05-AUG-77 12:59 PAGE 1-40
 DZMPL8.P11 05-AUG-77 12:57 BEGIN OF AREA TESTED (+20) WHEN PROGRAM RELOCATES.

SEQ 0042

2093	004152	122737	000005	000404	TST5:	CMPB	#5,2#STESTN	;CHECK FOR PROPER TEST SEQUENCE
2094	004160	001403				BEQ	15	
2095	004162	004767	010026			JSR	PC,SE0ERR	;*ERROR* REPORT ERROR MESSAGE AND HALT AT FATHLT
(1)	004166	000033				33		;*****ERROR NUMBER 33*****
(1)								
2096	004170	005077	176534		15:	CLR	2CSRADR	
2097	004174	012767	000001	176462	T5A:	MOV	#1,DATBUF	;INITIAL DATA
2098	004202	005067	176460			CLR	DATBUF+2	;32 BITS WORTH
2099	004206	012767	000001	176460	20\$:	MOV	#1,SBEMSK	;INITIAL ERROR MASK
2100	004214	005067	176456			CLR	SBEMSK+2	;32 BITS WORTH
2101	004210	016767	176440	176442	30\$:	MOV	DATBUF,TSTDAT	
2102	004226	016767	176434	176436		MOV	DATBUF+2,TSTDAT+2	;TO SAVE ORIG DATA
2103	004234	105737	000302			TSTB	2#PASFLG	;COMP DATA ON SECOND PASS ONLY
2104	004240	001404				BEQ	40\$	;BR IF FIRST PASS
2105	004242	005167	176422			COM	TSTDAT	;SECOND PASS, COMP BOTH WORDS
2106	004246	005167	176420			COM	TSTDAT+2	
2107	004252	016702	176412		40\$:	MOV	TSTDAT,R2	
2108	004256	016703	176410			MOV	TSTDAT+2,R3	
2109	004262	012767	002670	006522		MOV	#TSTDAT,SOURCE	;SET UP ADDRESS FOR CHKGEN
2110	004270	004767	006570			JSR	PC,CHKGEN	;GEN CHECKBITS ON TSTDAT
2111	004274	004567	007104			JSR	R5,BITCOM	;BIT COMPLEMENT ROUTINE
2112	004300	002674				.WORD	SBEMSK	;MASK
2113	004302	002670				.WORD	TSTDAT	;DATA
2114	004304	013701	000320		50\$:	MOV	2#MINMEM,R1	;FIRST TEST ADDRESS
2115	004310	016777	176374	176412		MOV	ECCDIS,2CSRADR	;FORCE VALIDATE WITH ECC DISABLED
2116	004316	016721	176346			MOV	TSTDAT,(R1)+	;WRITE FIRST 16 BITS
2117	004322	016700	006466			MOV	CHECK,R0	;GET CHECKBITS FROM CHKGEN
2118	004326	056700	176360			BIS	DIAGA,R0	;SET DIAGNOSTIC CHECK A
2119	004332	010077	176372			MOV	R0,2CSRADR	;LOAD CSR WITH IMAGE IN R0
2120	004336	016711	176330			MOV	TSTDAT+2,(R1)	;WRITE SECOND 16 BITS AND
2121								;CHECK BITS. WE NOW HAVE CHECKBITS
2122								;GENERATED ON DATBUF AND DATA WITH
2123								;ONE BIT IN ERROR (AS PER SBEMSK).
2124	004342	016777	176342	176360		MOV	ECCDIS,2CSRADR	;DISABLE ERROR CORRECTING
2125	004350	016700	176314			MOV	TSTDAT,R0	;SET UP GOOD DATA FOR ERROR REPT.
2126	004354	020041				CMP	R0,-(R1)	;READ THE LOW WORD
2127	004356	001403				BEQ	56\$	;BR IF OK
2128	004360	004767	007154			JSR	PC,ERROR	;*ERROR* REPORT ERROR MESSAGE
(1)	004364	000034				34		;*****ERROR NUMBER 34*****
(1)								
2129	004366	016700	176300		55\$:	MOV	TSTDAT+2,R0	;HIGH WORD
2130	004372	062701	000002			ADD	#2,R1	;POINT R1 TO SECOND 16 BITS
2131	004376	020011				CMP	R0,(R1)	;READ THE HIGH WORD
2132	004400	001403				BEQ	56\$	;BR IF OK
2133	004402	004767	007132			JSR	PC,ERROR	;*ERROR* REPORT ERROR MESSAGE
(1)	004406	000035				35		;*****ERROR NUMBER 35*****
(1)								
2134	004410	005077	176314		56\$:	CLR	2CSRADR	;ENABLE ECC
2135	004414	010200				MOV	R2,R0	;THIS IS ORIGINAL DATA
2136	004416	020041				CMP	R0,-(R1)	;SEE IF ITS BEEN CORRECTED
2137	004420	001403				BEQ	57\$	;IT SHOULD HAVE BEEN
2138	004422	004767	007112			JSR	PC,ERROR	;*ERROR* REPORT ERROR MESSAGE
(1)	004426	000036				36		;*****ERROR NUMBER 36*****
(1)								
2139	004430	032777	000020	176272	57\$:	BIT	#20,2CSRADR	;CHECK IF SBE INDICATE SET
2140	004436	001011				BNE	58\$	;BR IF IT IS SET

DZMLB.P11 05-AUG-77 12:57

BEGIN OF AREA TESTED (+20) WHEN PROGRAM RELOCATES.

Address	Hex	Dec	Label	Op	Op2	Op3	Op4	Op5	Op6	Op7	Op8	Op9	Op10	Op11	Op12	Op13	Op14	Op15	Op16	Op17	Op18	Op19	Op20	Op21	Op22	Op23	Op24	Op25	Op26	Op27	Op28	Op29	Op30	Op31	Op32	Op33	Op34	Op35	Op36	Op37	Op38	Op39	Op40	Op41	Op42	Op43	Op44	Op45	Op46	Op47	Op48	Op49	Op50	Op51	Op52	Op53	Op54	Op55	Op56	Op57	Op58	Op59	Op60	Op61	Op62	Op63	Op64	Op65	Op66	Op67	Op68	Op69	Op70	Op71	Op72	Op73	Op74	Op75	Op76	Op77	Op78	Op79	Op80	Op81	Op82	Op83	Op84	Op85	Op86	Op87	Op88	Op89	Op90	Op91	Op92	Op93	Op94	Op95	Op96	Op97	Op98	Op99	Op100	Op101	Op102	Op103	Op104	Op105	Op106	Op107	Op108	Op109	Op110	Op111	Op112	Op113	Op114	Op115	Op116	Op117	Op118	Op119	Op120	Op121	Op122	Op123	Op124	Op125	Op126	Op127	Op128	Op129	Op130	Op131	Op132	Op133	Op134	Op135	Op136	Op137	Op138	Op139	Op140	Op141	Op142	Op143	Op144	Op145	Op146	Op147	Op148	Op149	Op150	Op151	Op152	Op153	Op154	Op155	Op156	Op157	Op158	Op159	Op160	Op161	Op162	Op163	Op164	Op165	Op166	Op167	Op168	Op169	Op170	Op171	Op172	Op173	Op174	Op175	Op176	Op177	Op178	Op179	Op180	Op181	Op182	Op183	Op184	Op185	Op186	Op187	Op188	Op189	Op190	Op191	Op192	Op193	Op194	Op195	Op196	Op197	Op198	Op199	Op200	Op201	Op202	Op203	Op204	Op205	Op206	Op207	Op208	Op209	Op210	Op211	Op212	Op213	Op214	Op215	Op216	Op217	Op218	Op219	Op220	Op221	Op222	Op223	Op224	Op225	Op226	Op227	Op228	Op229	Op230	Op231	Op232	Op233	Op234	Op235	Op236	Op237	Op238	Op239	Op240	Op241	Op242	Op243	Op244	Op245	Op246	Op247	Op248	Op249	Op250	Op251	Op252	Op253	Op254	Op255	Op256	Op257	Op258	Op259	Op260	Op261	Op262	Op263	Op264	Op265	Op266	Op267	Op268	Op269	Op270	Op271	Op272	Op273	Op274	Op275	Op276	Op277	Op278	Op279	Op280	Op281	Op282	Op283	Op284	Op285	Op286	Op287	Op288	Op289	Op290	Op291	Op292	Op293	Op294	Op295	Op296	Op297	Op298	Op299	Op300	Op301	Op302	Op303	Op304	Op305	Op306	Op307	Op308	Op309	Op310	Op311	Op312	Op313	Op314	Op315	Op316	Op317	Op318	Op319	Op320	Op321	Op322	Op323	Op324	Op325	Op326	Op327	Op328	Op329	Op330	Op331	Op332	Op333	Op334	Op335	Op336	Op337	Op338	Op339	Op340	Op341	Op342	Op343	Op344	Op345	Op346	Op347	Op348	Op349	Op350	Op351	Op352	Op353	Op354	Op355	Op356	Op357	Op358	Op359	Op360	Op361	Op362	Op363	Op364	Op365	Op366	Op367	Op368	Op369	Op370	Op371	Op372	Op373	Op374	Op375	Op376	Op377	Op378	Op379	Op380	Op381	Op382	Op383	Op384	Op385	Op386	Op387	Op388	Op389	Op390	Op391	Op392	Op393	Op394	Op395	Op396	Op397	Op398	Op399	Op400	Op401	Op402	Op403	Op404	Op405	Op406	Op407	Op408	Op409	Op410	Op411	Op412	Op413	Op414	Op415	Op416	Op41
---------	-----	-----	-------	----	-----	-----	-----	-----	-----	-----	-----	-----	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	------

E04

DZMML MACY11 30(1046) 05-AUG-77 12:59 PAGE 1-42
DZMMLB.P11 05-AUG-77 12:57

BEGIN OF AREA TESTED (+20) WHEN PROGRAM RELOCATES.

SEQ 0044

2189	004672	062701	000004	35:	ADD	#4,R1	:MOVE TO NEXT WORD
2190	004676	020100			CMP	R1,R0	:END OF TEST 6 YET ?
2191	004700	103764			BLO	25	:NO, BRANCH
2192	004702	004467	006420	45:	JSR	R4,SAVOT4	:SAVE REGS R0 TO R4
2193	004706	005077	176016		CLR	ACSRADR	:CLEAR CSR
2194	004712	012767	000001	175744	MOV	#1,DATBUF	:INITIAL DATA
2195	004720	005067	175742		CLR	DATBUF+2	:32 BITS WORTH
2196	004724	012767	000001	175742	T6A: MOV	#1,SBEMSK	:INITIAL ERROR MASK
2197	004732	005067	175740		CLR	SBEMSK+2	:32 BITS WORTH
2198	004736	016767	175722	175724	T6B: MOV	DATBUF,TSTDAT	:SAVE ORIGINAL DATA
2199	004744	016767	175716	175720	MOV	DATBUF+2,TSTDAT+2	:BOTH WORDS
2200	004752	016767	175706	175724	MOV	DATBUF,DATSAV	:IN CASE PROG HAS SBE
2201	004760	016767	175702	175720	MOV	DATBUF+2,DATSAV+2	:DITTO
2202	004766	012767	002670	006016	MOV	#TSTDAT,SOURCE	:NEED ADDRESS FOR CHKGEN
2203	004774	004767	006064		JSR	PC,CHKGEN	:GENERATE CHECK BITS
2204	005000	004567	006400		JSR	R5,BITCOM	:BIT COM ROUTINE
2205	005004	002674			.WORD	SBEMSK	:MASK FOR COMPLEMENTING
2206	005006	002670			.WORD	TSTDAT	:WORD TO BE COMPLEMENTED
2207	005010	013701	000320	305:	MOV	#MINMEM,R1	:FIRST TEST ADDRESS
2208	005014	010104			MOV	R1,R4	:PUT IT IN R4 ALSO
2209	005016	016777	175666	175704	MOV	ECCDIS,ACSRADR	:INHIBIT ECC
2210	005024	016724	175640		MOV	TSTDAT,(R4)+	:WRITE 16 BITS
2211	005030	016703	005760		MOV	CHECK,R3	:GET CHECKBITS
2212	005034	056703	175652		BIS	DIAGA,R3	:SET DIAGNOSTIC A BIT
2213	005040	010377	175664		MOV	R3,ACSRADR	:LOAD CSR WITH NEW IMAGE
2214	005044	016714	175622		MOV	TSTDAT+2,(R4)	:WRITE HIGH WORD+CHECKBITS
2215	005050	005077	175654		CLR	ACSRADR	:IT'S DANGEROUS IF WE DON'T
2216	005054	012702	002674		MOV	#SBEMSK,R2	:ADDRESS OF ERROR MASK
2217	005060	066702	175634		ADD	BASE,R2	
2218	005064	162704	000002		SUB	#2,R4	:ADJUST ADDRESS
2219	005070	112714	177777	405:	MOVB	#-1,(R4)	:WRITE A BYTE OF 1'S
2220	005074	132712	177777		BITB	#-1,(R2)	
2221	005100	001476			BEQ	605	
2222	005102	005767	000432		TST	T6FLG	:SINGLE ERRORS IN TST6 AREA ?
2223	005106	001016			BNE	455	:BR IF YES
2224	005110	005077	175614		CLR	ACSRADR	:CLEAR CSR
2225	005114	105714			TSTB	(R4)	:READ
2226	005116	032777	000020	175604	BIT	#20,ACSRADR	:SINGLE ERROR INDICATE SET ?
2227	005124	001453			BEQ	505	
2228	005126	016701	175576		MOV	CSRADR,R1	:FOR PRINTOUT
2229	005132	005000			CLR	R0	
2230	005134	004767	006400		JSR	PC,ERROR	:*ERROR* REPORT ERROR MESSAGE
(1)	005140	000043			43		:*****ERROR NUMBER 43*****
(1)							
2231	005142	000444		455:	BR	505	:CONTINUE
2232	005144	010446			MOV	R4,-(SP)	:NEED A SCRATCH LOC
2233	005146	163716	000320		SUB	#MINMEM,(SP)	:TO FIND BYTE OFFSET
2234	005152	012703	002704		MOV	#DATSAV,R3	:GET ADDRESS
2235	005156	066703	175536		ADD	BASE,R3	:OFFSET FOR RELOC
2236	005162	062603			ADD	(SP)+,R3	:ADD BYTE COUNT
2237	005164	112713	177777		MOVB	#-1,(R3)	:WRITE BYTE OF 1'S LIKE TEST LOCATION
2238	005170	016777	175516	175532	MOV	DIAGA,ACSRADR	:DIAG BIT
2239	005176	105714			TSTB	(R4)	:READ THE BYTE TO GET THE CHECKBITS
2240							
2241	005200	017703	175524		MOV	ACSRADR,R3	:GET CHECKBITS
2242	005204	042703	170037		BIC	#170037,R3	:CLEAR OTHER BITS

F04

DZMML MACY11 30(1046) 05-AUG-77 12:59 PAGE 1-43

SEQ 0045

DZMMLB.P11 05-AUG-77 12:57

BEGIN OF AREA TESTED (+20) WHEN PROGRAM RELOCATES.

2243	005210	005077	175514		CLR	ACSRADR	
2244	005214	012767	002704	005570	MOV	#DATSAV, SOURCE	; FOR CHKGEN
2245	005222	004767	005636		JSR	PC, CHKGEN	; GENERATE CHECKBITS
2246	005226	020367	005562		CMP	R3, CHECK	; COMPARE ACTUAL VS. EXPECTED
2247	005232	001410			BEQ	50\$	; BR IF OK
2248	005234	010300			MOV	R3, R0	; GOOD DATA
2249	005236	012701	013014		MOV	#CHECK, R1	; ADDRESS OF CHECKBITS
2250	005242	066701	175452		ADD	BASE, R1	; PLUS OFFSET
2251	005246	004767	006266		JSR	PC, ERROR	; ERROR ROUTINE
2252	005252	000043			43		
2253	005254	122714	177777	50\$:	CMP9	#-1, (R4)	; CHECK DATA
2254	005260	001461			BEQ	70\$	; BR IF OK
2255	005262	010401			MOV	R4, R1	; GET R4
2256	005264	042701	000001		BIC	#1, R1	; TO MAKE EVEN ADDR FOR PRINTOUT
2257	005270	004767	006244		JSR	PC, ERROR	; *ERROR* REPORT ERROR MESSAGE
(1)	005274	000044			44		; *****ERROR NUMBER 44*****
(1)							
2258	005276	005767	000236	60\$:	TST	T6FLG	; SINGLE ERRORS IN TST6 AREA ?
2259	005302	001011			BNE	65\$	; BR IF YES
2260	005304	105714			TSTB	(R4)	; READ THE BYTE
2261	005306	032777	000020	175414	BIT	#20, ACSRADR	; SBE ERROR BIT SET ?
2262	005314	001357			BNE	50\$	; SHOULD BE SET, BR IF OK
2263	005316	004767	006216		JSR	PC, ERROR	; *ERROR* REPORT ERROR MESSAGE
(1)	005322	000045			45		; *****ERROR NUMBER 45*****
(1)							
2264	005324	000437			BR	70\$	; CONTINUE
2265	005326	010446		65\$:	MOV	R4, -(SP)	; SAVE R4
2266	005330	163716	000320		SUB	#MINMEM, (SP)	; FOR BYTE OFFSET
2267	005334	012703	002704		MOV	#DATSAV, R3	; ADDRESS OF DATA
2268	005340	066703	175354		ADD	BASE, R3	; FOR RELOC
2269	005344	062603			ADD	(SP)+, R3	; BYTE OFFSET
2270	005346	112713	177777		MOVB	#-1, (R3)	; WRITE A BYTE OF 1'S
2271	005352	016777	175334	175350	MOV	DIAGA, ACSRADR	; DIAG BIT
2272	005360	105714			TSTB	(R4)	; READ THE BYTE
2273	005362	017746	175342		MOV	ACSRADR, -(SP)	; GETCONTENTS
2274	005366	005077	175336		CLR	ACSRADR	; CLEAR THE CSR
2275	005372	042716	170037		BIC	#170037, (SP)	; ONLY WANT CHECKBITS
2276	005376	012767	002704	005406	MOV	#DATSAV, SOURCE	; FOR CHKGEN
2277	005404	004767	005454		JSR	PC, CHKGEN	; GENERATE CHECKBITS ON DATA
2278	005410	022667	005400		CMP	(SP)+, CHECK	; COMPARE ACTUAL VS. EXPECTED
2279	005414	001403			BEQ	70\$	; BR IF OK
2280	005416	004767	006116		JSR	PC, ERROR	; ERROR ROUTINE
2281	005422	000045			45		
2282	005424	132712	177777	70\$:	BITB	#-1, (R2)	; CHECK FOR LAST BYTE
2283	005430	001012			BNE	80\$	
2284	005432	005202			INC	R2	
2285	005434	005204			INC	R4	; MOVE TO NEXT BYTE
2286	005436	013701	000320		MOV	#MINMEM, R1	; FIRST TEST ADDRESS
2287	005442	032704	000002		BIT	#2, R4	; TEST FOR LOWER WORD
2288	005446	001610			BEQ	40\$	; BR IF IT'S LOW 16 BITS
2289	005450	062701	000002		ADD	#2, R1	; ADJUST POINTER FOR ERROR REPT.
2290	005454	000605			BR	40\$	
2291	005456	005737	000406	80\$:	TST	#SPASS	; FIRST PASS ?
2292	005462	001420			BEQ	100\$	; BR IF YES
2293	005464	005767	175206		TST	SBEMSK+2	; LAST ERROR BIT ?
2294	005470	100405			BMI	90\$	; MINUS MEANS BIT 31

DZMPL MACY11 30(1046) 05-AUG-77 12:59 PAGE 1-44
 DZMPLB.P11 05-AUG-77 12:57

BEGIN OF AREA TESTED (+20) WHEN PROGRAM RELOCATES.

