

PDP11

MEMORY EXERCISER
MD-11-DZKMA-B

EP-DZKMA-B-DL-A
COPYRIGHT © 1976
FICHE 1 OF 1

NOV 1976
digital
MADE IN USA

The microfiche card contains 60 frames of data arranged in a 10x6 grid. Each frame displays a different set of data, likely representing memory addresses and their contents. The data is organized into columns and rows, with some frames showing hex values and others showing alphanumeric strings. The frames are separated by small gaps, and the overall layout is consistent across the card.

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

:ERROR # 1 :[TSTRP] FATAL DATA ERROR
: LOCATIONS 0000-430 FAILED 1'S + 0'S TEST.
: RD = GOOD DATA
: RI = ADDRESS OF FAILING LOCATION.

:ERROR # 2 :[APTSIZ] APT FATAL ERROR
: APT MEMORY TABLES NOT SETUP CORRECTLY.
: CHECK LOCATIONS \$MAMS1 [430] TO \$MADR4 [446]
: , FOR CORRECT MEMORY SIZE DATA.

:ERROR # 3 :[TSTSIZ] OPERATOR FATAL ERROR
: SELECTED MEMORY SIZE GREATER THAN 28K, BUT
: SR BIT12 (10000) NOT SET.
: SET BIT12 AND RESTART AT 200.

:ERROR # 4 :[TSTSIZ] OPERATOR FATAL ERROR
: LOWEST SELECTED TEST LIMIT IS HIGHER THAN
: HIGHEST TEST LIMIT. SET LOCATIONS "LOWTWO" [322]
: TO "HIGHADD" [330] CORRECTLY AND RESTART
: AT 200.

:ERROR # 5 :[TST0] TEST SEQUENCE ERROR
: TST0 HAS BEEN ENTERED OUT OF SEQUENCE
: TESTN SHOULD = 00
: THE DIAGNOSTIC HAS BEEN CORRUPTED.
: IF POSSIBLE SELECT ANOTHER 4K BANK
: BANK 0 AND RERUN THE TEST ON THE FAILING MEMORY.

:ERROR # 6 :[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 # 7 :[TST0] ADDRESS AND DATA ERROR
: IDENTICAL TO PREVIOUS ERROR EXCEPT THE DATA
: WRITTEN INTO THE FAILING LOCATION WAS IN
: ERROR ALSO.

:ERROR # 10 :[TST0] DATA ERROR
: IF BAD DATA = 0000 COULD BE AN ADDRESSING
: ERROR , ELSE COMPARE GOOD AND BAD DATA FOR FAILING BITS.

:ERROR # 11 :[TST0] ADDRESSING ERROR
: THE FAILING ADDRESS RESPONDED BUT IS NON-
: EXISTENT. MAY BE A DUAL ADDRESSING PROBLEM.

:ERROR # 12 :[TST1] TEST SEQUENCE ERROR
: \$TEST [404] SHOULD = 01
: THE DIAGNOSTIC HAS BEEN CORRUPTED.

;ERROR # 13 ;[TST1] DATA ERROR
;COMPARE GOOD AND BAD PRINTED DATA, FAILING
;DATA BITS MAY SHORTED OR SWAPPED.

;ERROR # 14 ;[TST2] TEST SEQUENCE ERROR
;STESTN [404] SHOULD = 02
; THE DIAGNOSTIC HAS BEEN CORRUPTED.

;ERROR # 15 ;[TST2] ADDRESS OR DATA ERROR
;IF "ADR ERR" NOT PRINTED THEN THE BYTE SELECT
;CIRCUITRY PROBABLY FAILED.

;ERROR # 16 ;[TST3] TEST SEQUENCE ERROR
;STESTN [404] SHOULD = 03
; THE DIAGNOSTIC HAS BEEN CORRUPTED.

;ERROR # 17 ;[TST3] DUAL ADDRESSING ERROR
;DUAL ADDRESSING PROBLEM FOR BITS THAT DIFFER
;IN GOOD AND BAD DATA PRINTOUT.

;ERROR # 20 ;[TST3] DUAL ADDRESSING ERROR
;FOR THIS ERROR THE DATA PRINTED IS AN ADDRESS.
;THIS IS THE ADDRESS THAT WAS SELECTED WHEN THE
;SAME DATA WAS WRITTEN INTO THE FAILING LOCATION.

;ERROR # 21 ;[TST3] DUAL ADDRESSING ERROR
;SAME AS ERROR #20 EXCEPT DIFFERENT DATA
;(SWAPPED BAKPAT) WAS WRITTEN.

;ERROR # 22 ;[TST4] TEST SEQUENCE ERROR
;STESTN [404] SHOULD = 04.
; THE DIAGNOSTIC HAS BEEN CORRUPTED.

;ERROR # 23 ;[TST4] DUAL ADDRESSING ERROR
;IF PASFLG = 0 THEN THE FAILING LOCATION
;AND FAILING DATA ARE DUAL ADDRESSES.

;ERROR # 24 ;[TST5] TEST SEQUENCE ERROR
;STESTN [404] SHOULD = 05
; THE DIAGNOSTIC HAS BEEN CORRUPTED.

;ERROR # 25 ;[TST5] DATA ERROR
;DATA WRITE OR READ ERROR.

;ERROR # 26 ;[TST5] 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 # 27 ;[TST5] MARCHING 1'S AND 0'S DATA ERROR
;IDENTICAL TO PREVIOUS ERROR EXCEPT THE DATA IS
;CHECKED IMMEDIATELY AFTER BEING WRITTEN.

;ERROR # 30 ;[TST6] TEST SEQUENCE ERROR

```

;STESTN SHOULD = 06
;THE DIAGNOSTIC HAS BEEN CORRUPTED.

;ERROR # 31  ;[TST6] 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 # 32  ;[TST7] TEST SEQUENCE ERROR
;             ;STESTN SHOULD = 07
;             ;THE DIAGNOSTIC HAS BEEN CORRUPTED.

;ERROR # 33  ;[TST7] SHIFTING DIAGONAL DATA ERROR
;             ;IF PASFLG=0 BAKPAT WRITE OR READ ERROR.
;             ;IF PASFLG=1 BAKPAT READ CHECK ERROR
;             ;IF PASFLG= GREATER THAN 1 BUT EVEN VALUE THEN:
;             ;             ;THE FAILING LOCATION COULD NOT BE WRITTEN INTO.
;             ;IF PASFLG= GREATER THAN 1 BUT ODD VALUE THEN:
;             ;             ;THE FAILING LOCATION WAS WRITTEN CORRECTLY
;             ;             ;BUT LOST THE DATA.

;ERROR # 34  ;[TST10] TEST SEQUENCE ERROR
;             ;STESTN SHOULD = 10
;             ;THE DIAGNOSTIC HAS BEEN CORRUPTED.

;ERROR # 35  ;[TST10] BAKPAT DATA ERROR
;             ;BAKPAT WRITE OR READ ERROR INTO THE FAILING LOCATION.

;ERROR # 36  ;[TST10] READ RECOVERY DATA ERROR
;             ;THIS ERROR CAN BE REPORTED BY TST10 AND TST11.
;             ;(THEY SHARE CODE). SEE $TESTN [404] FOR WHICH TEST FAILED.
;             ;FOR BOTH TESTS COMPARE THE GOOD AND BAD DATA AT THE FAILING
;             ;LOCATION TO SEE WHICH BITS FAILED.

;ERROR # 37  ;[TST10] READ RECOVERY DATA ERROR
;             ;IDENTICAL TO THE PREVIOUS ERROR EXCEPT SWAPPED BAKPAT IS
;             ;USED AS WRITE AND READ DATA.

;ERROR # 40  ;[TST11] TEST SEQUENCE ERROR
;             ;STESTN SHOULD = 11
;             ;THE DIAGNOSTIC HAS BEEN CORRUPTED.

;ERROR # 41  ;[TST12] TEST SEQUENCE ERROR
;             ;STESTN SHOULD = 12
;             ;THE DIAGNOSTIC HAS BEEN CORRUPTED.

;ERROR # 42  ;[TST12] WORST CASE CORE TEST DATA ERROR
;             ;IF PASFLG=1 COMPARE GOOD AND BAD DATA FOR FAILING BITS.
;             ;IF PASFLG=2 THE FAILING LOCATION WAS WRITTEN AND READ
;             ;             ;WITH GOOD DATA,BUT FAILED READ CHECK

```

E01

DZKMA MACY11 27(732) 04-NOV-76 07:49 PAGE 5
DZKNAB.P11

```

;          READING IN THE MIN. TO MAX DIRECTION.
;IF PASFLG=3 SAME CONDITIONS AS PASFLG=2 EXCEPT FAILED
;          DOING THE READ CHECK FROM MAX TO MIN DIRECTION.
;
;ERROR # 43 ;[TST12] WORST CASE CORE TEST DATA ERROR
;          IDENTICAL TO PREVIOUS ERROR EXCEPT THE DATA WRITTEN
;AND READ IS COMPLEMENTED.
;
;ERROR # 44 ;[TST13] TEST SEQUENCE ERROR
;          $TESTN SHOOULD = 13
;          THE DIAGNOSTIC HAS BEEN CORRUPTED.
;
;ERROR # 45 ;[TST13] WRITE RECOVERY TEST DATA ERROR
;          IF PASFLG=0 COMPARE GOOD AND BAD DATA FOR FAILING BITS.
;          IF PASFLG=77400 DATA ERROR FOUND WHILE DOING A SECOND READ CHECK.
;          IF PASFLG=77402 DATA ERROR FOUND IN FAILING LOCATION AFTER
;          SMALL TEST PROGRAM RUN IN FAILING BANK.
;
;ERROR # 46 ;[TST13] WRITE RECOVERY TEST DATA ERROR
;          DATA ERROR FOUND JUST BEFORE THE SMALL TEST
;          WAS TO BE RUN IN THE FAILING BANK. TO AVOID "BLOWING" UP
;          WHEN THE SMALL TEST IS RUN TST13 IS ABORTED.
;
;ERROR # 47 ;[TST13] WRITE RECOVERY TEST DATA ERROR
;          IDENTICAL TO ERROR #XXX EXCEPT THE DATA WRITTEN
;          AND READ IS DIFFERENT. (177667).
;          177667 IS THE COMPLEMENT OF "JMP (RD)" (110) WHICH IS
;          THE ESCAPE FROM THE SMALL TEST PROGRAM RUN IN THE BANK
;          UNDER TEST.
;
;ERROR # 50 ;[PARERR] PARITY TRAP ERROR
;          PARITY TRAP TO 114 OCCURRED.
;          FOR THIS ERROR PRINTOUT THE "GOOD DATA" IS ACTUALLY
;          THE FAILING PARITY MODULE UNIBUS ADDRESS.
;          SAVLOC [352] CONTAINS THE PC WHERE THE TRAP OCCURRED.
;
;ERROR # 51 ;[PARITY] PARITY TRAP FATAL ERROR
;          A PARITY TRAP TO 114 OCCURRED, BUT NO PARITY MODULES COULD BE FOUND
;          WITH AN ERROR BIT (BIT15) SET.
;
;ERROR # 52 ;[NOMM] OPERATOR FATAL ERROR
;          TESTING ABOVE 28K WAS SELECTED, BUT NO MEMORY MANAGEMENT
;          OPTION WAS FOUND.
;          RESET SWITCH OPTIONS AND RESTART AT 200.
;
;ERROR # 53 ;[PARITY] OPERATOR FATAL ERROR
;          PARITY TESTING WAS SELECTED BUT NO PARITY MODULES
;          WERE FOUND.
;          RESET SWITCH OPTIONS AND START AT 200.

```

.REPT 0

16.3) 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" OR "PAR ERR" WILL BE TYPED OUT IF ADDRESS ERROR OR PARITY 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.

[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.)

[8.0] MISCELLANEOUS

[8.1] ADDRESS/BANK RANGES IN OCTAL AND DECIMAL

THIS REFERENCE TABLE CROSS REFERENCES THE MEMORY BANK NO.S, 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	NOT USED	
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	1 1600	772342
8	32K-36K	200000-217776	2 2000	772344
9	36K-40K	220000-237776	3 2200	772346
10	40K-44K	240000-257776	4 2400	772350
11	44K-48K	260000-277776	5 2600	772352
12	48K-52K	300000-317776	6 3000	772354
13	52K-56K	320000-337776	1 3200	
14	56K-60K	340000-357776	2 3400	
15	60K-64K	360000-377776	3 3600	
16	64K-68K	400000-417776	4 4000	
17	68K-72K	420000-437776	5 4200	
18	72K-76K	440000-457776	6 4400	
19	76K-80K	460000-477776	1 4600	
20	80K-84K	500000-517776	2 5000	
21	84K-88K	520000-537776	3 5200	
22	88K-92K	540000-557776	4 5400	
23	92K-96K	560000-577776	5 5600	
24	96K-100K	600000-617776	6 6000	
25	100K-104K	620000-637776	1 6200	
26	104K-108K	640000-657776	2 6400	
27	108K-112K	660000-677776	3 6600	
28	112K-116K	700000-717776	4 7000	
29	116K-120K	720000-737776	5 7200	
30	120K-124K	740000-757776	6 7400	
31	124K-128K	760000-777776	7 7600	772354

NOTES:

- 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. 1 WOULD EQUAL 2000, PAR 2 WOULD EQUAL 2200 ETC.

[8.2] EXECUTION TIME

HERE ARE SOME TYPICAL EXECUTION TIMES.

LSI-11 AND 4K:= 100 SECS.
LSI-11 AND 8K:= 5 MINUTES.

[8.2] PASS COUNT AND TEST NO. LOCATIONS

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

STESTN [404] = CURRENT TEST NO. AND RELOCATION, PARITY FLAGS.

WHERE:
LOW BYTE = TEST NO.
IF BIT15 = 1 TEST IS RELOCATED
IF BIT13 = 1 PARITY UNDER TEST.

[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 - 7744 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 "DZKMA-A" TITLE
2. [TSTRP] SAVE DATA FROM LOCATIONS 0-376

INTO 7744-10314.

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 WRITING INTO SUCCEEDING MEMORY LOCATIONS UNTIL TIMEOUT TRAP TO 4 OCCURS. ENABLE MEMORY MANAGEMENT AND SIZE MEMORY ABOVE 28K.
5. [TYPsiz] TYPE MEMORY TEST LIMITS.
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 !
```

IF GREATER THAN 4K UNDER TEST THE ABSOLUTE LOADER (300 ADDRESSES) IS APPENDED. IF GREATER THAN 4K AND UNDER XXDP CHAIN MODE 5674 (OCTAL) ADDRESSES ARE APPENDED TO THE TEST. THIS SAVES THE XXDP MONITOR, AND ALLOWS THE LOCATIONS OCCUPIED BY XXDP TO BE TESTED.

7. [CLRMEM] CALL "PARITY" ROUTINE AND IF SELECTED, ENABLE ALL PARITY MODULES. "PARMAP" [LOC. 352] CONTAINS A MAP OF PARITY MODULES FOUND. IF MODULE 172336 BIT 15 IS SET, IF #172334 FOUND BIT 14 IS SET ETC..
8. [CLRMEM] CLEAR MEMORY CURRENTLY UNDER TEST
9. [CONT] DISPATCH TO TST0
10. [TST0] EXECUTE TEST 0. SEE SECTION 10 FOR TEST DESCRIPTIONS.
11. [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.
12. [TST1-TST12] EXECUTE TST1-TST12 EACH TIME GOING TO STEP 9.

13. [TST13] TEST 13 IS DIFFERENT FROM TESTS 0-12, BECAUSE IT IS A SMALL PROGRAM ACTUALLY RUNNING IN THE MEMORY UNDER TEST. BEFORE THIS SMALL PROGRAM IS STARTED "TST13 BNK XX" IS TYPED. THIS IS DONE IN CASE THE PROGRAM FAILS. THE USER CAN THEN AT LEAST TELL WHICH BANK OF MEMORY FAILED.
14. [RELOC] THE PROGRAM RELOCATES TO HIGH MEMORY TO TEST THE LOCATIONS IT OCCUPIES. (430-ENDPRG). WHERE "ENDPRG" IS THE CONTENTS OF ENDSTK(306). I.E THE LAST PROGRAM ADDRESS. NOTE "RELOC" IS PRINTED JUST PRIOR TO THE ACTUAL RELOCATION.
15. TESTS 0-13 ARE RUN AS DESCRIBED ABOVE EXCEPT ONLY BANK 0 LOCATIONS 430-ENDPRG ARE TESTED.
16. [RELOER] RELOCATE THE PROGRAM BACK TO LOWER MEMORY.
17. [LOWER] IF CONTROL-C TYPED GO PRINT ERROR HISTORY.
18. [TSTMM] IF MEMORY MANAGEMENT SELECTED AND AVAILABLE, RUN TESTS 0-13 ON THE FIRST 24K SLICE ABOVE 28K.
19. [CONTMM] CALL "UPMM" TO UPDATE MEMORY MANAGEMENT PAR REGISTERS TO POINT TO THE NEXT 24K SLICE OF UPPER MEMORY.
20. [MAXADR] REPEAT STEPS 18 + 19 UNTIL ALL MEMORY ABOVE 28K IS TESTED.
21. [ENDPAS] PRINT ERROR HISTORY OF FAILING BITS
22. [SEOP] DISABLE PARITY MODULES.
PRINT "END PASS #XX"

[9.2] TEST TITLES

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

TEST 0: TEST FOR PROPER BANK SELECTION
 TEST 1: CHECK DATI/DATO LINES
 TEST 2: TEST MEMORY FOR HOLDING DATA AND BYTE SELECTION
 TEST 3: DUAL ADDRESS TEST A
 TEST 4: DUAL ADDRESS TEST B
 TEST 5: MARCHING 1'S AND 0'S
 TEST 6: CELLS' VOLATILITY TEST
 TEST 7: SHIFTING DIAGONAL
 TEST 10: READ RECOVERY GALLOPING TEST THROUGH EVERY 64TH CELL
 TEST 11: READ RECOVERY LONG GALLOPING/FAST GALLOPING TEST
 TEST 12: WORST CASE TESTING FOR CORE MEMORY

TEST 13: WRITE RECOVERY TEST

[10.0] RXDP & ACT11 & APT OPERATION

RXDP CHAIN MODE

OPERATION IS IDENTICAL TO STAND ALONE EXCEPT:

1. NO "DZKMA-B" TITLE IS PRINTED.
2. NO TEST 13 PRINTOUTS SUCH AS "TST13 BNK 00".
3. THE PROGRAM ALWAYS HALTS ON ERROR.
4. 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

L01

DZKMA MACY11 27(732) 04-NOV-76 07:49 PAGE 12
DZKMAB.P11

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
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

593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633

.LIST ME,BIN,SEQ,LOC
.TITLE DZKMA
;*COPYRIGHT (C) JANUARY 1976
;*DIGITAL EQUIPMENT CORP.
;*MAYNARD, MASS. 01754
*
*PROGRAM BY PERVEZ ZAKI
*
*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
*PACKAGE (MAINDEC-11-DZGAC-CO),MAR 21, 1976.
*
\$SWR=160000 ;;HALT ON ERROR, LOOP ON TEST, INHIBIT ERROR TYP0UT

160000

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

SCOPE =NOP
.=42
.WORD 0 ;FOR ACT/XXDP

000042

000240
000042
000000

.SBTTL ACT11 HOOKS

HOOKS REQUIRED BY ACT11

\$SVPC= ;SAVE PC
.=46 ;;1)SET LOC.46 TO ADDRESS OF \$ENDAD IN .SEOP
\$ENDAD
.=52 ;;2)SET LOC.52 TO 40000
.WORD 40000
.= \$SVPC ;; RESTORE PC

000046

000044
000046
000156
000052
040000
000044

000052

000070

000070
012737
000000

000136 000024

PWRDN: MOV #PWRUP, @#24
HALT

```

634
635
636          000104          . =104
637          . GET HERE IF AN ILLEGAL TRAP TO LOC. 4 OCCURRED.
638 000104 005237 000400  BUSER: INC @#SMSGTY ; TELL APT FATAL ERROR#000
639 000110 000000          HALT ; *ERROR* TRAP TO LOC. 4 OCCURRED.
640 000112 000000          HALT ; IN CASE CONTINUE PRESSED.
641          ; 114 AND 116 ARE RESERVED FOR PARITY TRAP VECTORS. SETUP IN
642          ; ROUTINE "BEGIN".
643          . =120
644
645
646
647          ; * WRITE MEMORY BACKGROUND
648          ; -----
649          ;
650          ; THIS ROUTINE IS USED TO WRITE THE MEMORY BACKGROUND TO
651          ; THE VALUE STORED AT LOCATION BAKPAT. THE ROUTINE ASSUMES
652          ; THAT R4 IS POINTING TO THE LOWEST LOCATION AND R5 TO THE
653          ; HIGHEST LOCATION TO BE WRITTEN. THE PROGRAM LEAVES THE
654          ; SUBROUTINE WITH R0 CONTAINING THE CONTENTS OF BAKPAT.
655          ;
656
657 000120 010401          WRTMEM: MOV R4,R1 ; SET R1 TO LOWEST LOCATION UNDER TEST
658 000122 013700 000316 2$: MOV @#BAKPAT,R0 ; LOAD R0 WITH THE CONTENTS OF LOCATION BAKPAT
659 000126 010021          ; STARTING FROM THE LOWEST LOCATION WRITE THE
660 000130 020105          ; MEMORY TO BACK GROUND PATTERN
661 000132 103775          ;
662 000134 000207          RTS PC ; RETURN FROM THE SUBROUTINE
663
664
665 000136 013706 000350  PWRUP: MOV @#SAVR6,SP ; RESTORE STACK POINTER
666 000142 012700 006066  MOV #PNTMES-BEGIN,R0
667 000146 060600          ADD SP,R0 ; GET THE INDIRECT ADDRESS OF LOCATION TPCRLF
668          ; RELATIVE TO LOCATION OF DIAGNOSTIC IN THE CORE
669 000150 004710          JSR PC,(R0) ; GO TO THE TYPE ROUTINE AND TYPE CR, LF AND A "P"
670 000152 000120          .ASCIZ /P/
671          .EVEN
672
673 000154 000411          BR START
674
675          ; * SERVICE XXDP/ACT11
676 000156 004710  $ENDAD: JSR PC,(R0) ; RETURN TO ACT11/XXDP MONITOR
677 000160 000240          NOP ; IF QUICK VERIFY=RESET ELSE NOP
678 000162 000240          NOP ; IF QUICK VERIFY=CLR #-1 ELSE INC #0
679 000164 000240          NOP ; IF QUICK VERIFY=BR -4 ELSE NOP
680 000166 000430          BR RESTR ; REPEAT TEST UNDER ACT11/XXDP
681
682          . =176
683 000176 000000  SWREG: .WORD 0
684
685
686          ; *****
687          ; SBTTL START AND RESTART ROUTINES
688          ; RESTART AT 200 TO CLEAR APT TABLES
689          ; *****
    