SEQ 0046

2295	005472	004567	005656		JSR	R5,DASHL	;DOUBLE ARITHMETIC SHIFT LEFT
2296	005476	002674			.WORD	SBEMSK	;ON THE ERROR MASK
2297	005500	000167	177232		JMP	T6B	
2298	005504	005767	175156	90\$:	TST	DATBUF+2	;LAST DATA BIT ?
2299	005510	100405			BMI	100\$	;MINUS = BIT 31
2300	005512	004567	005636		JSR	R5,DASHL	
2301	005516	002664			.WORD	DATBUF	;SHIFT IT LEFT
2302	005520	000167	177200		JMP	T6A	
2303	005524	004767	005610	100\$:	JSR	PC,RSTOT4	;RESTORE REGS R0 TO R4
2304	005530	005067	000004		CLR	T6FLG	;FOR NEXT TIME
2305	005534	000167	175046	END6:	JMP	TSTSCP	
2306	005540	000000		T6FLG:	.WORD	0	
2307							
2308							;DOUBLE BIT ERROR FORCE TO CHECK DOUBLE ERROR
2309							;STATUS BIT AND TRAP THROUGH LOCATION 114.
2310							
2311	005542	122737	000007	000404	TST7:	CMPB	07,00\$TESTN
2312	005550	001403			BEQ	1\$	
2313	005552	004767	006436		JSR	PC,SEGERR	;*ERROR* REPORT ERROR MESSAGE AND HALT AT FATHLT
(1)	005556	000046			46		;*****ERROR NUMBER 46*****
(1)							
2314	005560	005077	175144	1\$:	CLR	0CSRADR	;CLEAR CSR
2315	005564	013701	000320		MOV	0\$MINMEM,R1	
2316	005570	004467	005532		JSR	R4,SAVOT4	;SAVE REGS R0 TO R4
2317	005574	005067	175064	10\$:	CLR	DATBUF	;MAKE INITIAL DATA
2318	005600	005067	175062		CLR	DATBUF+2	;ALL ZEROS
2319	005604	012767	000001	175062	20\$:	01,SBEMSK	;INITIAL SINGLE ERROR MASK
2320	005612	005067	175060		CLR	SBEMSK+2	;SECOND WORD
2321	005616	012767	000001	175054	30\$:	01,DBEMSK	;INITIAL DOUBLE ERROR MASK
2322	005624	005067	175052		CLR	DBEMSK+2	;32 BITS HERE ALSO
2323	005630	016767	175030	175032	35\$:	MOV	DATBUF,TSTDAT
2324	005636	016767	175024	175026		MOV	DATBUF+2,TSTDAT+2
2325	005644	105737	000302		TSTB	0\$PASFLG	;NO COMPLEMENTING FIRST PASS
2326	005650	001404			BEQ	40\$	
2327	005652	005167	175012		COM	TSTDAT	;COMP FIRST WORD
2328	005656	005167	175010		COM	TSTDAT+2	;SECOND WORD
2329	005662	005077	175042	175004	40\$:	CLR	0CSRADR
2330	005666	026767	175002		CMP	SBEMSK,DBEMSK	;CAN'T HAVE THE SAME ERROR BIT SET
2331	005674	001004			BNE	45\$	;IN BOTH MASKS
2332	005676	026767	174774	174776	CMP	SBEMSK+2,DBEMSK+2	;COULD BE EQUAL IN SECOND WORD
2333	005704	001502			BEQ	65\$	;GO MAKE THEM NOT EQUAL
2334	005706	012767	002670	005076	45\$:	MOV	0TSTDAT,SOURCE
2335	005714	004767	005144		JSR	PC,CHKGEN	;GO GENERATE CHECK BITS
2336	005720	004567	005460		JSR	R5,BITCOM	;FORCE SINGLE ERROR
2337	005724	002674			.WORD	SBEMSK	;AS PER SBEMSK
2338	005726	002670			.WORD	TSTDAT	;ON DATA IN TSTDAT
2339	005730	004567	005450		JSR	R5,BITCOM	;FORCING DOUBLE ERROR
2340	005734	002700			.WORD	DBEMSK	;THIS MASK INDICATES SECOND BIT IN ERROR
2341	005736	002670			.WORD	TSTDAT	;SAME DATA WORD
2342	005740	016704	005050		MOV	CHECK,R4	;GET CHECKBITS
2343	005744	056704	174742		BIS	DIAGA,R4	;SET DIAGNOSTIC A BIT
2344	005750	016777	174734	174752	MOV	ECCDIS,0CSRADR	;INHIBIT ECC
2345	005756	016721	174706		MOV	TSTDAT,(R1)+	;WRITE 16 BITS
2346	005762	010477	174742		MOV	R4,0CSRADR	;LOAD CSR
2347	005766	016711	174700		MOV	TSTDAT+2,(R1)	;WRITE HIGH WORD
2348	005772	005077	174732		CLR	0CSRADR	;CLEAR CSR AGAIN

H04

DZMPL MACY11 30(1046) 05-AUG-77 12:59 PAGE 1-45

SEQ 0047

DZMPLB.P11 05-AUG-77 12:57

BEGIN OF AREA TESTED (+20) WHEN PROGRAM RELOCATES.

2349	005776	162701	000002		SUB	#2,R1	:ADJUST TEST ADDRESS
2350	006002	005711			TST	(R1)	:READ THE LOCATION
2351	006004	032777	100000	174716	BIT	#100000,ACSRADR	:LOOK FOR DBE STATUS BIT
2352	006012	001011			BNE	50\$	:IT SHOULD BE SET
2353	006014	010146			MOV	R1, -(SP)	:SAVE R1
2354	006016	016701	174706		MOV	CSRADR, R1	:GET CSR ADDRESS
2355	006022	012700	100000		MOV	#100000, R0	:SHOW BIT 15 SET
2356	006026	004767	005506		JSR	PC, ERROR	:*ERROR* REPORT ERROR MESSAGE
(1)	006032	000047			47		:*****ERROR NUMBER 47*****
(1)							
2357	006034	012601			MOV	(SP)+, R1	:RESTORE R1
2358	006036	012702	006100		MOV	#60\$, R2	:SET UP FOR BUS PBL
2359	006042	066702	174652		ADD	BASE, R2	
2360	006046	010237	000114		MOV	R2, #114	
2361	006052	052777	000001	174650	BIS	#1, ACSRADR	:SET DOUBLE ERROR INDICATE
2362							
2363	006060	005711			TST	(R1)	:READ THE TEST LOCATION AGAIN
2364	006062	005077	174642		CLR	ACSRADR	:CLEAR STATUS REG
2365	006066	005000			CLR	R0	:CLEAR R0
2366	006070	004767	005444		JSR	PC, ERROR	:*ERROR* REPORT ERROR MESSAGE
(1)	006074	000050			50		:*****ERROR NUMBER 50*****
(1)							
2367	006076	000402			BR	61\$	:IF ERROR, DON'T ADJUST STACK
2368	006100	062706	000004		ADD	#4, SP	:ADJUST STACK FROM TRAP
2369	006104	005737	000406		61\$: TST	2*\$PASS	:FIRST PASS ?
2370	006110	001424			BEQ	80\$	:BR IF YES
2371	006112	005767	174564		65\$: TST	DBEMSK+2	:CHECK MASK FOR LAST BIT
2372	006116	100404			BMI	70\$	:MINUS = BIT31
2373	006120	004567	005230		JSR	R5, DASHL	:DOUBLE SHIFT
2374	006124	002700			.WORD	DBEMSK	
2375	006126	000640			BR	35\$	
2376	006130	005767	174542		70\$: TST	SBEMSK+2	:CHECK SINGLE ERROR MASK TOO
2377	006134	100404			BMI	75\$	:BR IF DONE
2378	006136	004567	005212		JSR	R5, DASHL	:SHIFT
2379	006142	002674			.WORD	SBEMSK	
2380	006144	000624			BR	30\$	
2381	006146	105737	000302		75\$: TSTB	2\$PASFLG	:FIRST PASS
2382	006152	001003			BNE	80\$	:NON ZERO MEANS WE'RE DONE
2383	006154	105237	000302		INCB	2\$PASFLG	:FIRST PASS, NOT DONE
2384	006160	000605			BR	10\$	:KEEP GOING
2385	006162	005737	000406		80\$: TST	2*\$PASS	:CHECKING PASS AGAIN
2386	006166	001406			BEQ	82\$	:CHECK 1K ADDR FUNCTION ON
2387							:FIRST PASS ONLY
2388	006170	016777	174514	174532	MOV	ECCDIS, ACSRADR	:OTHERWISE DISABLE ECC
2389	006176	005021			CLR	(R1)+	:CLEAR THE DBE'S
2390	006200	005011			CLR	(R1)	
2391	006202	000545			BR	90\$	:AND EXIT
2392	006204	005077	174520		82\$: CLR	ACSRADR	:CLEAR CSR
2393	006210	005002			CLR	R2	:OUR COUNTER
2394	006212	005711			TST	(R1)	:READ THE LOCATION
2395	006214	017703	174510		MOV	ACSRADR, R3	:MOV CHECKBITS INTO R3
2396	006220	005077	174504		CLR	ACSRADR	:CLEAR CSR AGAIN
2397	006224	042703	170037		BIC	#170037, R3	:ONLY WANT CHECKBITS
2398	006230	006303			ASL	R3	:POSITION THEM
2399	006232	006303			ASL	R3	:INTO LOW BYTE
2400	006234	006303			ASL	R3	

DZMML MACY11 30(1046) 05-AUG-77 12:59 PAGE 1-46

SEQ 0048

DZMMLB.P11 05-AUG-77 12:57

BEGIN OF AREA TESTED (+20) WHEN PROGRAM RELOCATES.

2401	006236	000303		SWAB	R3	; HERE
2402	006240	004767	007142	JSR	PC, GET1K	; GET THE 1K BANK
2403	006244	020300		CMP	R3, R0	; BETTER BE THE SAME
2404	006246	001413		BEQ	85\$	; BR IF OK
2405	006250	010146		MOV	R1, -(SP)	; SAVE R1 FOR NOW
2406	006252	016777	174432 174450	MOV	ECCDIS, @CSRADR	; DISABLE ECC
2407	006260	010311		MOV	R3, (R1)	; PUT BAD DATA TO BE PRINTED INTO R1
2408	006262	005077	174442	CLR	@CSRADR	; RESTORE CSR
2409	006266	004767	005246	JSR	PC, ERROR	; *ERROR* REPORT ERROR MESSAGE
(1)	006272	000051		51		; *****ERROR NUMBER 51*****
(1)						
2410	006274	012601		MOV	(SP)+, R1	; RESTORE R1
2411	006276	016777	174406 174424 85\$:	MOV	ECCDIS, @CSRADR	; DISABLE ERROR CORRECTING
2412	006304	005021		CLR	(R1)+	; TO CLEAR ANY DOUBLE ERRORS
2413	006306	005011		CLR	(R1)	
2414	006310	005077	174414	CLR	@CSRADR	; RESTORE CSR
2415	006314	042701	003777	BIC	#3777, R1	; STRIP AWAY TO LAST 1K
2416	006320	062701	004000	ADD	#4000, R1	; MOVE UP TO NEXT 1K
2417	006324	020105		CMP	R1, R5	; OVER TOP YET ?
2418	006326	103073		BHIS	90\$	; BR IF YES
2419	006330	004767	007142 88\$:	JSR	PC, TSTADD	; CHECK FOR ERROR FREE LOC
2420	006334	005703		TST	R3	; ZERO MEANS ERROR FREE
2421	006336	001456		BEQ	86\$	; BR IF OK
2422	006340	005202		INC	R2	; COUNT ONE WORD CHECKED
2423	006342	020227	001000	CMP	R2, #1000	; TESTED WHOLE 1K SLICE YET ?
2424	006346	103447		BLO	87\$	; BR IF NOT
2425	006350	004767	006056	JSR	PC, TPCRLF	; PRINT ROUTINE
2426	006354	051524	033534 047055	.ASCIZ	/TST7-NO ERROR FREE LOC IN 1K SLICE AT /	
	006362	020117	051105 047522			
	006370	020122	051106 042505			
	006376	046040	041517 044440			
	006404	020116	045461 051440			
	006412	044514	042503 040440			
	006420	020124	000			
2427		006424		.EVEN		
2428	006424	042701	003776	BIC	#3776, R1	; WANT STARTING ADDRESS OF 1K
2429	006430	012703	006534	MOV	#SAV7+2, R3	; SCRATCH LOC FOR ADDRESS
2430	006434	066703	174260	ADD	BASE, R3	; FOR OFFSET
2431	006440	010146		MOV	R1, -(SP)	; SAVE R1
2432	006442	004767	006454	JSR	PC, PUTADR	; PUT ADDRESS IN SAV7 & SAV7+2
2433	006446	105277	171636	INCB	@TYPCNT	; PRINT ONE WORD
2434	006452	005741		TST	-(R1)	; ADJUST FOR PRINTOUT
2435	006454	004767	006122	JSR	PC, TYPOCT	; PRINT IT HERE
2436	006460	005002		CLR	R2	; RESTORE R2 FOR NEXT SLICE
2437	006462	012601		MOV	(SP)+, R1	; RESTORE R1
2438	006464	000704		BR	85\$	; GO LOOK FOR GOOD LOC
2439	006466	062701	000004 87\$:	ADD	#4, R1	; NEXT WORD
2440	006472	000716		BR	88\$	; GO CHECK IT
2441	006474	016721	174170 86\$:	MOV	TSTDAT, (R1)+	; WRITE FIRST WORD
2442	006500	010477	174224	MOV	R4, @CSRADR	; CSR IMAGE WITH BAD CHECKBITS
2443	006504	016711	174162	MOV	TSTDAT+2, (R1)	; WRITE 2ND WORD + CHECKBITS
2444	006510	162701	000002	SUB	#2, R1	; ADJUST R1
2445	006514	000633		BR	82\$	; CONTINUE TESTING
2446	006516	005077	174206 90\$:	CLR	@CSRADR	; CLEAR CSR TO NORMAL
2447	006522	004767	004612	JSR	PC, RSTOT4	; RESTORE THE REGS 0 TO 4
2448	006526	000167	174054	END7: JMP	TSTSCP	

J04

DZMML MACY11 30(1046) 05-AUG-77 12:59 PAGE 1-47

SEQ 0049

DZMMLB.P11 05-AUG-77 12:57 BEGIN OF AREA TESTED (+20) WHEN PROGRAM RELOCATES.

```

2449 006532 000000 000000 SAV7: .WORD 0,0
2450
2451 ;THIS TEST CHECKS THE WRITE INHIBIT ON DOUBLE
2452 ;BIT ERRORS DURING A DATIP OPERATION BY USE
2453 ;OF AN 'INC' INSTRUCTION.
2454
2455 006536 122737 000010 000404 TST10: CMPB #10,2*STESTN
2456 006544 001403 BEQ 1$
2457 006546 004767 005442 JSR PC,SEQERR ;*ERROR* REPORT ERROR MESSAGE AND HALT AT FATHLT
(1) 006552 000052 52 ;*****ERROR NUMBER 52*****
(1)
2458 006554 004467 004546 1$: JSR R4,SAVOT4 ;SAVE REGS R0 TO R4
2459 006560 005067 174100 10$: CLR DATBUF ;INITIAL DATA
2460 006564 005067 174076 CLR DATBUF+2 ;2 WORDS WORTH
2461 006570 012767 000001 174076 20$: MOV #1,SBEMSK ;INITIAL ERROR MASK
2462 006576 005067 174074 CLR SBEMSK+2
2463 006602 012767 000001 174070 30$: MOV #1,DBEMSK ;DOUBLE ERROR MASK
2464 006610 005067 174066 CLR DBEMSK+2 ;2 WORDS
2465 006614 016767 174044 174046 35$: MOV DATBUF,TSTDAT ;PRESERVE ORIG DATA
2466 006622 016767 174040 174042 MOV DATBUF+2,TSTDAT+2
2467 006630 105737 000302 TSTB 2*PASFLG ;SECOND PASS YET ?
2468 006634 001404 BEQ 40$ ;BR IF NO
2469 006636 005167 174026 COM TSTDAT ;COMPL DATA ON SECOND PASS
2470 006642 005167 174024 COM TSTDAT+2 ;
2471 006646 005077 174056 40$: CLR 2*CSRADR ;
2472 006652 026767 174022 174014 CMP DBEMSK,SBEMSK ;CHECK FOR SAME MASKS
2473 006660 001004 BNE 45$ ;BR IF OK
2474 006662 026767 174014 174006 CMP DBEMSK+2,SBEMSK+2
2475 006670 001476 BEQ 70$ ;BR IF THEY'RE EQUAL
2476 006672 012767 002670 004112 45$: MOV 2*TSTDAT,SOURCE ;SET UP ADDRESS FOR CHKGEN
2477 006700 004767 004160 JSR PC,CHKGEN ;GENERATE CHECK BITS
2478 006704 004567 004474 JSR R5,BITCOM ;COMPL ROUTINE
2479 006710 002674 .WORD SBEMSK ;MASK
2480 006712 002670 .WORD TSTDAT ;DATA
2481 006714 004567 004464 JSR R5,BITCOM ;TO MAKE DOUBLE ERROR
2482 006720 002700 .WORD DBEMSK
2483 006722 002670 .WORD TSTDAT
2484 006724 016702 004064 MOV CHECK,R2 ;GET CHECKBITS
2485 006730 056702 173756 BIS DIAG,R2 ;SET DIAGNOSTIC A BIT
2486 006734 013701 000320 50$: MOV 2*MINMEM,R1 ;TEST ADDRESS
2487 006740 016777 173744 173762 MOV ECCDIS,2*CSRADR ;INHIBIT ECC
2488 006746 016721 173716 MOV TSTDAT,(R1)+ ;WRITE FIRST 16 BITS
2489 006752 010277 173752 MOV R2,2*CSRADR ;LOAD CSR WITH IMAGE FROM R2
2490 006756 016711 173710 MOV TSTDAT+2,(R1) ;SECOND 16 BITS+CHECKBITS
2491 006762 105067 004064 CLRB UPPFLG ;INDICATE LOWER WORD
2492 006766 013703 000320 MOV 2*MINMEM,R3 ;TEST ADDRESS
2493 006772 005077 173732 55$: CLR 2*CSRADR ;CLEAR IT
2494 006776 005223 INC (R3)+ ;INC INSTRUCTION
2495 007000 026741 173664 CMP TSTDAT,-(R1) ;CHECK FOR UNCHANGED DATA
2496 007004 001405 BEQ 60$ ;SHOULD BE UNCHANGED
2497 007006 016700 173656 MOV TSTDAT,R0 ;FOR PRINTOUT
2498 007012 004767 004522 JSR PC,ERROR ;*ERROR* REPORT ERROR MESSAGE
(1) 007016 000053 53 ;*****ERROR NUMBER 53*****
(1)
2499 007020 062701 000002 60$: ADD #2,R1 ;POINT TO UPPER WORD
2500 007024 026711 173642 CMP TSTDAT+2,(R1) ;READ IT

```

K04

DZMML MACY11 30(1046) 05-AUG-77 12:59 PAGE 1-48

SEQ 0050

DZMMLB.P11 05-AUG-77 12:57

BEGIN OF AREA TESTED (+20) WHEN PROGRAM RELOCATES.

2501	007030	001434			BEQ	80\$		;BR IF UNCHANGED
2502	007032	016700	173634		MOV	TSTDAT+2,R0		
2503	007036	004767	004476		JSR	PC,ERROR		;*ERROR* REPORT ERROR MESSAGE
(1)	007042	000054			54			;*****ERROR NUMBER 54*****
(1)								
2504	007044	105767	004002	65\$:	TSTB	UPPFLG		;LOWER WORD
2505	007050	001003			BNE	66\$		;BR IF NO
2506	007052	105267	003774		INCB	UPPFLG		
2507	007056	000745			BR	55\$		
2508	007060	005737	000406	66\$:	TST	2\$SPASS		;CHECK PASS #
2509	007064	001424			BEQ	90\$		;BR IF FIRST
2510	007066	005767	173610	70\$:	TST	DBEMSK+2		;LAST BIT IN MASK ?
2511	007072	100404			BMI	75\$		;BR IF BIT 31
2512	007074	004567	004254		JSR	R5,DASHL		;SHIFT ROUTINE
2513	007100	002700			.WORD	DBEMSK		
2514	007102	000644			BR	35\$		
2515	007104	005767	173566	75\$:	TST	SBEMSK+2		;LAST BIT IN SINGLE ERROR MASK ?
2516	007110	100404			BMI	80\$		;BR IF YES
2517	007112	004567	004236		JSR	R5,DASHL		;SHIFT
2518	007116	002674			.WORD	SBEMSK		
2519								
2520	007120	000630			BR	30\$		
2521	007122	105737	000302	80\$:	TSTB	2\$PASFLG		;WHICH PASS
2522	007126	001003			BNE	90\$		;BR IF WE'RE DONE
2523	007130	105237	000302		INCB	2\$PASFLG		;INDICATE SECOND PASS COMING
2524	007134	000611			BR	10\$		;GO DO IT!
2525	007136	016777	173546	173564	90\$:	MOV	ECCDIS,2CSRADR	;INHIBIT ECC TO CLEAR DBE'S
2526	007144	005011			CLR	(R1)		
2527	007146	005041			CLR	-(R1)		
2528	007150	005077	173554		CLR	2CSRADR		;RESTORE CSR TO NORMAL
2529	007154	004767	004160		JSR	PC,RSTOT4		;RESTORE THE REGS
2530	007160	000167	173422	END10:	JMP	TSTSCP		
2531								
2532								;CHECK FOR WRITE INHIBIT DURING A WRITE BYTE.
2533								;CHECKS FOR UNCORRECTED DATA.
2534								
2535	007164	122737	000011	000404	TST11:	CMPB	111,2\$STESTN	
2536	007172	001403			BEQ	1\$		
2537	007174	004767	005014		JSR	PC,SEQERR		;*ERROR* REPORT ERROR MESSAGE AND HALT AT FATHLT
(1)	007200	000055			55			;*****ERROR NUMBER 55*****
(1)								
2538	007202	004467	004120	1\$:	JSR	R4,SAVOT4		;SAVE REGS R0 TO R4
2539	007206	005067	173452	10\$:	CLR	DATBUF		;INITIAL DATA
2540	007212	005067	173450		CLR	DATBUF+2		;32 BITS WORTH
2541	007216	012767	000001	173450	20\$:	MOV	1\$SBEMSK	;SINGLE ERROR MASK
2542	007224	005067	173446		CLR	SBEMSK+2		
2543	007230	012767	000001	173442	30\$:	MOV	1\$DBEMSK	;DOUBLE ERROR MASK
2544	007236	005067	173440		CLR	DBEMSK+2		
2545	007242	016767	173416	173420	35\$:	MOV	DATBUF,TSTDAT	;PRESERVE ORIG DATA
2546	007250	016767	173412	173414		MOV	DATBUF+2,TSTDAT+2	
2547	007256	105737	000302		TSTB	2\$PASFLG		;WHICH PASS ?
2548	007262	001404			BEQ	40\$		;FIRST PASS, NO COMPLEMENTING
2549	007264	005167	173400		COM	TSTDAT		
2550	007270	005167	173376		COM	TSTDAT+2		;SECOND PASS, COMPLEMENT TSTDAT
2551	007274	005077	173430	40\$:	CLR	2CSRADR		
2552	007300	026767	173370	173372	CMP	SBEMSK,DBEMSK		;CHECK FOR SAME MASKS

L04

 DZMML MACY11 30(1046) 05-AUG-77 12:59 PAGE 1-49
 DZMMLB.P11 05-AUG-77 12:57

BEGIN OF AREA TESTED (+20) WHEN PROGRAM RELOCATES.

SEQ 0051

2553	007306	001004			BNE	45\$	;BR IF NOT EQUAL
2554	007310	026767	173362	173364	CMP	SBEMSK+2,DBEMSK+2	;SECOND WORD ALSO
2555	007316	001473			BEQ	70\$	;BR TO MAKE THEM NOT EQUAL
2556	007320	012767	002670	003464	MOV	#TSTDAT,SOURCE	;ADDRESS FOR CHKGEN
2557	007326	004767	003532		JSR	PC,CHKGEN	;GO GENERATE CHECK BITS
2558	007332	004567	004046		JSR	R5,BITCOM	;BIT COMP ROUTINE
2559	007336	002674			.WORD	SBEMSK	;MASK
2560	007340	002670			.WORD	TSTDAT	;DATA WORD
2561	007342	004567	004036		JSR	R5,BITCOM	;MUST FORCE SECOND ERROR
2562	007346	002700			.WORD	DBEMSK	;MASK
2563	007350	002670			.WORD	TSTDAT	
2564	007352	016702	003436		MOV	CHECK,R2	;GET CHECKBITS
2565	007356	056702	173330		BIS	DIAGA,R2	;SET DIAGNOSTIC A BIT
2566	007362	013701	000320		MOV	#MINMEM,R1	;TEST LOCATION
2567	007366	016777	173316	173334	MOV	ECCDIS,#CSRADR	;DISABLE ECC
2568	007374	016721	173270		MOV	TSTDAT,(R1)+	;WRITE FIRST 16 BITS
2569	007400	010277	173324		MOV	R2,#CSRADR	;LOAD CSR WITH IMAGE FROM R2
2570	007404	016711	173262		MOV	TSTDAT+2,(R1)	;WRITE SECOND 16 BITS + CHECKBITS
2571	007410	005077	173314		CLR	#CSRADR	;CLEAR CSR
2572	007414	013702	000320		MOV	#MINMEM,R2	;GET ADDRESS OF TEST LOC
2573	007420	010203			MOV	R2,R3	;R2 DESIGNATES FIRST BYTE
2574	007422	062703	000003		ADD	#3,R3	;R3 DESIGNATES LAST BYTE
2575	007426	112722	000360		MOV	#360,(R2)+	;TRY WRITING A BYTE
2576	007432	013701	000320		MOV	#MINMEM,R1	
2577	007436	026711	173226		CMP	TSTDAT,(R1)	;CHECK FOR UNCHANGED DATA
2578	007442	001405			BEQ	60\$	;BR IF OK
2579	007444	016700	173220		MOV	TSTDAT,R0	;FOR ERROR MSG
2580	007450	004767	004064		JSR	PC,ERROR	;*ERROR* REPORT ERROR MESSAGE
(1)	007454	000056			56		;*****ERROR NUMBER 56*****
(1)							
2581	007456	062701	000002		ADD	#2,R1	;UPPER WORD
2582	007462	026711	173204		CMP	TSTDAT+2,(R1)	;READ SECOND WORD
2583	007466	001405			BEQ	65\$	;BR IF UNCHANGED
2584	007470	016700	173176		MOV	TSTDAT+2,R0	
2585	007474	004767	004040		JSR	PC,ERROR	;*ERROR* REPORT ERROR MESSAGE
(1)	007500	000057			57		;*****ERROR NUMBER 57*****
(1)							
2586	007502	020203			CMP	R2,R3	;TESTED LAST BYTE ?
2587	007504	001350			BNE	55\$	;BR IF NO
2588	007506	005737	000406		TST	#SPASS	;FIRST PASS ?
2589	007512	001424			BEQ	100\$	;BR IF YES
2590	007514	005767	173162		TST	DBEMSK+2	;CHECKING FOR LAST ERROR BIT
2591	007520	100404			BMI	80\$	;BR IF DONE HERE
2592	007522	004567	003626		JSR	R5,DASHL	;NOT DONE, SHIFT LEFT
2593	007526	002700			.WORD	DBEMSK	;DOUBLE ERROR MASK
2594	007530	000644			BR	35\$	
2595	007532	005767	173140		TST	SBEMSK+2	;LAST SBE MASK
2596	007536	100404			BMI	90\$	;BR IF DONE WITH THIS PASS
2597	007540	004567	003610		JSR	R5,DASHL	;DOUBLE WORD SHIFT ROUTINE
2598	007544	002674			.WORD	SBEMSK	
2599	007546	000630			BR	30\$	
2600	007550	105737	000302		TSTB	#PASFLG	;TEST PASS FLAG
2601	007554	001003			BNE	100\$	;NON ZERO MEANS WE'RE DONE
2602	007556	105237	000302		INCB	#PASFLG	;NOT DONR
2603	007562	000611			BR	10\$	
2604	007564	016777	173120	173136	MOV	ECCDIS,#CSRADR	;DISABLE ECC

M04

DZMML MACY11 30(1046) 05-AUG-77 12:59 PAGE 1-50

SEQ 0052

DZMMLB.P11 05-AUG-77 12:57

BEGIN OF AREA TESTED (+20) WHEN PROGRAM RELOCATES.

2605	007572	013701	000320		MOV	2#MINMEM,R1	;TEST LOCATION
2606	007576	005021			CLR	(R1)+	
2607	007600	005011			CLR	(R1)	;TO ERASE ANY DBE'S FROM TESTING
2608	007602	005077	173122		CLR	2CSRADR	;RESTORE CSR
2609	007606	004767	003526		JSR	PC,RSTOT4	;RESTORE REGS R0 TO R4
2610	007612	000167	172770	END11:	JMP	TSTSCP	
2611							
2612							;DOUBLE BIT ERROR WRITE CANCEL WITH
2613							WORD WRITE.
2614							;CHECKS WRITE INHIBIT WITH WORD WRITES TO
2615							WORD WITH DOUBLE ERROR.
2616							
2617	007616	122737	000012	000404	TST12:	CMPB	12,2#STESTN
2618	007624	001403				BEQ	15
2619	007626	004767	004362			JSR	PC,SEQERR
(1)	007632	000060				60	;*****ERROR NUMBER 60*****
(1)							
2620	007634	004467	003466		15:	JSR	R4,SAVOT4
2621	007640	005067	173020		T12A:	CLR	DATBUF
2622	007644	005067	173016			CLR	DATBUF+2
2623	007650	012767	000001	173016		MOV	1,SBEMSK
2624	007656	005067	173014			CLR	SBEMSK+2
2625	007662	012767	000001	173010	T12B:	MOV	1,DBEMSK
2626	007670	005067	173006			CLR	DBEMSK+2
2627	007674	016767	172764	172766	35\$:	MOV	DATBUF,TSTDAT
2628	007702	016767	172760	172762		MOV	DATBUF+2,TSTDAT+2
2629	007710	105737	000302			TSTB	2#PASFLG
2630	007714	001404				BEQ	40\$
2631	007716	005167	172746			COM	TSTDAT
2632	007722	005167	172744			COM	TSTDAT+2
2633	007726	026767	172742	172744	40\$:	CMP	SBEMSK,DBEMSK
2634	007734	001004				BNE	45\$
2635	007736	026767	172734	172736		CMP	SBEMSK+2,DBEMSK+2
2636	007744	001501				BEQ	70\$
2637	007746	012767	002670	003036	45\$:	MOV	1,TSTDAT,SOURCE
2638	007754	004767	003104			JSR	PC,CHKGEN
2639	007760	004567	003420			JSR	R5,BITCOM
2640	007764	002674				.WORD	SBEMSK
2641	007766	002670				.WORD	TSTDAT
2642	007770	004567	003410			JSR	R5,BITCOM
2643	007774	002700				.WORD	DBEMSK
2644	007776	002670				.WORD	TSTDAT
2645	010000	016700	003010			MOV	CHECK,R0
2646	010004	056700	172702			BIS	DIAGA,R0
2647	010010	013701	000320		50\$:	MOV	2#MINMEM,R1
2648	010014	016777	172670	172706		MOV	ECCDIS,2CSRADR
2649	010022	016721	172642			MOV	TSTDAT,(R1)+
2650	010026	010077	172676			MOV	R0,2CSRADR
2651	010032	016711	172634			MOV	TSTDAT+2,(R1)
2652	010036	105067	003010			CLRB	UPPFLG
2653	010042	162701	000002			SUB	2,R1
2654	010046	005077	172656		55\$:	CLR	2CSRADR
2655	010052	012721	177400			MOV	177400,(R1)+
2656	010056	013701	000320			MOV	2#MINMEM,R1
2657	010062	026711	172602			CMP	TSTDAT,(R1)
2658	010066	001405				BEQ	60\$

;CHECK FOR ORIGINAL DATA
;SHOULD BE UNCHANGED

N04

DZMML MACY11 30(1046) 05-AUG-77 12:59 PAGE 1-51

SEQ 0053

DZMMLB.P11 05-AUG-77 12:57

BEGIN OF AREA TESTED (+20) WHEN PROGRAM RELOCATES.

2659	010070	016700	172574		MOV	TSTDAT,RO	;FOR ERROR MSG
2660	010074	004767	003440		JSR	PC,ERROR	;*ERROR* REPORT ERROR MESSAGE
(1)	010100	000061			61		;*****ERROR NUMBER 61*****
(1)							
2661	010102	062701	000C02	60\$:	ADD	#2,R1	;UPPER WORD
2662	010106	026711	172560		CMP	TSTDAT+2,(R1)	;THIS SHOULD BE UNCHANGED ALSO
2663	010112	001405			BEQ	65\$	
2664	010114	016700	172552		MOV	TSTDAT+2,RO	
2665	010120	004767	003414		JSR	PC,ERROR	;*ERROR* REPORT ERROR MESSAGE
(1)	010124	000062			62		;*****ERROR NUMBER 62*****
(1)							
2666	010126	105767	002720	65\$:	TSTB	UPPFLG	;WHICH LOOP ?
2667	010132	001003			BNE	66\$	;SECOND, BR OUT
2668	010134	105267	002712		INCB	UPPFLG	;FIRST, KEEP GOING
2669	010140	000742			BR	55\$	
2670	010142	005737	000406	66\$:	TST	#\$SPASS	;FIRST PASS ?
2671	010146	001426			BEQ	85\$	;BR IF YES
2672	010150	005767	172526	70\$:	TST	DBEMSK+2	;LAST BIT ?
2673	010154	100404			BMI	75\$	;MINUS = BIT 31
2674	010156	004567	003172		JSR	R5,DASHL	;SHIFT ROUTINE
2675	010162	002700			.WORD	DBEMSK	;THIS 32 BIT WORD GETS SHIFTED
2676	010164	000643			BR	35\$	
2677	010166	005767	172504	75\$:	TST	SBEMSK+2	;LAST BIT IN THIS MASK ?
2678	010172	100405			BMI	80\$	;BR IF LAST BIT
2679	010174	004567	003154		JSR	R5,DASHL	
2680	010200	002674			.WORD	SBEMSK	
2681	010202	000167	177454		JMP	T12B	
2682	010206	105737	000302	80\$:	TSTB	#\$PASFLG	;FIRST PASS ?
2683	010212	001004			BNE	85\$	;BR IF SECOND
2684	010214	105237	000302		INCB	#\$PASFLG	;INDICATE SECOND PASS COMING
2685	010220	000167	177414		JMP	T12A	
2686	010224	016777	172460	172476	85\$:	MOV	ECCDIS,#CSRADR
2687	010232	013701	000320		MOV	#MINMEM,R1	;RESTORE TEST ADDRESS
2688	010236	005021			CLR	(R1)+	;CLEAR ANY DBE'S FROM TEST
2689	010240	005011			CLR	(R1)	
2690	010242	005077	172462		CLR	#CSRADR	;CLEAR CSR
2691	010246	004767	003066		JSR	PC,RSTOT4	;RESTORE REGS R0 TO R4
2692	010252	000167	172330	END12:	JMP	TSTSCP	
2693						;TEST DUAL ADDRESS TEST	
2694							
2695						;CHECKS FOR DUAL ADDRESSING BY WRITING	
2696						;AND READING THE ADDRESS IN THE LOCATION	
2697						;AND WRITING AND READING ITS COMPLEMENT	
2698	010256	122737	000013	000404	TST13:	CMPB	#13,#\$TESTN
2699	010264	001403			BEQ	6\$	
2700	010266	004767	003722		JSR	PC,SEQERR	;*ERROR* REPORT ERROR MESSAGE AND HALT AT FATHLT
(1)	010272	000063			63		;*****ERROR NUMBER 63*****
(1)							
2701							
2702	010274	005003		6\$:	CLR	R3	;R3 INDICATES FIRST PASS OR
2703	010276	010100		1\$:	MOV	R1,RO	;COMPLEMENT PASS
2704	010300	005703			TST	R3	;IF R3= ZERO, STORE ADDRESS
2705	010302	001401			BEQ	2\$	;IN THE LOCATION
2706	010304	005100			COM	RO	;OTHERWISE STORE COMPLEMENT
2707	010306	010021		2\$:	MOV	RO,(R1)+	;OF ADDRESS
2708	010310	020105			CMP	R1,R5	;UNTIL HIGHEST LOC IS REACHED

DZMML MACY11 30(1046) 05-AUG-77 12:59 PAGE 1-52
 DZMMLB.P11 05-AUG-77 12:57

BEGIN OF AREA TESTED (+20) WHEN PROGRAM RELOCATES.

SEQ 0054

```

2709 010312 103771          BLO 15
2710 010314 020041      3$: CMP RD,-(R1)      ;CHECK FOR CORRECT CONTENTS
2711 010316 001405          BEQ 4$          ;BRANCH IF OK
2712 010320 105237 000275  INCB 2$SADERR    ;PROBABLY AN ADDRESSING PROB.
2713 010324 004767 003210  JSR PC,ERROR    ;*ERROR* REPORT ERROR MESSAGE
(1) 010330 000064          64          ;*****ERROR NUMBER 64*****
(1)
2714
2715 010332 010100      4$: MOV R1,R0
2716 010334 162700 000002  SUB #2,R0      ;CHECK THAT ADDRESS IS STORED AT
2717 010340 005703      TST R3            ;LOCATION IF R3 IS 0
2718 010342 001401      BEQ 5$            ;OTHERWISE CHECK FOR
2719 010344 005100      COM R0            ;ADDRESS COMPLEMENT.
2720 010346 020104      5$: CMP R1,R4
2721 010350 101361      BHI 3$
2722 010352 112737 000001 000302  MOVB #1,2$PASFLG ;SET PASFLG FOR ERROR REPORT
2723 010360 005103      COM R3            ;COMPLEMENT R3
2724 010362 001345      BNE 1$            ;REPEAT TEST, COMPLEMENTING ADE
2725 010364 000167 172216  END13: JMP TSTSCP
2726
2727
2728 ;*(1) THIS TEST CHECKS THE MEMORY FOR THE CAPABILITY
2729 ;* OF HOLDING 1'S AND 0'S BY WRITING A BACKGROUND
2730 ;* OF BAKPAT AND READING IT
2731 ;*(2) MEMORY IS WRITTEN USING A BYTE AT A TIME
2732 ;*(3) STEPS 1 & 2 ARE REPEATED WITH A SWAPPED BACKGROUND PATTERN
2733
2734 010370 122737 000014 000404 TST14: CMPB #14,2$TESTN ;CHECK FOR PROPER TEST SEQUENCE
2735 010376 001403          BEQ 1$
2736 010400 004767 003610  JSR PC,SEQERR    ;*ERROR* REPORT ERROR MESSAGE AND HALT AT FATHLT
(1) 010404 000065          65          ;*****ERROR NUMBER 65*****
(1)
2737 010406 013700 000312      1$: MOV 2$BAKPAT,R0 ;GET THE PATTERN INTO R0
2738 010412 110021      MOVB R0,(R1)+ ;WRITE A BYTE
2739 010414 113721 000313  MOVB 2$BAKPAT+1,(R1)+ ;WRITE THE MEMORY WITH THE WORD STORED IN BAKPAT
2740 010420 020105      CMP R1,R5 ;COMPARE TEST LOC TO TOP + 2
2741 010422 103771      BLO 1$            ;BRANCH IF LOWER
2742
2743 010424 020041      2$: CMP RD,-(R1)      ;TEST THE MEMORY TO SEE IF IT CONTAINS
2744          ;THE WORD STORED IN BAKPAT
2745 010426 001416      BEQ 8$
2746 010430 062701 000002  ADD #2,R1
2747 010434 123741 000313  CMPB 2$BAKPAT+1,-(R1);CHECK FOR BYTE SELECTION PROBLEM
2748 010440 001402      BEQ 4$
2749 010442 120041      CMPB R0,-(R1)      ;AGAIN CHECK FOR BYTE SELECTION PROBLEM
2750 010444 001002      BNE 6$
2751 010446 105237 000275  4$: INCB 2$SADERR ;PREPARE TO INFORM THAT IT IS ADDRESSING ERROR
2752 010452 042701 000001  6$: BIC #1,R1 ;MAKE THE ADDRESS IN R1 EVEN
2753 010456 004767 003056  JSR PC,ERROR    ;*ERROR* REPORT ERROR MESSAGE
(1) 010462 000066          66          ;*****ERROR NUMBER 66*****
(1)
2754 010464 020104      8$: CMP R1,R4      ;KEEP ON TESTING THE MEMORY UNTIL
2755 010466 101356      BHI 2$            ;R1 EQUALS THE LOWEST ADDRESS
2756 010470 000337 000312  SWAB 2$BAKPAT ;CHANGE THE DATA PATTERN
2757 010474 001744      BEQ 1$            ;IF THE DATA PATTERN DOES NOT HAVE LOW
2758          ;BYTE =0 THEN FALL THRU

```


DZMML MACY11 30(1046) 05-AUG-77 12:59 PAGE 1-53
 DZMMLB.P11 05-AUG-77 12:57

BEGIN OF AREA TESTED (+20) WHEN PROGRAM RELOCATES.

SEQ 0055

```

2759 010476 000167 172104      END14: JMP    TSTSCP
2760
2761
2762
2763
2764
2765
2766
2767
2768
2769 010502 122737 000015 000404 TST15: CMPB    #15,2#STESTN ;CHECK FOR PROPER TEST SEQUENCE
2770 010510 001403          BEQ      1$
2771 010512 004767 003022      JSR      PC,ERROR ;#ERROR# REPORT ERROR MESSAGE
(1) 010516 000067          B7 ;*****ERROR NUMBER 67*****
(1)
2772 010520 010446          1$: MOV     R4,-(SP) ;SAVE R4
2773 010522 016703 172164      MOV     DIAGA,R3 ;DIAGNOSTIC CHECK MODE
2774 010526 052703 004000      BIS     #4000,R3 ;CHECK BIT CT
2775 010532 013701 000320      MOV     2#MINMEM,R1 ;FIRST TEST LOC
2776 010536 010146          MOV     R1,-(SP) ;SAVE IT FOR LATER
2777 010540 005004          CLR     R4
2778 010542 010377 172162      MOV     R3,2CSRADR ;SET UP CSR
2779 010546 010411          2$: MOV     R4,(R1) ;WRITE CHECKBITS BY WRITING LOC
2780 010550 012761 100000 000002 MOV     #100000,2(R1) ;SET BIT 31 TO MATCH CHECKBITS
2781 010556 062701 000004      ADD     #4,R1 ;POINT TO NEXT WORD
2782 010562 020105          CMP     R1,R5 ;TOP + 2
2783 010564 103770          BLO     2$ ;BR IF NOT
2784 010566 162701 000704      3$: SUB     #4,R1 ;ADJUST R1
2785 010572 005711          TST     (R1) ;READ
2786 010574 032777 003740 172126 BIT     #3740,2CSRADR ;SHOULD BE ALL ZEROS
2787 010602 001421          BEQ     4$ ;BR IF OK
2788 010604 012700 004000      MOV     #4000,R0 ;GOOD DATA
2789 010610 004767 005036      JSR     PC,BITCHK ;USE XOR TO COUNT BAD CHECKBITS
2790 010614 032777 004000 172110 BIT     #4000,2SWR ;ECC DIS ?
2791 010622 001004          BNE     33$ ;PRNIT SINGLE ERRORS
2792 010624 022767 000001 005102 CMP     #1,BITCNT ;MORE THAN 1 BAD BIT ?
2793 010632 001405          BEQ     4$ ;YES
2794 010634 012700 004000      33$: MOV     #4000,R0 ;FOR ERROR MSG
2795 010640 004767 002674      JSR     PC,ERROR ;#ERROR# REPORT ERROR MESSAGE
(1) 010644 000070          70 ;*****ERROR NUMBER 70*****
(1)
2796 010646 052777 003740 172054 4$: BIS     #3740,2CSRADR ;MAKE CHECKBITS ALL 1'S EXCEPT FOR CT
2797 010654 042777 004000 172046 BIC     #4000,2CSRADR ;MAKE CT = 0
2798 010662 010411          MOV     R4,(R1) ;WRITE THE CHECKBITS
2799 010664 010461 000002      MOV     R4,2(R1) ;BOTH WORDS
2800 010670 020116          CMP     R1,(SP) ;BOTTOM YET ?
2801 010672 001335          BNE     3$ ;BR IF NO
2802 010674 010377 172030      5$: MOV     R3,2CSRADR ;RESTORE CSR
2803 010700 005711          TST     (R1) ;READ THE LOC AGAIN
2804 010702 017700 172022      MOV     2CSRADR,R0 ;GET CSR CONTENTS INTO R0
2805 010706 042700 170037      BIC     #170037,R0 ;ONLY WANT BITS 11-5
2806 010712 022700 003740      CMP     #3740,R0 ;READ FOR ALL 1'S
2807 010716 001421          BEQ     6$ ;BR IF OK
2808 010720 012700 003740      MOV     #3740,R0 ;FOR ERROR MSG
2809 010724 004767 004722      JSR     PC,BITCHK
2810 010730 032777 004000 171774 BIT     #4000,2SWR ;ECC DIS ?

```

DZMML MACY11 30(1046) 05-AUG-77 12:59 PAGE 1-54
 DZMMLB.P11 05-AUG-77 12:57 BEGIN OF AREA TESTED (+20) WHEN PROGRAM RELOCATES.