```

```

690 000200 013706 000350      START:  MOV    Q#SAVR6,SP      ;SETUP STACK POINTER.
691 000204 012703 000412      MOV    #SUNIT,R3      ;CLEAR THE APT MAILBOX FROM SMAIL TO SDEVCT
692 000210 005043              IS:    CLR    -(R3)      ;CLEAR A MAILBOX LOCATION
693 000212 022703 000400      CMP    #SMAIL,R3     ;DONE?
694 000216 001374              BNE    IS             ;BRANCH IF NO
695 000220 105737 000042      TSTB  Q#42           ;ACT11 MODE?
696 000224 001011              BNE    RESTR         ;BRANCH IF YES
697 000226 105737 000405      TSTB  Q#STESTN+1     ;ARE WE RELOCATED?
698 000232 100406              BMI    RESTR         ;BR IF YES- SINCE TPCRLF IS RELOCATED ALSO-
699 000234 004767 006320      JSR    PC,TPCRLF     ;PRINT TITLE
700 000240 055104 046513 026501 .ASCIZ /DZKMA-B/
701 000246 000102
  
```

```

702
703
704 000250 012704 007744      RESTR: MOV    #ENDPRG,R4 ;LOAD R4 WITH THE ADDRESS OF THE END OF THE PROGRAM
705 000254 012703 000346      MOV    #SAVR5,R3     ;CAUSE R3 TO POINT TO THE LOCATION SAVR5
706 000260 012305              MOV    (R3)+,R5      ;RESTORE R5
707 000262 012306              MOV    (R3)+,SP      ;AND RESTORE R6 JUST IN CASE IT IS A RESTART
708 000264 010600              MOV    SP,R0         ;PLACE THE STARTING ADDRESS OF THE TEST IN R0
709 000266 012746 000340      MOV    #340,-(SP)    ;SET HIGH PRIORITY FOR RT?
710 000272 010046              MOV    R0,-(SP)
711 000274 000002              RTI
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
  
```

.SBTTL APT PARAMETER BLOCK

```

*****
;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.
  
```

```

736 000276
737 000276 000000
738 000300 000400
739 000302 000012
740 000304 000156
741 000306 000000
742 000310 000024
743
744
745
  
```

```

$APTHD:
$HIBTS: .WORD 0 ;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
$MBADR: .WORD $MAIL ;ADDRESS OF APT MAILBOX (BITS 0-15)
$TSTM: .WORD 10 ;RUN TIM OF LONGEST TEST
$PASTM: .WORD 110 ;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
$UNITM: .WORD ;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
        .WORD SETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
  
```

```

REL=$TESTN+1 ;IT WILL BE 0 IF THE PROGRAM IS IN THE LOWER
  
```


802				
803	000330	000000	HIGHTWO: 0	; HOLDS BITS 17:16 OF HIGH TEST ADDRESS
804	000332	037776	HIGHADD: 37776	; HOLDS BITS 15:0 OF HIGH TEST ADDRESS
805			;*****	
806				
807	000334	000000	\$HIMAX: 0	; HOLDS BITS 17:16 OF MAXIMUM AVAILABLE MEMORY
808	000336	017776	\$MAXM: 17776	; HOLDS BITS 15:0 OF MAXIMUM AVAILABLE MEMORY
809				
810	000340	000000	MAXMEM: .WORD	; MAXIMUM CURRENT VIRTUAL MEMORY UNDER TEST
811				
812	000342	000000	SAVMAX: .WORD	
813	000344	000000	SAVR4: .WORD	
814	000346	000000	SAVR5: .WORD	
815				
816			;* SAVR6 POINTS TO WHERE THE PROGRAM STARTS EVEN WHEN RELOCATED.	
817	000350	000500	SAVR6: .WORD	BEGIN ; CONTAINS START ADDRESS WHEN RELOCATED ALSO.
818	000352	000000	PARMAP: 0	; MAP OF PARITY MODULES UNDER TEST.
819	000354	000000	SAVLOC: 0	; TEST 6 STORES ERROR INFO HERE
820	000356	000000	PARSP: 0	; SAVE SP DURING PARITY ERROR TRAP.
821	000360	000000	PARPS: 0	; SAVE PSM DURING PARITY ERROR TRAP.
822				; NOTE-PARSP +PARPS ARE NEEDED SINCE THERE IS
823				; IS NOT ENOUGH ROOM ON THE STACK (500-452) AND
824				; SO THE STACK MUST BE RESET IN THE PARERR ROUTINE.
825				; IN THIS CRUDE FASHION.
826				
827				
828				

;*364-400 IS USED AS A STACK AREA BY ENRCHK ROUTINE FOR ERROR HISTORY PRINTOUT

829 000400
830
831
832
833
834 000400
835 000400 000000
836 000402 000000
837 000404 000000
838 000406 000000
839 000410 000000
840 000412 000000
841 000414 000000
842 000416 000000
843 000420
844 000420 000
845 000421 000
846 000422 000000
847 000424 000000
848 000426 000000
849
850
851
852
853
854
855 000430 000
856 000431 000
857
858
859
860
861 000432 000000
862
863 000434 000
864 000435 000
865 000436 000000
866 000440 000
867 000441 000
868 000442 000000
869 000444 000
870 000445 000
871 000446 000000
872 000450
873
874
875
876

.=400
.SBTTL APT MAILBOX-ETABLE
:*****
:EVEN
\$MAIL: APT MAILBOX
\$MSGTY: .WORD AMSGTY :MESSAGE TYPE CODE
\$FATAL: .WORD AFATAL :FATAL ERROR NUMBER
\$TESTN: .WORD ATESTN :TEST NUMBER
\$PASS: .WORD APASS :PASS COUNT
\$DEVCT: .WORD ADEVCT :DEVICE COUNT
\$UNIT: .WORD AUNIT :I/O UNIT NUMBER
\$MSGAD: .WORD AMSGAD :MESSAGE ADDRESS
\$MSGLG: .WORD AMSGLG :MESSAGE LENGTH
\$ETABLE: APT ENVIRONMENT TABLE
\$ENV: .BYTE AENV :ENVIRONMENT BYTE
\$ENVM: .BYTE AENVM :ENVIRONMENT MODE BITS
\$SWREG: .WORD ASWREG :APT SWITCH REGISTER
\$USWR: .WORD AUSWR :USER SWITCHES
\$CPUOP: .WORD ACPUOP :CPU TYPE, OPTIONS
:BIT 15-11=CPU TYPE
: 11/04=01, 11/05=02, 11/20=03, 11/40=04, 11/45=05
: 11/70=06, PDQ=07, Q=10
:BIT 10=REAL TIME CLOCK
:BIT 9=FLOATING POINT PROCESSOR
:BIT 8=MEMORY MANAGEMENT
\$MAMS1: .BYTE AMAMS1 :HIGH ADDRESS, M.S. BYTE
\$MTYP1: .BYTE AMTYP1 :MEM. TYPE, BLK#1
:MEM. TYPE BYTE -- (HIGH BYTE)
: 900 NSEC CORE=001
: 300 NSEC BIPOLAR=002
: 500 NSEC MOS=003
\$MADR1: .WORD AMADR1 :HIGH ADDRESS, BLK#1
:MEM. LAST ADDR.=3 BYTES, THIS WORD AND LOW OF "TYPE" ABOVE
\$MAMS2: .BYTE AMAMS2 :HIGH ADDRESS, M.S. BYTE
\$MTYP2: .BYTE AMTYP2 :MEM. TYPE, BLK#2
\$MADR2: .WORD AMADR2 :MEM. LAST ADDRESS, BLK#2
\$MAMS3: .BYTE AMAMS3 :HIGH ADDRESS, M.S. BYTE
\$MTYP3: .BYTE AMTYP3 :MEM. TYPE, BLK#3
\$MADR3: .WORD AMADR3 :MEM. LAST ADDRESS, BLK#3
\$MAMS4: .BYTE AMAMS4 :HIGH ADDRESS, M.S. BYTE
\$MTYP4: .BYTE AMTYP4 :MEM. TYPE, BLK#4
\$MADR4: .WORD AMADR4 :MEM. LAST ADDRESS, BLK#4
\$ETEND:
.MEXIT
:*****
.SBTTL BEGIN OF AREA TESTED (+20) WHEN PROGRAM RELOCATES.

```

877      ;:*****
878 000450 177570      SWR: 177570 ;CHANGES TO SWREG IF NO HARDWARE SWITCH REGISTER
879
880      ;=500
881 000500 010706      BEGIN: MOV PC,SP ;SET UP STACK POINTER TO EQUAL BEGIN ADDRESS
882
883 000502 005746      TST -(SP)
884 000504 010637 000350      MOV SP, @#SAVR6 ;SAVE SP FOR FUTURE USE
885 000510 012737 000070 000024      MOV #PWRDN, @#24 ;PREPARE FOR ANY FUTURE POWER DOWN
886 000516 005037 000300      CLR @#SPRERR
887 000522 005037 000314      CLR @#TYPCNT
888 000526 012700 000114      MOV #114, RO ;PREPARE TO SETUP PARITY TRAP VECTOR
889 000532 012710 005456      MOV #PARERR--6, (RO)
890 000536 060720      ADD PC, (RO)+ ;TO PARERR
891 000540 012710 000340      MOV #340, (RO) ;AND PSW OF 340
892 000544 105737 000405      TSTB @#REL ;IS THIS CODE RELOCATED?
893 000550 100002      BPL ONEPAS ;BRANCH IF NO
894 000552 000167 000542      JMP TSTREL ;THIS CODE IS RELOCATED SO GET TEST SIZE.
895
896 000556 005737 000406      ONEPAS: TST @#SPASS ;IS THIS THE FIRST PASS?
897 000562 001402      BEQ TSTRP ;BRANCH IF YES (TEST TRAP CATCHER ADDRESSES)
898 000564 000167 000374      JMP SETSTK ;GET THE TEST SIZE
899 000570 012704 007744      TSTRP: MOV #ENDPRG, R4 ;LOAD R4 WITH THE ADDRESS OF THE END OF THE PROGRAM
900 000574 012700 000377      MOV #377, RO
901 000600 010037 000316      MOV RO, @#BAKPAT
902 000604 005001      CLR R1
903 000606 012124 000400      2$: MOV (R1)+, (R4)+ ;SAVE FROM 0000 TO BEGIN-30 AT END OF PROGRAM FOR NOW
904 000610 020127      CMP R1, #SMAIL
905 000614 103774      BLO 2$
906 000616 005741      3$: TST -(R1) ;PREPARE TO TEST THE TRAP VECTORS
907 000620 010011      4$: MOV RO, (R1) ;CHECK THE TRAP VECTORS FOR THE CAPABILITY
908 ;OF HOLDING 0'S & 1'S
909 000622 020011      CMP RO, (R1) ;IS THE DATA OK?
910 000624 001403      BEQ 6$ ;BRANCH IF YES
911
912 000626 004767 005304      JSR PC, FATERR ;*ERROR* REPORT ERROR MESSAGE AND HALT AT FATHLT
913 000632 000001      I ;*****ERROR NUMBER !*****
914
915 000634 000300      6$: SWAB RO
916 000636 001370      BNE 4$
917 000640 005701      TST R1 ;IF WE HAVE NOT REACHED THE LOWEST MEMORY LOCATION
918 000642 001365      BNE 3$ ;THEN REPEAT FROM 3$
919 000644 012701 000400      MOV #SMAIL, R1
920 000650 014441      8$: MOV -(R4), -(R1) ;RESTORE TRAP CATCHER ETC.
921 000652 005701      TST R1
922 000654 001375      BNE 8$
923 000656 012700 000006      SETSWR: MOV #6, RO
924 000662 012710 000340      MOV #340, (RO) ;SET UP TIME OUT TRAP PSW
925 000666 012740 000700      MOV #4$, -(RO) ;AND THE RETURN ADDRESS
926 000672 005777 177552      2$: TST @SWR ;DOES THE SWITCH REGISTER POINTED BY SWR EXIST ?
927 000676 000404      BR 5$ ;BRANCH IF YES
928 000700 022626      4$: CMP (SP)+, (SP)+ ;RESTORE THE STACK POINTER
929 000702 012737 000176 000450      MOV #SWREG, @#SWR ;AND PLACE THE ADDRESS OF THE SWITCH REGISTER
930 ;DESIGNED FOR THE COMPUTERS NOT HAVING HARDWARE
931 ;SWITCH REGISTER AND RUNNING STAND ALONE
932 000710 105737 000420      5$: TSTB @#SENV ;RUNNING UNDER APT?
    
```

DZKMA MACY11 27(732) 04-NOV-76 07:49 PAGE 20
 DZKMAB.P11 BEGIN OF AREA TESTED (+20) WHEN PROGRAM RELOCATES.

```

933 000714 001403          BEQ      APTSIZ      ;BRANCH IF NO
934 000716 012737 000422 000450      MOV      #SSWREG,2#SWR ;SET SWR EQUAL TO APT SWITCH REGISTER.
935
936
937
938
939
940
941
942
943
944
945
946
947 000724 012703 000340          APTSIZ: MOV      #MAXMEM,R3      ;POINT R3 TO MAXMEM.
948 000730 013737 000330 000334      MOV      2#HIGHTWO,2#SHIMAX    ;IN CASE NO SELF SIZING DONE.
949 000736 013737 000332 000336      MOV      2#HIGHADD,2#SMAXM    ;IN CASE NO SELF SIZING DONE.
950 000744 105737 000421          TSTB    2#SENVN              ;DOES APT ALLOW SELF SIZING?
951 000750 100021          BPL      TRYSR              ;BRANCH IF YES
952
953 000752 012701 000451          IS:     MOV      #SMTYP4+4,R1   ;POINT R1 TO BLOCK TYPE 4(+4)
954 000756 162701 000004          SUB      #4,R1              ;POINT R1 TO NEXT BLOCK TYPE.
955 000762 105711          TSTB    (R1)                ;IS THE BLOCK TYPE NON ZERO?
956 000764 001006          BNE      2$                 ;BRANCH IF YES (MEMORY EXISTS)
957 000766 020127 000431          CMP      R1,#SMTYP1         ;ALL APT BLOCK TYPES BEEN CHECKED?
958 000772 101371          BHI      1$                 ;BRANCH IF NO
959
960 000774 004767 005136          JSR      PC,FATERR          ;*ERROR* REPORT ERROR MESSAGE AND HALT AT FATHLT
961 001000 000002          2       ;*****ERROR NUMBER 2*****
962
963 001002 004767 006302          2$:     JSR      PC,GETADR    ;GO SET MAXIMUM APT ADDRESS INTO $MAXM + $SHIMAX
964 001006 004767 006276          JSR      PC,GETADR    ;GO SET MAXIMUM APT ADDRESS INTO HIGHADD+HIGHTWO
965 001012 000446          BRTPSZ: BR      TYP$IZ     ; TYPE THE SIZE OF MEMORY UNDER TEST
966
967 001014 032777 000100 177426      TRYSR:  BIT      #100,2#SWR   ;USER DEFINED MEMORY TEST BOUNDARIES??
968 001022 001042          BNE      TYP$IZ           ;BRANCH IF YES (DON'T SIZE MEMORY)
969
970
971
972
973
974 001024 010401          SLFSIZ: MOV      R4,R1        ;SETUP R1 AND R4 TO THE LOWEST ADDRESS OF MEMORY
975 001026 012710 001042          MOV      #4$(,R0)          ;SET UP RETURN ADDRESS FROM TIME OUT TRAP TO 4$
976 001032 011111          2$:     MOV      (R1),(R1)    ;WRITE A MEMORY LOCATION INTO ITSELF AND TRAP IF NONEXIS
977
978 001034 062701 000002          ADD      #2,R1             ;ADD 2 TO THE ADDRESS POINTER
979 001040 000774          BR      2$                 ;KEEP ON SIZING UP THE MEMORY UNTIL
980
981
982 001042 022626          4$:     CMP      (SP)+,(SP)+  ;RESTORE THE STACK POINTER
983 001044 004767 005770          JSR      PC,MMNG          ; SERVICE MEMORY MANAGEMENT IF IT IS AVAILABLE
984
985 001050 105737 000276          TSTB    2#MMAVA          ;AND IF IT HAS TO BE TESTED
986 001054 001414          BEQ      12$              ;SEE IF MEMORY MANAGEMENT HAS TO BE TESTED
987 001056 012710 001070          6$:     MOV      #8$(,R0)    ;IF NO MEM. MANG. THEN GO TO 12$
988 001062 012701 020000          MOV      #20000,R1        ;SET UP THE RETURN ADDRESS FROM TRAP TO 8$
;BEGIN CHECKING MEMORY ABOVE 28K

```

DZKMA MACY11 27(732) 04-NOV-76 07:49 PAGE 21
 DZKNAB.P11 BEGIN OF AREA TESTED (+20) WHEN PROGRAM RELOCATES.

989	001066	000761			BR	2\$			
990	001070	022626		8\$:	CMP	(SP)+,(SP)+			;RESTORE STACK POINTER
991	001072	022701	160000		CMP	#160000,R1			;IF R1 DID NOT READ ALL THE LOCATIONS POINTED BY
992									;PAGE ADDRESS REGISTER 6 THEN IT HAS REACHED THE
993									;MAXIMUM AVAILABLE MEMORY
994	001076	001003			BNE	12\$;IN WHICH CASE GO TO 12\$
995	001100	004767	006106		JSR	PC,UPMM			;OTHERWISE GO TO UPDATE MEM. MANG. REGISTERS
996	001104	000764			BR	6\$			
997	001106	024341		12\$:	CMP	-(R3),-(R1)			;CAUSE R3 TO POINT TO LOCATION SMAXM AND R1
998									;TO THE MAXIMUM AVAILABLE MEMORY
999	001110	004767	006104		JSR	PC,PUTADR			;GO TO THE SUBROUTINE TO PLACE THE ADDRESS IN R1
1000									;AT LOCATIONS SMAXM AND SHIMAX
1001	001114	024343			CMP	-(R3),-(R3)			;MAKE R3 POINT TO HIGHADD
1002	001116	004767	006076		JSR	PC,PUTADR			;PLACE THE ADDRESS IN R1 AT LOCATIONS HIGHADD
1003									;AND HIGHTWO
1004	001122	005743			TST	-(R3)			
1005	001124	005043			CLR	-(R3)			;CLEAR THE LOCATION LOWADD
1006	001126	005043			CLR	-(R3)			;AND LOWTWO
1007	001130	012720	000104	TYPsiz:	MOV	#BUSER,(R0)+			;SET UP VECTOR FOR ANY FUTURE TRAP
1008	001134	010403			MOV	R4,R3			;SET R3 TO POINT TO THE LOWEST AVAILABLE MEMORY
1009									;LOCATION
1010	001136	012701	000324		MOV	#LOWTWO,R1			
1011	001142	004767	005400		JSR	PC,PCRLF			;TYPE CR/LF
1012	001146	004767	005546		JSR	PC,OCTTYP			;TYPE LOW TEST ADDRESSE (LOWTWO+LOWADD)
1013	001152	004767	005302	TYPMEM:	JSR	PC,\$TYPE			;TYPE "--"
1014	001156	000055			.ASCIZ	/-/			
1015					.EVEN				
1016	001160	004767	005534		JSR	PC,OCTTYP			;TYPE HIGHEST TEST ADDRESS (HIGHTWO+HIGHADD)
1017	001164	012703	000330	SETSTK:	MOV	#HIGHTWO,R3			;MAKE R3 POINT TO THE HIGH ORDER BITS OF TOP ADDRESS
1018	001170	004767	006130		JSR	PC,\$GTSIZ			;GET THE BITS 13-17 OF THE TOP ADDRESS
1019									;PLACED IN BITS 0-4 OF R2
1020	001174	010401			MOV	R4,R1			;SET R1 TO LOWEST TEST ADDRESS
1021									
1022	001176	062704	000022	4\$:	ADD	#18.,R4			;APPEND THE ERROR STACK FOR THE MEMORY UNDER
1023									;TEST TO THE END OF THE PROGRAM
1024	001202	005302			DEC	R2			
1025	001204	002374			BGE	4\$			
1026	001206	010437	000310		MOV	R4,#ENDSTK			;SAVE THE ADDRESS OF THE END OF THE ERROR STACK
1027	001212	005021		6\$:	CLR	(R1)+			;CLEAR THE ERROR STACK
1028	001214	020104			CMP	R1,R4			
1029	001216	101775			BLOS	6\$			
1030	001220	012737	157776 000340		MOV	#157776,#MAXMEM			;SET MAXMEM TO MAXIMUM VIRTUAL ADDRESS
1031	001226	005723			TST	(R3)+			;TESTING MEMORY MANAGEMENT?
1032	001230	001004			BNE	SAVLDR			;BRANCH IF YES (GO SAVE LOADERS AT TOP OF VIRTUAL MEMORY
1033	001232	021300			CMP	(R3),R0			;IS THE VIRTUAL ADDRESS ABOVE 157776?
1034	001234	103002			BHIS	SAVLDR			;BRANCH IF YES (GO SAVE LOADERS)
1035	001236	011363	000002		MOV	(R3),2(R3)			;OTHERWISE MAKE THE CONTENTS OF LOCATION MAXMEM
1036									;EQUAL TO THE MAXIMUM AVAILABLE MEMORY
1037									;AND FALL INTO SAVE LOADERS.
1038									
1039	001242	004767	006136	SAVLDR:	JSR	PC,CLMM			;DISABLE THE MEMORY MANAGEMENT UNIT
1040	001246	005723			TST	(R3)+			;MAKE R3 TO POINT TO THE LOCATION MAXMEM
1041	001250	011305			MOV	(R3),R5			;R5 CONTAINS THE ADDRESS OF MAXIMUM AVAILABLE MEM.
1042									
1043									
1044									

;IF ONLY 4K BEING TESTED DON'T SAVE LOADERS

DZKMA MACY11 27(732) 04-NOV-76 07:49 PAGE 22
 DZKMA8.P11 BEGIN OF AREA TESTED (+20) WHEN PROGRAM RELOCATES.

1045	001252	020527	017776		CMP	R5, #17776	; ONLY TESTING 4K MAX?	
1046	001256	103416			BLO	4\$; BRANCH IF YES (DON'T SAVE LOADERS)	
1047								
1048	001260	162705	000276	3\$:	SUB	#276, R5	; PREPARE TO SAVE 300 BYTES OF THE LOADERS	
1049	001264	005737	000042		TST	#42	; IS THE PROGRAM RUNNING UNDER ACT OR XXDP ?	
1050	001270	001406			BEQ	2\$; IF NOT THEN GO TO 2\$	
1051	001272	023737	000042	000046	CMP	#42, #46	; ARE WE RUNNING UNDER XXDP CHAIN MODE?	
1052	001300	001402			BEQ	2\$; BRANCH IF NO	
1053	001302	162705	005674		SUB	#(1502.*2), R5	; SAVE 1500. WORDS FOR XXDP CHAIN MODE	
1054	001306	012524		2\$:	MOV	(R5)+, (R4)+	; SAVE LOADER	
1055	001310	020513			CMP	R5, (R3)		
1056	001312	101775			BLOS	2\$		
1057	001314	012323		4\$:	MOV	(R3)+, (R3)+	; SAVE THE CONTENTS OF LOCATION MAXMEM IN SAVMAX	
1058	001316	010423			MOV	R4, (R3)+	; AND THE CONTENTS OF R4 AT SAVR4	
1059								
1060	001320	010537	000346		TSTREL: MOV	R5, #SAVR5	; SAVE HIGHEST VIRTUAL ADDRESS+2	
1061	001324	004767	006054		TSTSIZ: JSR	PC, CLMM	; GO TO DISABLE MEMORY MANAGEMENT UNIT	
1062	001330	005745			TST	-(R5)	; SET R5 BACK TO HIGHEST VIRTUAL ADDRESS	
1063	001332	012703	000324	1\$:	MOV	#LOWTWO, R3	; PREPARE TO LOAD R4 AND R5 WITH THE MEMORY BOUNDRIES	
1064	001336	005723			TST	(R3)+	; IF THE BITS 16,17 OF THE LOWEST LOCATION UNDER	
1065							; TEST ARE NON ZERO	
1066	001340	001003			BNE	2\$; THEN GO TO 2\$	
1067	001342	021327	157776		CMP	(R3), #157776	; IF THE LOWEST LOCATION UNDER TEST IS HIGHER THAN	
1068							; 157776 THEN GO TO TEST MEMORY MANAGEMENT	
1069	001346	103411			BLO	4\$		
1070	001350	032777	010000	177072	2\$:	BIT	#10000, #SWR	; IS MEMORY MANAGEMENT SELECTED?
1071	001356	001003			BNE	3\$; YES ALL IS WELL	
1072	001360	004767	004552		JSR	PC, FATERR	; *ERROR* REPORT ERROR MESSAGE AND HALT AT FATHLT.	
1073	001364	000003				3	; *****ERROR NUMBER 3*****	
1074								
1075	001366	000167	003526	3\$:	JMP	TSTM	; GO TO TEST MEMORY MANAGEMENT	
1076	001372	020423		4\$:	CMP	R4, (R3)+		
1077	001374	103002			BHIS	6\$		
1078	001376	016304	177776		MOV	-2(R3), R4	; ADJUST R4 TO POINT TO THE LOWEST LOCATION UNDER TEST	
1079	001402	005723		6\$:	TST	(R3)+	; IF BITS 16-17 OF HIGHEST LOCATION TO BE TESTED	
1080	001404	001003			BNE	8\$; ARE NON ZERO THEN GO TO 8\$	
1081	001406	021305			CMP	(R3), R5	; OTHERWISE SEE IF THE HIGHEST LOCATION TO BE	
1082							; TESTED IS HIGHER THAN 157776	
1083	001410	101001			BHI	8\$; IF SO THEN GO TO 8\$	
1084	001412	011305			MOV	(R3), R5	; MODIFY R5	
1085	001414	105737	000405	8\$:	TSTB	#REL	; ARE WE RELOCATED.?	
1086	001420	100014			BPL	10\$; BRANCH IF NO	
1087	001422	013704	000322		MOV	#RELBOT, R4	; SET BOTTOM TEST ADDRESS WHEN RELOCATED.	
1088	001426	020527	017776		CMP	R5, #17776	; ARE WE RELOCATED IN BANK 0?	
1089	001432	103402			BLO	9\$; BRANCH IF YES	
1090	001434	012705	017776		MOV	#17776, R5	; ELSE SET HIGH MEMORY UNDER TEST=4K	
1091								
1092	001440	020405		9\$:	CMP	R4, R5	; IS LOW LIMIT LOWER THAN HIGH LIMIT?	
1093	001442	103403			BLO	10\$; BRANCH IF YES	
1094	001444	004767	004466		JSR	PC, FATERR	; *ERROR* REPORT ERROR MESSAGE AND HALT AT FATHLT.	
1095	001450	000004				4	; *****ERROR NUMBER 4*****	
1096								
1097	001452	012703	000342	10\$:	MOV	#SAVMAX, R3	; RESTORE THE CONTENTS OF MAXMEM	
1098	001456	011343			MOV	(R3), -(R3)	; MAKE THE CONTENTS OF MAXMEM = MAXIMUM AVAILABLE	
1099	001460	062713	000002	MEMTST:	ADD	#2, (R3)	; MEMORY +2	
1100								

```

1101 001464 005725          TST      (R5)+          ;AND SET R5=MAX MEMORY+2
1102
1103          ;CLEAR MEMORY UNDER TEST
1104
1105 001466 012702 000001  CLRMEM: MOV      #1,R2          ;SET R2 TO ENABLE PARITY MODULE CODE.
1106 001472 004767 006006          JSR      PC,PARITY          ;ENABLE PARITY IF WANTED AND AVAILABLE.
1107 001476 010500          MOV      R5,R0
1108 001500 005040          2$:    CLR      -(R0)          ;BEGIN CLEARING THE MEMORY FROM THE TOP
1109 001502 020004          CMP      R0,R4          ;UNTIL THE BOTTOM IS REACHED
1110 001504 101375          BHI     2$
1111 001506 012702 000316          MOV      #BAKPAT,R2
1112 001512 012212          MOV      (R2)+,(R2)          ;WRITE SWAPPED BAKPAT IN LOCATION SWAPAT
1113 001514 000312          SWAB    (R2)
1114 001516 017702 176726          MOV      @SWR,R2          ;LOAD R2 WITH THE OPTIONS STORED AT $SWREG
1115 001522 042702 177760          BIC     #177760,R2          ;ONLY LEAVE THE LOWER 4 BITS OF $SWREG IN R2 TO GO TO
1116
1117
1118
1119
1120          ;ENTER HERE FROM TSTSCP ROUTINE AT END OF SUBTEST
1121
1122 001526 005037 000306  CONT:  CLR      @#PASFLG          ;INIT SUBTEST PASS FLAG.
1123 001532 110237 000404          MOVB    R2,@#STESTN          ;SET UP STESTN WITH THE TEST NUMBER GOING
1124
1125 001536 010401          LOOP:  MOV      R4,R1          ;TO BE EXECUTED
1126 001540 010246          MOV      R2,-(SP)          ;LOAD R1 WITH THE LOWEST LOCATION UNDER TEST
1127 001542 012703 000376          MOV      #376,R3          ;SAVE R2 ON THE STACK
1128 001546 004767 005446          JSR      PC,PUTADR          ;POINT R3 TO SCRATCH STACK
1129
1130          ;GO TO GENERATE 18 BIT ADDRESS OUT OF THE ADDRESS
1131 001552 005743          TST     -(R3)          ;STORED IN R1 AND STORE IT IN LOCATIONS (R3)
1132          ;AND (R3-2)
1133 001554 004767 005544          JSR      PC,$GTSIZ          ;CAUSE R3 TO POINT TO THE HIGH ORDER BITS OF THE
1134          ;18 BIT ADDRESS
1135 001560 010400          MOV      R4,R0          ;PLACE BITS 13-17 OF THE ADDRESS IN BITS
1136          ;0-4 OF R2
1137 001562 010401          MOV      R4,R1          ;PLACE THE ADDRESS OF THE LOWEST LOCATION UNDER
1138 001564 010403          MOV      R4,R3          ;TEST IN R0
1139 001566 012602          MOV      (SP)+,R2          ;IN R1
1140 001570 006302          ASL     R2          ;AND IN R3
1141 001572 060702          ADD     PC,R2          ;RESTORE R2
1142 001574 066207 000004          ADD     TBL,-(R2),PC          ;GO TO THE TEST #
1143          ;STORED IN BITS 0-3 OF SWITCH REGISTER
1144
1145
1146 001600 000102          TBL:   TST0-TBL          ;RELATIVE ADDRESS OF TEST # 0
1147 001602 000334          TST1-TBL          ;RELATIVE ADDRESS OF TEST # 1
1148 001604 000434          TST2-TBL          ;RELATIVE ADDRESS OF TEST # 2
1149 001606 000544          TST3-TBL          ;RELATIVE ADDRESS OF TEST # 3
1150 001610 001012          TST4-TBL          ;RELATIVE ADDRESS OF TEST # 4
1151 001612 001122          TST5-TBL          ;RELATIVE ADDRESS OF TEST # 5
1152 001614 001270          TST6-TBL          ;RELATIVE ADDRESS OF TEST # 6
1153 001616 001424          TST7-TBL          ;RELATIVE ADDRESS OF TEST # 7
1154 001620 001646          TST10-TBL          ;RELATIVE ADDRESS OF TEST # 10
1155 001622 002174          TST11-TBL          ;RELATIVE ADDRESS OF TEST # 11
1156 001624 002246          TST12-TBL          ;RELATIVE ADDRESS OF TEST # 12

```

K02

DZKMA MACY11 27(732) 04-NOV-76 07:49 PAGE 24
DZKMA8.P11 BEGIN OF AREA TESTED (+20) WHEN PROGRAM RELOCATES.

1157 001626 002520
1158 001630 003146
1159
1160
1161
1162
1163

TST13-TBL
RELOC-TBL

;RELATIVE ADDRESS OF TEST # 13
;RELATIVE ADDRESS OF ROUTINE 'RELOC'

;R5 IS POINTING TO THE TOP OF THE MEMORY TO BE TESTED+2
;R4 & R0 ARE POINTING TO THE LOWEST ADDRESS OF MEMORY TO BE TESTED

```

1164          : *          SCOPE ROUTINE
1165          : *          -----
1166          : *
1167          : *
1168          : *          PROGRAM COMES TO THIS ROUTINE AFTER COMPLETION OF EACH TEST AND
1169          : *          IF CNTRL-C TYPED GOTO ERROR HISTORY TYPE ROUTINE.
1170          : *          IF SR= 2000 (BIT10) THEN HALT
1171          : *          IF SR= 40000 (BIT14) THEN LOOP ON TEST DEFINED BY SR BITS<3:0>
1172          : *          ELSE CONTINUE TO NEXT TEST.
1173          : *
1174          : *
1175          : *
1176 001632 105737 000420      TSTSCP: TSTB      @#SENV      ;ARE WE RUNNING UNDER APT?
1177 001636 001002              BNE          CNTSCP      ;IF SO THEN GO TO CNTSCP
1178 001640 004767 006020      JSR          PC,CHECKC ;TEST FOR CONTROL-C AND IF TYPED GO
1179          : *          PRINT ERROR HISTORY AND HALT AT FATHLT.
1180 001644 113702 000404      CNTSCP: MOVB      @#$TESTN,R2 ;PLACE THE TEST NUMBER IN THE LOWER BYTE OF R2
1181          : *          SINCE THERE ARE LESS THAN 377 TESTS UPPER BYTE
1182          : *          OF R2 WILL BE 0
1183 001650 005237 000410              INC          @#$DEVCT ;TELL APT WE ARE STILL RUNNING OKAY
1184 001654 032777 002000 176566      BIT          #2000,@SWR ;IS THE PROGRAM GOING TO HALT AFTER EACH TEST?
1185 001662 001401              BEQ          TSTGO      ;IF NOT THEN GO TO 2$
1186 001664 000000              SWHALT: HALT      ;HALT AT END OF TEST SWITCH SET.
1187          : *
1188 001666 032777 040000 176554      TSTGO: BIT      #40000,@SWR ;IS THE PROGRAM GOING TO LOOP ON TEST
1189 001674 001320              BNE          LOOP      ;IF SO THEN GO TO THE STARTING OF THE SAME TEST
1190 001676 105202              INCB         R2
1191 001700 000712              BR          CONT      ;GO TO CONT AND CONTINUE EXECUTING THE NEXT TEST

```

```

1192                                     ;:*****
1193                                     ;*TEST 0 TEST FOR PROPER BANK SELECTION
1194                                     ;*(1) THIS TEST ASSUMES THAT THE MEMORY IS IN A STATE
1195                                     ;* OF ALL 0'S AND R0 HAS THE ADDRESS OF THE LOWEST
1196                                     ;* LOCATION UNDER TEST
1197                                     ;*(2) IT CHECKS FOR PROPER BANK SELECTION BY WRITING
1198                                     ;* 1'S IN A LOCATION AND CHECKING FOR 0'S IN THE SAME
1199                                     ;* LOCATIONS OF OTHER 4K BANKS OF THE MEMORY
1200                                     ;* [I.E. LOCATIONS LIKE 7766 AND 27766 ETC.]
1201                                     ;*(3) THIS TEST ALSO CHECKS TO SEE THAT NONE OF THE NON EXIST-
1202                                     ;* ING BANK RESPOND WHEN THEY ARE ADDRESSED
1203                                     ;:*****
1204 001702 105737 000404 1ST0: TSTB 2#STESTN ;CHECK FOR PROPER TEST SEQUENCE
1205 001706 001403 BEQ .+10
1206 001710 004767 004222 JSR PC,SEGERR ;*ERROR* REPORT ERROR MESSAGE AND HALT AT FATHLT
1207 001714 000005 5 ;*****ERROR NUMBER 5*****
1208
1209 001716 012703 177777 MOV #177777,R3
1210 001722 010401 1S: MOV R4,R1 ;R1 = ADDRESS OF LOWEST LOCATION OF MEMORY UNDER TEST
1211 001724 010310 MOV R3,(R0) ;SET ALL THE BITS AT (R0)
1212 001726 020001 2S: CMP R0,R1 ;IS R0 POINTING TO THE SAME MEMORY LOCATION AS R1
1213 001730 001417 BEQ 4S ;IN WHICH CASE CHECK FOR ALL 1'S AT (R1)
1214 001732 005711 TST (R1) ;OTHERWISE CHECK (R1) FOR ALL 0'S
1215 001734 001430 BEQ 5S
1216 001736 020311 CMP R3,(R1) ;IF R1 IS NOT EQUAL TO R0 AND (R1)
1217 ; DOES NOT CONTAIN ALL 0'S THEN
1218 ; CHECK TO SEE IF (R0) = (R1)
1219 001740 001004 BNE 3S
1220 001742 012767 000006 000042 MOV #6,12S ;*ERROR* SETUP ERROR NO. IN 12S
1221 ;*****ERROR NUMBER #6*****
1222 001750 000403 BR 10S
1223 001752 3S:
1224 001752 012767 000007 000032 MOV #7,12S ;*ERROR* SETUP ERROR NO. IN 12S
1225 ;*****ERROR NUMBER #7*****
1226 001760 010046 10S: MOV R0,-(SP) ;SAVE R0 ON STACK
1227 001762 105237 000301 INCB 2#ADERR ;AN ADDRESSING ERROR IS SUSPECTED
1228 001766 000407 BR 11S
1229 001770 020311 4S: CMP R3,(R1) ;CHECK (R1) FOR ALL 1'S
1230 001772 001411 BEQ 5S
1231 001774 012767 000010 000010 MOV #10,12S ;*ERROR* SETUP ERROR NO. IN 12S
1232 ;*****ERROR NUMBER #10*****
1233 002002 010046 MOV R0,-(SP) ;SAVE R0 ON STACK
1234 002004 010300 MOV R3,R0
1235 002006 004767 003566 11S: JSR PC,ERROR ;GO TO THE ERROR SUBROUTINE
1236 002012 000000 12S: .WORD ;ERROR NUMBER TO BE REPORTED WILL BE PLACED HERE
1237 002014 012600 MOV (SP)+,R0 ;RESTORE R0
1238
1239 002016 013706 000350 5S: MOV 2#SAVR6,SP ;RESTORE THE STACK POINTER
1240 002022 062701 020000 ADD #20000,R1 ;CAUSE R1 TO POINT TO THE SAME CHIP
1241 ; LOCATION IN THE NEXT 4K BANK OF MEMORY
1242 ; BY ADDING 1 TO THE 14TH BIT OF ADDRESS IN R1
1243 002026 020105 CMP R1,R5 ;COMPARE R1 WITH THE HIGHEST MEMORY
1244 ; LOCATION WHICH IS STORED IN R5
1245 002030 103736 BLO 2S ;IF R1 LESS THAN R5 THEN REPEAT THE TEST FROM 2S
1246
1247 002032 105737 000421 TSTB 2#SENVN ;HAS APT INHIBITED SIZING?
    
```


B03

DZKMA MACY11 27(732) 04-NOV-76 07:49 PAGE 28
 DZKMA8.P11 T1 CHECK DI/DO LINES

```

1279          ;*****
1280          ;*TEST 1      CHECK DI/DO LINES
1281          ;*(1)      THIS TEST CHECKS THE DATI/DATO LINES BY SHIFTING
1282          ;*          A 1 IN THE WORD DIRECTION
1283          ;*****
1284 002134 122737 000001 000404 †ST1:  CMPB  #1,2#STESTN  ;CHECK FOR PROPER TEST SEQUENCE
1285
1286
1287 002142 001403          BEQ    +10
1288 002144 004767 003766  JSR    PC,SEGERR  ;*ERROR* REPORT ERROR MESSAGE AND HALT AT FATHLT
1289 002150 000012          12      ;*****ERROR NUMBER 12*****
1290
1291 002152 012700 000001  1$:   MOV    #1,R0
1292 002156 010002          MOV    R0,R2      ;SET R2=1
1293 002160 010011          2$:   MOV    R0,(R1) ;MOV 1 AT LOCATION (R1)
1294 002162 020011          3$:   CMP    R0,(R1)  ;COMPARE R1 WITH THE CONTENTS OF LOCATION (R1)
1295 002164 001403          BEQ    4$
1296 002166 004767 003406  JSR    PC,ERROR   ;*ERROR* REPORT ERROR MESSAGE
1297 002172 000013          13      ;*****ERROR NUMBER 13*****
1298
1299
1300 002174 005702          4$:   TST    R2      ;ARE WE SHIFTING A 0 IN DATA DIRECTION?
1301 002176 001406          BEQ    5$      ;IF SO THEN GO TO 5$
1302 002200 006300          ASL    R0      ;SHIFT THE 1 BROUGHT IN AT 1$ IN
1303                          ;DATA DIRECTION
1304 002202 103366          BCC    2$      ;IF THE 1 HAS NOT BEEN SHIFTED THRU
1305                          ;THE 16 DATA BITS THEN REPEAT FROM 2$
1306 002204 005002          CLR    R2      ;INITIATE SHIFTING OF 0 IN DATA DIRECTION
1307 002206 012700 177776  MOV    #177776,R0
1308 002212 000762          BR     2$
1309
1310 002214 000261          5$:   SEC
1311 002216 006100          ROL    R0      ;SET C BIT
1312 002220 103757          BCS    2$      ;SHIFT A 0 16 TIMES IN DATA DIRECTION
1313                          ;IF THE 0 HAS NOT BEEN SHIFTED THRU
1314 002222 062701 020000  ADD    #20000,R1 ;THE 16 DATA BITS THEN REPEAT FROM 2$
1315                          ;OTHERWISE GO TO THE NEXT BANK OF
1316 002226 020105          CMP    R1,R5   ;4K MEMORY AND REPEAT THE TEST
1317 002230 103750          BLO    1$
1318 002232 000737          END1: BR     ENDO
1319

```

```

1320      ;*****
1321      ;*TEST 2      TEST MEMORY FOR HOLDING DATA AND BYTE SELECTION
1322      ;*(1)      THIS TEST CHECKS THE MEMORY FOR THE CAPABILITY
1323      ;*          OF HOLDING 1'S AND 0'S BY WRITING A BACKGROUND
1324      ;*          OF BAKPAT AND READING IT
1325      ;*(2)      MEMORY IS WRITTEN USING A BYTE AT A TIME
1326      ;*(3)      STEPS 1 & 2 ARE REPEATED WITH A SWAPPED BACKGROUND PATTERN
1327      ;*****
1328 002234 122737 000002 000404 TST2:  CMPB    #2,#STESTN    ;CHECK FOR PROPER TEST SEQUENCE
1329
1330      BEQ     .+10
1331 002242 001403      JSR     PC,SEQERR    ;*ERROR* REPORT ERROR MESSAGE AND HALT AT FATHLT
1332 002250 000014      14      ;*****ERROR NUMBER 14*****
1333
1334 002252 013700 000316 1S:    MOV     @#BAKPAT,R0
1335 002256 110021      MOVB   R0,(R1)+
1336 002260 113721 000317      MOVB   @#BAKPAT+1,(R1)+;WRITE THE MEMORY WITH THE WORD STORED IN BAKPAT
1337 002264 020105      CMP    R1,R5
1338 002266 103771      BLO   1S
1339
1340 002270 020041 2S:    CMP    R0,-(R1)    ;TEST THE MEMORY TO SEE IF IT CONTAINS
1341      ;THE WORD STORED IN BAKPAT
1342 002272 001416      BEQ   8S
1343 002274 062701 000002      ADD   #2,R1
1344 002300 123741 000317      CMPB   @#BAKPAT+1,-(R1);CHECK FOR BYTE SELECTION PROBLEM
1345 002304 001402      BEQ   4S
1346 002306 120041      CMPB   R0,-(R1)    ;AGAIN CHECK FOR BYTE SELECTION PROBLEM
1347 002310 001002      BNE   6S
1348 002312 105237 000301 4S:    INCB   @#SADERR    ;PREPARE TO INFORM THAT IT IS ADDRESSING ERROR
1349 002316 042701 000001 6S:    BIC   #1,R1      ;MAKE THE ADDRESS IN R1 EVEN
1350 002322 004767 003252      JSR   PC,ERROR    ;*ERROR* REPORT ERROR MESSAGE
1351 002326 000015      15      ;*****ERROR NUMBER 15*****
1352
1353 002330 020104 8S:    CMP    R1,R4      ;KEEP ON TESTING THE MEMORY UNTIL
1354 002332 101356      BHI   2S          ;R1 EQUALS THE LOWEST ADDRESS
1355 002334 000337 000316      SWAB  @#BAKPAT    ;CHANGE THE DATA PATTERN
1356 002340 001744      BEQ   1S          ;IF THE DATA PATTERN DOES NOT HAVE LOW
1357      ;BYTE =0 THEN FALL THRU
1358 002342 070733  END2:  BR     END1
1359
1360      ;THE TEST LEAVES BAKPAT LOCATION THE SAME AS IT WAS IN THE BEGINNING
1361

```