SEQ 0056

2811	010736	001004			BNE	55\$		
2812	010740	022767	000001	004766	CMP	#1,BITCNT		;MORE THAN ONE ?
2813	010746	001405			BEQ	6\$		;NO
2814	010750	012700	003740		MOV	#3740,R0		;FOR MSG
2815	010754	004767	002560		JSR	PC,ERROR		;*ERROR* REPORT ERROR MESSAGE
(1)	010760	000071			71			;*****ERROR NUMBER 71*****
(1)								
2816	010762	010377	171742		MOV	R3,ACSRADR		;RESTORE CSR
2817	010766	010411			MOV	R4,(R1)		;ACCESS THE LOC
2818	010770	012761	100000	000002	MOV	#100000,2(R1)		;SECOND WORD
2819	010776	062701	000004		ADD	#4,R1		;POINT TO NEXT ADDRESS
2820	011002	020105			CMP	R1,R5		;TOP + 2 YET?
2821	011004	103733			BLO	5\$		;BR IF NOT
2822	011006	011601			MOV	(SP),R1		;RESTORE MINMEM
2823	011010	010377	171714		MOV	R3,ACSRADR		;RESTORE CSR
2824	011014	005711			TST	(R1)		;READ
2825	011016	032777	003740	171704	BIT	#3740,ACSRADR		;SHOULD BR 0
2826	011024	001421			BEQ	8\$		
2827	011026	012700	004000		MOV	#4000,R0		;GOOD DATA
2828	011032	004767	004614		JSR	PC,BITCHK		;CHECK # OF BAD BITS
2829	011036	032777	004000	171666	BIT	#4000,ASMR		;ECC DIS ?
2830	011044	001004			BNE	77\$		;YES
2831	011046	022767	000001	004660	CMP	#1,BITCNT		;MORE THAN ONE
2832	011054	001405			BEQ	8\$		;NO
2833	011056	012700	004000		MOV	#4000,R0		;FOR PRINTOUT
2834	011062	004767	002452		JSR	PC,ERROR		;*ERROR* REPORT ERROR MESSAGE
(1)	011066	000072			72			;*****ERROR NUMBER 72*****
(1)								
2835	011070	052777	003740	171632	BIS	#3740,ACSRADR		;COMP CHECKBITS
2836	011076	042777	004000	171624	BIC	#4000,ACSRADR		;CLEAR CT
2837	011104	010411			MOV	R4,(R1)		;WRITE
2838	011106	010461	000002		MOV	R4,2(R1)		
2839	011112	062701	000004		ADD	#4,R1		;NEXT WORD
2840	011116	020105			CMP	R1,R5		;TOP + 2 ?
2841	011120	103733			BLO	7\$		;BR IF NO
2842	011122	010377	171602		MOV	R3,ACSRADR		;RESTORE CSR
2843	011126	162701	000004		SUB	#4,R1		;ADJUST R1
2844	011132	005711			TST	(R1)		;READ
2845	011134	017700	171570		MOV	ACSRADR,R0		;GET CSR CONTENTS
2846	011140	042700	170037		BIC	#170037,R0		;ONLY WANT CHECKBITS
2847	011144	022700	003740		CMP	#3740,R0		;ALL ONES ?
2848	011150	001421			BEQ	10\$		;BR IF OK
2849	011152	012700	003740		MOV	#3740,R0		;FOR ERROR MSG
2850	011156	004767	004470		JSR	PC,BITCHK		;COUNT BAD BITS
2851	011162	032777	004000	171542	BIT	#4000,ASMR		;ECC DIS
2852	011170	001004			BNE	99\$		;YES
2853	011172	022767	000001	004534	CMP	#1,BITCNT		;MORE THAN ONE ?
2854	011200	001405			BEQ	10\$		;NO
2855	011202	012700	003740		MOV	#3740,R0		;FOR MSG.
2856	011206	004767	002326		JSR	PC,ERROR		;*ERROR* REPORT ERROR MESSAGE
(1)	011212	000073			73			;*****ERROR NUMBER 73*****
(1)								
2857	011214	010377	171510		MOV	R3,ACSRADR		;RESTORE CSR
2858	011220	010411			MOV	R4,(R1)		;WRITE NEW CHECKBITS
2859	011222	012761	100000	000002	MOV	#100000,2(R1)		;BIT 31
2860	011230	020116			CMP	R1,(SP)		;BOTTOM YET ?

E05

DZMPL MACY11 30(1046) 05-AUG-77 12:59 PAGE 1-55
 DZMPLB.P11 05-AUG-77 12:57 BEGIN OF AREA TESTED (+20) WHEN PROGRAM RELOCATES.

SEQ 0057

```

2861 011232 001335      BNE      9$          ;BR IF NOT
2862 011234 005711      TST      (R1)        ;RERAD
2863 011236 032777 003740 171464 11$: BIT      #3740,2CSRADR ;SHOULD BE ALL 0'S
2864 011244 001421      BEQ      12$        ;BR IF OK
2865 011246 012700 004000      MOV      #4000,R0 ;GOOD DATA
2866 011252 004767 004374      JSR      PC,BITCHK ;COUNT BAD BITS
2867 011256 032777 004000 171446      BIT      #4000,2SWR ;ECC DIS ?
2868 011264 001004      BNE      111$       ;YES
2869 011266 022767 000001 004440      CMP      #1,BITCNT ;MORE THAN 1
2870 011274 001405      BEQ      12$        ;BR NO
2871 011276 012700 004000 111$: MOV      #4000,R0 ;FOR MSG
2872 011302 004767 002232      JSR      PC,ERROR ;*ERROR* REPORT ERROR MESSAGE
(1) 011306 000074      74          ;*****ERROR NUMBER 74*****
(1)
2873 011310 062701 000004 171410 12$: ADD      #4,R1 ;POINT TO NEXT WORD
2874 011314 010377      MOV      R3,2CSRADR ;RESTORE CSR
2875 011320 020105      CMP      R1,R5 ;TOP + 2 YET?
2876 011322 103744      BLO      11$        ;BR IF NOT
2877 011324 005726      TST      (SP)+ ;FINALLY RESTORE STACK
2878 011326 012604      MOV      (SP)+,R4 ;RESTORE R4
2879 011330 000167 171252 END15: JMP      TSTSCP ;AND EXIT
2880
2881
2882 ;*(1) THIS TEST WRITES A BACK GROUND OF THE WORD STORED
2883 ;* AT BAKPAT.
2884 ;*(2) READS EVERY LOCATION FOR CORRECT DATA, SWAPS BYTES
2885 ;* AT THE LOCATION AND PROCEEDS IN MAX. TO MIN
2886 ;* DIRECTION OF MEMORY LOCATIONS.
2887 ;*(3) READS EVERY LOCATION FOR SWAPPED BAKPAT PATTERN
2888 ;* WRITES BAKPAT BACKGROUND IN THE LOCATION AND PROCEEDS
2889 ;* IN MIN. TO MAX. DIRECTION
2890 ;*(4) REPEATS STEP 2 GOING IN MIN. TO MAX. DIRECTION
2891 ;*(5) REPEATS STEP 3 GOING IN MAX. TO MIN. DIRECTION
2892
2893
2894 011334 122737 000016 000404 TST16: CMPB     #16,2$TESTN ;CHECK FOR PROPER TEST SEQUENCE
2895 011342 001403      BEQ      1$          ;
2896 011344 004767 002644      JSR      PC,SEQERR ;*ERROR* REPORT ERROR MESSAGE AND HALT AT FATHLT
(1) 011350 000075      75          ;*****ERROR NUMBER 75*****
(1)
2897 011352 004737 000120 1$: JSR      PC,2$WRTMEM ;GO TO WRITE THE MEMORY WITH A BACKGROUND OF THE
2898 ;WORD STORED IN BAKPAT
2899 011356 020041 2$: CMP      R0,-(R1) ;READ THE CONTENTS OF LOCATION POINTED BY R1
2900 011360 001403      BEQ      3$          ;TO SEE IF IT HAS THE SAME VALUE AS R0
2901 011362 004767 002152      JSR      PC,ERROR ;*ERROR* REPORT ERROR MESSAGE
(1) 011366 000076      76          ;*****ERROR NUMBER 76*****
(1)
2902 011370 000300 3$: SWAB      R0 ;SWAP THE BYTES AT (R1)
2903 011372 010011      MOV      R0,(R1) ;READ (R1) FOR CORRECT VALUE
2904 011374 021100      CMP      (R1),R0
2905 011376 001403      BEQ      4$          ;
2906 011400 004767 002134      JSR      PC,ERROR ;*ERROR* REPORT ERROR MESSAGE
(1) 011404 000077      77          ;*****ERROR NUMBER 77*****
(1)
2907
2908 011406 000300 4$: SWAB      R0 ;SWAP THE BYTES OF THE REGISTER

```

F05

DZMML MACY11 30(1046) 05-AUG-77 12:59 PAGE 1-56
 DZMMLB.P11 05-AUG-77 12:57

SEQ 0058

BEGIN OF AREA TESTED (+20) WHEN PROGRAM RELOCATES.

```

2909          011410 001023          BNE      9$      ;CONTAINING BACKGROUND PATTERN
2910          ;IF THE LOWER BYTE OF THE REGISTER
2911          ;IS NOT 0 THEN THE PROGRAM IS READING
2912          ;THE MEMORY TO CONTAIN A BACK GROUND OF
2913          ;BAKPAT AND WRITING THE SWAPPED WORD
2914          ;IN WHICH CASE GO TO 9$
2915
2916
2917
2918          011412 005703          5$:      TST      R3      ;R3 WAS 0 WHEN THE PROGRAM ENTERED
2919          ;THIS TEST, AND IT IS NOT ALTERED UNTIL PASFLG=3
2920          ;IF R3 EQUAL 0 THEN THE PROGRAM IS
2921          ;READING/WRITING MIN. TO MAX. OTHERWISE
2922          ;IT IS GOING IN MAX. TO MIN. DIRECTION
2923          011414 001023          BNE      10$      ;IF R3 IS NOT CLEAR THEN GO TO 10$
2924          011416 062701 000002      6$:      ADD      #2,R1      ;OTHERWISE ADD 2 TO THE CONTENTS OF R1
2925          011422 020105          CMP      R1,R5      ;COMPARE R1 WITH THE MAX. MEMORY LOCATION TO
2926          ;BE TESTED
2927          011424 103006          BHS      8$      ;IF R1>R5 THEN GO TO 8$ OTHERWISE
2928          011426 020011          7$:      CMP      R0,(R1)      ;READ (R1) FOR THE CORRECT DATA
2929          011430 001757          BEQ      3$      ;WRITE COMPLEMENT OF THE DATA FOUND AT (R1)
2930          ;AND REPEAT UNTIL R1 > R5
2931          011432 004767 002102      JSR      PC,ERROR      ;*ERROR* REPORT ERROR MESSAGE
2932          (1) 011436 000100          100      ;*****ERROR NUMBER 100*****
2933          (1)
2934          011440 000753          BR      3$
2935          011442 105237 000302      8$:      INCB      2#PASFLG
2936          011446 000300          SWAB      R0
2937          011450 001742          BEQ      2$
2938          ;IF THE LOWER BYTE OF R0 IS ALL 0'S
2939          ;THEN BEGIN READING BAKPAT SWAPPED WRITING BAKPAT
2940          ;AND READING BAKPAT GOING FROM MAX. TO MIN.[PASFLG=4]
2941          ;OTHERWISE CLEAR R0
2942          ;PUT THE LOWEST TESTING ADDRESS IN R1
2943          ;AND BEGIN READING 0'S, WRITING 1'S AND
2944          ;READING 1'S IN MIN. TO MAX. DIRECTION [PASFLG=3]
2945          011452 005103          COM      R3
2946          011454 010401          MOV      R4,R1
2947          011456 000763          BR      7$
2948
2949          011460 005703          9$:      TST      R3
2950          011462 001353          BNE      5$
2951          ;THEN READ BAKPAT, WRITE
2952          ;SWAPPED BAKPAT AND READ SWAPPED BAKPAT
2953          ;IN MIN. TO MAX. DIRECTION
2954          ;OTHERWISE TEST IS PROCEEDING IN MAX. TO
2955          ;MIN. DIRECTION.
2956          ;KEEP ON LOOPING UNTIL R1=R4
2957          011464 020104          10$:      CMP      R1,R4
2958          011466 101333          BHI      2$
2959          011470 105237 000302      INCB      2#PASFLG
2960          011474 000300          SWAB      R0
2961          011476 001753          BEQ      7$
2962          ;IF R0 SWAPPED HAS LOWER BYTE=0
2963          ;THEN READ BAKPAT SWAPPED, WRITE BAKPAT,
2964          ;AND READ BAKPAT GOING FROM MIN. TO MAX.
2965
2966          011500 000167 171102      END16:  JMP      TSTSCP
2967
2968          ;*(1) THIS TEST WRITES THE MEMORY WITH A BACK GROUND OF BAKPAT
2969          ;*(2) WITH PASFLG=0 THE TEST READS THE MEMORY FOR BAKPAT
2970          ;* AND THEN INCREMENTS PASFLG
2971          ;*(3) IT THEN READS/SWAPS BYTES/WRITES A LOCATION X FOR
2972          ;* OVER 2 MSEC AND THEN READS THE MEMORY FOR BAKPAT

```


G05

DZMPL MACY11 30(1046) 05-AUG-77 12:59 PAGE 1-57

SEQ 0059

DZMPLB.P11 05-AUG-77 12:57

BEGIN OF AREA TESTED (+20) WHEN PROGRAM RELOCATES.

```

2963      ;*(4) REPEATS STEP 3 WITH X=X+4K UNTIL END OF MEMORY IS ENCOUNTERED
2964      ;*(5) IT THEN INCREMENTS PASFLG AND WRITES THE MEMORY TO
2965      ;*(6) BAKPAT AND WITH PASFLG=2 IT READS MEMORY FOR ALL
2966      ;*(6) SWAPPED BAKPAT AFTER WHICH PASFLG IS INCREMENTED TO 3
2967      ;*(6) REPEATS STEPS 3 AND 4 READING THE MEMORY FOR SWAPPED
2968      ;*(6) BAKPAT INSTEAD OF BAKPAT.
2969 011504 122737 000017 000404 TST17: CMPB  #17,2#TESTN ;CHECK FOR PROPER TEST SEQUENCE
2970 011512 001403 BEQ  .+10
2971 011514 004767 002474 JSR  PC,SEQERR ;*ERROR* REPORT ERROR MESSAGE AND HALT AT FATHLT
(1) 011520 000101 101 ;*****ERROR NUMBER 101*****
(1)
2972 011522 004737 000120 RPT17: JSR  PC,2#WRTMEM ;GO TO WRITE THE MEMORY WITH A BACKGROUND OF THE
2973 ;WORD STORED AT LOCATION BAKPAT
2974 011526 005037 000302 CLR  2#PASFLG
2975 011532 010403 15: MOV  R4,R3 ;SET R3
2976 011534 010401 25: MOV  R4,R1 ;AND R1 TO THE STARTING ADDRESS OF MEMORY UNDER TEST
2977 011536 020011 35: CMP  R0,(R1) ;CHECK (R1) FOR CORRECT DATA
2978 011540 001403 BEQ  45
2979 011542 004767 001772 JSR  PC,ERROR ;*ERROR* REPORT ERROR MESSAGE
(1) 011546 000102 102 ;*****ERROR NUMBER 102*****
(1)
2980 011550 062701 000002 45: ADD  #2,R1 ;INCREMENT R1 BY 2
2981 011554 020105 CMP  R1,R5 ;SEE IF R1 HAS REACHED THE MAX. OF MEMORY
2982 011556 103767 BLO  35
2983 011560 132737 000001 000302 BITB  #1,2#PASFLG ;CHECK TO SEE IF PASFLG=0 OR 2
2984 011566 001002 BNE  55
2985 011570 105237 000302 INCB  2#PASFLG ;IN WHICH CASE INCREMENT PASFLG COUNTER BY 1
2986
2987 011574 020305 55: CMP  R3,R5 ;SEE IF R3 HAS REACHED THE MAX. OF THE MEMORY
2988 011576 103012 BHIS  75
2989 011600 012702 037776 MOV  #37776,R2 ;WRITE INTO 1 LOC FOR >2MS (ABOUT 100MS)
2990 011604 000313 65: SWAB (R3)
2991 011606 005302 DEC  R2
2992 011610 001375 BNE  65
2993 011612 010337 000350 MOV  R3,2#SAVLOC ;SAVE LOCATION WRITTEN FOR 2MS FOR ERROR REPORT.
2994 011616 062703 020000 ADD  #20000,R3 ;BY ADDING 1 TO THE 14TH ADDRESS BIT CAUSE
2995 ;R3 TO POINT TO A LOCATION IN THE NEXT
2996 ;4K BANK OF MEMORY
2997 011622 000744 BR  25
2998 011624 105237 000302 75: INCB  2#PASFLG ;MAKE PASFLG=2
2999 011630 000337 000312 SWAB  2#BAKPAT ;IF BAKPAT IS NOT BEING SWAPPED FOR THE 2ND
3000 011634 001732 BEQ  RPT17 ;THEN GO BACK TO THE LOCATION RPT17
3001 011636 000167 170744 END17: JMP  TSTSCP
3002
3003
3004
3005
3006
3007
3008
3009

```

H05

 DZMML MACY11 30(1046) 05-AUG-77 12:59 PAGE 1-58
 DZMMLB.P11 05-AUG-77 12:57 RELOCATE?

SEQ 0060

```

3014 011642 005077 171062 RELOC: CLR 2CSRADR
3015 011646 012737 000377 000312 MOV 2377,2#BAKPAT
3016 011654 105737 000272 TSTB 2#MMAVA ;IS THE MEMORY MANAGEMENT BEING TESTED?
3017 011660 001170 BNE CONTMM ;IF SO THEN GO TO CONTMM AND CONTINUE TESTING
3018 ;MEMORY MANAGEMENT
3019 011662 005767 171030 TST LDPLG ;ONLY RELOCATE IF IN MEM UNDER TEST
3020 011666 001542 BEQ CKDONE ;DON'T RELOC, BRANCH
3021 011670 032777 001000 171034 BIT 21000,2SWR ;RELOCATION WANTED?
3022 011676 001136 BNE CKDONE ;BRANCH IF NO
3023 011700 005767 171022 TST INHREL ;RELOCATION ALLOWED?
3024 011704 001133 BNE CKDONE ;NO, SINGLE ERROR IN LOC 0-500, BRANCH
3025 011706 105737 000170 TSTB 2#REL ;IF THE PROGRAM HAS ALREADY BEEN RELOCATED THEN ALSO
3026 011712 100473 BMI RELOER ;PLACE THE PROGRAM BACK IN LOWER CORE
3027 011714 112737 000200 000170 MOVB 2200,2#REL ;OTHERWISE PREPARE TO RELOCATE
3028
3029 ;RELOCATE THE DIAGNOSTIC TO SECOND 16K
3030
3031
3032 011722 012701 100500 MOV 2100000+BEGIN,R1 ;START OF RELOCATED VERSION
3033 011726 012702 116132 MOV 2100000+ENDPRG,R2 ;END
3034 011732 005711 3S: TST (R1) ;READ A WORD
3035 011734 032777 100000 170766 BIT 2100000,2CSRADR ;CHECK FOR DBE
3036 011742 001005 BNE 5S ;BR IF ERROR
3037 011744 062701 000004 ADD 24,R1 ;NEXT WORD
3038 011750 020102 CMP R1,R2 ;END YET?
3039 011752 103767 BLO 3S ;BR IF NOT
3040 011754 000405 BR 4S ;BR IF ALL IS OK
3041 011756 105037 000170 5S: CLRB 2#REL ;INDICATE NO RELOC
3042 011762 004767 002226 JSR PC,FATERR ;*ERROR* REPORT ERROR MESSAGE AND HALT AT FATHLT
(1) 011766 000103 103 ;*****ERROR NUMBER 103*****
(1)
3043 011770 032777 000040 170734 4S: BIT 240,2SWR ;PRINTING WANTED
3044 011776 001005 BNE 1S ;BR IF NO
3045 012000 004767 002434 JSR PC,PNTMES ;TYPE "RELOC"
3046 012004 042522 047514 000103 .ASCIZ /RELOC/
3047 .EVEN
3048 012012 004767 003270 1S: JSR PC,CLMM ;CLEAR MM REGS
3049 012016 012737 000500 000320 MOV 2BEGIN,2#MINMEM ;NEW TEST ADDRESS
3050 012024 052767 000010 170656 BIS 210,ECCDIS ;PROTECT 2ND 16K
3051 012032 052767 000010 170652 BIS 210,DIAGA ;DITTO
3052 012040 012705 100000 MOV 210000,R5 ;START OF 2ND 16K
3053 012044 010567 170650 MOV R5,BASE ;BASE FOR RELOC
3054 012050 012704 000430 MOV 2BEGIN-50,R4 ;FIRST LOC TO BE RELOCATED
3055 012054 060405 ADD R4,R5
3056 012056 012425 2S: MOV (R4)+(R5)+ ;MOV A WORD
3057 012060 020437 000342 CMP R4,2#SAVR4 ;LAST LOC YET?
3058 012064 103774 BLO 2S ;BR IF NO
3059 012066 012704 000430 MOV 2BEGIN-50,R4 ;FIRST TEST ADDRESS
3060 012072 016705 170622 MOV BASE,R5
3061 012076 000137 100500 JMP 2#BEGIN+100000 ;JUMP TO RELOCATED VERSION
3062
3063 ;*RELOCATE THE DIAGNOSTIC BACK TO LOWER MEMORY
3064
3065
3066 012102 105737 000170 RELOER: TSTB 2#REL ;IS DIAGNOSTIC IN RELOCATED STATE?
3067 012106 100032 BPL CKDONE ;BRANCH IF NO

```


DZMML MACY11 30(1046) 05-AUG-77 12:59 PAGE 1-59
 DZMMLB.P11 05-AUG-77 12:57 RELOCATE?