```

1362          ;*****
1363          ;*TEST 3          DUAL ADDRESS TEST A
1364
1365          ;*(1) THIS TEST CHECKS FOR DUAL ADDRESSING PROBLEMS BY WRITING A
1366          ;*          BACK GROUND OF BAKPAT.
1367          ;*(2) STARTING FROM THE LOWEST LOCATION IN THE BANK THE TEST WRITES A
1368          ;*          LOCATION WITH SWAPPED BAKPAT
1369          ;*(3) READS THE MEMORY FOR PROPER CONTENTS
1370          ;*(4) SHIFTS A 1 ALONG THE ADDRESS DIRECTION AND REPEATS STEPS 1-3
1371          ;*(5) REPEATS STEP 1-4 FOR EACH 4K BANK
1372          ;*****
1373 002344 122737 000003 000404 †ST3: CMPB  #3,‡#STESTN ;CHECK FOR PROPER TEST SEQUENCE
1374 002352 001403          BEQ    +10
1375 002354 004767 003556          JSR   PC,‡#SEQERR ;*ERROR* REPORT ERROR MESSAGE AND HALT AT FATHLT
1376 002360 000016          16      ;*****ERROR NUMBER 16*****
1377
1378 002362 005003          CLR   R3
1379 002364 004737 000120 2$: JSR   PC,‡#WRTMEM ; WRITE MEMORY WITH THE BACKGROUND STORED
1380          ;AT LOCATION BAKPAT
1381 002370 005002          CLR   R2
1382 002372 050302          BIS   R3,R2
1383 002374 020204          CMP   R2,R4
1384 002376 103465          BLO  16$
1385 002400 020205          CMP   R2,R5
1386 002402 103077          BHIS 20$
1387 002404 000312          SWAB (R2)
1388          ;MAKE R2 POINT TO THE MEMORY BANK POINTED BY R3
1389          ;IF R2 IS LESS THAN R4
1390          ;THEN DO NOTHING
1391          ;IF R2 IS HIGHER THAN THE HIGHEST LOCATION TO BE
1392          ;TESTED THEN EXIT THE TEST
1393          ;OTHERWISE WRITE THE COMPLEMENT OF BAKPAT IN
1394          ;THE LOCATION POINTED BY R2
1395          ;IF R1 IS POINTING TO A LOCATION LOWER THAN R4
1396          ;THEN GO TO 12$
1397 002406 005001          CLR   R1
1398 002410 050301          BIS   R3,R1
1399 002412 020104          CMP   R1,R4
1400 002414 103445          BLO  12$
1401 002416 020105          CMP   R1,R5
1402 002420 103053          BHIS 15$
1403 002422 020102          CMP   R1,R2
1404 002424 001431          BEQ  10$
1405 002426 020011          CMP   R0,(R1)
1406          ;CHECK THE MEMORY FOR CORRECT DATA
1407          ;IF R1 IS NOT = TO R2 THEN (R1) SHOULD HAVE
1408          ;THE SAME WORD AS BAKPAT
1409          ;IN WHICH CASE GO BACK TO 12$
1410          ;*ERROR* SETUP ERROR NO. IN 22$
1411          ;*****ERROR NUMBER #17*****
1412          ;PLACE R0 ON THE STACK
1413 002430 001437          BEQ  12$
1414 002432 012767 000017 000032 MOV   #17,22$
1415          ;IF (R1) IS NOT = R0 THEN SEE IF IT IS SAME
1416          ;AS A SWAPPED R0
1417          ;IF NOT THEN A SUSPECTED DUAL ADDRESSING PROBLEM
1418          ;FOR THE BITS THAT ARE DIFFERENT IN R0 AND (R1)
1419          ;OTHERWISE THERE IS DUAL ADDRESSING FOR THE
1420          ;ENTIRE WORD
1421          ;*ERROR* SETUP ERROR NO. IN 22$
1422          ;*****ERROR NUMBER #20*****
1423          ;ADDRESSING PROBLEM IS DETECTED
1424          ;SAVE R0
1425          ;SET R0=GOOD ADDRESS FOR ERROR REPORT
1426          ;GO TO THE ERROR SUBROUTINE
1427          ;ERROR NUMBER TO BE REPORTED WILL BE PLACED HERE
1428          ;RESTORE R0
1429 002440 010046          8$: MOV   R0,-(SP)
1430 002442 000316          SWAB (SP)
1431 002444 022611          CMP   (SP)+,(R1)
1432          ;IF (R1) IS NOT = R0 THEN SEE IF IT IS SAME
1433          ;AS A SWAPPED R0
1434          ;IF NOT THEN A SUSPECTED DUAL ADDRESSING PROBLEM
1435          ;FOR THE BITS THAT ARE DIFFERENT IN R0 AND (R1)
1436          ;OTHERWISE THERE IS DUAL ADDRESSING FOR THE
1437          ;ENTIRE WORD
1438          ;*ERROR* SETUP ERROR NO. IN 22$
1439          ;*****ERROR NUMBER #20*****
1440          ;ADDRESSING PROBLEM IS DETECTED
1441          ;SAVE R0
1442          ;SET R0=GOOD ADDRESS FOR ERROR REPORT
1443          ;GO TO THE ERROR SUBROUTINE
1444          ;ERROR NUMBER TO BE REPORTED WILL BE PLACED HERE
1445          ;RESTORE R0
1446 002446 001003          BNE  9$
1447
1448 002450 012767 000020 000014 MOV   #20,22$
1449          ;*ERROR* SETUP ERROR NO. IN 22$
1450          ;*****ERROR NUMBER #20*****
1451          ;ADDRESSING PROBLEM IS DETECTED
1452          ;SAVE R0
1453          ;SET R0=GOOD ADDRESS FOR ERROR REPORT
1454          ;GO TO THE ERROR SUBROUTINE
1455          ;ERROR NUMBER TO BE REPORTED WILL BE PLACED HERE
1456          ;RESTORE R0
1457 002456 105237 000301 9$: INCB ‡#SADERR
1458 002462 010046          MOV   R0,-(SP)
1459 002464 010200          MOV   R2,R0
1460 002466 004767 003106          JSR   PC,ERROR
1461 002472 000000          22$: .WORD
1462 002474 012600          MOV   (SP)+,R0
    
```

E03

DZKMA MACY11 27(732) 04-NOV-76 07:49 PAGE 31
 DZKNAB.P11 T3 DUAL ADDRESS TEST A

1418	002476	010011			MOV	RO,(R1)	;RESTORE (R1)
1419	002500	020037	000316		CMP	RO,0#BAKPAT	;IF THE CONTROL CAME HERE FROM 15\$-2 THEN
1420	002504	001411			BEQ	12\$	
1421	002506	000407			BR	11\$;RETURN TO 11\$
1422	002510	000300		10\$:	SWAB	RO	;MAKE RO SAME AS SWAPPED BAKPAT
1423	002512	020011			CMP	RO,(R1)	;IF R1 = R2 THEN (R1) SHOULD CONTAIN A WORD
1424							;EQUAL TO SWAPPED RO
1425	002514	001404			BEQ	11\$;IN WHICH CASE GO BACK TO 11\$
1426	002516	012767	000021 177746		MOV	#21,22\$;*ERROR* SETUP ERROR NO. IN 22\$
1427							;*****ERROR NUMBER #21*****
1428	002524	000745			BR	8\$;AND GO TO 8\$
1429	002526	000300		11\$:	SWAB	RO	;RESTORE RO TO BAKPAT
1430	002530	040301		12\$:	BIC	R3,R1	;TAKE OUT THE BANK ADDRESS FROM THE ADDRESS IN R1
1431	002532	005701			TST	R1	;IF R1 IS 0 THEN PLACE A 1 IN R1
1432	002534	001001			BNE	13\$;OTHERWISE GO TO 13\$
1433	002536	005201			INC	R1	
1434	002540	006101		13\$:	ROL	R1	
1435	002542	020127	020000		CMP	R1,#20000	;IF R1 IS LESS THAN A 4K BOUNDARY
1436	002546	103720			BLO	7\$;THEN REPEAT FROM 7\$
1437	002550	000312		15\$:	SWAB	(R2)	;RESTORE (R2) TO BAKPAT
1438	002552	040302		16\$:	BIC	R3,R2	;TAKE OUT THE BANK ADDRESS FROM THE ADDRESS
1439							;STORED IN R2
1440	002554	005702			TST	R2	;IF R2 = 0 THEN MOVE A 1 TO R2
1441	002556	001001			BNE	18\$;OTHERWISE GO TO 18\$
1442	002560	005202			INC	R2	
1443	002562	006102		18\$:	ROL	R2	;SHIFT A ONE IN THE ADDRESS WORD
1444	002564	020227	020000		CMP	R2,#20000	;IS THE ADDRESS IN R2 MORE THAN THE BOUNDARY
1445							;OF 4K
1446	002570	103700			BLO	6\$;IF NOT THEN GO TO 6\$
1447	002572	060203			ADD	R2,R3	;OTHERWISE MAKE R3 POINT TO THE NEXT 4K BANK
1448	002574	020337	000340		CMP	R3,0#MAXMEM	;IF R3 IS POINTING TO A BANK THAT IS LOWER
1449							;THAN MAXMEM
1450	002600	103673			BLO	4\$;THEN REPEAT FROM 4\$
1451	002602	000337	000316	20\$:	SWAB	0#BAKPAT	
1452	002606	001656			BEQ	TST3	;REPEAT THE TEST WITH SWAPPED BAKPAT ONLY IF
1453							;THE LOWER BYTE OF BAKPAT IS 0
1454	002610	000654		END3:	BR	END2	

```

1455 ;*****
1456 ;*TEST 4 DUAL ADDRESS TEST B
1457 ;*(1) THIS TEST CHECKS FOR DUAL ADDRESSING BY WRITING
1458 ;* AND READING THE ADDRESS IN THE LOCATION AND THEN
1459 ;* WRITING AND READING ADDRESS COMPLEMENT
1460 ;*****
1461 002612 122737 000004 000404 †ST4: CMPB #4,‡#STESTN ;CHECK FOR PROPER TEST SEQUENCE
1462 002620 001403 BEQ .+10
1463 002622 004767 003310 JSR PC,SEQERR ;*ERROR* REPORT ERROR MESSAGE AND HALT AT FATHLT
1464 002626 000022 22 ;*****ERROR NUMBER 22*****
1465
1466 002630 005003 CLR R3
1467 002632 010100 1$: MOV R1,R0
1468 002634 005703 TST R3 ;IF R3 IS NOT 0 THEN STORE THE ADDRESS
1469 002636 001401 BEQ 2$ ;IN THE LOCATION
1470 002640 005100 COM R0 ;OTHERWISE STORE COMPLEMENT
1471 002642 010021 2$: MOV R0,(R1)+ ;OF THE ADDRESS
1472 002644 020105 CMP R1,R5 ;UNTIL THE HIGHEST MEMORY LOCATION IS REACHED
1473 002646 103771 BLO 1$
1474
1475 002650 020041 3$: CMP R0,-(R1) ;CHECK THE LOCATION FOR THE CORRECT CONTENTS
1476 002652 001405 BEQ 4$
1477 002654 105237 000301 INCB ‡#$ADERR ;THIS IS PROBABLY ADDRESS PROBLEM RATHER THAN
1478 ;BIT PROBLEM
1479 002660 004767 002714 JSR PC,ERROR ;*ERROR* REPORT ERROR MESSAGE
1480 002664 000023 23 ;*****ERROR NUMBER 23*****
1481
1482 002666 010100 4$: MOV R1,R0
1483 002670 162700 000002 SUB #2,R0 ;CHECK THAT THE ADDRESS IS STORED AT
1484 002674 005703 TST R3 ;LOCATION IF R3 IS NOT 0
1485 002676 001401 BEQ 5$ ;OTHERWISE CHECK FOR
1486 002700 005100 COM R0 ;ADDRESS COMPLEMENT
1487 002702 020104 5$: CMP R1,R4
1488 002704 101361 BHI 3$
1489 002706 112737 000001 000306 MOVB #1,‡#PASFLG ;SET PASFLG FOR ERROR REPORT.
1490 002714 005103 COM R3 ;COMPLEMENT THE CONTENTS OF R3
1491 002716 001345 BNE 1$ ;REPEAT TST3 IF R3, IS NON 0, ENABLING ADDRESS
1492 ;COMPLEMENT TO BE WRITTEN AND READ, OTHERWISE FALL THRU
1493 002720 000733 END4: BR END3
1494

```

```

1495          ;*****
1496          ;*TEST 5 MARCHING 1'S AND 0'S
1497          ;*(1) THIS TEST WRITES A BACK GROUND OF THE WORD STORED
1498          ;* AT BAKPAT.
1499          ;*(2) READS EVERY LOCATION FOR CORRECT DATA, SWAPS BYTES
1500          ;* AT THE LOCATION AND PROCEEDS IN MAX. TO MIN
1501          ;* DIRECTION OF MEMORY LOCATIONS.
1502          ;*(3) READS EVERY LOCATION FOR SWAPPED BAKPAT PATTERN
1503          ;* WRITES BAKPAT BACKGROUND IN THE LOCATION AND PROCEEDS
1504          ;* IN MIN. TO MAX. DIRECTION
1505          ;*(4) REPEATS STEP 2 GOING IN MIN. TO MAX. DIRECTION
1506          ;*(5) REPEATS STEP 3 GOING IN MAX. TO MIN. DIRECTION
1507
1508          ;*****
1509 002722 122737 000005 000404 †STS:  CMPB  #5,2#STESTN ;CHECK FOR PROPER TEST SEQUENCE
1510
1511 002730 001403          BEQ  .+10
1512 002732 004767 003200  JSR  PC,SEQERR ;*ERROR* REPORT ERROR MESSAGE AND HALT AT FATHLT
1513 002736 000024          JSR  24 ;*****ERROR NUMBER 24*****
1514
1515 002740 004737 000120  1$:  JSR  PC,2#WRTMEM ;GO TO WRITE THE MEMORY WITH A BACKGROUND OF THE
1516          ;WORD STORED IN BAKPAT
1517 002744 020041          2$:  CMP  R0,-(R1) ;READ THE CONTENTS OF LOCATION POINTED BY R1
1518 002746 001403          BEQ  3$ ;TO SEE IF IT HAS THE SAME VALUE AS R0
1519 002750 004767 002624  JSR  PC,ERROR ;*ERROR* REPORT ERROR MESSAGE
1520 002754 000025          JSR  25 ;*****ERROR NUMBER 25*****
1521
1522 002756 000300          3$:  SWAB R0
1523 002760 010011          MOV  R0,(R1) ;SWAP THE BYTES AT (R1)
1524 002762 021100          CMP  (R1),R0 ;READ (R1) FOR CORRECT VALUE
1525 002764 001403          BEQ  4$
1526 002766 004767 002606  JSR  PC,ERROR ;*ERROR* REPORT ERROR MESSAGE
1527 002772 000026          JSR  26 ;*****ERROR NUMBER 26*****
1528
1529
1530 002774 000300          4$:  SWAB R0 ;SWAP THE BYTES OF THE REGISTER
1531          ;CONTAINING BACKGROUND PATTERN
1532 002776 001023          BNE  9$ ;IF THE LOWER BYTE OF THE REGISTER
1533          ;IS NOT 0 THEN THE PROGRAM IS READING
1534          ;THE MEMORY TO CONTAIN A BACK GROUND OF
1535          ;BAKPAT AND WRITING THE SWAPPED WORD
1536          ;IN WHICH CASE GO TO 9$
1537
1538
1539
1540 003000 005703          5$:  TST  R3 ;R3 WAS 0 WHEN THE PROGRAM ENTERED
1541          ;THIS TEST, AND IT IS NOT ALTERED UNTIL PASFLG=3
1542          ;IF R3 EQUAL 0 THEN THE PROGRAM IS
1543          ;READING/WRITING MIN. TO MAX. OTHERWISE
1544          ;IT IS GOING IN MAX. TO MIN. DIRECTION
1545 003002 001023          BNE  10$ ;IF R3 IS NOT CLEAR THEN GO TO 10$
1546 003004 062701 000002  6$:  ADD  #2,R1 ;OTHERWISE ADD 2 TO THE CONTENTS OF R1
1547 003010 020105          CMP  R1,R5 ;COMPARE R1 WITH THE MAX. MEMORY LOCATION TO
1548          ;BE TESTED
1549 003012 103006          BHS  8$ ;IF R1>R5 THEN GO TO 8$ OTHERWISE
1550 003014 020011          7$:  CMP  R0,(R1) ;READ (R1) FOR THE CORRECT DATA

```



```

1581                                     ;*****
1582                                     ;*TEST 6          CELLS' VOLATILITY TEST
1583
1584                                     ;*(1)  THIS TEST WRITES THE MEMORY WITH A BACK GROUND OF BAKPAT
1585                                     ;*(2)  WITH PASFLG=0 THE TEST READS THE MEMORY FOR BAKPAT
1586                                     ;*      AND THEN INCREMENTS PASFLG
1587                                     ;*(3)  IT THEN READS/SWAPS BYTES/WITES A LOCATION X FOR
1588                                     ;*      OVER 2 MSEC AND THEN READS THE MEMORY FOR BAKPAT
1589                                     ;*(4)  REPEATS STEP 3 WITH X=X+4K UNTIL END OF MEMORY IS ENCOUNTERED
1590                                     ;*(5)  IT THEN INCREMENTS PASFLG AND WRITES THE MEMORY TO
1591                                     ;*      BAKPAT AND WITH PASFLG=2 IT READS MEMORY FOR ALL
1592                                     ;*      SWAPPED BAKPAT AFTER WHICH PASFLG IS INCREMENTED TO 3
1593                                     ;*(6)  REPEATS STEPS 3 AND 4 READING THE MEMORY FOR SWAPPED
1594                                     ;*      BAKPAT INSTEAD OF BAKPAT.
1595
1596                                     ;*****
1597 003070 122737 000006 000404 †ST6:  CMPB    #6,2#STESTN    ;CHECK FOR PROPER TEST SEQUENCE
1598
1599
1600 003076 001403                BEQ     .+10
1601 003100 004767 003032        JSR     PC,SEQERR    ;*ERROR* REPORT ERROR MESSAGE AND HALT AT FATHLT
1602 003104 000030                30                    ;*****ERROR NUMBER 30*****
1603
1604 003106 004737 000120        RPT6:  JSR     PC,2#WRTMEM ;GO TO WRITE THE MEMORY WITH A BACKGROUND OF THE
1605                                     ;WORD STORED AT LOCATION BAKPAT
1606 003112 005037 000306                CLR     2#PASFLG
1607 003116 010403                1$:    MOV     R4,R3    ;SET R3
1608 003120 010401                2$:    MOV     R4,R1    ;AND R1 TO THE STARTING ADDRESS OF MEMORY UNDER TEST
1609 003122 020011                3$:    CMP     R0,(R1)  ;CHECK (R1) FOR CORRECT DATA
1610 003124 001403                BEQ     4$
1611 003126 004767 002446        JSR     PC,ERROR    ;*ERROR* REPORT ERROR MESSAGE
1612 003132 000031                31                    ;*****ERROR NUMBER 31*****
1613
1614 003134 062701 000002        4$:    ADD     #2,R1    ;INCREMENT R1 BY 2
1615 003140 020105                CMP     R1,R5        ;SEE IF R1 HAS REACHED THE MAX. OF MEMORY
1616 003142 103767                BLO    3$
1617 003144 132737 000001 000306  BITB   #1,2#PASFLG  ;CHECK TO SEE IF PASFLG=0 OR 2
1618 003152 001002                BNE    5$
1619 003154 105237 000306                INCB   2#PASFLG    ;IN WHICH CASE INCREMENT PASFLG COUNTER BY 1
1620
1621 003160 020305                5$:    CMP     R3,R5        ;SEE IF R3 HAS REACHED THE MAX. OF THE MEMORY
1622 003162 103012                BHIS   7$
1623 003164 012702 037776        MOV     #37776,R2   ;WRITE INTO 1 LOC FOR >2MS (ABOUT 100MS)
1624 003170 000313                6$:    SWAB   (R3)
1625 003172 005302                DEC     R2
1626 003174 001375                BNE    6$
1627 003176 010337 000354        MOV     R3,2#SAVLOC ;SAVE LOCATION WRITTEN FOR 2MS FOR ERROR REPORT.
1628 003202 062703 020000        ADD     #20000,R3  ;BY ADDING 1 TO THE 14TH ADDRESS BIT CAUSE
1629                                     ;R3 TO POINT TO A LOCATION IN THE NEXT
1630                                     ;4K BANK OF MEMORY
1631 003206 000744                BR     2$
1632 003210 105237 000306        7$:    INCB   2#PASFLG ;MAKE PASFLG=2
1633 003214 000337 000316        SWAB   2#BAKPAT    ;IF BAKPAT IS NOT BEING SWAPPED FOR THE 2ND
1634 003220 001732                BEQ     RPT6        ;THEN GO BACK TO THE LOCATION RPT6
1635 003222 000721                END6:  BR     ENDS
1636
    
```

J03

DZKMA MACY11 27(732) 04-NOV-76 07:49 PAGE 36
DZKMA.B.F11 T6 CELLS' VOLATILITY TEST

1637

```

1638                                     ;:*****
1639                                     ;*TEST 7          SHIFTING DIAGONAL
1640
1641                                     ;*(1)  THIS TEST WRITES THE MEMORY WITH A BACKGROUND OF BAKPAT
1642                                     ;*(2)  IT WRITES A DIAGONAL OF SWAPPED BAKPAT THROUGH EACH MEMORY BANK
1643                                     ;*(3)  READS THE MEMORY FOR CORRECT DATA
1644                                     ;*(4)  SHIFTS THE DIAGONAL AND REPEATS STEP 3 UNTIL THE
1645                                     ;*     DIAGONAL HAS BEEN SHIFTED 64 TIMES
1646                                     ;*(5)  WRITES A BACKGROUND OF SWAPPED BAKPAT, A DIAGONAL OF
1647                                     ;*     BAKPAT AND REPEATS FROM STEP 3
1648                                     ;:*****
1649 003224 122737 000007 000404 †ST7:  CMPB  #7,‡#STESTN  ;CHECK FOR PROPER TEST SEQUENCE
1650
1651 003232 001403          BEQ  .+10
1652 003234 004767 002676      JSR  PC,SEQERR  ;*ERROR* REPORT ERROR MESSAGE AND HALT AT FATHLT
1653 003240 000032          32      ;*****ERROR NUMBER 32*****
1654
1655 003242 005037 000306      2$:  CLR  ‡#PASFLG
1656 003246 010337 000304      MOV  R3,‡#LOWBNK  ;LOWBNK CONTAINS ADDRESS OF THE LOWEST LOCATION
1657                                     ;IN THE 4K BANK THAT CAN BE TESTED
1658 003252 010302          MOV  R3,R2
1659 003254 052702 017776      BIS  #17776,R2  ;R2 CONTAINS THE ADDRESS OF THE TOP OF THE BANK
1660 003260 005722          TST  (R2)+      ;ADD 2 TO R2
1661 003262 020502          CMP  R5,R2
1662 003264 103001          BHIS 4$        ; IF R2 IS GREATER THAN R5 THEN GO TO 4$
1663 003266 010502          MOV  R5,R2      ;NOW R2 CONTAINS THE ADDRESS OF THE HIGHEST LOCATION
1664                                     ;THAT CAN BE TESTED
1665 003270 010337 000302      4$:  MOV  R3,‡#STRTDI  ;LOAD STRTDI WITH THE STARTING ADDRESS OF THE
1666                                     ;DIAGONAL
1667 003274 013701 000304      MOV  ‡#LOWBNK,R1  ;R1 IS NOW POINTING TO THE LOWEST LOCATION IN THE 4K
1668                                     ;BANK
1669 003300 013700 000316      6$:  MOV  ‡#BAKPAT,R0  ;STORE THE CONTENTS OF BAKPAT IN R0
1670 003304 020103          CMP  R1,R3      ;IS R1 POINTING TO A LOCATION IN THE DIAGONAL ?
1671 003306 001010          BNE  10$        ;IF NOT THEN GO TO 10$
1672 003310 062703 000002      ADD  #2,R3      ;THE FOLLOWING CODE IS USED TO PLACE THE
1673 003314 032703 000176      BIT  #176,R3    ;ADDRESS OF THE NEXT LOCATION IN THE DIAGONAL
1674 003320 001402          BEQ  8$         ;IN R3
1675 003322 062703 000200      ADD  #200,R3
1676 003326 000300      8$:  SWAB R0        ;DIAGONAL WILL CONTAIN SWAPPED BACKGROUND PATTERN
1677 003330 132737 000001 000306 10$: BITB #1,‡#PASFLG  ;CONTENTS OF LOCATION PASFLG WILL BE EVEN IF THE
1678                                     ;MEMORY IS BEING WRITTEN AND IT WILL BE ODD
1679                                     ;IF IT IS ONLY BEING READ
1680 003336 001001          BNE  12$        ;IF IT IS BEING READ ONLY THEN GO TO 12$
1681 003340 010011          MOV  R0,(R1)    ;OTHERWISE WRITE THE MEMORY WITH THE CONTENTS
1682                                     ;OF R0
1683 003342 020011      12$:  CMP  R0,(R1)    ;CHECK THE LOCATION POINTED BY R1 TO CONTAIN
1684                                     ;PROPER DATA
1685 003344 001403          BEQ  14$        ;IF IT IS OK THEN GO TO 14$
1686 003346 004767 002226      JSR  PC,ERROR   ;*ERROR* REPORT ERROR MESSAGE
1687 003352 000033          33      ;*****ERROR NUMBER 33*****
1688
1689 003354 062701 000002      14$:  ADD  #2,R1      ;CAUSE R1 TO POINT TO THE NEXT MEMORY LOCATION
1690 003360 020102          CMP  R1,R2      ;IS IT THE END OF THE BANK ?
1691 003362 103746          BLO  6$         ;IF NOT THEN GO TO 6$
1692 003364 005237 000410      16$:  INC  ‡#SDEVCT  ;TELL APT WE ARE STIL RUNNING OKAY
1693 003370 105237 000306      INCB ‡#PASFLG

```

DZKMA - MACY11 27(732) 04-NOV-76 07:49 PAGE 38
 DZKNAB.P11 T7 SHIFTING DIAGONAL

1694	003374	013703	000302		MOV	2#STRTDI,R3	;LOAD R3 WITH THE STARTING ADDRESS OF THE DIAGONAL
1695	003400	132737	000001	000306	BITB	#1,2#PASFLG	;HAS THE READ OF THE MEMORY BEEN DONE ?
1696	003406	001330			BNE	4\$;IF NOT THEN GO TO 4\$
1697	003410	005723			TST	(R3)+	;ADD 2 TO THE STARTING ADDRESS OF THE DIAGONAL
1698	003412	020302			CMP	R3,R2	;AND UNLESS THE END OF THE BANK IS REACHED
1699	003414	103003			BHIS	18\$	
1700	003416	105737	000306		TSTB	2#PASFLG	;OR THE DIAGONAL HAS BEEN ROTATED 64 TIMES
1701	003422	100322			BPL	4\$;REPEAT FROM 4\$
1702	003424	013703	000304	18\$:	MOV	2#LOWBNK,R3	;MAKE R3 POINT TO THE LOWEST LOCATION IN THE
1703							;IN THE BANK UNDER TEST
1704	003430	000337	000316		SWAB	2#BAKPAT	
1705	003434	001715			BEQ	4\$;AND IF THE TEST HAS NOT BEEN PERFORMED WITH THE
1706							;SWAPPED BACK GROUND PATTERN THEN GO TO 4\$
1707	003436	010203			MOV	R2,R3	;MAKE THE PRESENT HIGH BOUNDARY AS THE NEXT
1708							;LOW BOUNDARY
1709	003440	020205			CMP	R2,R5	;UNLESS THE PRESENT HIGH BOUNDARY IS ALSO THE
1710							;HIGH BOUNDARY FOR THE MEMORY UNDER TEST
1711	003442	103677			BLO	2\$	
1712	003444	000666		END7:	BR	END6	

N03

DZKMA MACY11 27(732) 04-NOV-76 07:49 PAGE 40
 DZKMA8.P11 T10 READ RECOVERY GALLOPING TEST THROUGH EVERY 64TH CELL

1769	003544	010011							MOV	RO, (R1)	;RESTORE THE CONTENTS OF (R1)
1770	003546	010100							MOV	R1, RO	;RESTORE RO
1771											
1772	003550	000310							8\$: SWAB	(RO)	
1773	003552	031011							10\$: BIT	(RO), (R1)	;CHECK TO SEE THAT NONE OF THE BITS SET ;IN (RO) ARE SET IN (R1) AND VICE VERSA
1774											;THE ONLY EXCEPTION TO THIS WILL BE WHEN RO=R1
1775	003554	020001							CMP	RO, R1	
1776											
1777	003556	001412							BEQ	12\$	
1778	003560	021137	000316						CMP	(R1), @#BAKPAT	;CHECK THAT (R1) HAS BAKPAT IN IT
1779	003564	001407							BEQ	12\$	
1780	003566	010046							MOV	RO, -(SP)	;SAVE RO ON STACK
1781	003570	013700	000316						MOV	@#BAKPAT, RO	;PLACE THE PATTERN WORD IN RO
1782	003574	004767	002000						JSR	PC, ERROR	;*ERROR* REPORT ERROR MESSAGE
1783	003600	000036							36		;*****ERROR NUMBER 36*****
1784											
1785	003602	012600							MOV	(SP)+, RO	;RESTORE RO
1786	003604	021037	000320						12\$: CMP	(RO), @#SWAPAT	;CHECK THAT (RO) HAS SWAPPED BAKPAT IN IT
1787	003610	001412							BEQ	14\$	
1788	003612	010146							MOV	R1, -(SP)	;SAVE R1 ON THE STACK
1789	003614	010001							MOV	RO, R1	;MAKE R1 POINT TO THE FAILING LOCATION
1790	003616	013700	000320						MOV	@#SWAPAT, RO	;LOAD RO WITH THE EXPECTED RESULT IN (R1)
1791	003622	004767	001752						JSR	PC, ERROR	;*ERROR* REPORT ERROR MESSAGE
1792	003626	000037							37		;*****ERROR NUMBER 37*****
1793											
1794	003630	010011							MOV	RO, (R1)	;RECOVER (R1) FROM THE ERROR
1795	003632	010100							MOV	R1, RO	;RESTORE RO
1796	003634	012601							MOV	(SP)+, R1	;AND RESTORE R1
1797	003636	122737	000011	000404					14\$: CMPB	#11, @#STESTN	;IS THE PROGRAM EXECUTING TEST # 11 ?
1798	003644	001402							BEQ	16\$;IF SO THEN GO TO 16\$
1799	003646	062701	000176						ADD	#176, R1	
1800	003652	062701	000002						16\$: ADD	#2, R1	;MAKE R1 POINT TO THE NEXT ADJACENT CELL
1801	003656	020102							CMP	R1, R2	;AND IF R1 HAS NOT REACHED THE END OF THE BOUNDARY
1802	003660	103734							BLO	10\$;THEN REPEAT FROM 10\$
1803	003662	000320							SWAB	(RO)+	;RESTORE THE LOCATION FOR WHICH THE GALLOPING TEST ;WAS BEING PERFORMED
1804											
1805	003664	122737	000011	000404					CMPB	#11, @#STESTN	;IS IT TEST 11 ?
1806	003672	001407							BEQ	17\$;IF SO THEN GO TO 17\$
1807	003674	005723							TST	(R3)+	;OTHERWISE INCREMENT R3 BY 2
1808	003676	062716	000002						ADD	#2, (SP)	;FOR EVERY ROW/COLUMN TESTED ADD 2
1809	003702	105716							TSTB	(SP)	
1810	003704	100002							BPL	17\$;UNTIL (SP) IS 200
1811	003706	161603							SUB	(SP), R3	;SUBTRACT 200 FROM R3
1812	003710	005016							CLR	(SP)	
1813	003712	032700	000177						17\$: BIT	#177, RO	;AT A 64TH CALL BOUNDARY?
1814	003716	001002							BNE	18\$;BRANCH IF NO
1815	003720	005237	000410						INC	@#SDEVCT	;TELL APT WE ARE STILL RUNNING
1816	003724	020002							18\$: CMP	RO, R2	;IF RO HAS NOT REACHED THE END OF THE BOUNDARY
1817	003726	103674							BLO	6\$;THEN REPEAT FROM 6\$
1818	003730	162603							SUB	(SP)+, R3	;RESTORE SP AND R3
1819	003732	000337	000320						SWAB	@#SWAPAT	
1820	003736	000337	000316						SWAB	@#BAKPAT	
1821	003742	001660							BEQ	2\$;IF THE LOWER BYTE OF BAKPAT IS 0 THEN REPEAT FROM 2\$
1822	003744	010203							MOV	R2, R3	;OTHERWISE MAKE THE PRESENT HIGH BOUNDARY AS THE ;NEXT LOW BOUNDARY
1823											
1824	003746	020205							CMP	R2, R5	

804

DZKMA MACY11 27(732) 04-NOV-76 07:49 PAGE 41
DZKMA8.P11 T10 READ RECOVERY GALLOPING TEST THROUGH EVERY 64TH CELL

1825	003750	001410		BEQ	END10		: IF PREVIOUS HIGH BOUNDARY WAS THE END OF THE
1826							: TEST BOUNDARY THEN EXIT THE TEST
1827	003752	032702	017776	BIT	#17776,R2		: WAS IT A 4K BOUNDARY ?
1828	003756	001025		BNE	RPT11		: IF NOT THEN WE WERE PERFORMING TEST 11 WITH LONG
1829							: GALLOPING TEST DISABLED
1830	003760	122737	000011 000404	CMPB	#11,0#STESTN		: IF IT IS TEST # 11 THEN GC TO REPEAT TEST 11
1831	003766	001421		BEQ	RPT11		
1832	003770	000636		BR	RPT10		: OTHERWISE REPEAT TEST 10
1833	003772	000624		BR	END7		
1834				END10:			
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
    
```

 ;*TEST 11 READ RECOVERY LONG GALLOPING/FAST GALLOPING TEST
 ;*(1) THIS TEST WRITES MEMORY WITH BAKPAT
 ;*(2) THE TEST BEGINS AT THE LOWEST LOCATION BEING TESTED
 ;* (LETS NAME IT 'B')
 ;*(3) 'A'+ 'B' [MOVE THE ADDRESS OF 'B' TO THE POINTER FOR LOCATION 'A']
 ;*(4) SWAPS BYTES FOR LOCATION 'A'
 ;*(5) READS 'A', READS 'B'
 ;*(6) 'B'='B'+3
 ;*(7) IF GALLOPING OPTION BIT AT \$SWREG IS HIGH THEN STEPS 4 AND 5
 ;* ARE REPEATED UNTIL 'B' REACHES THE HIGHEST MEMORY LOCATION
 ;* OF THE 4K BANK IN WHICH 'A' IS RESIDING, THEN 'A' IS
 ;* DECREMENTED BY 2 AND AFTER MAKING 'B' TO POINT TO THE LOWEST
 ;* LOCATION OF THE 4K MEMORY BANK CONTAINING 'A' STEPS 3,4,5 AND
 ;* 6 ARE REPEATED UNTIL 'A' EQUALS THE END OF THE ENTIRE MEMORY
 ;*(8) IF GALLOPING OPTION BIT IS NOT HIGH THEN STEPS 4 AND 5 ARE
 ;* REPEATED UNTIL 'B' IS POINTING TO A CELL IN THE NEXT COLUMN
 ;* IF SEQUENTIAL CELLS LIE ALONG THE ROW, OR THE NEXT ROW
 ;* IF SEQUENTIAL CELLS LIE ALONG THE COLUMN, AT WHICH TIME
 ;* STEPS 2,3,4,5 AND 7 ARE REPEATED UNTIL THE END OF THE MEMORY
 ;*(9) TEST IS REPEATED FOR THE OPPOSITE BACKGROUND DATA
 ;*(10) IN THIS TEST R0 POINTS TO LOCATION 'A', R1 TO LOCATION
 ;* 'B', R2 TO THE HIGHEST LOCATION AND R3 TO THE LOWEST
 ;* LOCATION IN A 64/4K CELL BOUNDARY
 ;*(11) MOST OF THE CODE USED BY TEST 10 IS ALSO USED BY THIS TEST

```

1865 003774 122737 000011 000404 †ST11: CMPB #11,0$STESTN ;CHECK FOR PROPER TEST SEQUENCE
1866
1867 004002 001403 BEQ +10
1868 004004 004767 002126 JSR PC,SEQERR ;*ERROR* REPORT ERROR MESSAGE AND HALT AT FATHLT
1869 004010 000040 40 ;*****ERROR NUMBER 40*****
1870
1871 004012 010402 MOV R4,R2 ;MAKE R2 TO POINT TO THE LOWEST LOCATION
1872 ;UNDER TEST
1873 004014 105777 174430 TSTB @SWR ;LONG GALLOP ENABLED?
1874 004020 100004 BPL RPT11 ;BRANCH IF NO
1875 004022 004767 002540 JSR PC,PNTMES ;TYPE "GLP"
1876 004026 046107 000120 .ASCIZ /GLP/
1877 004032 105777 174412 RPT11: TSTB @SWR ;LONG GALLOPING ENABLED?
1878 004036 100613 BMI RPT10 ;BRANCH IF YES
1879 ;TO RPT10
1880 004040 052702 000176 BIS #176,R2 ;OTHERWISE SET THE LOW ORDER BITS OF THE ADDRESS
1881 ;TO GET THE HIGH BOUNDARY
1882
1883 004044 000612 BR GALLOP ; PERFORM GALLOPING TEST
    
```

```

1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906 004046 122737 000012 000404
1907 004054 001403
1908 004056 004767 002054
1909 004062 000041
1910
1911
1912 004064 012702 000001
1913 004070 012703 000400
1914 004074 112737 000001 000306 15:
1915 004102 010401 25:
1916
1917 004104 013700 000316 45:
1918 004110 030201
1919 004112 001004
1920 004114 030301
1921 004116 001404
1922 004120 005100 65:
1923
1924
1925
1926
1927 004122 000402
1928 004124 030301 85:
1929
1930 004126 001774
1931 004130 132737 000002 000306 125:
1932 004136 001001 145

```

;TEST 12 WORST CASE TESTING FOR CORE MEMORY
;(1) STARTING FROM THE LOWEST LOCATION UNDER TEST THE MEMORY
; IS WRITTEN WITH A BACKGROUND OF BAKPAT, HOWEVER LOCATIONS
; HAVING ADDRESS SUCH THAT EXCLUSIVE OR OF ADDRESS BITS 1 &
; 8 = 1 ARE WRITTEN TO A VALUE OF SWAPPED BAKPAT
;(2) STARTING FROM THE LOWEST LOCATION THE MEMORY IS CHECKED
; TO CONTAIN THE CORRECT DATA AS EXPLAINED IN STEPS 3 & 4,
; UNTILL THE HIGHEST LOCATION UNDER TEST IS REACHED
;(3) READ EACH LOCATION FOR THE CORRECT CONTENT
;(4) COMPLEMENT THE LOCATION AND READ IT; COMPLEMENT THE LOCATION
; BACK TO ITS ORIGINAL VALUE AND READ IT AGAIN
;(5) STARTING FROM THE HIGHEST LOCATION UNDER TEST REPEAT STEPS
; 3 & 4 UNTIL THE LOWEST LOCATION UNDER TEST IS REACHED
;(6) REPEAT STEPS 1-5, HOWEVER THIS TIME LOCATIONS WITH XOR
; OF ADDRESS BITS 8 & 13 = 1 ARE WRITTEN TO SWAPPED BAKPAT
;(7) REPEAT STEPS 1-5, HOWEVER THIS TIME LOCATIONS WITH XOR
; OF ADDRESS BITS 3 & 9 = 1 ARE WRITTEN TO SWAPPED BAKPAT
;(8) REPEAT STEPS 1-7 WITH A BACKGROUND OF SWAPPED BAKPAT AND
; THE LOCATIONS TO BE WRITTEN TO SWAPPED BAKPAT WRITTEN TO
; BAKPAT.

TST12: CMPB #12,2#TESTN ;CHECK FOR PROPER TEST SEQUENCE
BEQ .+10
JSR PC,SEQERR ;*ERROR* REPORT ERROR MESSAGE AND HALT AT FATHLT
41 ;*****ERROR NUMBER 41*****
;PREPARE TO TAKE THE EXCLUSIVE OR OF ADDRESS BITS 1
;AND 8
;INITIALIZE THE COUNTER FOR THE SUBTEST
;PLACE THE STARTING ADDRESS OF MEMORY UNDER
;TEST IN R1
;CHECK TO SEE IF ADDRESS BIT STORED IN R2 IS SET
;IF IT IS SET THEN GO TO 85
;CHECK TO SEE IF ADDRESS BIT POINTED BY R3 IS SET
;IF IT IS NOT SET THEN GO TO 125
;COME HERE ONLY IF EXCLUSIVE OR OF ADDRESS BITS
;POINTED BY R2 & POINTED BY R3 = 1 IN WHICH
;CASE PREPARE TO WRITE THE LOCATION
;WITH A COMPLEMENT OF LOCATIONS NOT MEETING
;THIS CONDITION
;COME HERE IF ADDRESS BIT POINTED BY R2 IS 1 AND
;CHECK ADDRESS BIT POINTED BY R3
;IF ADDRESS BIT POINTED BY R3 IS 0 THEN GO TO 65
;IS IT 2ND OR 3RD PASS OF THE SUBTEST ?
;IF SO THEN READ THE MEMORY

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

004320 122737 000013 000404
 004326 001403
 004330 004767 001602
 004334 000044
 004336 012702 010247
 004342 012700 177667
 004346 010546
 004350 010446
 004352 000241
 004354 006005
 004356 006004
 004360 160405
 004362 006005
 004364 103002
 004366 062716 000002
 004372 012604
 004374 012605
 004376 010403
 004400 000405

```

*****
*TEST 13 WRITE RECOVERY TEST
* THIS TEST DIFFERS FROM 0-12 IN THAT IT CONSISTS OF A SMALL TEST PROGRAM
* ACTUALLY RUNNING IN THE 4K BANK UNDER TEST.
* THE PROGRAM IS SELF MODIFYING AND MAY BE DIFFICULT TO DEBUG.
* TO AID IN THE DEBUG, BEFORE A A BANK IS ENTERED "TST13 BANK XX"
* IS TYPED. THIS WILL ALLOW THE USER TO AT LEAST SEE WHICH MEMORY
* BANK FAILED.
* THE TEST CONSISTS OF 1/2 OF THE BANK STORED WITH "MOV R2, -(PC)"
* AND THE OTHER 1/2 CONTAINING "177667". "177667" IS THE COMPLEMENT
* OF "JMP (R0)" INSTRUCTION.
* R2 CONTAINS "COM -(R1)" INSTRUCTION ON ENTRY TO THE BANK AND R1 CONTAINS
* THE HIGHEST TEST ADDRESS IN THAT BANK. THE HIGHEST TEST ADDRESS IS
* USUALLY ON 4K BOUNDARIES. WHEN TESTING BANK 0 RELOCATED, HOWEVER
* R1 CONTAINS THE FIRST FREE TEST ADDRESS BELOW THE DIAGNOSTIC.
* IF YOU UNDERSTAND THIS SO FAR THE REST IS EASY.
* THE TEST EXECUTION IS AS FOLLOWS:
* 1. THE "MOV R2, -(PC)" INSTRUCTION EXECUTES STORING
* THE CONTENTS OF R2 IN THE ADDRESS IT VACATED (DUE TO -(PC)).
* 2. SINCE R2 CONTAINS A "COM -(R1)" INSTRUCTION IT COMPLEMENTS
* THE HIGHEST ADDRESS UNDER TEST. THIS ADDRESS CONTAINED
* "177667" SO AFTER THE COM -(R1) IT EQUALS 110
* CLEVERLY THIS IS THE "JMP (R0)" INSTRUCTION.
* 3. THIS SEQUENCE CONTINUES UNTIL THE "MOV R2, -(PC) INSTRUCTIONS
* REACH THE MIDDLE OF THE TEST BANK. THEN THE "JMP (R0)" INSTRUCTION IS
* AND EXECUTED. R0 CONTAINED THE RETURN ADDRESS BACK
* TO TEST 13.
* 4. THESE STEPS ARE REPEATED FOR EACH BANK UNDER TEST.
*****
TST13: CMPB #13, #STESTN ;CHECK FOR PROPER TEST SEQUENCE
        BEQ .+10
        JSR PC, SEQERR ;*ERROR* REPORT ERROR MESSAGE AND HALT AT FATHLT
        44 ;*****ERROR NUMBER 44*****
15: MOV #10247, R2 ;PLACE THE OP CODE OF INSTRUCTION MOV R2, -(PC)
        ;IN R2.
        MOV #177667, R0 ;PLACE THE COMPLEMENT OF THE INSTRUCTION
        ;JMP (R0) IN R0
        ;INSURE LOWEST TEST ADDRESS TO END OF 4K SEGMENT IS MULTIPLE OF 2
        ;SINCE THE TEST STORES "MOV R2, -(PC) IN 1/2 AND 177667 IN THE OTHER 1/2
25: MOV R5, -(SP) ;SAVE R5
        MOV R4, -(SP) ;STORE LOWEST ADDRESS ON STACK
295: CLC
        ROR R5 ;MAKE POSITIVE BYTE COUNT OF HIGH ADDRESS
        ROR R4 ;DO SAME FOR LOWEST ADDRESS
        SUB R4, R5 ;GET DIFFERENCE OF LOWEST ADDRESS AND HIGHEST
        ROR R5 ;IF DIFFERENCE IS ODD THEN R4 IS AT LOWEST ADDRESS
        BCC 30$ ;BRANCH IF R4 IS AT LOWEST TEST ADDRESS.
        ADD #2, (SP) ;INCREASE LOWEST TEST ADDRESS BY 2
30$: MOV (SP)+, R4 ;RESTORE R4 (POSSIBLY INCREASED BY 2 FROM ENTRY)
        MOV (SP)+, R5 ;RESTORE HIGHEST TEST ADDRESS
        MOV R4, R3 ;PLACE THE LOWEST LOCATION UNDER TEST
        ;IN R3
BR 28$ ;LEAVE LOW BITS OF R3 ALONE FIRST TIME IN CASE BANK 0
    
```