SEQ 0061

3068	012110	005067	170604		CLR	BASE	
3069	012114	042767	000010	170566	BIC	#10,ECCDIS	
3070	012122	042767	000010	170562	BIC	#10,DIAGA	
3071							
3072	012130	016737	170566	000320	MOV	SAVMIN,2#MINMEM	;RESTORE TEST ADDRESS FOR ECC
3073	012136	012704	000430		MOV	#BEGIN-50,R4	;PREPARE TO RELOCATE THE PROGRAM TO LOWER CORE
3074	012142	010405			MOV	R4,R5	
3075	012144	062705	100000		ADD	#100000,R5	
3076	012150	012524		25:	MOV	(R5)+,(R4)+	
3077	012152	020437	000342		CMP	R4,2#SAVR4	
3078	012156	103774			BLO	25	
3079	012160	105037	000170		CLRB	2#REL	
3080	012164	012706	000500		MOV	#BEGIN,SP	;RESET STACK TO LOWER MEMORY
3081	012170	010637	000346		MOV	SP,2#SAVR6	; "BEGIN" USES THIS TO RESET THE STACK.
3082	012174	000137	012200	CKDONE:	JMP	2#LOWER	;TRANSFER THE PROGRAM CONTROL TO THE LOWER CORE
3083							
3084							
3085							
3086	012200	105737	000402	LOWER:	TSTB	2#SFATAL	;FATAL ERROR
3087	012204	001405			BEQ	15	;BR IF NOT
3088	012206	005737	000042		TST	2#42	;APT ?
3089	012212	001402			BEQ	15	;NO KEEP TESTING
3090	012214	000167	002040		JMP	APTHLT	
3091	012220	105737	000311	15:	TSTB	2#SAVKBB	;HERE DUE TO IC TYPED?
3092	012224	001141			BNE	STPSTK	;BRANCH IF YES (TYPE ERROR STACK)
3093	012226	004767	002504	TSTMM:	JSR	PC,MEMMNG	;SET THE REGISTERS IF THE MEMORY MANAGEMENT
3094							IS AVAILABLE
3095	012232	105737	000272		TSTB	2#MMAVA	IS MEM. MANAG. AVAILABLE ?
3096	012236	001474			BEQ	ENDPAS	BRANCH IF NO
3097	012240	000406			BR	SCNTMM	BEGIN TESTING ABOVE 28K
3098	012242	022737	007400	172352	CONTMM:	CMP	#7400,2#172352
3099	012250	001464			BEQ	ENDMAX	DON'T WANT TO WRAP AROUND
3100	012252	004767	002636		JSR	PC,UPMM	GO TO END-OF-PASS
3101	012256	012703	000322	SCNTMM:	MOV	#LOWTWO,R3	GO TO UPDATE MEM. MANAG. REGISTERS
3102	012262	004767	002744		JSR	PC,GETSIZ	MAKE R3 POINT TO THE LOCATION LOWTWO
3103							LOAD BITS 6-10 OF R2 WITH THE BITS 13-17
3104	012266	012704	040000		MOV	#40000,R4	OF THE LOWEST ADDRESS UNDER TEST
3105							MAKE R4 POINT TO THE LOWEST LOCATION IN THE BANK
3106	012272	020237	172344		CMP	R2,2#172344	POINTED BY PAGE ADDRESS REGISTER 2 (PAR2)
3107							IS THE CONTENT OF R2 LOWER THAN THE CONTENT OF
3108	012276	103405			BLO	25	PAR2 ?
3109	012300	050104			BIS	R1,R4	IF SO THEN GO 25
3110							SUBROUTINE GETSIZ LOADED R1 WITH BITS 0-12
3111	012302	162702	000200		SUB	#200,R2	OF LOWADD WHICH HAVE NOW BEEN LOADED IN R4
3112	012306	004767	002430		JSR	PC,MMREG	
3113	012312	010437	000320	25:	MOV	R4,2#MINMEM	;SET MEM. MANAG. REGISTERS
3114	012316	004767	002710		JSR	PC,GETSIZ	NEW MINMEM FOR ECC TESTS
3115							PLACE BITS 13-17 OF HIGHEST LOCATION TO BE TESTED
3116							IN BITS 6-10 OF R2, #160000 IN R0 AND BITS 0-12
3117	012322	004767	000020		JSR	PC,MAXADR	OF LOCATION HIGHADD IN R1
3118	012326	010005			MOV	R0,R5	GET THE ADDRESS OF MAX. MEM. UNDER TEST
3119	012330	004767	002676		JSR	PC,GETSIZ	
3120	012334	004767	000006		JSR	PC,MAXADR	PREPARE TO SET UP LOCATION MAXMEM
3121	012340	010013			MOV	R0,(R3)	GET THE MAXIMUM ADDRESS OF AVAILABLE MEMORY
3122	012342	000167	167664		JMP	CLMEM	AND STORE INTO "MAXMEM"
3123							GO TEST A 20K SLICE ABOVE 28K.

DZMML MACY11 30(1046) 05-AUG-77 12:59 PAGE 1-60
 DZMMLB.P11 05-AUG-77 12:57 RELOCATE?

SEQ 0062

```

3124
3125
3126
3127
3128
3129
3130
3131 012346 010046
3132 012350 012700 172356
3133
3134 012354 162716 020000
3135 012360 050116
3136 012362 020240
3137 012364 001411
3138 012366 020027 172340
3139 012372 101370
3140
3141 012374 062700 000004
3142 012400 021002
3143 012402 003006
3144 012404 012716 157776
3145 012410 012600
3146 012412 062700 000002
3147 012416 000207
3148
3149 012420 022626
3150
3151 012422 016737 170274 000320

```

```

;MAXADR - SUBROUTINE TO GET CURRENT 20K SLICE OF MEMORY ADDRESSES ABOVE 28K.
;REGISTERS:
;R0= ON ENTRY= #160000 AND ON EXIT=HIGHEST VIRTUAL ADDR. UNDER TEST
;R1= LOW ORDER 12 BITS OF VIRTUAL TEST ADDRESS
;R2= PAR BLOCK NO. CURRENTLY UNDER TEST.

MAXADR: MOV R0, -(SP) ;PUT MAXIMUM AVAILABLE ADDRESS ON STACK
        MOV #172356, R0 ;R0=PAR7 UNIBUS ADDRESS
; **BEGIN LOOP TO FIND PAR ADDRESS UNDER TEST
2$: SUB #20000, (SP) ;DECREMENT VIRTUAL ADDRESS BY 4K
    BIS R1, (SP) ;SET BITS 11:0 TO MAXIMUM VIRTUAL TEST ADDRESS
    CMP R2, -(R0) ;DOES CURRENT PAR= TEST BLOCK NO.?
    BEQ 3$ ;BRANCH IF YES
    CMP R0, #172340 ;ARE WE AT PAR0?
    BHI 2$ ;NO KEEP TRYING
; **END LOOP TO FIND PAR ADDRESS UNDER TEST
    ADD #4, R0 ;SET TO CURRENT PAR
    CMP (R0), R2 ;IS THE PAR BLOCK UNDER TEST GTR THAN ALLOWED?
    BGT 4$ ;BRANCH IF YES (FALL INTO ENDPAS)
    MOV #157776, (SP) ;EXIT WITH MAXADR= 28K SEGMENT TEST SIZE
3$: MOV (SP)+, R0 ;SET R0 TO MAXIMUM VIRTUAL TEST ADDRESS
    ADD #2, R0 ;MAKE MAXIMUM MEMORY+2
    RTS PC ;AND EXIT MAXADR ROUTINE

4$: CMP (SP)+, (SP)+ ;FIXUP STACK
    ;AND FALL THRU TO ENDPAS.
ENDMAX: MOV SAVMIN, @#MINMEM ;BECAUSE WE WON'T SIZE AGAIN

```


L05

DZMML MACY11 30(1046) 05-AUG-77 12:59 PAGE 1-62
DZMMLB.P11 05-AUG-77 12:57 TYPE ROUTINE FOR ERROR STACK

SEQ 0064

3212 012644 012703 000376
3213 012650 000743

MOV #376,R3
BR RETSTK

;RESET SCRATCH STACK FOR EACH BIT PRINTED.


```

3218
3219
3220
3221
3222
3223
3224
3225
3226
3227
3228
3229 012652 005002
3230 012654 105737 000311
3231 012660 001050
3232 012662 005237 000406
3233 012666 005067 170032
3234 012672 032777 000040 170032
3235 012700 001015
3236 012702 004767 001524
3237 012706 047105 020104 040520
      012714 051523 021440 000
3238
3239 012722 013737 000406 000306
3240 012730 004767 001532
3241 012734 013700 000042
3242 012740 001405
3243 012742 004767 000012
3244 012746 000005
3245
3246
3247
3248 012750 000137 000156
3249
3250 012754 000137 000250
3251
3252 012760 004767 002322
3253 012764 013704 000342
3254 012770 014445
3255 012772 020437 000304
3256 012776 101374
3257 013000 000207
3258
3259
3260
3261 013002 004767 177752
3262 013006 000167 001246
3263
3264
3265
3266
3267
3268
3269
3270
3271
3272

```

N05

DZMML MACY11 30(1046) 05-AUG-77 12:59 PAGE 1-64
 DZMMLB.P11 05-AUG-77 12:57 END OF PASS

SEQ 0066

3273	013012	000000	SOURCE: .WORD	0	
3274	013014	000000	CHECK: .WORD	0	
3275	013016	125252	CHKTAB: .WORD	125252	:C1 BIT TABLE
3276	013020	125252	.WORD	125252	:C2 BIT TABLE
3277	013022	146314	.WORD	146314	:C4 BIT TABLE
3278	013024	146314	.WORD	146314	:C8 BIT TABLE
3279	013026	170360	.WORD	170360	:C16 BIT TABLE
3280	013030	170360	.WORD	170360	:C32 BIT TABLE
3281	013032	177777	.WORD	177777	:CT BIT TABLE
3282	013034	177400	.WORD	177400	
3283	013036	000377	.WORD	377	
3284	013040	177777	.WORD	177777	
3285	013042	177400	.WORD	177400	
3286	013044	177777	.WORD	177777	
3287	013046	064551	.WORD	64551	
3288	013050	113151	.WORD	113151	
3289					
3290	013052	000	UPPFLG: .BYTE	0	
3291					
3292	013053	000	CHKCNT: .BYTE	0	:C1 PARITY COUNTER
3293	013054	000	.BYTE	0	:C2 PARITY COUNTER
3294	013055	000	.BYTE	0	:C4 PARITY COUNTER
3295	013056	000	.BYTE	0	:C8 PARITY COUNTER
3296	013057	000	.BYTE	0	:C16 PARITY COUNTER
3297	013060	000	.BYTE	0	:C32 PARITY COUNTER
3298	013061	000	.BYTE	0	:CT PARITY COUNTER
3299	013062	077	PARBYT: .BYTE	77	:ODD/EVEN PARITY FLAGS
3300					:0=EVEN
3301					:1=ODD
3302					:CT C32 C16 C8 C4 C2 C1
3303	013064	000236	CHKGEN: .EVEN		
3304	013064	004467	JSR	R4, SAVOT4	:SAVE R0 TO R4
3305	013070	005000	CLR	R0	:CLEARING OUT PARITY COUNTERS
3306	013072	012701	MOV	#CHKCNT, R1	:START 7 COUNTERS
3307	013076	066701	ADD	BASE, R1	
3308	013102	022700	10\$: CMP	#7, R0	:FINISHED?
3309	013106	001403	BEQ	20\$	:BRANCH IF YES
3310	013110	105021	CLRB	(R1)+	:NO, DO THE NEXT BYTE
3311	013112	005200	INC	R0	:BUMP THE COUNT
3312	013114	000772	BR	10\$	:LOOP
3313	013116	012701	20\$: MOV	#1, R1	:SET THE TEST BIT
3314	013122	012702	30\$: MOV	#CHKTAB, R2	:START OF CHECK BIT TABLE
3315	013126	066702	ADD	BASE, R2	
3316	013132	012703	MOV	#CHKCNT, R3	:START ADDRESS OF PARITY COUNTERS
3317	013136	066703	ADD	BASE, R3	
3318	013142	012704	MOV	#6, R4	:FOR SEVEN PARITY COUNTERS
3319	013146	105067	40\$: CLRB	UPPFLG	:INDICATE LOWER WORD
3320	013152	016700	MOV	SOURCE, R0	:SET ADDRESS OF DATA IN R0
3321	013156	066700	ADD	BASE, R0	
3322	013162	030110	42\$: BIT	R1, (R0)	:CHECK LOWER 16 BITS FOR TEST BIT
3323	013164	001403	BEQ	50\$	:BRANCH IF NOT SET
3324	013166	030112	BIT	R1, (R2)	:LOOK FOR TEST BIT IN CHECK BIT TABLE
3325	013170	001401	BEQ	50\$	:BRANCH IF NOT FOUND
3326	013172	105213	INCB	(R3)	:FOUND A MATCH, COUNT IT
3327	013174	062700	50\$: ADD	#2, R0	:R0 NOW = SOURCE +2
3328	013200	062702	ADD	#2, R2	:POINT TO SECOND WORD IN TABLE

DZMPL MACY11 30(1046) 05-AUG-77 12:59 PAGE 1-65
 DZMPLB.P11 05-AUG-77 12:57 END OF PASS

SEQ 0067

3329	013204	105767	177642	TSTB	UPPFLG	:CHECK FOR UPPER/LOWER
3330	013210	001003		BNE	60\$	:BRANCH IF UPPER
3331	013212	105267	177634	INCB	UPPFLG	:CHANGE FLAG TO UPPER
3332	013216	000761		BR	42\$	:CHECK SECOND 16 BITS
3333	013220	005704		60\$: TST	R4	:LOOK FOR LAST PARITY COUNTER
3334	013222	001403		BEQ	70\$	:BRANCH IF DONE
3335	013224	005203		INC	R3	:ADVANCE PARITY COUNTER POINTER
3336	013226	005304		DEC	R4	:ADVANCE CHECK BIT FLAG COUNTER
3337	013230	000746		BR	40\$	:GO CHECK NEXT CHECK BIT IN TABLE
3338	013232	006301		70\$: ASL	R1	:SHIFT THE BIT OVER
3339	013234	103332		BCC	30\$	:ADJUST THE PARITY AND ASSEMBLE
3340						:CHECK BIT WORD
3341	013236	012700	000001	MOV	#1,R0	:SET FIRST BIT IN R1
3342	013242	012701	013053	MOV	#CHKCNT,R1	:ADDRESS OF C1 PARITY COUNTER
3343	013246	066701	167446	ADD	BASE,R1	
3344	013252	130067	177604	80\$: BITB	R0,PARBYT	:CHECKING FOR ODD OR EVEN PARITY
3345	013256	001401		BEQ	90\$	:BRANCH IF EVEN PARITY
3346	013260	105211		INCB	(R1)	:ODD, INC THE PARITY COUNTER
3347	013262	132711	000001	90\$: BITB	#1,(R1)	:TEST PARITY COUNTER FOR SET BIT
3348	013266	001401		BEQ	100\$	
3349	013270	050004		BIS	R0,R4	:SET THE CHECK BIT IN CHECK
3350	013272	005201		100\$: INC	R1	:NEXT COUNTER
3351	013274	006300		ASL	R0	:SHIFT TEST BIT TO NEXT BIT
3352	013276	032700	000200	BIT	#200,R0	:CT YET?
3353	013302	001763		BEQ	80\$	:BRANCH IF NO
3354	013304	000304		SWAB	R4	
3355	013306	006004		ROR	R4	
3356	013310	006004		ROR	R4	
3357	013312	006004		ROR	R4	:POSITION CHECK BITS IN BITS 11-5
3358	013314	010467	177474	MOV	R4,CHECK	
3359	013320	004767	000014	JSR	PC,RSTOT4	:RESTORE R0 TO R4
3360	013324	000207		RTS	PC	
3361						
3362						
3363				:CALL	JSR R4,SAVOT4	
3364	013326	010346		SAVOT4: MOV	R3,-(SP)	:SAVE R3
3365	013330	010246		MOV	R2,-(SP)	:SAVE R2
3366	013332	010146		MOV	R1,-(SP)	:SAVE R1
3367	013334	010046		MOV	R0,-(SP)	:SAVE R0
3368	013336	010407		MOV	R4,PC	:R4 IS ALREADY SAVED
3369						
3370				:CALL	JSR PC,RSTOT4	
3371	013340	012604		RSTOT4: MOV	(SP)+,R4	:RETURN ADDRESS
3372	013342	012600		MOV	(SP)+,R0	:RESTORE R0
3373	013344	012601		MOV	(SP)+,R1	:R1
3374	013346	012602		MOV	(SP)+,R2	:R2
3375	013350	012603		MOV	(SP)+,R3	:R3
3376	013352	000204		RTS	R4	:RESTORE R4 AND RETURN
3377						
3378						
3379						
3380						
3381						
3382	013354	010046		DASHL: MOV	R0,-(SP)	:SAVE R0
3383	013356	012500		MOV	(R5)+,R0	:PICK UP ARGUMENT
3384	013360	066700	167334	ADD	BASE,R0	:

DZMPL MACY11 30(1046) 05-AUG-77 12:59 PAGE 1-66
 DZMPLB.P11 05-AUG-77 12:57 END OF PASS

SEQ 0068

```

3385 013364 006360 000002      ASL      2(R0)      ;SHIFT HIGH 16 BITS
3386 013370 006310              ASL      (R0)      ;SHIFT LOW 16 BITS
3387 013372 103002              BCC      10$      ;BR IF LOW WORD BIT 15 = 0
3388 013374 005260 000002      INC      2(R0)      ;IT WAS = 1 INC HI WORD
3389 013400 012600              MOV      (SP)+,R0    ;RESTORE R0
3390 013402 000205              RTS      R5
3391
3392 013404 012567 000112      BITCOM: MOV      (R5)+,MSKADR ;FIRST ARG IS ADDR OF MASK
3393 013410 012567 000110      MOV      (R5)+,DATADR ;SECOND IS ADDR OF DATA
3394 013414 010046              MOV      R0,-(SP)
3395 013416 010146              MOV      R1,-(SP)
3396 013420 010246              MOV      R2,-(SP)
3397 013422 066767 167272 000072 ADD      BASE,MSKADR ;SAVE THESE REGS
3398 013430 066767 167264 000066 ADD      BASE,DATADR
3399 013436 017700 000060      MOV      2MSKADR,R0 ;GET MASK
3400 013442 017701 000056      MOV      2DATADR,R1 ;GET DATA
3401 013446 004767 000054      JSR      PC,XOR ;DO THE XOR
3402 013452 010277 000046      MOV      R2,2DATADR ;PUT RESULT BACK AS DATA
3403 013456 062767 000002 000036 ADD      #2,MSKADR ;NEXT WORD
3404 013464 062767 000002 000032 ADD      #2,DATADR ;DITTO
3405 013472 017700 000024      MOV      2MSKADR,R0
3406 013476 017701 000022      MOV      2DATADR,R1
3407 013502 004767 000020      JSR      PC,XOR ;DO THE SAME FOR SECOND WORD
3408 013506 010277 000012      MOV      R2,2DATADR ;PUT RESULT BACK TOO
3409 013512 012602              MOV      (SP)+,R2
3410 013514 012601              MOV      (SP)+,R1
3411 013516 012600              MOV      (SP)+,R0
3412 013520 000205              RTS      R5
3413
3414
3415 013522 000000              MSKADR: .WORD 0
3416 013524 000000              DATADR: .WORD 0
3417
3418
3419
3420
3421
3422
3423
3424
3425 013526 010102              XOR:  MOV      R1,R2 ;SAVE B OPERAND
3426 013530 040002              BIC      R0,R2 ;A NOT B
3427 013532 040100              BIC      R1,R0 ;B NOT A
3428 013534 050002              BIS      R0,R2 ;A XOR B
3429 013536 000207              RTS      PC ;EXIT AND RETURN
3430
3431

```

;;*****


```

3436      * ERROR HANDLING ROUTINE
3437      * -----
3438      *
3439      * PROGRAM COMES HERE EACH TIME AN ERROR IS ENCOUNTERED THIS
3440      * ROUTINE TYPES OUT THE ERROR MESSAGE IN THE FORMAT GIVEN EARLIER
3441
3442
3443 013540 017637 000000 000402 ERROR: MOV      2(SP), 2#SFATAL ;LOAD THE LOCATION SFATAL WITH THE ERROR NUMBER
3444 013546 005767 167152          TST      ERRFLG      ;FIRST ERROR ?
3445 013552 001050          BNE      1$              ;BR IF NOT FIRST ERROR
3446 013554 032777 020000 167150 BIT      20000, 2SWR ;ERROR PRINTOUTS DESIRED ?
3447 013562 001044          BNE      1$              ;BR IN NO
3448 013564 004767 000642          JSR      PC, TPCRLF ;TYPE ERROR HEADER FOR FIRST ERROR
3449 013570 040440 042104 020122 .ASCII / ADDR GOOD BAD PC ERR # PASFLG XOR /
3450 013576 020040 020040 047507
3451 013604 042117 020040 020040
3452 013612 041040 042101 020040
3453 013620 020040 020040 050040
3454 013626 020103 020040 020040
3455 013634 051105 020122 020043
3456 013642 020040 050040 051501
3457 013650 046106 020107 020040
3458 013656 054040 051117 020040
3459 013664 005015 000          .ASCIIZ (15)(12)
3460 013670 005267 167030          EVEN
3461 013674 010346          INC      ERRFLG      ;INDICATE NOT FIRST ERROR
3462 013676 010046          MOV      R3, -(SP) ;SAVE R3
3463                                     MOV      R0, -(SP) ;AND R0 ON THE STACK
3464                                     ;SETUP BANK NO. IN FATAL FOR APT
3465
3466 013700 010103          MOV      R1, R3      ;GET VIRTUAL ADDRESS UNDER TEST FOR GETBNK
3467 013702 004767 001420          JSR      PC, GETBNK ;GET BANK NO. UNDER TEST INTO PBNK
3468 013706 013703 000306          MOV      2#PBANK, R3 ;GET BANK UNDER TEST
3469 013712 110337 000403          MOVB    R3, 2#SFATAL+1 ;STORE FAILING BANK NO. FOR APT
3470
3471 ;
3472
3473 013716 010346          MOV      R3, -(SP) ;TEMPORARILY STORE R3
3474 013720 012703 000376          MOV      2#376, R3 ;MAKE R3 AS THE STACK POINTER
3475 013724 013743 000302          MOV      2#PASFLG, -(R3) ;OUTPUT THE WORD STORED AT
3476 013730 005043          CLR      -(R3) ;
3477 013732 113713 000402          MOVB    2#SFATAL, (R3) ;PUT ERROR NO. ON ERROR STACK
3478 013736 016643 000006          MOV      6(SP), -(R3) ;PLACE THE RETURN PC AT (R3)
3479 013742 011143          MOV      (R1), -(R3) ;PLACE BAD DATA
3480 013744 010043          MOV      R0, -(R3) ;AND GOOD DATA ON THE STACK
3481 013746 005043          CLR      -(R3) ;
3482 013750 016313 000004          MOV      4(R3), (R3) ;TAKE THE
3483 013754 040013          BIC      R0, (R3) ;EXCLUSIVE OR OF GOOD AND BAD DATA
3484 013756 046300 000004          BIC      4(R3), R0 ;TO FIND THE BITS THAT FAILED
3485 013762 050013          BIS      R0, (R3) ;AND PLACE IT ON THE STACK
3486 013764 011367 000744          MOV      (R3), BDCHIP ;TO PRINT OUT XOR OF GOOD VS BAD
3487 013770 012700 002112          MOV      2#ENDPROG--24., R0 ;THIS CODE BRINGS THE RELATIVE ADDRESS
3488 013774 060700          ADD      PC, R0 ;OF THE STARTING OF THE ERROR STACK
3489 013776 062700          ADD      2#18., R0 ;FOR THE SPECIFIC 4K BANK
3490 014002 005316          DEC      (SP)

```

3483	014004	002374		BGE	6S		
3484	014006	005726		TST	(SP)+		;RESTORE THE STACK POINTER
3485							
3486	014010	105037	000273	ERRTYP:	CLRB	2#TYPENB	;DISABLE ANY TYPE OUT
3487	014014	105737	000274	1S:	TSTB	2#SPRERR	;IF THIS IS PARITY PROBLEM
3488	014020	001007		BNE	3S		;THEN GO TO 3S
3489	014022	105720		TSTB	(R0)+		;OTHERWISE INCREMENT THE ERROR STACK POINTER BY 1
3490	014024	105737	000275	TSTB	2#SADERR		;IF THIS IS ADDRESSING PROBLEM
3491	014030	001003		BNE	3S		;THEN GO TO 3S
3492	014032	105720		TSTB	(R0)+		;INCREMENT THE POINTER R0 BY 1
3493	014034	005713		2S:	TST	(R3)	;IS BIT 15 OF (R3) SET?
3494	014036	100015		BPL	4S		;IF NOT THEN GO TO 4S
3495	014040	122710	000377	3S:	CMPB	#377,(R0)	;OTHERWISE SEE IF THIS ERROR HAS OCCURED 377 TIMES
3496	014044	001401		BEQ	5S		;IF SO DON'T BUMP ERROR COUNT
3497	014046	105210		INCB	(R0)		;INCREMENT THE ERROR COUNTER BY 1
3498	014050	122710	000001	5S:	CMPB	#1,(R0)	;MORE THAN 1 ERROR OCCURRED ON THIS BIT?
3499	014054	001404		BEQ	7S		;BRANCH IF NO
3500	014056	032777	000400	166646	BIT	#400,2SWR	;STOP ERROR PRINTOUT AFTER 1 WANTED?
3501	014064	001002		BNE	4S		;BRANCH IF YES (DON'T TYPE ERROR)
3502	014066	105237	000273	7S:	INCB	2#TYPENB	;ENABLE THE TYPE OUT ROUTINE
3503	014072	105737	000275	4S:	TSTB	2#SADERR	;ADDRESS ERROR?
3504	014076	001403		BEQ	6S		;BRANCH IF NO
3505	014100	004767	000166		JSR	PC,TPADERR	;PRINT "ADR ERR"
3506	014104	000403		BR	8S		
3507	014106	105720		6S:	TSTB	(R0)+	;POINT TO NEXT ENTRY IN ERROR STACK
3508	014110	006313		ASL	(R3)		;IS THERE STILL AN ERROR BIT SET IN ERROR.
3509	014112	001350		BNE	2S		;BR IF YES - KEEP FILLING ERROR STACK
3510	014114	112737	000006	000310	8S:	MOVB	#6,2#TYPCNT
3511							;TELL TYPOCT TO TYPE 6 WORDS OF ERROR STACK.
3512	014122	004767	000774		JSR	PC,PUTADR	;THE STACK POINTED BY R3
3513							;GO TO THE SUBROUTINE TO PLACE THE ADDRESS IN R1
3514	014126	004767	000416		JSR	PC,TYPERR	;AT LOCATIONS (R3) AND (R3-2)
3515							;TYPE ERROR STACK (7 WORDS)
3516	014132	005037	000274	10S:	CLR	2#SPRERR	;CLEAR ADDRESS ERROR FLAG
3517	014136	012600		MOV	(SP)+,R0		;RESTORE R0
3518	014140	012603		MOV	(SP)+,R3		;AND R3
3519	014142	105737	000420	FNDERR:	TSTB	2#SENV	;ARE WE RUNNING UNDER APT?
3520	014146	001404		BEQ	2S		;IF NOT THEN TEST FOR HALT
3521	014150	012737	000001	000400	MOV	#1,2#MSGTY	;OTHERWISE INFORM THE APT
3522	014156	000443		BR	FATHLT		;GOTO FATHLT AND WAIT FOR APT.
3523							
3524	014160	010246		2S:	MOV	R2,-(SP)	;SAVE R2 TEMP
3525	014162	005777	166544		TST	2SWR	;DOES THE OPERATOR REQUIRE THE PROGRAM TO HALT
3526							;ON ERROR
3527	014166	100405			BMI	4S	;IF SO THEN HALT ON ERROR
3528							
3529							;CHECK FOR CONTROL-C KEY
3530	014170	004767	001660		JSR	PC,CHECKC	;IF CONTROL-C TYPED THEN PRINT ERROR HISTORY
3531							;AND HALT AT FATHLT.
3532	014174	105737	000042	7S:	TSTB	2#42	;ARE WE RUNNING UNDER ACT?
3533	014200	001401		BEQ	6S		;BRANCH IF NO
3534							
3535	014202	000777		4S:	BR	4S	;PROGRAM HAS HALTED ON ERROR, R1 IS POINTING
3536							;TO A LOCATION WHICH SHOULD HAVE CONTAINED
3537							;THE WORD STORED IN R0
3538	014204	012602		6S:	MOV	(SP)+,R2	;RESTORE R2

Address	Hex 1	Hex 2	Hex 3	Hex 4	Label	Operation	Comments
3539	014206	062716	000002		ADD	#2, (SP)	;RESTORE THE RETURN ADDRESS
3540	014212	000207			RTS	PC	;RETURN FROM THE SUBROUTINE
3541							
3542							
3543							
3544	014214				FATERR:		
3545	014214	004767	000212		SEQERR:	JSR PC,TPCRLF	;TYPE "ERROR #"
3546	014220	051105	047522	020122		.ASCIZ /ERROR #/	
3547	014226	000043				.EVEN	
3548							
3549	014230	017637	000000	000402		MOV #2(SP),#SFATAL	;LOAD THE LOCATION SFATAL WITH THE ERROR NUMBER
3550	014236	105237	000310			INCB #STPCNT	;TELL STPNUM TO TYPE 1 WORD
3551	014242	012703	000376			MOV #376,R3	;STPNUM USES R3 AS STACK
3552	014246	013743	000402			MOV #SFATAL,-(R3)	;PUT ERROR NO. ON STACK
3553	014252	005743				TST -(R3)	;STPNUM REQUIRES THIS
3554	014254	004767	000356			JSR PC,FATYP	;TYPE ERROR NO.
3555	014260	105737	000420		APTHLT:	TSTB #SENV	;RUNNING UNDER APT?
3556	014264	001326				BNE FNDERR	;BRANCH IF YES
3557	014266	000240			FATHLT:	NOP	;FATAL ERROR OR IC HALT.
3558	014270	000776				BR FATHLT	;RESTART AT 250 IF DESIRED
3559							
3560	014272	105737	000273		TPADER:	TSTB #TYPEENB	
3561	014276	001406				BEQ 1\$	;BR IF NO TYPE ENABLED
3562	014300	004767	000134			JSR PC,PNTMES	;PRINT MESSAGE ROUTINE
3563	014304	042101	020122	051105		.ASCIZ /ADR ERR/	
3564	014312	000122				.EVEN	
3565	014314	000207			1\$:	RTS PC	
3566							
3567							
3568							
3569							
3570							
3571							
3572							
3573							
3574							
3575	014316	010146			NOTYP:	MOV R1,-(SP)	
3576	014320	016601	000002			MOV 2(SP),R1	
3577	014324	105721			4\$:	TSTB (R1)+	;IF THIS TYPE OUT HAS BEEN SUPRESSED THEN
3578	014326	001376				BNE 4\$	;PREPARE TO RETURN
3579	014330	000412				BR RETTYP	
3580	014332	010146			STYPE:	MOV R1,-(SP)	;SAVE R1
3581	014334	010046				MOV R0,-(SP)	;AND R0 ON THE STACK
3582	014336	016601	000004			MOV 4(SP),R1	;PLACE THE ADDRESS OF MESSAGE TO BE TYPED IN R1
3583	014342	112100			2\$:	MOVB (R1)+,R0	;PLACE THE BYTE TO BE TYPED IN R0
3584	014344	001403				BEQ 4\$	;IF IT IS END OF MESSAGE THEN GO TO 4\$
3585	014346	004767	000022			JSR PC,STPCHR	;OTHERWISE GO TO TYPE THE CONTENTS OF R0
3586	014352	000773				BR 2\$	
3587	014354	012600			4\$:	MOV (SP)+,R0	;RESTORE R0
3588	014356	005201			RETTYP:	INC R1	;CAUSE R1 TO
3589	014360	042701	000001			BIC #1,R1	;POINT TO EVEN ADDRESS
3590	014364	010166	000002			MOV R1,2(SP)	;MODIFY THE RETURN ADDRESS
3591	014370	012601				MOV (SP)+,R1	;RESTORE R1
3592	014372	000416				BR EXTYP	;AND RETURN VIA RTS PC

G06

DZMML MACY11 30(1046) 05-AUG-77 12:59 PAGE 1-70
 DZMMLB.P11 05-AUG-77 12:57 ERROR HANDLING ROUTINE

SEQ 0072

```

3593
3594 014374 132737 000040 000421 STPCHR: BITB #40,2#SENVN ;HAVE TYPE OUTS BEEN DISABLED?
3595 014402 001005 BNE 4$ ;IF SO THEN RETURN FROM THE SUBROUTINE
3596 014404 105737 177564 2$: TSTB 2#STPS ;WAIT HERE
3597 014410 100375 BPL 2$ ;UNTIL THE PRINTER IS READY
3598 014412 110037 177566 MOVB RD,2#STPB ;LOAD DATA TO BE TYPED INTO DATA REG.
3599 014416 000404 4$: BR EXTYP ;RETURN
3600
3601 014420 004767 177706 PCRLF: JSR PC,STYPE
3602 014424 005015 000 .ASCIZ <15><12> ;CR/LF
3603 014430 000207 EXTYP: RTS PC ;RETURN
3604
3605 014432 004767 177762 TPCRLF: JSR PC,PCRLF
3606 014436 000735 BR STYPE ;TYPE CR/LF
3607 ;NOW GO TO TYPE THE REST OF THE MESSAGE
3608
3609
3610 014440 032777 000020 166264 PNTMES: BIT #20,2$WR ;PRINTOUTS ALLOWED?
3611 014446 001323 BNE NOTYP ;BRANCH IF NO
3612 014450 123737 000042 000046 CMPB 2#42,2#46 ;RUNNING UNDER ACT 11?
3613 014456 001717 BEQ NOTYP ;BRANCH IF YES -NOT PRINTOUT-
3614 014460 000764 BR TPCRLF ;SEND CR/LF AND TYPE MESSAGE.

```


H06

DZMML MACY11 30(1046) 05-AUG-77 12:59 PAGE 1-71
 DZMMLB.P11 05-AUG-77 12:57 ROUTINE TO TYPE OUT A DECIMAL NUMBER

SEQ 0073

```

3619
3620
3621
3622
3623
3624
3625
3626
3627 014462 004767 177732      TYPDEC: JSR      PC,PCRLF      ;TYPE CR/LF
3628
3629 014466 005046
3630 014470 013746 000306      STPDEC: CLR      -(SP)
3631
3632 014474 162716 000012      MOV      @#DECWRD,-(SP) ;GET THE WORD THAT HAS TO BE CONVERTED TO A
3633 014500 002403
3634
3635 014502 005266 000002      2$: SUB      #10.,(SP) ;DECIMAL NUMBER
3636 014506 000772
3637 014510 062716 000012      BLT      4$           ;IF THE NUMBER IN (SP) WAS LESS THAN 10. THEN
3638 014514 052716 000060      INC      2(SP)       ;GO TO 4$
3639 014520 112667 000020      BR       2$          ;OTHERWISE ADD 1 TO THE LOCATION STORING 10'S DIGIT
3640 014524 052716 000060      4$: ADD      #10.,(SP) ;AND RETURN TO 2$
3641 014530 112667 000007      BIS      #60,(SP)     ;MAKE THE CONTENTS OF (SP) A DECIMAL NUMBER
3642 014534 004767 177572      MOVB     (SP)+,6$-2   ;PLACE THE 1'S DIGIT TO BE TYPED
3643
3644 014540 020040 030040 000060  BIS      #60,(SP)     ;MAKE THE CONTENTS OF (SP) A DECIMAL NUMBER
3645
3646 014546 000207
3647
3648
3649
3650
3651
3652
3653
3654
3655
3656
3657
3658
3659
3660
3661
3662
3663
3664
3665
3666
3667
3668
3669
3670
3671
3672
3673
3674
3675
3676
3677
3678
3679
3680
3681
3682
3683
3684
3685
3686
3687
3688
3689
3690
3691
3692
3693
3694
3695
3696
3697
3698
3699
3700
3701
3702
3703
3704
3705
3706
3707
3708
3709
3710
3711
3712
3713
3714
3715
3716
3717
3718
3719
3720
3721
3722
3723
3724
3725
3726
3727
3728
3729
3730
3731
3732
3733
3734
3735
3736
3737
3738
3739
3740
3741
3742
3743
3744
3745
3746
3747
3748
3749
3750
3751
3752
3753
3754
3755
3756
3757
3758
3759
3760
3761
3762
3763
3764
3765
3766
3767
3768
3769
3770
3771
3772
3773
3774
3775
3776
3777
3778
3779
3780
3781
3782
3783
3784
3785
3786
3787
3788
3789
3790
3791
3792
3793
3794
3795
3796
3797
3798
3799
3800
3801
3802
3803
3804
3805
3806
3807
3808
3809
3810
3811
3812
3813
3814
3815
3816
3817
3818
3819
3820
3821
3822
3823
3824
3825
3826
3827
3828
3829
3830
3831
3832
3833
3834
3835
3836
3837
3838
3839
3840
3841
3842
3843
3844
3845
3846
3847
3848
3849
3850
3851
3852
3853
3854
3855
3856
3857
3858
3859
3860
3861
3862
3863
3864
3865
3866
3867
3868
3869
3870
3871
3872
3873
3874
3875
3876
3877
3878
3879
3880
3881
3882
3883
3884
3885
3886
3887
3888
3889
3890
3891
3892
3893
3894
3895
3896
3897
3898
3899
3900
3901
3902
3903
3904
3905
3906
3907
3908
3909
3910
3911
3912
3913
3914
3915
3916
3917
3918
3919
3920
3921
3922
3923
3924
3925
3926
3927
3928
3929
3930
3931
3932
3933
3934
3935
3936
3937
3938
3939
3940
3941
3942
3943
3944
3945
3946
3947
3948
3949
3950
3951
3952
3953
3954
3955
3956
3957
3958
3959
3960
3961
3962
3963
3964
3965
3966
3967
3968
3969
3970
3971
3972
3973
3974
3975
3976
3977
3978
3979
3980
3981
3982
3983
3984
3985
3986
3987
3988
3989
3990
3991
3992
3993
3994
3995
3996
3997
3998
3999
4000
4001
4002
4003
4004
4005
4006
4007
4008
4009
4010
4011
4012
4013
4014
4015
4016
4017
4018
4019
4020
4021
4022
4023
4024
4025
4026
4027
4028
4029
4030
4031
4032
4033
4034
4035
4036
4037
4038
4039
4040
4041
4042
4043
4044
4045
4046
4047
4048
4049
4050
4051
4052
4053
4054
4055
4056
4057
4058
4059
4060
4061
4062
4063
4064
4065
4066
4067
4068
4069
4070
4071
4072
4073
4074
4075
4076
4077
4078
4079
4080
4081
4082
4083
4084
4085
4086
4087
4088
4089
4090
4091
4092
4093
4094
4095
4096
4097
4098
4099
4100
4101
4102
4103
4104
4105
4106
4107
4108
4109
4110
4111
4112
4113
4114
4115
4116
4117
4118
4119
4120
4121
4122
4123
4124
4125
4126
4127
4128
4129
4130
4131
4132
4133
4134
4135
4136
4137
4138
4139
4140
4141
4142
4143
4144
4145
4146
4147
4148
4149
4150
4151
4152
4153
4154
4155
4156
4157
4158
4159
4160
4161
4162
4163
4164
4165
4166
4167
4168
4169
4170
4171
4172
4173
4174
4175
4176
4177
4178
4179
4180
4181
4182
4183
4184
4185
4186
4187
4188
4189
4190
4191
4192
4193
4194
4195
4196
4197
4198
4199
4200
4201
4202
4203
4204
4205
4206
4207
4208
4209
4210
4211
4212
4213
4214
4215
4216
4217
4218
4219
4220
4221
4222
4223
4224
4225
4226
4227
4228
4229
4230
4231
4232
4233
4234
4235
4236
4237
4238
4239
4240
4241
4242
4243
4244
4245
4246
4247
4248
4249
4250
4251
4252
4253
4254
4255
4256
4257
4258
4259
4260
4261
4262
4263
4264
4265
4266
4267
4268
4269
4270
4271
4272
4273
4274
4275
4276
4277
4278
4279
4280
4281
4282
4283
4284
4285
4286
4287
4288
4289
4290
4291
4292
4293
4294
4295
4296
4297
4298
4299
4300
4301
4302
4303
4304
4305
4306
4307
4308
4309
4310
4311
4312
4313
4314
4315
4316
4317
4318
4319
4320
4321
4322
4323
4324
4325
4326
4327
4328
4329
4330
4331
4332
4333
4334
4335
4336
4337
4338
4339
4340
4341
4342
4343
4344
4345
4346
4347
4348
4349
4350
4351
4352
4353
4354
4355
4356
4357
4358
4359
4360
4361
4362
4363
4364
4365
4366
4367
4368
4369
4370
4371
4372
4373
4374
4375
4376
4377
4378
4379
4380
4381
4382
4383
4384
4385
4386
4387
4388
4389
4390
4391
4392
4393
4394
4395
4396
4397
4398
4399
4400
4401
4402
4403
4404
4405
4406
4407
4408
4409
4410
4411
4412
4413
4414
4415
4416
4417
4418
4419
4420
4421
4422
4423
4424
4425
4426
4427
4428
4429
4430
4431
4432
4433
4434
4435
4436
4437
4438
4439
4440
4441
4442
4443
4444
4445
4446
4447
4448
4449
4450
4451
4452
4453
4454
4455
4456
4457
4458
4459
4460
4461
4462
4463
4464
4465
4466
4467
4468
4469
4470
4471
4472
4473
4474
4475
4476
4477
4478
4479
4480
4481
4482
4483
4484
4485
4486
4487
4488
4489
4490
4491
4492
4493
4494
4495
4496
4497
4498
4499
4500
4501
4502
4503
4504
4505
4506
4507
4508
4509
4510
4511
4512
4513
4514
4515
4516
4517
4518
4519
4520
4521
4522
4523
4524
4525
4526
4527
4528
4529
4530
4531
4532
4533
4534
4535
4536
4537
4538
4539
4540
4541
4542
4543
4544
4545
4546
4547
4548
4549
4550
4551
4552
4553
4554
4555
4556
4557
4558
4559
4560
4561
4562
4563
4564
4565
4566
4567
4568
4569
4570
4571
4572
4573
4574
4575
4576
4577
4578
4579
4580
4581
4582
4583
4584
4585
4586
4587
4588
4589
4590
4591
4592
4593
4594
4595
4596
4597
4598
4599
4600
4601
4602
4603
4604
4605
4606
4607
4608
4609
4610
4611
4612
4613
4614
4615
4616
4617
4618
4619
4620
4621
4622
4623
4624
4625
4626
4627
4628
4629
4630
4631
4632
4633
4634
4635
4636
4637
4638
4639
4640
4641
4642
4643
4644
4645
4646
4647
4648
4649
4650
4651
4652
4653
4654
4655
4656
4657
4658
4659
4660
4661
4662
4663
4664
4665
4666
4667
4668
4669
4670
4671
4672
4673
4674
4675
4676
4677
4678
4679
4680
4681
4682
4683
4684
4685
4686
4687
4688
4689
4690
4691
4692
4693
4694
4695
4696
4697
4698
4699
4700
4701
4702
4703
4704
4705
4706
4707
4708
4709
4710
4711
4712
4713
4714
4715
4716
4717
4718
4719
4720
4721
4722
4723
4724
4725
4726
4727
4728
4729
4730
4731
4732
4733
4734
4735
4736
4737
4738
4739
4740
4741
4742
4743
4744
4745
4746
4747
4748
4749
4750
4751
4752
4753
4754
4755
4756
4757
4758
4759
4760
4761
4762
4763
4764
4765
4766
4767
4768
4769
4770
4771
4772
4773
4774
4775
4776
4777
4778
4779
4780
4781
4782
4783
4784
4785
4786
4787
4788
4789
4790
4791
4792
4793
4794
4795
4796
4797
4798
4799
4800
4801
4802
4803
4804
4805
4806
4807
4808
4809
4810
4811
4812
4813
4814
4815
4816
4817
4818
4819
4820
4821
4822
4823
4824
4825
4826
4827
4828
4829
4830
4831
4832
4833
4834
4835
4836
4837
4838
4839
4840
4841
4842
4843
4844
4845
4846
4847
4848
4849
4850
4851
4852
4853
4854
4855
4856
4857
4858
4859
4860
4861
4862
4863
4864
4865
4866
4867
4868
4869
4870
4871
4872
4873
4874
4875
4876
4877
4878
4879
4880
4881
4882
4883
4884
4885
4886
4887
4888
4889
4890
4891
4892
4893
4894
4895
4896
4897
4898
4899
4900
4901
4902
4903
4904
4905
4906
4907
4908
4909
4910
4911
4912
4913
4914
4915
4916
4917
4918
4919
4920
4921
4922
4923
4924
4925
4926
4927
4928
4929
4930
4931
4932
4933
4934
4935
4936
4937
4938
4939
4940
4941
4942
4943
4944
4945
4946
4947
4948
4949
4950
4951
4952
4953
4954
4955
4956
4957
4958
4959
4960
4961
4962
4963
4964
4965
4966
4967
4968
4969
4970
4971
4972
4973
4974
4975
4976
4977
4978
4979
4980
4981
4982
4983
4984
4985
4986
4987
4988
4989
4990
4991
4992
4993
4994
4995
4996
4997
4998
4999
5000
5001
5002
5003
5004
5005
5006
5007
5008
5009
5010
5011
5012
5013
5014
5015
5016
5017
5018
5019
5020
5021
5022
5023
5024
5025
5026
5027
5028
5029
5030
5031
5032
5033
5034
5035
5036
5037
5038
5039
5040
5041
5042
5043
5044
5045
5046
5047
5048
5049
5050
5051
5052
5053
5054
5055
5056
5057
5058
5059
5060
5061
5062
5063
5064
5065
5066
5067
5068
5069
5070
5071
5072
5073
5074
5075
5076
5077
5078
5079
5080
5081
5082
5083
5084
5085
5086
5087
5088
5089
5090
5091
5092
5093
5094
5095
5096
5097
5098
5099
5100
5101
5102
5103
5104
5105
5106
5107
5108
5109
5110
5111
5112
5113
5114
5115
5116
5117
5118
5119
5120
5121
5122
5123
5124
5125
5126
5127
5128
5129
5130
5131
5132
5133
5134
5135
5136
5137
5138
5139
5140
5141
5142
5143
5144
5145
5146
5147
5148
5149
5150
5151
5152
5153
5154
5155
5156
5157
5158
5159
5160
5161
5162
5163
5164
5165
5166
5167
5168
5169
5170
5171
5172
5173
5174
5175
5176
5177
5178
5179
5180
5181
5182
5183
5184
5185
5186
5187
5188
5189
5190
5191
5192
5193
5194
5195
5196
5197
5198
5199
5200
5201
5202
5203
5204
5205
5206
5207
5208
5209
5210
5211
5212
5213
5214
5215
5216
5217
5218
5219
5220
5221
5222
5223
5224
5225
5226
5227
5228
5229
5230
5231
5232
5233
5234
5235
5236
5237
5238
5239
5240
5241
5242
5243
5244
5245
5246
5247
5248
5249
5250
5251
5252
5253
5254
5255
5256
5257
5258
5259
5260
5261
5262
5263
5264
5265
5266
5267
5268
5269
5270
5271
5272
5273
5274
5275
5276
5277
5278
5279
5280
5281
5282
5283
5284
5285
5286
5287
5288
5289
5290
5291
5292
5293
5294
5295
5296
5297
5298
5299
5300
5301
5302
5303
5304
5305
5306
5307
5308
5309
5310
5311
5312
5313
5314
5315
5316
5317
5318
5319
5320
5321
5322
5323
5324
5325
5326
5327
5328
5329
5330
5331
5332
5333
5334
5335
5336
5337
5338
5339
5340
5341
5342
5343
5344
5345
5346
5347
5348
5349
5350
5351
5352
5353
5354
5355
5356
5357
5358
5359
5360
5361
5362
5363
5364
5365
5366
5367
5368
5369
5370
5371
5372
5373
5374
5375
5376
5377
5378
5379
5380
5381
5382
5383
5384
5385
5386
5387
5388
5389
5390
5391
5392
5393
5394
5395
5396
5397
5398
5399
5400
5401
5402
5403
5404
5405
5406
5407
5408
5409
5410
5411
5412
5413
5414
5415
5416
5417
5418
5419
5420
5421
5422
5423
5424
5425
5426
5427
5428
5429
5430
5431
5432
5433
5434
5435
5436
5437
5438
5439
5440
5441
5442
5443
5444
5445
5446
5447
5448
5449
5450
5451
5452
5453
5454
5455
5456
5457
5458
5459
5460
5461
5462
5463
5464
5465
5466
5467
5468
5469
5470
5471
5472
5473
5474
5475
5476
5477
5478
5479
5480
5481
5482
5483
5484
5485
5486
5487
5488
5489
5490
5491
5492
5493
5494
5495
5496
5497
5498
5499
5500
5501
5502
5503
5504
5505
5506
5507
5508
5509
5510
5511
5512
5513
5514
5515
5516
5517
5518
5519
5520
5521
5522
5523
5524
5525
5526
5527
5528
5529
5530
5531
5532
5533
5534
5535
5536
5537
5538
5539
5540
5541
5542
5543
5544
5545
5546
5547
5548
5549
5550
5551
5552
5553
5554
5555
5556
5557
5558
5559
5560
5561
5562
5563
5564
5565
5566
5567
5568
5569
5570
5571
5572
5573
5574
5575
5576
5577
5578
5579
5580
5581
5582
5583
5584
5585
5586
5587
5588
5589
5590
5591
5592
5593
5594
5595
5596
5597
5598
5599
5600
5601
5602
5603
5604
5605
5606
5607
5608
5609
5610
5611
5612
5613
5614
5615
5616
5617
5618
5619
5620
5621
5622
5623
5624
5625
5626
5627
5628
5629
5630
5631
5632
5633
5634
5635
5636
5637
5638
5639
5640
5641
5642
5643
5644
5645
5646
5647
5648
5649
5650
5651
5652
5653
5654
5655
5656
5657
5658
5659
5660
5661
5662
5663
5664
5665
5666
5667
5668
5669
5670
5671
5672
5673
5674
5675
5676
5677
5678
5679
5680
5681
5682
5683
5684
5685
5686
5687
5688
5689
5690
5691
5692
5693
5694
5695
5696
5697
5698
5699
5700
5701
5702
5703
5704
5705
5706
5707
5708
5709
5710
5711
5712
5713
5714
5715
5716
5717
5718
5719
5720
5721
5722
5723
5724
5725
5726
5727
5728
5729
5730
5731
5732
5733
5734
5735
5736
5737
5738
5739
5740
5741
5742
5743
5744
5745
5746
5747
5748
5749
5750
5751
5752
5753
5754
5755
5756
5757
5758
5759
5760
5761
5762
5763
5764
5765
5766
5767
5768
5769
5770
5771
5772
5773
5774
5775
5776
5777
5778
5779
5780
5781
5782
5783
5784
5785
5786
5787
5788
5789
5790
5791
5792
5793
5794
5795
5796
5797
5798
5799
5800
5801
5802
5803
5804
5805
5806
5807
5808
5809
5810
5811
5812
5813
5814
5815
5816
5817
5818
5819
5820
5821
5822
5823
5824
5825
5826
5827
5828
5829
5830
5831
5832
5833
5834
5835
5836
5837
5838
5839
5840
5841
5842
5843
5844
5845
5846
5847
5848
5849
5850
5851
5852
5853
5854
5855
5856
5857
5858
5859
5860
5861
5862
5863
5864
5865
5866
5867
5868
5869
5870
5871
5872
5873
5874
5875
5876
5877
5878
5879
5880
5881
5882
5883
5884
5885
5886
5887
5888
5889
5890
5891
5892
5893
5894
5895
5896
5897
5898
5899
5900
5901
5902
5903
5904
5905
5906
5907
5908
5909
5910
5911
5912
5913
5914
5915
5916
5917
5918
5919
5920
5921
5922
5923
5924
5925
5926
5927
5928
5929
5930
5931
5932
5933
5934
5935
5936
5937
5938
5939
5940
5941
5942
5943
5944
5945
5946
5947
5948
5949
5950
595
```