DZKMA MACY11 27(732) 04-NOV-76 07:49 PAGE 46
 DZKNAB.P11 T13 WRITE RECOVERY TEST

```

2033 004402 042703 017776      3$:   BIC      #17776,R3      ;CAUSE R3 TO POINT TO THE LOWEST LOCATION
2034                                     ;IN THE 4K BANK UNDER TEST
2035 004406 105737 000405      TSTB     @#REL      ;ARE WE RELOCATED?
2036 004412 100504          BMI      14$      ;BRANCH IF YES-TEST BANK0 ONLY-
2037 004414 020305      28$:   CMP      R3,R5      ;IF R3 IS HIGHER THAN THE HIGHEST LOCATION
2038 004416 103102          BHIS     14$      ;UNDER TEST THEN EXIT
2039                                     ;IF R5 LESS THAN 20000 THEN WE ARE TESTING BANK0 RELOCATED IN BANK0
2040 004420 020527 020000      CMP      R5,#20000 ;IS HIGHEST TEST ADDRESS BELOW 4K?
2041 004424 103002          BHIS     31$      ;BRANCH IF NO
2042 004426 010501          MOV      R5,R1      ;SET R1 TO HIGHEST TEST ADDRESS IN BANK0
2043 004430 000405          BR       32$
2044
2045 004432 010301      31$:   MOV      R3,R1      ;SET R1 TO LOWEST CURRENT TEST ADDRESS
2046 004434 042701 017776      BIC      #17776,R1 ;CLEAR LOW ORDER ADDRESS BITS
2047 004440 062701 020000      ADD      #20000,R1 ;CAUSE R1 TO POINT TO THE HIGHEST LOCATION+2
2048                                     ;OF THE 4K BANK BEING POINTED BY R3
2049 004444 020137 000340      32$:   CMP      R1,@#MAXMEM ;IF R1 IS HIGHER THAN MAX. OF THE
2050 004450 101065          BHI      14$      ;MEMORY+2 ALTHOUGH R3 IS LESS THAN R5
2051                                     ;THEN THE HIGHEST LOCATION UNDER
2052                                     ;TEST IS NOT IN A 4K BANK EXIT
2053
2054 004452 132737 000001 000306      BITB     #1,@#PASFLG ;IS THE LOWEST BIT OF LOCATION PASFLG
2055 004460 001101          BNE      16$      ;SET? IN WHICH CASE BACK GROUND HAS
2056                                     ;ALREADY BEEN WRITTEN AND WRITE RECOVERY
2057                                     ;TEST IS BEING PERFORMED
2058
2059 004462 020304      4$:   CMP      R3,R4      ;OTHERWISE WRITE THE BACKGROUND
2060 004464 103430          BLO      8$      ;DEFINED AT STEP 3.
2061 004466 105737 000307      TSTB     @#PASFLG+1 ;IS THE TEST JUST DOING READ, I.E.
2062 004472 001002          BNE      6$      ;IS THE PASFLG+1 LOCATION NON ZERO? IF SO
2063                                     ;THEN GO TO 6$
2064 004474 012713 010247      MOV      #10247,(R3) ;WRITE THE LOCATION WITH THE OP CODE FOR MOV R2,-(PC)
2065 004500 020213      6$:   CMP      R2,(R3) ;READ (R3) TO CONTAIN CORRECT DATA
2066 004502 001421          BEQ      8$
2067 004504 010046          MOV      R0,-(SP) ;SAVE R0
2068 004506 010146          MOV      R1,-(SP) ;AND R1 ON THE STACK
2069 004510 010301          MOV      R3,R1
2070 004512 010200          MOV      R2,R0
2071 004514 004767 001060      JSR      PC,ERROR ;SET R0= GOOD DATA FOR ERROR PRINTOUT
2072 004520 000045          45      ;*ERROR* REPORT ERROR MESSAGE
2073                                     ;*****ERROR NUMBER 45*****
2074
2075 004522 012601          MOV      (SP)+,R1 ;RESTORE R1
2076 004524 012600          MOV      (SP)+,R0 ;AND R0
2077 004526 105737 000306      TSTB     @#PASFLG ;IF PASFLG IS 0 AND THE MEMORY DOES NOT HAVE
2078                                     ;THE PROPER DATA THEN WE DON'T WANT TO GO AND
2079                                     ;EXECUTE THE INSTRUCTIONS STORED IN MEMORY UNDER
2080 004532 001005          BNE      8$      ;TEST
2081                                     ;BRANCH IF PASFLG NOT =0
2082 004534 010200          MOV      R2,R0 ;SAVE FOR ERROR REPORT
2083 004536 004767 001036      JSR      PC,ERROR ;*ERROR* REPORT ERROR MESSAGE
2084 004542 000046          46      ;*****ERROR NUMBER 46*****
2085
2086 004544 000664          BR       END12    ;ABORT TST 13.
2087
2088 004546 062703 000002      8$:   ADD      #2,R3      ;INCREMENT R3 BY 2

```

H04

DZKMA MACY11 27(732) 04-NOV-76 07:49 PAGE 47
 DZKMA8.P11 T13 WRITE RECOVERY TEST

2089	004552	162701	000002		SUB	#2,R1	;DECREMENT R1 BY 2
2090	004556	020105			CMP	R1,R5	;WRITE THE BACKGROUND DEFINED AT STEP 4.
2091	004560	103014			BHIS	12\$	
2092	004562	020103			CMP	R1,R3	;HAS STORING THE 177667 REACHED WHERE "MOV R2,-(PC) IS?
2093	004564	103405			BLO	10\$;BRANCH IF YES DON'T DESTROY THE MOV R2,-(PC) IS.
2094	004566	105737	000307		TSTB	2#PASFLG+1	;IS THE THE READ ONLY CHECK PASS?
2095	004572	001002			BNE	10\$;BRANCH IF YES
2096	004574	012711	177667		MOV	#177667,(R1)	;WRITE THE LOCATION WITH THE COMPLEMENT OF THE
2097							;OP CODE JMP (R0)
2098	004600	020011		10\$:	CMP	R0,(R1)	;READ R1 TO CONTAIN CORRECT DATA
2099	004602	001403			BEQ	12\$	
2100	004604	004767	000770		JSR	PC,ERROR	;*ERROR* REPORT ERROR MESSAGE
2101	004610	000047				47	;*****ERROR NUMBER 47*****
2102							
2103	004612	020301		12\$:	CMP	R3,R1	;IF WE HAVE NOT REACHED THE MIDDLE OF 4K BANK
2104	004614	103722			BLO	4\$;THEN REPEAT FROM 4\$
2105							
2106							;RETURN HERE AFTER PROGRAM RUN IN BANK UNDER TEST
2107							
2108	004616	062703	020000	13\$:	ADD	#20000,R3	;OTHERWISE GO TO THE NEXT 4K BANK
2109	004622	000667			BR	3\$	
2110							
2111	004624	122737	000001 000306	14\$:	CMPB	#1,2#PASFLG	;THE PROGRAM CONTROL COMES HERE AS FOLLOWS
2112							;1-PASFLG=0, PROGRAM HAS JUST COMPLETED A
2113							;WRITE/READ CYCLE FOR THE BACK GROUND
2114							;AND WANTS TO BEGIN THE WRITE RECOVERY TEST
2115	004632	001440			BEQ	24\$;2-PASFLG=1, PROGRAM HAS JUST COMPLETED
2116							;THE WRITE RECOVERY TEST AND WANTS TO
2117							;READ MEMORY FOR CORRECT DATA
2118	004634	103630			BLO	END12	;3-PASFLG=2, PROGRAM HAS CORRECTLY READ THE
2119							;MEMORY AND WANTS TO GO THE NEXT TEST.
2120							
2121	004636	105137	000307		COMB	2#PASFLG+1	;ENTER HERE WITH PASFLG=0, ON THE FIRST ENTRY
2122							;ENABLE READ ONLY FOR THE MEMORY AND ON THE SECOND
2123							;ENTRY DISABLE READ ONLY
2124	004642	001241			BNE	2\$	
2125	004644	012702	005141		MOV	#5141,R2	;PLACE THE OP CODE FOR INSTRUCTION COM -(R1)
2126							;IN R2
2127	004650	012700	177740		MOV	#13\$-.-6,R0	;PLACE THE RETURN ADDRESS IN R0 AS 13\$
2128	004654	060700			ADD	PC,R0	;THUS WHEN THE READ RECOVERY TEST REACHES
2129							;THE MIDDLE OF THE 4K MEMORY THEN THE
2130							;INSTRUCTION EXECUTED WILL BE JMP (R0)
2131							;BRANCHING THE PROGRAM TO 13\$
2132	004656	105237	000306	15\$:	INCB	2#PASFLG	;INCREMENT PASFLG BY 1.
2133	004662	000631			BR	2\$	
2134							
2135	004664	032777	000020 173556	16\$:	BIT	#20,2SWR	;HAS THE PRINTOUTS BEEN SUPRESSED ?
2136	004672	001017			BNE	18\$;IF SO THEN GO TO 18\$
2137	004674	105737	000042		TSTB	2#42	;IS THE PROGRAM RUNNING UNDER ACT?
2138	004700	001014			BNE	18\$;BRANCH IF YES
2139	004702	004767	001660		JSR	PC,PNTMES	;TYPE THE BANK UNDER TEST
2140	004706	051524	030524 020063		.ASCIZ	/TST13 BANK/	
2141	004714	040502	045516 000				
2142		004722			.EVEN		
2143	004722	004767	002476		JSR	PC,GETBNK	;GET BANK NO. UNDER TEST INTO DECDRD FOR PRINT.
2144	004726	004767	001662		JSR	PC,\$TPDEC	;TYPE BANK NO. UNDER TEST

2156	004746	012737	000377	000316	RELOC:	MOV	#377, @#BAKPAT	
2157	004754	105737	000276			TSTB	@#MMAVA	; IS THE MEMORY MANAGEMENT BEING TESTED ?
2158	004760	001065				BNE	CONTMM	; IF SO THEN GO TO CONTMM AND CONTINUE TESTING
2159								; MEMORY MANAGEMENT
2160	004762	032777	001000	173460		BIT	#1000, @SWR	; RELOCATION WANTED?
2161	004770	001046				BNE	CKDONE	; BRANCH IF NO
2162	004772	105737	000405			TSTB	@#REL	; IF THE PROGRAM HAS ALREADY BEEN RELOCATED THEN ALSO
2163	004776	100420				BMI	RELOER	; PLACE THE PROGRAM BACK IN LOWER CORE
2164	005000	112737	000200	000405		MOVB	#200, @#REL	; OTHERWISE PREPARE TO RELOCATE
2165								
2166								; RELOCATE THE DIAGNOSTIC TO HIGHEST AVAILABLE MEMORY
2167								
2168								
2169	005006	004767	001554			JSR	PC, PNTMES	; TYPE "RELOC"
2170	005012	042522	047514	000103		.ASCIZ	/RELOC/	
2171						.EVEN		
2172	005020	013705	000340			MOV	@#MAXMEM, R5	; PREPARE TO LOAD THE PROGRAM IN THE HIGHEST
2173								; AVAILABLE MEMORY
2174	005024	014445			2S:	MOV	-(R4), -(R5)	; RELOCATE THE PROGRAM
2175	005026	020427	000430			CMF	R4, #BEGIN-50	; NEITHER RELOCATE NOR TEST LOCATIONS LOWER THAN BEGIN-50
2176	005032	101374				BHI	2S	
2177	005034	000165	000050			JMP	50(R5)	
2178								
2179								; *RELOCATE THE DIAGNOSTIC BACK TO LOWER MEMORY
2180								
2181								
2182	005040	013705	000346		RELOER:	MOV	@#SAVR5, R5	; RESTORE R5
2183	005044	105737	000405			TSTB	@#REL	; IS DIAGNOSTIC IN RELOCATED STATE?
2184	005050	100016				BPL	CKDONE	; BRANCH IF NO
2185								
2186	005052	012704	000430			MOV	#BEGIN-50, R4	; PREPARE TO RELOCATE THE PROGRAM TO LOWER CORE
2187	005056	012524			2S:	MOV	(R5)+, (R4)+	
2188	005060	020537	000340			CMF	R5, @#MAXMEM	
2189	005064	103774				BLO	2S	
2190	005066	105037	000405			CLRB	@#REL	
2191	005072	010537	000346			MOV	R5, @#SAVR5	; SAVE R5
2192	005076	012706	000500			MOV	#BEGIN, SP	; RESET STACK TO LOWER MEMORY
2193	005102	010637	000350			MOV	SP, @#SAVR6	; "BEGIN" USES THIS TO RESET THE STACK.
2194	005106	000137	005112		CKDONE:	JMP	@#LOWER	; TRANSFER THE PROGRAM CONTROL TO THE LOWER CORE
2195								
2196								
2197								
2198	005112	105737	000315		LOWER:	TSTB	@#SAVKBB	; HERE DUE TO IC TYPED?
2199	005116	001073				BNE	\$TPSTK	; BRANCH IF YES (TYPE ERROR STACK)
2200	005120	004767	001714		TSTM:	JSR	PC, MEMMNG	; SET THE REGISTERS IF THE MEMORY MANAGEMENT
2201								; IS AVAILABLE
2202	005124	105737	000276			TSTB	@#MMAVA	; IS MEM. MANAG. AVAILABLE ?
2203	005130	001462				BEQ	ENDPAS	; BRANCH IF NO
2204	005132	000402				BR	\$CNTMM	; BEGIN TESTING ABOVE 28K
2205	005134	004767	002052		CONTMM:	JSR	PC, UPMM	; GO TO UPDATE MEM. MANAG. REGISTERS
2206	005140	012703	000324		SCNTMM:	MOV	#LOWTWO, R3	; MAKE R3 POINT TO THE LOCATION LOWTWO
2207	005144	004767	002160			JSR	PC, GETSIZ	; LOAD BITS 6-10 OF R2 WITH THE BITS 13-17
2208								; OF THE LOWEST ADDRESS UNDER TEST
2209	005150	012704	020000			MOV	#20000, R4	; MAKE R4 POINT TO THE LOWEST LOCATION IN THE BANK
2210								; POINTED BY PAGE ADDRESS REGISTER 1 (PAR1)
2211	005154	020237	172342			CMP	R2, @#172342	; IS THE CONTENT OF R2 LOWER THAN THE CONTENT OF


```

2255                                     ; * TYPE ROUTINE FOR ERROR STACK
2256                                     ; *
2257                                     ; *
2258                                     ; *
2259                                     ; *
2260                                     ; *
2261                                     ; *
2262                                     ; *
2263 005276 032777 000020 173144 ENDPAS: BIT #20,JSWR ; ARE WE GOING TO TYPE THE ERROR STACK AND END OF PASS?
2264 005304 001055 SEOP ; IF NOT THEN GO TO SEOP
2265 005306 012746 177777 STPSTK: MOV #-1,-(SP) ; THE PROGRAM HAS REACHED THE END AND ERROR
2266 ; STACK AND END OF PASS WILL BE TYPED OUT
2267 005312 012701 007744 MOV #ENDPRG,R1 ; PLACE THE STARTING ADDRESS OF THE ERROR STACK
2268 ; FOR 0 TO 4K MEMORY IN R1
2269 005316 012703 000376 TYPSTK: MOV #376,R3
2270 005322 005216 INC (SP) ; IF WE HAVE GONE THRU THE ENTIRE
2271 005324 020137 000310 CMP R1,#ENDSTK ; HAS THE END OF THE ERROR STACK BEEN REACHED ?
2272 005330 103043 BHIS SEOP ; THEN GO TO TYPE END OF PASS
2273 005332 112702 000022 MOVB #18.,R2
2274 005336 105302 RETSTK: DECB R2 ; IF ALL 16 BITS OF THIS BANK HAVE BEEN CHECKED.
2275 005340 002766 BLT TYPSTK ; BEEN CHECKED FOR ERROR THEN SEE IF THERE
2276 ; IS ANY MORE 4K MEMORY BANK
2277 005342 105721 TSTB (R1)+ ; OTHERWISE CHECK THE BYTE STORED AT (R1)
2278 005344 001774 BEQ RETSTK ; IF IT IS 0 WE WILL NOT TYPE IT
2279 005346 020227 000020 CMP R2,#16. ; IS THE POINTER POINTING TO ERROR STACK BYTE
2280 ; MEANT FOR COLLECTING ADDRESS FAILURES FOR
2281 ; THE SPECIFIC MEMORY BANK
2282 005352 103404 BLO 25 ; IF NOT THEN GO TO TYPE BIT NUMBER
2283 005354 101026 BHI PARFL ; IF IT IS POINTING TO THE STACK LOCATION INTENDED
2284 ; TO COLLECT PARITY FAILURES THEN GO TO PARFL
2285 005356 004767 001012 JSR PC,TPADER ; OTHERWISE TYPE "ADDRESS ERROR"
2286 005362 000404 BR FAILNM
2287 005364 010237 000312 25: MOV R2,#DECDWRD ; PREPARE TO TYPE THE NUMBER OF THE FAILING BIT
2288 ; IN DECIMAL
2289 005370 004767 001214 JSR PC,TYPDEC ; GO TO TYPE THE BIT NUMBER IN DECIMAL
2290 005374 011637 000312 FAILNM: MOV (SP),#DECDWRD ; PREPARE TO TYPE THE PAGE NUMBER
2291 005400 004767 001210 JSR PC,STPDEC ; IN DECIMAL
2292 005404 005043 CLR -(R3)
2293 005406 114113 MOVB -(R1),(R3) ; PREPARE TO PRINTOUT THE NUMBER OF TIMES THIS
2294 ; FAILURE OCCURED
2295 005410 105021 CLRB (R1)+ ; CLEAR THE ERROR STACK
2296 005412 005043 CLR -(R3)
2297 005414 105237 000314 INCB #TYPCNT ; ENABLE THE TYPE OUT OF 1 WORDS
2298 005420 004767 001330 JSR PC,RPTOCT ; TYPE THE 4K BANK AND THE NUMBER OF TIMES
2299 ; THIS FAILURE WAS SEEN
2300 005424 012703 000376 MOV #376,R3 ; RESET SCRATCH STACK FOR EACH BIT PRINTED.
2301 005430 000742 BR RETSTK
2302 005432 004767 000762 PARFL: JSR PC,TPPRER ; TYPE "PAR ERR"
2303 005436 000756 BR FAILNM
  
```



```

2352                                     ;* ERROR HANDLING ROUTINE
2353                                     ;* -----
2354                                     ;*
2355                                     ;* PROGRAM COMES HERE EACH TIME AN ERROR IS ENCOUNTERED THIS
2356                                     ;* ROUTINE TYPES OUT THE ERROR MESSAGE IN THE FORMAT GIVEN EARLIER
2357                                     ;*
2358
2359 005600 017637 000000 000402 ERROR: MOV  2(SP),2#$FATAL ;LOAD THE LOCATION $FATAL WITH THE ERROR NUMBER
2360 005606 010346 1$:      MOV  R3,-(SP) ;SAVE R3
2361 005610 010046          MOV  R0,-(SP) ;AND R0 ON THE STACK
2362
2363                                     ;SETUP BANK NO. IN FATAL FOR APT
2364
2365 005612 010103          MOV  R1,R3 ;GET VIRTUAL ADDRESS UNDER TEST FOR GETBNK
2366 005614 004767 001604     JSR  PC,GETBNK ;GET BANK NO. UNDER TEST INTO PBNK
2367 005620 013703 000312     MOV  2#$PBNK,R3 ;GET BANK UNDER TEST
2368 005624 110337 000403     MOV  R3,2#$FATAL+1 ;STORE FAILING BANK NO. FOR APT
2369
2370 ;
2371
2372 005630 010346          MOV  R3,-(SP) ;TEMPORARILY STORE R3
2373 005632 012703 000376     MOV  #376,R3 ;MAKE R3 AS THE STACK POINTER
2374 005636 013743 000306     MOV  2#$PASFLG,-(R3) ;OUTPUT THE WORD STORED AT
2375 005642 005043 2$:      CLR  -(R3)
2376 005644 113713 000402     MOV  2#$FATAL,(R3) ;PUT ERROR NO. ON ERROR STACK
2377 005650 016643 000006     MOV  6(SP),-(R3) ;PLACE THE RETURN PC AT (R3)
2378 005654 011143          MOV  (R1),-(R3) ;PLACE BAD DATA
2379 005656 010643          MOV  R0,-(R3) ;AND GOOD DATA ON THE STACK
2380 005660 005043          CLR  -(R3)
2381 005662 016313 000004     MOV  4(R3),(R3) ;TAKE THE
2382 005666 040013          BIC  R0,(R3) ;EXCLUSIVE OR OF GOOD AND BAD DATA
2383 005670 046300 000004     BIC  4(R3),R0 ;TO FIND THE BITS THAT FAILED
2384 005674 050013          BIS  R0,(R3) ;AND PLACE IT ON THE STACK
2385 005676 012700 002016     MOV  #ENDPRG--24.,R0 ;THIS CODE BRINGS THE RELATIVE ADDRESS
2386 005702 060700          ADD  PC,R0 ;OF THE STARTING OF THE ERROR STACK
2387 005704 062700 000022     6$:  ADD  #18.,R0 ;FOR THE SPECIFIC 4K BANK
2388 005710 005316          DEC  (SP)
2389 005712 002374          BGE  6$
2390 005714 005726          TST  (SP)+ ;RESTORE THE STACK POINTER
2391
2392 005716 105037 000277     ERRTYP: CLR  2#$TYPENB ;DISABLE ANY TYPE OUT
2393 005722 105737 000300     1$:  TST  2#$SPRERR ;IF THIS IS PARITY PROBLEM
2394 005726 001007          BNE  3$ ;THEN GO TO 3$
2395 005730 105720          TST  (R0)+ ;OTHERWISE INCREMENT THE ERROR STACK POINTER BY 1
2396 005732 105737 000301     TST  2#$ADERR ;IF THIS IS ADDRESSING PROBLEM
2397 005736 001003          BNE  3$ ;THEN GO TO 3$
2398 005740 105720          TST  (R0)+ ;INCREMENT THE POINTER R0 BY 1
2399 005742 005713 2$:      TST  (R3) ;IS BIT 15 OF (R3) SET?
2400 005744 100015          BPL  4$ ;IF NOT THEN GO TO 4$
2401 005746 122710 000377     3$:  CMP  #377,(R0) ;OTHERWISE SEE IF THIS ERROR HAS OCCURED 377 TIMES
2402 005752 001401          BEQ  5$ ;IF SO DON'T BUMP ERROR COUNT
2403 005754 105210          INCB (R0) ;INCREMENT THE ERROR COUNTER BY 1
2404 005756 122710 000001     5$:  CMP  #1,(R0) ;MORE THAN 1 ERROR OCCURRED ON THIS BIT?
2405 005762 001404          BEQ  7$ ;BRANCH IF NO
2406 005764 032777 000400 172456 BIT  #400,2$SWR ;STOP ERROR PRINTOUT AFTER 1 WANTED?
2407 005772 001002          BNE  4$ ;BRANCH IF YES (DON'T TYPE ERROR)

```

```

2408 005774 105237 000277 7S: INCB @#TYPENB ;ENABLE THE TYPE OUT ROUTINE
2409 006000 105737 000300 4S: TSTB @#SPRERR ;PARITY ERROR?
2410 006004 001411 BEQ 6S ;BRANCH IF NO
2411 006006 004767 000406 JSR PC,TPPRER ;ELSE TYPE "PAR ERR"
2412 006012 000411 BR 8S ;AND DON'T TEST INDIVIDUAL BIT FAILURES.
2413 006014 105737 000301 TSTB @#SADERR ;ADDRESS ERROR?
2414 006020 001403 BEQ 6S ;BRANCH IF NO
2415 006022 004767 000346 JSR PC,TPADERR ;PRINT "ADR ERR"
2416 006026 000403 BR 8S
2417 006030 105720 6S: TSTB (R0)+ ;POINT TO NEXT ENTRY IN ERROR STACK
2418 006032 006313 ASL (R3) ;IS THERE STILL AN ERROR BIT SET IN ERROR.
2419 006034 001342 BNE 2S ;BR IF YES - KEEP FILLING ERROR STACK
2420 006036 112737 000006 000314 8S: MOVB #6,@#TYPCNT ;TELL TYPCNT TO TYPE 6 WORDS OF ERROR STACK.
2421 ;THE STACK POINTED BY R3
2422 006044 004767 001150 JSR PC,PUTADR ;GO TO THE SUBROUTINE TO PLACE THE ADDRESS IN R1
2423 ;AT LOCATIONS (R3) AND (R3-2)
2424 006050 004767 000622 JSR PC,TYPERR ;TYPE ERROR STACK (7 WORDS)
2425
2426 006054 005037 000300 10S: CLR @#SPRERR ;CLEAR ADDRESS/PARITY ERROR FLAGS
2427 006060 012600 MOV (SP)+,R0 ;RESTORE R0
2428 006062 012603 MOV (SP)+,R3 ;AND R3
2429 006064 105737 000420 FNDERR: TSTB @#SENV ;ARE WE RUNNING UNDER APT?
2430 006070 001404 BEQ 2S ;IF NOT THEN TEST FOR HALT
2431 006072 012737 000001 000400 MOV #1,@#MSGTY ;OTHERWISE INFORM THE APT
2432 006100 000443 BR FATHLT ;GOTO FATHLT AND WAIT FOR APT.
2433
2434 006102 010246 2S: MOV R2,-(SP) ;SAVE R2 TEMP
2435 006104 005777 172340 TST @#WR ;DOES THE OPERATOR REQUIRE THE PROGRAM TO HALT
2436 ;ON ERROR
2437 006110 100405 BMI 4S ;IF SO THEN HALT ON ERROR
2438 ;CHECK FOR CONTROL-C KEY
2439
2440 006112 004767 001546 JSR PC,CHECKC ;IF CONTROL-C TYPED THEN PRINT ERROR HISTORY
2441 ;AND HALT AT FATHLT.
2442 006116 105737 000042 7S: TSTB @#42 ;ARE WE RUNNING UNDER ACT?
2443 006122 001401 BEQ 6S ;BRANCH IF NO
2444
2445 006124 000000 4S: HALT ;PROGRAM HAS HALTED ON ERROR, R1 IS POINTING
2446 ;TO A LOCATION WHICH SHOULD HAVE CONTAINED
2447 ;THE WORD STORED IN R0
2448 006126 012602 6S: MOV (SP)+,R2 ;RESTORE R2
2449 006130 062716 000002 ADD #2,(SP) ;RESTORE THE RETURN ADDRESS
2450 006134 000207 RTS PC ;RETURN FROM THE SUBROUTINE
2451
2452
2453
2454 006136 FATERR:
2455 006136 004767 000416 SEQERR: JSR PC,TPCRLF ;TYPE "ERROR #"
2456 006142 051105 047522 020122 .ASCIZ /ERROR #/
2457 006150 000043 .EVEN
2458
2459
2460 006152 017637 000000 000402 MOV @#(SP),@#SFATAL ;LOAD THE LOCATION SFATAL WITH THE ERROR NUMBER
2461 006160 105237 000314 INCB @#TYPCNT ;TELL STPNUM TO TYPE 1 WORD
2462 006164 012703 000376 MOV #376,R3 ;STPNUM USES R3 AS STACK
2463 006170 013743 000402 MOV @#SFATAL,-(R3) ;PUT ERROR NO. ON STACK