DZMPL MACY11 30(1046) 05-AUG-77 12:59 PAGE 1-72
 DZMPLB.P11 05-AUG-77 12:57 OCTAL TYPE OUT ROUTINE

SEQ 0074

```

3651
3652
3653
3654
3655
3656
3657
3658
3659
3660
3661
3662
3663
3664
3665
3666 014550 032777 020000 166154 TYPERR: BIT      #20000,JSWR      ;ERROR PRINTOUT WANTED?
3667 014556 001065                      BNE      OCTXT          ;BRANCH IF NO
3668 014560 004767 177634                JSR      PC,PCRLF        ;TYPE CR/LF
3669 014564 004767 000012                JSR      PC,TYPOCT       ;TYPE OCTAL NO.
3670 014570 000460                      BR       OCTXT          ;RETURN VIA RTS PC
3671 014572 012123                      OCTTYP: MOV      (R1)+,(R3)+  ;PLACE THE HIGH ORDER BITS AT LOCATION POINTED
3672                                     ;BY R3
3673 014574 012113                      MOV      (R1)+,(R3)    ;AND NOW PLACE THE LOW ORDER BITS
3674 014576 105237 000310                INCB     #TYPCNT      ;ENABLE THE TYPE OUT OF ONE OCTAL WORD
3675 014602 052743 000004                TYPOCT: BIS      #4,-(R3)
3676 014606 106113 25:                   ROLB     (R3)
3677 014610 103376                      BCC      25
3678 014612 005000                      CLR      RO
3679 014614 106113                      ROLB     (R3)          ;GET BITS 17 & 16 INTO RO
3680 014616 006100                      ROL      RO
3681 014620 106113                      ROLB     (R3)
3682 014622 006100                      ROL      RO
3683 014624 000405                      BR       $TPNUM
3684 014626 004767 177500                RPTOCT: JSR      PC,$TYPE
3685 014632 020040 000040                .ASCIZ  / /
3686                                     .EVEN
3687 014636 005000                      FATYP: CLR      RO
3688 014640 012723 000006                $TPNUM: MOV     #6,(R3)+  ;ENABLE THE TYPE OUT OF 6 OCTAL DIGITS
3689 014644 000241                      4$:      CLC
3690 014646 006113                      ROL      (R3)
3691 014650 006100                      ROL      RO
3692 014652 052700 000060                BIS      #60,RO      ;PLACE THE CARRY FROM (R3) IN RO
3693 014656 004767 177512                JSR      PC,$TPCHR    ;OR THE CONTENTS OF RO WITH AN ASCII 0
3694 014662 005000                      CLR      RO      ;TYPE THE OCTAL NUMBER STORED IN RO
3695 014664 006113                      ROL      (R3)
3696 014666 006100                      ROL      RO
3697 014670 006113                      ROL      (R3)
3698 014672 006100                      ROL      RO
3699 014674 105363 177776                DECB     -2(R3)      ;PLACE THE CARRY FROM (R3) IN RO
3700 014700 001361                      BNE      4$          ;IF WE HAVEN'T TYPED THE 6 OCTAL DIGITS
3701 014702 105337 000310                DECB     #TYPCNT      ;THEN REPEAT FROM 4$
3702 014706 100411                      BMI      OCTXT      ;IF ALL THE WORDS REQUIRED HAVE NOT BEEN
3703 014710 001346                      BNE      RPTOCT      ;BR IF = -1
3704 014712 032777 000200 166012        BIT      #200,JSWR    ;TYPED THEN REPEAT FROM RPTOCT
3705 014720 001404                      BEQ      OCTXT      ;PRINT OUT XOR
3706 014722 016743 000006                MOV      BDCHIP,-(R3) ;BR IF NOT WANTED
                                     ;PUT IN LAST USED STACK LOC TO BE PRINTED

```


J06

DZMML MACY11 30(1046) 05-AUG-77 12:59 PAGE 1-73
DZMMLB.P11 05-AUG-77 12:57 OCTAL TYPE OUT ROUTINE

SEQ 0075

3707 014726 005743
3708 014730 000736
3709 014732 000207
3710
3711 014734 000000

TST -(R3)
BR RPTOCT
OCTXT: RTS PC
BDCHIP: .WORD 0

;ADJUST POINTER
;PRINT LAST 6 OCTAL DIGITS

;XOR OF GOOD VS BAD DATA

K06

DZMML MACY11 30(1046) 05-AUG-77 12:59 PAGE 1-74
 DZMMLB.P11 05-AUG-77 12:57 SUBROUTINE FOR MEMORY MANAGEMENT

SEQ 0076

```

3716
3717
3718
3719
3720
3721
3722
3723
3724
3725 014736 012702 001400 MEMMNG: MOV #1400,R2
3726 014742 105037 000272 MMREG: CLRB @MMAVA
3727
3728 014746 032777 010000 165756 BIT #10000,@SWR
3729 014754 001043 BNE RETMM
3730 014756 012700 000004 MOV #4,R0
3731 014762 012720 015066 MOV @NOMM,(R0)+
3732 014766 012710 000340 MOV #340,(R0)
3733 014772 005037 177572 CLR @SRO
3734 014776 105237 000272 INCB @MMAVA
3735
3736 015002 012701 172340 MOV #172340,R1
3737 015006 005021 CLR (R1)+
3738 015010 012721 000200 MOV #200,(R1)+
3739 015014 062702 000200 2S: ADD #200,R2
3740 015020 010221 MOV R2,(R1)+
3741 015022 020127 172356 CMP R1,#172356
3742 015026 103772 BLO 2S
3743 015030 012711 007600 MOV #7600,(R1)
3744 015034 012701 172300 MOV #172300,R1
3745 015040 012721 077406 4S: MOV #77406,(R1)+
3746 015044 020127 172316 CMP R1,#172316
3747 015050 101773 BLOS 4S
3748 015052 005237 177572 INC @SRO
3749 015056 005010 $RETMM: CLR (R0)
3750 015060 012740 000352 MOV @BUSER,-(R0)
3751 015064 000207 RETMM: RTS PC
3752
3753 015066 022626 NOMM: CMP (SP)+,(SP)+
3754 015070 004767 177336 JSR PC,TPCRLF
3755 015074 047516 046440 043516 .ASCIZ /NO MNG/
3756 015102 000 .EVEN
3757 015104 004767 177104 JSR PC,FATERR
3758 015112 000761 (1) 104
3759 (1) $RETMM
3760 015114 013702 172354 UPMM: MOV @172354,R2
3761 015120 000710 BR MMREG

```

 PROGRAM CONTROL COMES HERE TO DETERMINE IF THE MEMORY MANAGEMENT
 IS AVAILABLE OR NOT, AND IF IT IS AVAILABLE THEN WHETHER
 THE MEMORY ABOVE 28K IS REQUIRED TO BE TESTED OR NOT.

; CLEAR THE BYTE THAT IS SUPPOSED TO INDICATE
 ; THAT MEM. MANAG. IS AVAILABLE FOR TESTING
 ; HAS THE OPERATOR ASKED TO CHECK MEMORY MANAG. ?
 ; IF NOT THEN RETURN FROM THE SUBROUTINE
 ; PREPARE TO SETUP TIME OUT VECTOR
 ; RETURN ADDRESS TO NOMM
 ; AND WITH A PSW OF 340
 ; TRY TO REACH MEM. MANAG. SRO
 ; IF IT IS AVAILABLE THEN SET MEM. MANAG. AVAILABLE
 ; BYTE
 ; R1 IS POINTING TO PAR0
 ; PAR0 WILL POINT TO BANK 0
 ; SET UP PAR 1
 ;
 ; SETUP PAR1-PAR6
 ; ADDRESS OF PAR7
 ;
 ; PAR7 IS POINTING TO THE I/O PAGE
 ;
 ; SETUP PDR0-PDR7
 ;
 ; ENABLE MEM. MANAG.
 ; RESTORE TIME OUT TRAP VECTOR FOR ANY FUTURE TRAP
 ;
 ; RESTORE STACK POINTER
 ; TYPE "NO MEMORY MANAGEMENT MESSAGE
 ;
 ; *ERROR* REPORT ERROR MESSAGE AND HALT AT FATHLT
 ; *****ERROR NUMBER 104*****
 ;
 ; RESTORE TIME OUT TRAP VECTOR
 ;
 ; PREPARE TO UPDATE MEMORY MANAG. REGISTERS


```

3766
3767
3768
3769
3770
3771
3772
3773
3774
3775
3776
3777 015122 005063 177776
3778 015126 010113
3779 015130 105737 000272
3780 015134 001425
3781 015136 010146
3782 015140 042701 017777
3783 015144 040113
3784 015146 052701 004000
3785 015152 006001
3786 015154 103376
3787 015156 062701 172340
3788 015162 011101
3789 015164 052701 010000
3790 015170 006101
3791 015172 103376
3792 015174 006301
3793 015176 006143
3794 015200 006101
3795 015202 006123
3796 015204 050113
3797 015206 012601
3798 015210 000207
3802
3803
3804
3805
3806
3807
3808
3809
3810
3811
3812
3813
3814
3815 015212 016143 000001
3816 015216 005043
3817
3818 015220 116113 177777
3819 015224 000207

```

```

; * 18 BIT ADDRESS GENERATOR
; * -----
; *
; * THIS SUBROUTINE IS USED TO PLACE THE ADDRESS STORED IN R1
; * IN THE LOCATION POINTED BY R3. THE ADDRESS IN R1 IS CONVERTED
; * TO AN 18 BIT ADDRESS ONLY IF MEM. MANAG. IS AVAILABLE IN WHICH
; * CASE THE HIGH ORDER BITS OF THE ADDRESS ARE PLACED IN LOCATION
; * POINTED BY R3-2
; *
PUTADR: CLR      -2(R3)
        MOV      R1,(R3)
        TSTB     2(R3)
        BEQ      6$
        MOV      R1,-(SP)
        BIC      R1,17777,R1
        BIC      R1,(R3)
        BIS      R1,4000,R1
2$:      ROR      R1
        BCC      2$
        ADD      R1,172340,R1
        MOV      R1,(R1),R1
        BIS      R1,10000,R1
4$:      ROL      R1
        BCC      4$
        ASL      R1
        ROL      R1,-(R3)
        ROL      R1
        ROL      R1,(R3)+
        BIS      R1,(R3)
        MOV      R1,(SP)+,R1
6$:      RTS      PC

; PLACE THE ADDRESS STORED IN R1 IN LOCATION (R3)
; IS THE MEM. MANAG. AVAILABLE ?
; IF NOT THEN RETURN FROM THE SUBROUTINE
; SAVE R1
; CLEAR BITS 0-12 OF THE ADDRESS IN R1
; LEAVE BITS 0-12 OF THE ADDRESS IN (R3)
; PREPARE TO SHIFT R1 BY 12 PLACES
; GET THE NUMBER OF PAR IN R1
; GET THE ADDRESS OF PAR IN R1
; LOAD R1 WITH THE CONTENTS OF PAR
; PLACE THE ADDRESS BITS 13-17 IN BITS 11-15 OF R1
; SO WE DON'T PICK UP C BIT
; PLACE BIT 17 IN LOCATION POINTED BY R3-2
; PLACE BIT 16 OF THE ADDRESS
; PLACE BITS 13-15 OF THE ADDRESS IN LOCATION (R3)
; RESTORE R1
; RETURN FROM THE SUBROUTINE

; * GET ADDRESS FROM THE APT MAILBOX
; * -----
; *
; * THIS SUBROUTINE IS USED TO GET ADDRESS FROM APT MAILBOX AND
; * PLACE IT IN THE LOCATION USED BY THE PROGRAM TO DEFINE THE
; * MEMORY BOUNDRIES.
; * PROGRAM CONTROL SHOULD COME TO THIS SUBROUTINE WITH R1 POINT-
; * ING TO THE MEMORY TYPE IN THE APT MAILBOX AND R3 POINTING TO
; * THE LOCATION+2 WHERE THE LOW ORDER BITS OF THE ADDRESS HAVE
; * TO BE PLACED
; *
GETADR: MOV      1(R1),-(R3)
        CLR      -(R3)
        MOV      R1,-1(R1),(R3)
2$:      RTS      PC

; PLACE THE LOW ORDER BITS OF THE ADDRESS
; CLEAR THE LOCATION WHERE THE HIGH ORDER BITS
; HAVE TO BE PLACED
; PLACE BITS 16 & 17
; RETURN FROM THE SUBROUTINE