```

2464	006174	005743		TST	-(R3)		:STPNUM REQUIRES THIS
2465	006176	004767	000562	JSR	PC,FATYP		:TYPE ERROR NO.
2466	006202	105737	000420	APTHLT: TSTB	J#SENV	:RUNNING	:UNDER APT?
2467	006206	001326			FNDERR		:BRANCH IF YES
2468	006210	000000		FATHLT: HALT			:FATAL ERROR OR IC HALT.
2469	006212	000137	000250	JMP	J#RESTR		:RESTART TST BUT DCN'T CLEAR PASS COUNT
2470							:IN CASE IC RESTART.

```

:PARERR
:   PARITY TRAP HANDLER
:   COME HERE FROM A TRAP TO 114.
:   THIS ROUTINE SEARCHES THE AVAILABLE PARITY MODULES AND IF ONE
:   HAS A PARITY ERROR BIT SET THE GET THE PARITY ERROR ADDRESS
:   AND CALL THE "ERROR" ROUTINE TO PRINT ERROR MESSAGE.
:   IF NO PARITY ERROR BITS CAN BE FOUND A FATAL ERROR IS DONE.
:
:REGISTER US AGE.
:RO= HOLDS PARITY MODULE ADDRESSES
:RI= GETS ERROR ADDRESS FOR "ERROR" CALL.
    
```

2485	006216	012637	000356	PARERR: MOV	(SP)+,J#PARSP		:SET PARSP TO RETURN ADDRESS
2486	006222	011637	000360	MOV	(SP),J#PARPS		:SAVE PSW FOR RETURN
2487	006226	013706	000350	MOV	J#SAVR6,SP		:AND RESET THE SP SINCE NOT ENOUGH STACK ROOM
2488							:TO COMPLETE THE ERROR SERVICE ROUTINE.
2489	006232	010067	000132	MOV	RO,SAVR0		:SAVE RO DURING PARITY SERVICE
2490	006236	010167	000130	MOV	RI,SAVR1		:SAVE RI DURING PARITY SERVICE
2491	006242	013701	000352	MOV	J#PARMAP,R1		:GET PARITY AVAILABLE MAP
2492	006246	012700	172100	MOV	#172100,RO		:RO= FIRST PARITY ADDRESS.
2493							
2494	006252	005701		TST	R1		:ANY PARITY MODULES AVAILABLE?
2495	006254	001442		BEQ	4\$:BR IF NO -FATAL ERROR-
2496	006256	000241		CLC			
2497	006260	006001		15: ROR	R1		:SHIFT PARITY MAP BIT INTO C BIT.
2498	006262	103005		BCC	2\$:BRANCH IF THIS PARITY MODULE NOT AVAILABLE.
2499	006264	005710		TST	(RO)		:PARITY MODULE ERROR BIT SET?
2500	006266	100406		BMI	3\$:BRANCH IF YES -CALL "ERROR" ROUTINE
2501	006270	020027	172136	CMP	RO,#172136		:DONE ALL PARITY MODULES?
2502	006274	002032		BGE	4\$:BR IF YES- GO TO FATAL ERROR CALL-
2503	006276	062700	000002	25: ADD	#2,RO	:POINT TO NEXT PARITY ADDRESS	
2504	006302	000766		BR	1\$:AND KEEP TRYING
2505	006304	042710	100000	35: BIC	#100000,(RO)		:CLEAR PARITY ERROR BIT.
2506	006310	011001		MOV	(RO),R1		:GET PARITY MODULE CSR
2507	006312	006101		ROL	R1		:SHIFT ERROR ADDRESS BITS 11-5 INTO 15-9
2508	006314	006101		ROL	R1		
2509	006316	006101		ROL	R1		
2510	006320	006101		ROL	R1		
2511	006322	042701	000777	BIC	#777,R1	:SAVE ERROR ADDRESS ONLY	
2512	006326	105237	000300	INCB	J#SPERR		:TELL "ERROR" PARITY ERROR CALL.
2513	006332	004767	177242	JSR	PC,ERROR		:#ERROR# REPORT ERROR MESSAGE
2514	006336	000050		50			:*****ERROR NUMBER 50*****
2515							
2516	006340	016700	000024	MOV	SAVR0,RO		:RESTORE RO
2517	006344	016701	000022	MOV	SAVR1,R1		:RESTORE R1
2518	006350	013746	000360	MOV	J#PARPS,-(SP)		:SET RETURN PSW ON STACK
2519	006354	013746	000356	MOV	J#PARSP,-(SP)		:AND SET RETURN ADDRESS ON STACK

```

2520 006360 000002 RTI ;RETURN TO TEST WHERE PARITY TRAP OCCURRED.
2521
2522
2523 006362 ;COME HERE IF NO PARITY ERROR FLAG FOUND SET
2524 006362 004767 177550 4S:
2525 006366 000051 JSR PC,FATERR ;*ERROR* REPORT ERROR MESSAGE AND HALT AT FATHLT
2526 ;*****ERROR NUMBER 51*****
2527
2528 ;RO+R1 ARE SAVED HERE FOR PARITY TRAP DUE TO INSUFFICIENT
2529 ;STACK SPACE BETWEEN 500-450.
2530 006370 000000 SAVRO: 0 ;SAVE RO DURING PARITY TRAP SERVICE
2531 006372 000000 SAVR1: 0 ;SAVE R1 DURING PARITY TRAP SERVICE
2532
2533 006374 105737 000277 TPADER: TSTB @#TYPENB ;TYPE ERROR?
2534 006400 001406 BEQ 1$ ;BRANCH IF NO
2535 006402 004767 000160 JSR PC,PNTMES ; TYPE CR, LF AND "ADR ER"
2536 006406 042101 020122 051105 .ASCIZ /ADR ERR/
2537 006414 000122
2538
2539 .EVEN
2540 006416 000207 1$: RTS PC
2541
2542 006420 105737 000277 TPPER: TSTB @#TYPENB ;ERROR PRINTOUTS ALLOWED?
2543 006424 001406 BEQ 1$ ;BRANCH IF NO
2544 006426 004767 000134 JSR PC,PNTMES ;GO TO TYPE CR, LF AND "PAR ERR"
2545 006432 040520 020122 051105 .ASCIZ /PAR ERR/
2546 006440 000122
2547
2548 .EVEN
006442 000207 1$: RTS PC

```

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

;* TYPE OUT ROUTINE

 ;*
 ;* THIS ROUTINE IS USED BY THE PROGRAM TO TYPE OUT ANY CHARACTER
 ;*

```

NOTYP: MOV R1, -(SP)
006444 010146 000002 4$: MOV 2(SP), R1
006446 016601 000002 4$: TSTB (R1)+ ; IF THIS TYPE OUT HAS BEEN SUPRESSED THEN
006452 105721 000002 4$: BNE 4$ ; PREPARE TO RETURN
006454 001376 000002 4$: BR RETTYP
006456 000412 000002 4$: MOV R1, -(SP) ; SAVE R1
006460 010146 000002 4$: MOV RO, -(SP) ; AND RO ON THE STACK
006462 010046 000002 4$: MOV 4(SP), R1 ; PLACE THE ADDRESS OF MESSAGE TO BE TYPED IN R1
006464 016601 000004 2$: MOVVB (R1)+, RO ; PLACE THE BYTE TO BE TYPED IN RO
006470 112100 000002 4$: BEQ 4$ ; IF IT IS END OF MESSAGE THEN GO TO 4$
006472 001403 000022 4$: JSR PC, STPCHR ; OTHERWISE GO TO TYPE THE CONTENTS OF RO
006474 004767 000022 4$: BR 2$
006500 000773 000002 4$: MOV (SP)+, RO ; RESTORE RO
006502 012600 000001 RETTYP: INC R1 ; CAUSE R1 TO
006504 005201 000002 RETTYP: BIC #1, R1 ; POINT TO EVEN ADDRESS
006506 042701 000001 RETTYP: MOV R1, 2(SP) ; MODIFY THE RETURN ADDRESS
006512 010166 000002 RETTYP: MOV (SP)+, R1 ; RESTORE R1
006516 012601 000002 RETTYP: BR EXTYP ; AND RETURN VIA RTS PC
006522 132737 000040 000421 STPCHR: BITB #40, 2$ENVM ; HAVE TYPE OUTS BEEN DISABLED?
006530 001005 000040 000421 STPCHR: BNE 4$ ; IF SO THEN RETURN FROM THE SUBROUTINE
006532 105737 177564 2$: TSTB 2$STPS ; WAIT HERE
006536 100375 177566 2$: BPL 2$ ; UNTIL THE PRINTER IS READY
006540 110037 177566 4$: MOVVB RO, 2$STPB ; LOAD DATA TO BE TYPED INTO DATA REG.
006544 000404 177566 4$: BR EXTYP ; RETURN
006546 004767 177706 PCRLF: JSR PC, STYPE
006552 005015 000 000 PCRLF: .ASCIZ <15><12> ; CR/LF
006556 000207 177706 PCRLF: RTS PC ; RETURN
006560 004767 177762 TPCRLF: JSR PC, PCRLF
006564 000735 177762 TPCRLF: BR STYPE ; NOW GO TO TYPE THE REST OF THE MESSAGE
006566 032777 000020 171654 PNTMES: BIT #20, 2$SWR ; PRINTOUTS ALLOWED?
006574 001323 000042 000046 PNTMES: BNE NOTYP ; BRANCH IF NO
006576 123737 000042 000046 PNTMES: CMPB 2$42, 2$46 ; RUNNING UNDER ACT 11?
006604 001717 000042 000046 PNTMES: BEQ NOTYP ; BRANCH IF YES -NOT PRINTOUT-
006606 000764 000042 000046 PNTMES: BR TPCRLF ; SEND CR/LF AND TYPE MESSAGE.
    
```



```

2678
2679
2680
2681
2682
2683
2684
2685
2686
2687 007040 012702 001400 MEMMNG: MOV #1400,R2
2688 007044 105037 000276 MMREG: CLRB @#MMAVA ;CLEAR THE BYTE THAT IS SUPPOSED TO INDICATE
2689 ; THAT MEM. MANAG. IS AVAILABLE FOR TESTING
2690 007050 032777 010000 171372 BIT #10000,@SWR ;HAS THE OPERATOR ASKED TO CHECK MEMORY MANAG. ?
2691 007056 001441 BEQ RETMM ;IF NOT THEN RETURN FROM THE SUBROUTINE
2692 007060 012700 000004 MOV #4,R0 ;PREPARE TO SETUP TIME OUT VECTOR
2693 007064 012720 007164 MOV #NOMM,(R0)+ ;RETURN ADDRESS TO NOMM
2694 007070 012710 000340 MOV #340,(R0) ;AND WITH A PSM OF 340
2695 007074 005037 177572 CLR @#SRO ;TRY TO REACH MEM. MANAG. SRO
2696 007100 105237 000276 INCB @#MMAVA ;IF IT IS AVAILABLE THEN SET MEM. MANAG. AVAILABLE
2697 ;BYTE
2698 007104 012701 172340 MOV #172340,R1 ;R1 IS POINTING TO PAR0
2699 007110 005021 CLR (R1)+ ;PAR0 WILL POINT TO BANK 0
2700 007112 062702 000200 25: ADD #200,R2
2701 007116 010221 MOV R2,(R1)+ ;SETUP PAR1-PAR6
2702 007120 020127 172356 CMP R1,#172356
2703 007124 103772 BLO 25
2704 007126 012711 007600 MOV #7600,(R1) ;PAR7 IS POINTING TO THE I/O PAGE
2705 007132 012701 172300 MOV #172300,R1
2706 007136 012721 077406 45: MOV #77406,(R1)+ ;SETUP PDR0-PDR7
2707 007142 020127 172316 CMP R1,#172316
2708 007146 101773 BLOS 45
2709 007150 005237 177572 INC @#SRO ;ENABLE MEM. MANAG.
2710 007154 005010 SRETMM: CLR (R0) ;RESTORE TIME OUT TRAP VECTOR FOR ANY FUTURE TRAP
2711 007156 012740 000104 MOV #BUSER,-(R0)
2712 007162 000207 RETMM: RTS PC
2713
2714 007164 022626 NOMM: CMP (SP)+,(SP)+ ;RESTORE STACK POINTER
2715 007166 004767 177366 JSR PC,TPCRLF ;TYPE "NO MEMORY MANAGEMENT MESSAGE"
2716 007172 047516 046440 043516 .ASCIZ /NO MNG/
2717 007200 000 .EVEN
2718 007202 004767 176730 JSR PC,FATERR ;*ERROR* REPORT ERROR MESSAGE AND HALT AT FATHLT
2719 007206 000052 52 ;*****ERROR NUMBER 52*****
2720
2721
2722 007210 000761 BR SRETMM ; RESTORE TIME OUT TRAP VECTOR
2723
2724 007212 013702 172354 UPMM: MOV @#172354,R2 ;PREPARE TO UPDATE MEMORY MANAG. REGISTERS
2725 007216 000712 BR MMREG

```

```

2726
2727
2728
2729
2730
2731
2732
2733
2734
2735
2736
2737 007220 005063 177776 PUTADR: CLR -2(R3)
2738 007224 010113 MOV R1,(R3) ;PLACE THE ADDRESS STORED IN R1 IN LOCATION (R3)
2739 007226 105737 000276 TSTB @#MMAVA ;IS THE MEM. MANAG. AVAILABLE ?
2740 007232 001425 BEQ 6$ ;IF NOT THEN RETURN FROM THE SUBROUTINE
2741 007234 010146 MOV R1,-(SP) ;SAVE R1
2742 007236 042701 017777 BIC #17777,R1 ;CLEAR BITS 0-12 OF THE ADDRESS IN R1
2743 007242 040113 BIC R1,(R3) ;LEAVE BITS 0-12 OF THE ADDRESS IN (R3)
2744 007244 052701 004000 BIS #4000,R1 ;PREPARE TO SHIFT R1 BY 12 PLACES
2745 007250 006001 2$: ROR R1
2746 007252 103376 BCC 2$ ;GET THE NUMBER OF PAR IN R1
2747 007254 062701 172340 ADD #172340,R1 ;GET THE ADDRESS OF PAR IN R1
2748 007260 011101 MOV (R1),R1 ;LOAD R1 WITH THE CONTENTS OF PAR
2749 007262 052701 010000 BIS #10000,R1
2750 007266 006101 4$: ROL R1
2751 007270 103376 BCC 4$ ;PLACE THE ADDRESS BITS 13-17 IN BITS 11-15 OF R1
2752 007272 006101 ROL R1
2753 007274 006143 ROL -(R3) ;PLACE BIT 17 IN LOCATION POINTED BY R3-2
2754 007276 006101 ROL R1
2755 007300 006123 ROL (R3)+ ;PLACE BIT 16 OF THE ADDRESS
2756 007302 050113 BIS R1,(R3) ;PLACE BITS 13-15 OF THE ADDRESS IN LOCATION (R3)
2757 007304 012601 MOV (SP)+,R1 ;RESTORE R1
2758 007306 000207 6$: RTS PC ;RETURN FROM THE SUBROUTINE
2759
2760
2761
2762
2763
2764
2765
2766
2767
2768
2769
2770
2771
2772 007310 016143 000001 GETADR: MOV 1(R1),-(R3) ;PLACE THE LOW ORDER BITS OF THE ADDRESS
2773 007314 005043 CLR -(R3) ;CLEAR THE LOCATION WHERE THE HIGH ORDER BITS
2774 ;HAVE TO BE PLACED
2775 007316 116113 177777 MOVB -1(R1),(R3) ;PLACE BITS 16 & 17
2776 007322 000207 2$: RTS PC ;RETURN FROM THE SUBROUTINE
    
```

* 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 AVAILBLE IN WHICH
 * CASE THE HIGH ORDER BITS OF THE ADDRESS ARE PLACED IN LOCATION
 * POINTED BY R3-2
 *

* 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
 *

```

2777
2778
2779
2780
2781
2782
2783
2784
2785
2786
2787 007324 105237 000315          $GTSIZ: INCB      @#SAVKBB          ;PREPARE TO PLACE ADDRESS BITS 13-17 IN BITS
2788                                     ;0-4 OF R2
2789
2790 007330 012301          GETSIZ: MOV      (R3)+,R1
2791 007332 011302          MOV      (R3),R2          ;LOAD R2 WITH THE LOW ORDER BITS OF THE ADDRESS
2792 007334 042702 017777          BIC      #17777,R2        ;CLEAR ADDRESS BITS 0-12
2793 007340 052702 000040          2$:     BIS      #40,R2
2794 007344 006001          4$:     ROR      R1
2795 007346 006002          ROR      R2          ;ROTATE R1 AND R2 7 TIMES
2796 007350 103375          BCC      4$
2797 007352 105737 000315          TSTB     @#SAVKBB
2798 007356 001405          BEQ      6$
2799 007360 105037 000315          CLRB     @#SAVKBB
2800 007364 052702 000100          BIS      #100,R2
2801 007370 000765          BR       4$
2802 007372 012301          6$:     MOV      (R3)+,R1          ;PLACE THE LOW ORDER ADDRESS BITS IN R1
2803 007374 012700 160000          MOV      #160000,R0
2804 007400 040001          BIC      R0,R1          ;LEAVE BITS 0-12 OF THE ADDRESS IN R1
2805 007402 000207          RTS      PC          ;RETURN FROM THE SUBRORNE
2806
2807
2808
2809
2810
2811
2812
2813
2814
2815 007404 105737 000276          ;* SUBROUTINE TO DISABLE MEMORY MANAGEMENT
2816 007410 001404          ;* -----
2817 007412 005037 177572          ;*
2818 007416 105037 000276          ;* THIS SUBROUTINE IS CALLED TO DISABLE THE MEMORY MANAGEMENT
2819 007422 000207          ;* UNIT
2820
2821
2822
2823
2824
2825
2826
2827
2828
2829 007424 010046          CLRMM: TSTB     @#MMAVA          ;WAS THE MEMORY MANAGEMENT ENABLED ?
2830 007426 010346          BEQ      1$          ;IF NOT THEN GO TO 1$
2831 007430 042703 017777          CLR      @#SRO          ;DISABLE THE MEMORY MANAGEMENT
2832 007434 052703 010000          CLRB     @#MMAVA          ;AND DO NOT ATTEMPT TO TEST MEM. MANAG.
2833                                     1$:     RTS      PC          ;RETURN FROM THE SUBROUTINE
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
    
```

DZKMA MACY11 27(732) 04-NOV-76 07:49 PAGE 63
 DZKMAB.P11 SUBROUTINE TO DISABLE MEMORY MANAGEMENT

```

2833 007440 000241          CLC
2834 007442 006003          1$: ROR      R3          ;SHIFT A BANK BIT
2835 007444 103376          BCC      1$          ;UNTIL IN BITS <2:0> OF R3
2836 007446 105737 000276  TSTB    @#MMAVA     ;MEMORY MANAGEMENT UNDER TEST?
2837 007452 001407          BEQ      2$          ;NO EXIT
2838
2839          ;GET PAR ADDRESS AND PHYSICAL BANK NO.
2840 007454 006303          ASL      R3          ;MAKE R3 PAR ADDRESS OFFSET.
2841 007456 062703 172340  ADD     #172340,R3  ;MAKE FULL PAR ADDRESS.
2842 007462 011300          MOV     (R3),R0     ;GET PAR CONTENTS
2843 007464 006300          ASL      R0
2844 007466 000300          SWAB    R0          ;SHIFT BANK BITS TO BITS <7:0>
2845 007470 110003          MOVB    R0,R3      ;SET R3 TO PHYSICAL BANK NO.
2846 007472 010337 000312  2$: MOV     R3,@#PBK  ;STORE PHYSICAL BANK NO.
2847 007476 012603          MOV     (SP)+,R3   ;RESTORE R3
2848 007500 012600          MOV     (SP)+,R0   ;RESTORE R0
2849 007502 000207          RTS      PC        ;RETURN TO CALLER
2850
2851
2852
2853          ; PARITY ENABLE/DISABLE ROUTINE
2854
2855          ;
2856          ; THIS ROUTINE ENABLES OR DISABLES PARITY MODULES AND PRINTS ASSOCIATED MEESSAGES.
2857          ; IF PARITY AVAILABLE THEN BIT13 OF "REL" IS SET AND "PAR"ITY IS PRINTED.
2858          ; ALSO THE BACKGROUND TEST PATTERN (LOC. BAKPAT) IS SET=376
2859
2860          ; REGISTER USAGE.
2861          ; R0= POINTS TO BUS TIMEOUT TRAP VECTOR (LOC. 4)
2862          ; R1= HOLDS PARITY MODULE UNIBUS ADDRESS.
2863          ; R2= ON ENTRY HOLDS ENABLE/DISABLE CODE .
2864          ; IF R2=0 THEN DISABLE
2865          ; IF R2=1 THEN ENABLE
2866          ; R3= SCRATCH TO SETUP LOC. PARMAP WITH A MAP OF PARITY MODULES PRESENT.
2867
2868          ; CALL IS
2869          ;
2870          ;
2871
2872 007504 032777 004000 170736  PARITY: BIT    #4000,@SWR  ;PARITY TEST WANTED?
2873 007512 001460          BEQ      6$          ;BRANCH IF NO
2874
2875 007514 012700 000004          MOV     #4,R0      ;POINT R0 TO BUS TIMEOUT ADDRESS.
2876 007520 012710 000122          MOV     #5$-6,(R0) ;SET RETURN FROM TIMEOUT TRAP TO 5$
2877 007524 060710          ADD     PC,(R0)    ;IN THE CURRENT BANK.
2878 007526 005037 000352          1$: CLR     @#PARMAP  ;CLEAR PARITY MAP HOLDER.
2879 007532 012701 172140          MOV     #172140,R1 ;SET R1 TO LAST PARITY MODULE ADDRESS+2
2880 007536 012703 100000          MOV     #100000,R3 ;SET R3 TO PARMAP AVAILABLE CODE BEGIN.
2881 007542 010241          2$: MOV     R2,-(R1) ;ENABLE A PARITY MODULE+TRAP IF NOT AVAILABLE.
2882 007544 050337 000352          BIS     R3,@#PARMAP ;NO TRAP TO 5$, SO SET PARITY AVAILABLE.
2883 007550 000241          CLC
2884 007552 006003          3$: ROR      R3          ;SETUP NEXT PARMAP BIT
2885 007554 103372          BCC     2$          ;BRANCH IF NOT DONE ALL PARITY ADDRESSES.
2886 007556 012710 000104          MOV     #BUSER,(R0) ;RESET BUS TIMEOUT TRAP VECTOR
2887 007562 005702          TST     R2          ;IS THIS A DISABLE CALL?
2888 007564 001433          BEQ     6$          ;BRANCH IF YES (EXIT)