```

```

3824
3825
3826
3827
3828
3829
3830
3831
3832
3833
3834 015226 105237 000311      $GTSIZ: INCB      2#SAVKBB      ;PREPARE TO PLACE ADDRESS BITS 13-17 IN BITS
3835                                     ;0-4 OF R2
3836
3837 015232 012301      GETSIZ: MOV      (R3)+,R1
3838 015234 011302      MOV      (R3),R2      ;LOAD R2 WITH THE LOW ORDER BITS OF THE ADDRESS
3839 015236 042702 017777      BIC      #17777,R2      ;CLEAR ADDRESS BITS 0-12
3840 015242 052702 000040      2$: BIS      #40,R2
3841 015246 006001      4$: ROR      R1
3842 015250 006002      ROR      R2      ;ROTATE R1 AND R2 7 TIMES
3843 015252 103375      BCC      4$
3844 015254 105737 000311      TSTB     2#SAVKBB
3845 015260 001405      BEQ      6$
3846 015262 105037 000311      CLRB     2#SAVKBB
3847 015266 052702 000100      BIS      #100,R2
3848 015272 000765      BR       4$
3849 015274 012301      6$: MOV      (R3)+,R1      ;PLACE THE LOW ORDER ADDRESS BITS IN R1
3850 015276 012700 160000      MOV      #160000,R0
3851 015302 040001      BIC      R0,R1      ;LEAVE BITS 0-12 OF THE ADDRESS IN R1
3852 015304 000207      RTS       PC      ;RETURN FROM THE SUBROUTINE
3853
3854
3858
3859
3860
3861
3862
3863
3864
3865 015306 105737 000272      CLRMM: TSTB     2#MMAVA      ;WAS THE MEMORY MANAGEMENT ENABLED ?
3866 015312 001404      BEQ      1$      ;IF NOT THEN GO TO 1$
3867 015314 005037 177572      CLR      2#SRO      ;DISABLE THE MEMORY MANAGEMENT
3868 015320 105037 000272      CLRB     2#MMAVA      ;AND DO NOT ATTEMPT TO TEST MEM. MANAG.
3869 015324 000207      1$: RTS       PC      ;RETURN FROM THE SUBROUTINE
3870
3871
3872
3873
3874
3875
3876
3877
3878
3879 015326 010046      ;* GET BANK NO. UNDER TEST
3880 015330 010346      ;CALLED BY ERRYP TO GET BANK NO. UNDER TEST INTO PBNK.
3881 015332 042703 017777      ;REGISTERS
3882 015336 052703 010000      ;R0=POINTER TO PAR UNDER TEST
3883                                     ;R3=VIRTUAL ADDRESS ON ENTRY
3884                                     ;R0+R3 ARE RESTORED ON EXIT.
3885
3886
3887
3888
3889
3890
3891
3892
3893
3894
3895
3896
3897
3898
3899
3900
3901
3902
3903
3904
3905
3906
3907
3908
3909
3910
3911
3912
3913
3914
3915
3916
3917
3918
3919
3920
3921
3922
3923
3924
3925
3926
3927
3928
3929
3930
3931
3932
3933
3934
3935
3936
3937
3938
3939
3940
3941
3942
3943
3944
3945
3946
3947
3948
3949
3950
3951
3952
3953
3954
3955
3956
3957
3958
3959
3960
3961
3962
3963
3964
3965
3966
3967
3968
3969
3970
3971
3972
3973
3974
3975
3976
3977
3978
3979
3980
3981
3982
3983
3984
3985
3986
3987
3988
3989
3990
3991
3992
3993
3994
3995
3996
3997
3998
3999

```



```

3883 015342 000241
3884 015344 006003
3885 015346 103376
3886 015350 105737 000272
3887 015354 001407
3888
3889
3890 015356 006303
3891 015360 062703 172340
3892 015364 011300
3893 015366 006300
3894 015370 000300
3895 015372 110003
3896 015374 010337 000306
3897 015400 012603
3898 015402 012600
3899 015404 000207
3900
3901
3902
3903
3904
3905
3906
3907
3908
3909 015406 010146
3910 015410 042701 003777
3911 015414 052701 002000
3912 015420 000241
3913 015422 006001
3914 015424 103376
3915 015426 010100
3916 015430 105737 000272
3917 015434 001416
3918
3919 015436 010146
3920 015440 006201
3921 015442 042701 000001
3922 015446 062701 172340
3923 015452 011100
3924 015454 006300
3925 015456 006300
3926 015460 006300
3927 015462 000300
3928 015464 042716 000034
3929 015470 052600
3930 015472 012601
3931 015474 000207
3932
3933
3934
3935
3936
3937
3938

1$:      CLC
        ROR      R3
        BCC      1$
        TSTB     2$MMAVA
        BEQ      2$
;SHIFT A BANK BIT
;UNTIL IN BITS <2:0> OF R3
;MEMORY MANAGEMENT UNDER TEST?
;NO EXIT

;GET PAR ADDRESS AND PHYSICAL BANK NO.
        ASL      R3
        ADD      #172340,R3
        MOV      (R3),R0
        ASL      R0
        SWAB     R0
        MOVB     R0,R3
2$:      MOV      R3,2$PBANK
        MOV      (SP)+,R3
        MOV      (SP)+,R0
        RTS      PC
;MAKE R3 PAR ADDRESS OFFSET.
;MAKE FULL PAR ADDRESS.
;GET PAR CONTENTS
;SHIFT BANK BITS TO BITS <7:0>
;SET R3 TO PHYSICAL BANK NO.
;STORE PHYSICAL BANK NO.
;RESTORE R3
;RESTORE R0
;RETURN TO CALLER

;GET THE 1K BANK UNDER TEST
;CALLED BY TEST 7
;REGISTERS
;R1 = VIRTUAL ADDRESS ON ENTRY
;R0 = POINTER TO 1K UNDER TEST ON EXIT
;R1 WILL BE RESTORED ON EXIT

GET1K:  MOV      R1,-(SP)
        BIC      #3777,R1
        BIS      #2000,R1
1$:      ROR      R1
        BCC      1$
        MOV      R1,R0
        TSTB     2$MMAVA
        BEQ      2$
;SAVE R1
;ONLY WANT 1K BIT COUNT
;SET UP SHIFT BIT
;CLEAR CARRY BIT
;ROTATE THE BANK
;UNTIL THEY'RE IN BITS 4:0
;IN CASE NO MEM MGNT
;MEM MGNT AVAILABLE ?
;BR IF NO

;GET PAR AND PHYSICAL BANK
        MOV      R1,-(SP)
        ASR      R1
        BIC      #1,R1
        ADD      #172340,R1
        MOV      (R1),R0
        ASL      R0
        ASL      R0
        ASL      R0
        SWAB     R0
        BIC      #34,(SP)
        BIS      (SP)+,R0
2$:      MOV      (SP)+,R1
        RTS      PC
;SAVE 1K BANK
;MAKE R1 A PAR ADDRESS
;PAR ADDRESSES ARE 4K BOUNDARIES
;MAKE IT A FULL PAR ADDR.
;GET THE PAR CONTENTS
;SHIFT R0
;ONLY WANT THE 2 LSB'S
;SET THE IN PAR
;RESTORE R1
;AND EXIT

;THIS ROUTINE CHECKS THE ADDRESS IN R1
;TO MAKE SURE IT IS ERROR FREE, INCLUDING
;THE CHECKBITS.
;R3 = 0 IF NO ERROR
;R3 = NONZERO IF ERROR IS FOUND

```

```

3939
3940 015476 016777 165206 165224 TSTADD: MOV ECCDIS, JCSRADR ;DISABLE ECC
3941 015504 005011 CLR (R1) ;CLEAR A LOC
3942 015506 005061 000002 CLR 2(R1) ;UPPER WORD TOO
3943 015512 005711 TST (R1) ;READ ZEROS
3944 015514 001047 BNE 1$ ;BR IF ERROR
3945 015516 005761 000002 TST 2(R1) ;SHOULD BE ZEROS ALSO
3946 015522 001044 BNE 1$ ;
3947 015524 005111 COM (R1) ;COMPLEMENT OF 0 = 177777
3948 015526 005161 000002 COM 2(R1) ;BOTH WORDS
3949 015532 022711 177777 CMP 0-1, (R1) ;READ ALL ONES
3950 015536 001036 BNE 1$ ;BR IF NO
3951 015540 022761 177777 000002 CMP 0-1, 2(R1) ;READ AGAIN
3952 015546 001032 BNE 1$ ;BR IF ERROR
3953 015550 016703 165154 MOV CSRADR, R3 ;GET ADDRESS OF CSR
3954 015554 016713 165132 MOV DIAGA, (R3) ;SET DIAGNOSTIC BIT
3955 015560 052713 000002 BIS #2, (R3) ;AND ECCDIS BIT
3956 015564 012711 000000 MOV #0, (R1) ;TO WRITE ALL ZERO CHECKBITS
3957 015570 005761 000002 TST 2(R1) ;READ AND RECORD CHECKBITS
3958 015574 032713 007740 BIT #7740, (R3) ;NO CHECKBIT SHOULD BE SET
3959 015600 001015 BNE 1$ ;BR IF SET
3960 015602 052713 007740 BIS #7740, (R3) ;MAKE CHECKBITS ALL ONES
3961 015606 012711 177777 MOV 0-1, (R1) ;WRITE THEM LIKE THIS
3962 015612 042713 007740 BIC #7740, (R3) ;SO WE KNOW IF WE READ THEM BACK
3963 015616 005761 000002 TST 2(R1) ;READ THEM
3964 015622 042713 170037 BIC #170037, (R3) ;ONLY WANT CHECKBITS
3965 015626 022713 007740 CMP #7740, (R3) ;SHOULD BE ALL ONES
3966 015632 001403 BEQ 2$ ;BR IF OK
3967 015634 012703 000001 1$: MOV #1, R3 ;INDICATE ERROR FOUND
3968 015640 000401 BR 3$ ;AND EXIT
3969 015642 005003 2$: CLR R3 ;CLEARED = GOOD LOC
3970 015644 005077 165060 3$: CLR JCSRADR ;CLEAR CSR BEFORE LEAVING
3971 015650 000207 RTS PC ;AND EXIT

```

;THIS ROUTINE IS USED TO COUNT THE BAD CHECKBITS
 ;FROM TEST 15. IT USES THE ROUTINE 'XOR' AND THEN
 ;COUNTS THE BAD ONES

```

3972
3973
3974
3975
3976
3977
3978 015652 005067 000056 BITCHK: CLR BITCNT ;WANT TO START CLEAN
3979 015656 010146 MOV R1, -(SP) ;SAVE ADDRESS
3980 015660 017701 165044 MOV JCSRADR, R1 ;GET CONTENTS OF CSR
3981 015664 042701 170037 BIC #170037, R1 ;ONLY WANT CHECK BITS
3982 015670 010146 MOV R1, -(SP) ;SAVE THIS TOO
3983 015672 004767 175630 JSR PC, XOR ;RO XOR R1 = R2
3984 015676 012701 000020 MOV #20, R1 ;GETTING READY TO COUNT
3985 015702 006301 1$: ASL R1 ;SHIFT TO NEXT POSITION
3986 015704 020127 010000 CMP R1, #10000 ;FINISHED YET ?
3987 015710 001405 BEQ 2$ ;BR IF DONE
3988 015712 030102 BIT R1, R2 ;SEE IF THIS BIT (R1) IS BAD
3989 015714 001772 BEQ 1$ ;BR IF NOT BAD
3990 015716 005267 000012 INC BITCNT ;IT IS, COUNT IT
3991 015722 000767 BR 1$ ;NOT DONE YET, KEEP GOING
3992 015724 012600 2$: MOV (SP)+, R0 ;FIRST GET DATA
3993 015726 012601 MOV (SP)+, R1 ;THEN ADDRESS
3994 015730 010011 MOV R0, (R1) ;PUT BAD DATA AWAY FOR PRINTOUT

```


C07

DZMML MACY11 30(1046) 05-AUG-77 12:59 PAGE 1-79
DZMMLB.P11 05-AUG-77 12:57

SEQ 0081

SUBROUTINE TO DISABLE MEMORY MANAGEMENT

```

3995 015732 000207          RTS      PC          ;THEN EXIT
3996
3997 015734 000000          BITCNT: .WORD 0          ;# OF BAD CHECKBITS FOUND ABOVE
3998
3999
4000          ;THIS ROUTINE PRINTS THE TITLE AND MESSAGE
4001
4002 015736 004767 176470    PRITITL: JSR      PC,TPCRLF          ;PRINT ROUTINE
4003 015742 055104 046515 026514 .ASCII  /DZMML-A/
4004 015750 101
015751 040 051525 020105 .ASCII  / USE TO TEST MF11S-K ECC MEMORY/
015756 047524 052040 051505
015764 020124 043115 030461
015772 026523 020113 041505
016000 020103 042515 047515
016006 054522
4005 016010 005015 054524 042520 .ASCIZ  <15><12>/TYPE <CONTROL C> TO EXIT/
016016 036040 047503 052116
016024 047522 020114 037103
016032 052040 020117 054105
016040 052111 000
4006 016044 012767 172136 164656 .EVEN
4007 016052 000207          MOV      #172136,CSRADR ;ORIGINAL CSR ADDRESS
4008
4009          RTS      PC          ;RETURN TO START ROUTINE
4010
4011
4012
4013
4014
4015          ;CHECKC
4016          ;THIS ROUTINE CHECKS IF CONTROL-C WAS TYPED AT THE END OF EACH
4017          ;TEST OR IN THE ERROR TYPE ROUTINE.
4018          ;IF CONTROL-C TYPED THE PROGRAM IS RETURNED TO LOWER MEMORY IF IT WAS
4019          ;RELOCATED AND THE ERROR HISTORY IS PRINTED OUT.
4020          ;FINALLY IT HALTS AT FATHLT.
4021 016054 105037 000311    CHECKC: CLRB      @#SAVKBB          ;INIT CONTROL-C FLAG.
4022 016060 105737 177560    TSTB      @#TKS          ;ANY CHAR. TYPED?
4023 016064 100021          BPL      EXITC          ;BR IF NO-EXIT VIA RTS PC-
4024 016066 113702 177562    MOV      @#SKBB,R2          ;GET THE CHAR TYPED.
4025 016072 042702 000200    BIC      @200,R2          ;CLEAR THE PARITY BIT.
4026 016076 122702 000003    CMP      @3,R2          ;IS IT CONTROL-C?
4027 016102 001012          BNE      EXITC          ;BRANCH IF NO -EXIT VIA RTS PC-
4028 016104 110237 000311    MOV      R2,@#SAVKBB          ;ELSE STORE THE CHAR. FOR USE AS A FLAG.
4029 016110 004767 176316    JSR      PC,TPCRLF          ;PRINT "tC"
4030 016114 041536 000          .ASCIZ  /tC/
4031 016120          .EVEN
4032 016120 005067 164600    CLR      ERRFLG          ;CLEAR ERROR HEADER FLAG FOR NEXT START
4033 016124 000167 173752    JMP      RELOER          ;GO RETURN PROGRAM TO LOWER CORE IF RELOCATED.
4034 016130 000207          EXITC: RTS      PC
4035
4036 016132 000000          ENDPROG: 0
4037
4038          ;THIS BEGINS THE STORAGE FOR THE ERROR HISTORY
4039          ;STACK FOR EACH 4K BANK 18. BYTES ARE SAVED.
4040          ;ALSO THE ABSOLUTE LOADER AND XXDP CODE IS SAVED
4040          ;AFTER THE ERROR STACK.
          .END

```

ABASE = 000000	BDCHIP 014734	EXITC 016130	RLDER 012760	TST3 003544
ACDW1 = 000000	BEGIN 000500	EXTYP 014430	RPTOCT 014626	TST4 003776
ACDW2 = 000000	BITCHK 015652	FAILNM 012614	RPT17 011522	TST5 004152
ACPUOP = 000000	BITCNT 015734	FATERR 014214	RSTOT4 013340	TST6 004622
ACT11 012734	BITCOM 013404	FATHLT 014266	SAVKBB 000311	TST7 005542
ADDW0 = 000000	BRTPSZ 001276	FATYP 014636	SAVLDR 002034	TYPCNT 000310
ADDW1 = 000000	BUSER 000352	FNDERR 014142	SAVLOC 000350	TYPDEC 014462
ADDW10 = 000000	CHECK 013014	GETADR 015212	SAVMAX 000340	TYPENB 000273
ADDW11 = 000000	CHECKC 016054	GETBNK 015326	SAVMIN 002722	TYPEOP 012702
ADDW12 = 000000	CHKCNT 013053	GETSIZ 015232	SAVR4 000342	TYPERR 014550
ADDW13 = 000000	CHKGEN 013064	GETIK 015406	SAVR5 000344	TYPMEM 001742
ADDW14 = 000000	CHKTAB 013016	HIGHAD 000330	SAVR6 000346	TYPCT 014602
ADDW15 = 000000	CKDONE 012174	HIGHTW 000326	SAVOT4 013326	TYPSIZ 001650
ADDW2 = 000000	CLAMEM 002232	INHREL 002726	SAV7 006532	TYPSTK 012540
ADDW3 = 000000	CLAMM 015306	LDFLG 002716	SBEMSK 002674	T12A 007640
ADDW4 = 000000	CNTSCP 002624	LOOP 002460	SCOPE = 000240	T12B 007662
ADDW5 = 000000	CONT 002422	LOWADD 000324	SEQERR 014214	T5A 004174
ADDW6 = 000000	CONTHM 012242	LOWBNK 000300	SETSTK 001754	T6A 004724
ADDW7 = 000000	CSRADR 002730	LOWER 012200	SETSWR 001126	T6B 004736
ADDW8 = 000000	CTLC 013002	LOWTWO 000322	SLFSIZ 001310	T6FLG 005540
ADDW9 = 000000	DASHL 013354	M = 000200	SOURCE 013012	UPMM 015114
ADEVCT = 000000	DATADR 013524	MAXADR 012346	SRO = 177572	UPPFLG 013052
ADEVN = 000000	DATBUF 002664	MAXMEM 000336	START 000200	WRTMEM 000120
AENV = 000000	DATSAV 002704	MEMING 014736	STRTDI 000276	XOR 013526
AENVN = 000000	DBEMSK 002700	MENTST 002224	SWAPAT 000314	SA = 000001
AFATAL = 000000	DECHRD 000306	MINMEM 000320	SWHALT 002644	SADERR 000275
AMADR1 = 000000	DIAGA 002712	MMAVA 000272	SWR 002732	SAPTHD 000272
AMADR2 = 000000	DISPLA 002734	MMREG 014742	SWREG 000176	SCNTMM 012256
AMADR3 = 000000	DSPREG 000174	MSKADR 013522	SW11 = 004000	SCPUOP 000426
AMADR4 = 000000	ECCDIS 002710	MSYES 002714	TBL 002544	SDEVCT 000410
AMAMS1 = 000000	ENDMAX 012422	N = 000105	TKS = 177560	SDOAGN 012754
AMAMS2 = 000000	ENDPAS 012430	NOMM 015066	TPADER 014272	SENDAD 000156
AMAMS3 = 000000	ENDPRO 016132	NOTYP 014316	TPCRLF 014432	SENV 000420
AMAMS4 = 000000	ENDSTK 000304	OCTTYP 014572	TRYSR 001300	SENVN 000421
AMSGAO = 000000	END0 003070	OCTXT 014732	TSTADD 015476	SEOP 012652
AMSGLG = 000000	END1 003360	ONEPAS 001030	TSTDAT 002670	SETABL 000420
AMSGTY = 000000	END10 007160	PABYT 013062	TSTGO 002646	SETEND 000450
AMTYP1 = 000000	END11 007612	PASFLG 000302	TSTMN 012226	SFATAL 000402
AMTYP2 = 000000	END12 010252	PBNK 000306	TSTREL 002104	SGTSIZ 015226
AMTYP3 = 000000	END13 010364	PCRLF 014420	TSTRP 001044	SHD = 000002
AMTYP4 = 000000	END14 010476	PNTMES 014440	TSTSCP 002606	SHIBTS 000272
APASS = 000000	END15 011330	PRTITL 015736	TSTSIZ 002110	SHIMAX 000332
APRIOR = 000000	END16 011500	PUTADR 015122	TST0 002736	SKBB = 177562
APTHLT 014260	END17 011636	PWRON 000044	TST1 003072	SMADR1 000432
APTSIZ 001210	END2 003540	PWRUP 000136	TST10 006536	SMADR2 000436
ASWREG = 000000	END3 003772	REL 000170	TST11 007164	SMADR3 000442
ATESTN = 000000	END4 004146	RELBOT 000316	TST12 007616	SMADR4 000446
AUNIT = 000000	END5 004616	RELOC 011642	TST13 010256	SMAIL 000400
AUSWR = 000000	END6 005534	RELOER 012102	TST14 010370	SMAMS1 000430
AVECT1 = 000000	END7 006526	RESTRT 000250	TST15 010502	SMAMS2 000434
AVECT2 = 000000	ERRFLG 002724	RETMN 015064	TST16 011334	SMAMS3 000440
BAKPAT 000312	ERROR 013540	RETSTK 012560	TST17 011504	SMAMS4 000444
BASE 002720	ERRTYP 014010	RETTYP 014356	TST2 003364	SMAXM 000334

E07

DZMLB MACY11 30(1046) 05-AUG-77 12:59 PAGE 2-1
DZMLB.P11 05-AUG-77 12:57 SYMBOL TABLE

SEQ 0083

\$MBAOR	000274	\$MTYP4	000445
\$MSGAD	000414	\$PASS	000406
\$MSGLG	000416	\$PASTM	000300
\$MSGTY	000400	\$PRERR	000274
\$MTYP1	000431	\$RETMH	015056
\$MTYP2	000435	\$SVPC	= 000044
\$MTYP3	000441	\$SWR	= 000000

\$SWREG	000422
\$TESTN	000404
\$TN	= 000000
\$TPB	= 177566
\$TPCHR	014374
\$TPDEC	014466
\$TPNUM	014640

\$TPS	= 177564
\$TPSTK	012530
\$TSTM	000276
\$TYPE	014332
\$UNIT	000412
\$UNITM	000302
\$USWR	000424

\$Z	= 000362
.	= 016134
\$X	= 000272

. ABS. 016134 000

ERRORS DETECTED: 0

DZMLB, DZMLB/DOC=DZMLB
RUN-TIME: 8 9 .3 SECONDS
RUN-TIME RATIO: 151/17=8.4
CORE USED: 12K (23 PAGES)

DOCUMENT PAGES: 83