```

DZKMA MACY11 27(732) 04-NOV-76 07:49 PAGE 64
 DZKMAB.P11 SUBROUTINE TO DISABLE MEMORY MANAGEMENT

```

2889 007566 005737 000352          TST      @#PARMAP      ;WERE ANY PARITY MODULES FOUND?
2890 007572 001011                    BNE      4$           ;BRANCH IF YES
2891 007574 004767 176760          JSR      PC,TPCRLF    ;PRINT "NO PAR"
2892 007600 047516 050040 051101  .ASCIZ   /NO PAR/
2893 007606          000
2894          007610          .EVEN
2895 007610 004767 176322          JSR      PC,FATERR    ;*ERROR* REPORT ERROR MESSAGE AND HALT AT FATHLT
2896 007614 000053          53           ;*****ERROR NUMBER 53*****
2897
2898
2899 007616 152737 000040 000405 4$:  BISB     #40,@#REL    ;SET PARITY UNDER TEST FLAG
2900 007624 012737 000376 000316      MOV      #376,@#BAKPAT ;SET BACKGROUND PATTERN TO
2901                                ;WORST CASE PARITY CODE.
2902 007632 004767 176722          JSR      PC,TPCRLF    ;PRINT "TST PARITY"
2903 007636 040520 044522 054524  .ASCIZ   /PARITY/
2904 007644          000
2905          007646          .EVEN
2906 007646 000405          BR       EXITC        ;AND EXIT VIA RTS PC
2907
2908                                ;GET HERE IF PARITY ADDRESS TIMED OUT TO LOC. 4
2909

```

M05

DZKMA MACY11 27(732) 04-NOV-76 07:49 PAGE 65
 DZKMAB.P11 SUBROUTINE TO DISABLE MEMORY MANAGEMENT

```

2910 007650 022626          5$:    CMP      (SP)+,(SP)+    ;RESET STACK FROM TRAP
2911 007652 000737          BR      3$          ;KEEP TRYING PARITY ADDRESSES.
2912
2913 007654 142737 000040 000405 6$:    BICB     #40,0#REL    ;CLEAR PARITY TESTING FLAG
2914 007662          EXITC:
2915 007662 000207          7$:    RTS      PC          ;RETURN TO CALLER
2916
2917
2918
2919
2920
2921          ;CHECKC
2922          ; THIS ROUTINE CHECKS IF CONTROL-C WAS TYPED AT THE END OF EACH
2923          ; TEST OR IN THE ERROR TYPE ROUTINE.
2924          ; IF CONTROL-C TYPED THE PROGRAM IS RETURNED TO LOWER MEMORY IF IT WAS
2925          ; RELOCATED AND THE ERROR HISTORY IS PRINTED OUT.
2926          ; FINALLY IT HALTS AT FATHLT.
2927 007664 105037 000315    CHECKC: CLRB     0#SAVKBB    ;INIT CONTROL-C FLAG.
2928 007670 105737 177560    TSTB     0#TKS          ;ANY CHAR. TYPED?
2929 007674 100372          BPL      EXITC          ;BR IF NO-EXIT VIA RTS PC-
2930 007676 113702 177562    MOVB     0#SKBB,R2     ;GET THE CHAR TYPED.
2931 007702 042702 000200    BIC      #200,R2      ;CLEAR THE PARITY BIT.
2932 007706 122702 000003    CMPB     #3,R2        ;IS IT CONTROL-C?
2933 007712 001363          BNE      EXITC          ;BRANCH IF NO -EXIT VIA RTS PC-
2934 007714 110237 000315    MOVE     R2,0#SAVKBB  ;ELSE STORE THE CHAR. FOR USE AS A FLAG.
2935 007720 004767 176634    JSR      PC,TPCRLF    ;PRINT "tC"
2936 007724 041536          .ASCIZ  /tC/
2937 007730          .EVEN
2938 007730 000167 175104    JMP      RELOER       ;GO RETURN PROGRAM TO LOWER CORE IF RELOCATED.
2939
2940          .=7744
2941 007744 000000    ENDPRG: 0
2942
2943
2944
2945
2946          000001          .END
  
```

```

; THIS BEGINS THE STORAGE FOR THE ERROR HISTORY
; STACK.FOR EACH 4K BANK 18. BYTES ARE SAVED.
; ALSO THE ABSOLUTE LOADER AND XXDP CODE IS SAVED
; AFTER THE ERROR STACK.
; FOR 4K MEMORY SIZE THEN PROGRAM=7744+22=7776
  
```


CHECKC	007664	1178	2440	2927#																	
CKDONE	005106	2161	2184	2194#																	
CLRMEN	001466	1105#	2226																		
CLRMN	007404	1039	1061	2339	2815#																
CNTSCP	001644	1177	1180#																		
CONT	001526	1122#	1191																		
CONTRM	005134	2158	2205#																		
CTLC	005570	2318	2348#																		
DECMRD	000312	778#	779	2287*	2290*	2326*	2608														
ENDPAS	005276	2203	2263#																		
ENDPRG	007744	704	899	2267	2385	2941#															
ENDSTK	000310	775#	776	1026*	2271	2342															
END0	002132	1276#	1318																		
END1	002232	1318#	1358																		
END10	003772	1825	1833#	1976																	
END12	004316	1976#	2086	2118																	
END2	002342	1358#	1454																		
END3	002610	1454#	1493																		
END4	002720	1493#	1579																		
END5	003066	1579#	1635																		
END6	003222	1635#	1712																		
END7	003444	1712#	1833																		
ERROR	005600	1235	1296	1350	1415	1479	1519	1526	1553	1611	1686	1766	1782	1791							
		1936	1944	2071	2083	2100	2359#	2513													
ERRTYP	005716	2392#																			
EXITC	007662	2906	2914#	2929	2933																
EXTYP	006556	2574	2581	2586#																	
FAILNM	005374	2286	2290#	2303																	
FATERR	006136	912	960	1072	1094	1262	2454#	2524	2719	2895											
FATHLT	006210	2432	2468#																		
FATYP	006764	2465	2661#																		
FNDERR	006064	2429#	2467																		
GALLOP	003472	1749#	1883																		
GETADR	007310	963	964	2772#																	
GETBNK	007424	2143	2366	2829#																	
GETSIZ	007330	2207	2218	2223	2790#																
HIGHAD	000332	804#	949																		
HIGHTM	000330	803#	948	1017																	
LOOP	001536	1125#	1189																		
LOWADD	000326	801#																			
LOWBNK	000304	768#	769	1656*	1667	1702															
LOWER	005112	2194	2198#																		
LOWTMO	000324	800#	1010	1063	2206																
M =	000200	613#	2100																		
MAXADR	005224	2221	2224	2235#																	
MAXMEM	000340	810#	947	1030*	1252	1448	2049	2172	2188												
MEMING	007040	983	2200	2687#																	
MEMTST	001460	1099#																			
MMAVA	000276	750#	753	985	2157	2202	2688*	2696*	2739	2815	2818*	2836									
MMREG	007044	2217	2688#	2725																	
N =	000054	613#	912	915#	960	963#	1072	1075#	1094	1097#	1206	1209#	1220	1222#							
		1223	1226#	1231	1233#	1262	1265#	1288	1291#	1296	1299#	1331	1334#	1350							
		1353#	1375	1378#	1400	1402#	1410	1412#	1426	1428#	1463	1466#	1479	1482#							
		1512	1515#	1519	1522#	1526	1529#	1553	1556#	1601	1604#	1611	1614#	1652							
		1655#	1686	1689#	1743	1746#	1766	1769#	1782	1785#	1791	1794#	1868	1871#							
		1908	1911#	1936	1939#	1944	1947#	2009	2012#	2071	2074#	2083	2086#	2100							

		2492*	2499	2501	2503*	2505*	2506	2516*	2563	2565*	2569*	2580	2652*	2654*
		2656*	2661*	2665*	2666*	2668*	2670*	2672*	2692*	2693*	2694*	2710*	2711*	2803*
		2804	2829	2842*	2843*	2844*	2845	2848*	2875*	2876*	2877*	2886*		
R1	=%000001	657*	659*	660	902*	903	904	906	907*	909	917	919*	920*	921
		953*	954*	955	957	974*	976*	978*	988*	991	997	1010*	1020*	1027*
		1028	1125*	1137*	1210*	1212	1214	1216	1229	1240*	1243	1252	1259	1261*
		1293*	1294	1314*	1316	1335*	1336*	1337	1340	1343*	1344	1346	1349*	1353
		1389*	1390*	1391	1393	1395	1397	1404	1418*	1423	1430*	1431	1433*	1434*
		1435	1467	1471*	1472	1475	1482	1487	1517	1523*	1524	1546*	1547	1550
		1563*	1571	1608*	1609	1614*	1615	1667*	1670	1681*	1683	1689*	1690	1759*
		1764*	1769*	1770	1773	1775	1778	1788	1789*	1794*	1795	1796*	1799*	1800*
		1801	1915*	1918	1920	1928	1933*	1934	1941*	1942	1947*	1953*	1955	1960*
		1962	2042*	2045*	2046*	2047*	2049	2068	2069*	2074*	2089*	2090	2092	2096*
		2098	2103	2214	2239	2267*	2271	2277	2293	2295*	2365	2378	2490	2491*
		2494	2497*	2506*	2507*	2508*	2509*	2510*	2511*	2517*	2557	2558*	2559	2562
		2564*	2565	2570*	2571*	2572	2573*	2645	2647	2698*	2699*	2701*	2702	2704*
		2705*	2706*	2707	2738	2741	2742*	2743	2744*	2745*	2747*	2748*	2749*	2750*
		2752*	2754*	2756	2757*	2772	2775	2790*	2794*	2802*	2804*	2879*	2881*	
R2	=%000002	1024*	1105*	1111*	1112*	1113*	1114*	1115*	1123	1126	1139*	1140*	1141*	1142
		1180*	1190*	1255*	1256*	1257*	1258*	1266*	1267*	1292*	1300	1306*	1381*	1382*
		1383	1385	1387*	1395	1414	1437*	1438*	1440	1442*	1443*	1444	1447	1623*
		1625*	1658*	1659*	1660	1661	1663*	1690	1698	1707	1709	1746*	1747*	1749*
		1750	1752*	1754	1801	1816	1822	1824	1827	1871*	1880*	1912*	1918	1964*
		1970*	2012*	2065	2070	2082	2125*	2211	2216*	2240	2246	2273*	2274*	2279
		2287	2315*	2434	2448*	2687*	2700*	2701	2724*	2791*	2792*	2793*	2795*	2800*
		2881	2887	2930*	2931*	2932	2934							
R3	=%000003	691*	692*	693	705*	706	707	947*	997	1001	1004	1005*	1006*	1008*
		1017*	1031	1033	1035*	1040	1041	1055	1057*	1058*	1063*	1064	1067	1076
		1078	1079	1081	1084	1097*	1098*	1099*	1127*	1131	1138*	1209*	1211	1216
		1229	1234	1378*	1382	1390	1430	1438	1447*	1448	1466*	1468	1484	1490*
		1540	1562*	1567	1607*	1621	1624*	1627	1628*	1656	1658	1665	1670	1672*
		1673	1675*	1694*	1697	1698	1702*	1707*	1757	1759	1807	1811*	1818*	1822*
		1913*	1920	1928	1971*	2030*	2033*	2037	2045	2059	2064*	2065	2069	2088*
		2092	2103	2108*	2146	2206*	2225*	2269*	2292*	2293*	2296*	2300*	2360	2365*
		2367*	2368	2372	2373*	2374*	2375*	2376*	2377*	2378*	2379*	2380*	2381*	2382*
		2383	2384*	2399	2418*	2428*	2462*	2463*	2464	2645*	2647*	2649*	2650*	2653*
		2655*	2662*	2664*	2669*	2671*	2673*	2737*	2738*	2743*	2753*	2755*	2756*	2772*
		2773*	2775*	2790	2791	2802	2830	2831*	2832*	2834*	2840*	2841*	2842	2845*
		2846	2847*	2880*	2882	2884*								
R4	=%000004	657	704*	899*	903*	920	974	1008	1020	1022*	1025	1028	1054*	1058
		1076	1078*	1087*	1092	1109	1125	1135	1137	1138	1210	1353	1383	1391
		1487	1563	1571	1607	1608	1746	1871	1915	1962	2020	2023*	2024	2028*
		2030	2059	2174	2175	2186*	2187*	2209*	2214*	2340*	2341	2342		
R5	=%000005	660	706*	1041*	1045	1048*	1053*	1054	1055	1060	1062	1081	1084*	1088
		1090*	1092	1101	1107	1243	1273	1316	1337	1385	1393	1472	1547	1615
		1621	1661	1663	1709	1750	1752	1824	1955	2019	2022*	2024*	2025*	2029*
		2037	2040	2042	2090	2172*	2174*	2177	2182*	2187	2188	2191	2222*	2341*
		782*	2198	2317	2787*	2797	2799*	2927*	2934*					
SAVKBB	000315													
SAVLDR	001242	1032	1034	1039*										
SAVLOC	000354	819*	1627*											
SAVMAX	000342	812*	1097											
SAVRO	006370	2489*	2516	2530*										
SAVR1	006372	2490*	2517	2531*										
SAVR4	000344	813*	2340											
SAVR5	000346	705	814*	1060*	2182	2191*								
SAVR6	000350	665	690	817*	884*	1239	2193*	2487						

UPNM	007212	995	2205	2724#														
WRTMEN	000120	657#	1379	1515	1604													
SA =	000001	1#	3	4#	6	7#	11	12#	16	17#	21	22#	27	28#				
		34	35#	40	41#	45	46#	49	50#	53	54#	57	58#	61				
		62#	65	66#	69	70#	73	74#	77	78#	82	83#	86	87#				
		90	91#	94	95#	98	99#	100	101#	108	109#	112	113#	116				
		117#	127	128#	131	132#	140	141#	144	145#	147	148#	153	154#				
		157	158#	161	162#	165	166#	173	174#	177	178#	181	182#	186				
		187#	191	192#	198	199#	204	205#	208	209#	213	214#	217#					
SADERR	000301	762#	765	1227*	1348*	1412*	1477*	2396	2413									
SAPTHD	000276	730	736#	749														
SCNTMM	005140	2204	2206#															
SCPUOP	000426	848#																
SDEVCT	000410	839#	1183*	1692*	1815*													
SDDAGN	005542	2329	2337#															
SENDAD	000156	625	676#	2335														
SENV	000420	844#	932	1176	2429	2466												
SEVM	000421	845#	950	1247	2576													
SEOP	005440	2264	2272	2315#														
SETABL	000420	843#																
SETEND	000450	742	872#															
SFATAL	000402	836#	2359*	2368*	2376	2460*	2463											
SGTSIZ	007324	1018	1133	2787#														
SHD =	000002	604																
SHIBTS	000276	737#																
SHIMAX	000334	807#	948*															
SKBB =	177562	789#	2930															
SMADR1	000432	861#																
SMADR2	000436	865#																
SMADR3	000442	868#																
SMADR4	000446	871#																
SMAIL	000400	693	738	742	834#	904	919											
SMAMS1	000430	855#																
SMAMS2	000434	863#																
SMAMS3	000440	866#																
SMAMS4	000444	869#																
SMAAM	000336	808#	949*															
SMBADR	000300	738#																
SMSGAD	000414	841#																
SMSGLC	000416	842#																
SMSGTY	000400	638#	835#	2431*														
SHTYP1	000431	856#	957															
SHTYP2	000435	864#																
SHTYP3	000441	867#																
SHTYP4	000445	870#	953															
SNWTST=	000001	1192#	1194	1279#	1281	1320#	1322	1362#	1364	1455#	1457	1495#	1497	1581#				
		1583	1638#	1640	1713#	1715	1837#	1839	1884#	1886	1977#	1979						
SPASS	000406	838#	896	2319*	2326													
SPASTM	000304	740#																
SPRERR	000300	758#	761	886*	2393	2409	2426*	2512*										
SRETM	007154	2710#	2722															
SSVPC =	000044	623#	628															
SSMR =	000000	604#	613#	1205	1285	1329	1374	1462	1510	1598	1650	1741	1866	1907				
		2008																
SSWREG	000422	846#	934															
STESTN:	000404	697	745	837#	1123*	1180	1204	1284	1328	1373	1461	1509	1597	1649				

DZKMA MACY11 27(732) 04-NOV-76 07:49 PAGE 76
DZKMAB.P11 CROSS REFERENCE TABLE -- MACRO NAMES

.SEOP	18
.SERRO	18
.SERRT	18
.SMULT	18
.SPOWE	18
.SRAND	18
.SRDDE	18
.SRDOC	18
.SREAD	18
.SR2AZ	18
.SSAVE	18
.SSB2D	18
.SSB2O	18
.SSCOP	18
.SSIZE	18
.SSUPR	18
.STRAP	18
.STYPB	18
.STYPD	18
.STYPE	18
.STYPO	18
.S4QCA	18
.1170	18

.ABS	591															
.ASCIZ	670	700	1014	1876	2140	2170	2323	2456	2537	2545	2584	2622	2659	2716	2892	
.BYTE	2903	2936														
.ENABL	780	782	844	845	855	856	863	864	866	867	869	870				
.END	1															
.ENDC	2946															
	599	622	626	628	687	690	724	726	733	799	805	826	833	855	863	
	866	869	874	876	878	1193	1194	1203	1204	1205	1250	1281	1283	1284	1285	
	1321	1322	1327	1328	1329	1363	1364	1372	1373	1374	1456	1457	1460	1461	1462	
	1496	1497	1508	1509	1510	1582	1583	1596	1597	1598	1639	1640	1648	1649	1650	
	1714	1715	1739	1740	1741	1838	1839	1864	1865	1866	1885	1886	1905	1906	1907	
	1978	1979	2006	2007	2008	2940										
.EVEN	671	702	785	833	1015	2142	2171	2325	2458	2539	2547	2585	2623	2660	2718	
	2894	2905	2937													
.IF	595	621	624	626	686	689	723	725	732	798	805	826	832	855	863	
	866	869	872	874	875	877	1192	1194	1203	1205	1279	1281	1283	1285	1320	
	1322	1327	1329	1362	1364	1372	1374	1455	1457	1460	1462	1495	1497	1508	1510	
	1581	1583	1596	1598	1638	1640	1648	1650	1713	1715	1739	1741	1837	1839	1864	
	1866	1884	1886	1905	1907	1977	1979	2006	2008	2940						
.IFF	622	626	687	690	724	726	733	799	806	833	876	878	1192	1193	1194	
	1204	1205	1279	1280	1281	1284	1285	1320	1321	1322	1328	1329	1362	1363	1364	
	1373	1374	1455	1456	1457	1461	1462	1495	1496	1497	1509	1510	1581	1582	1583	
	1597	1598	1638	1639	1640	1649	1650	1713	1714	1715	1740	1741	1837	1838	1839	
	1865	1866	1884	1885	1886	1906	1907	1977	1978	1979	2007	2008				
.IIF	594	599	604	833												
.IRP	1192	1279	1320	1362	1455	1495	1581	1638	1713	1837	1884	1977				
.LIST	1	4	7	12	17	22	28	35	41	46	50	54	58	62	66	
	70	74	78	83	87	91	95	99	101	109	113	117	128	132	141	
	145	148	154	158	162	166	174	178	182	187	192	199	205	209	214	
	217	593	594	611	613	646	826	833	915	963	1075	1097	1164	1192	1205	
	1209	1222	1226	1233	1265	1279	1285	1291	1299	1320	1329	1334	1353	1362	1374	
	1378	1402	1412	1428	1455	1462	1466	1482	1495	1510	1515	1522	1529	1556	1581	
	1598	1604	1614	1638	1650	1655	1689	1713	1741	1746	1769	1785	1794	1837	1866	
	1871	1884	1907	1911	1939	1947	1977	2008	2012	2074	2086	2100	2103	2156	2255	
	2304	2352	2516	2527	2549	2597	2625	2678	2722	2726	2759	2777	2808	2898	2940	
.MACR	609															
.MACRO	1	2	609	1192	1279	1320	1362	1455	1495	1581	1638	1713	1837	1884	1977	
.MCALL	594															
.MEXIT	873															
.NLIST	1	4	7	12	17	22	28	35	41	46	50	54	58	62	66	
	70	74	78	83	87	91	95	99	101	109	113	117	128	132	141	
	145	148	154	158	162	166	174	178	182	187	192	199	205	209	214	
	217	592	594	611	613	646	826	833	915	963	1075	1097	1164	1192	1205	
	1209	1222	1226	1233	1265	1279	1285	1291	1299	1320	1329	1334	1353	1362	1374	
	1378	1402	1412	1428	1455	1462	1466	1482	1495	1510	1515	1522	1529	1556	1581	
	1598	1604	1614	1638	1650	1655	1689	1713	1741	1746	1769	1785	1794	1837	1866	
	1871	1884	1907	1911	1939	1947	1977	2008	2012	2074	2086	2100	2103	2156	2255	
	2304	2352	2516	2527	2549	2597	2625	2678	2722	2726	2759	2777	2808	2898	2940	
.PAGE	634	829	1164	1192	1279	1320	1362	1455	1495	1581	1638	1713	1837	1884	1977	
	2156	2255	2304	2352	2549	2597	2625	2678	2726	2777						
.REPT	1	218	611													
.SBTTL	619	646	687	721	830	876	1164	1192	1279	1320	1362	1455	1495	1581	1638	
	1713	1837	1884	1977	2156	2255	2304	2352	2549	2597	2625	2678	2726	2759	2777	
	2808															
.TITLE	594															
.WORD	617	627	683	737	738	739	740	741	742	793	795	810	812	813	814	

DZKMA MACY11 27(732) 04-NOV-76 07:49 PAGE 81
DZKMAB.P11 CROSS REFERENCE TABLE -- PERMANENT SYMBOLS

817	835	836	837	838	839	840	841	842	846	847	848	861	865	868
871	1236	1416												

ERRORS DETECTED: 0
DEFAULT GLOBALS GENERATED: 0

*,DZKMAB.SEQ/SOL/CRF/PAGNUM/NL:TOC=SYSMAC.CO,DZKMAB.P11
RUN-TIME: 34 44 4 SECONDS
RUN-TIME RATIO: 135/84=1.6
CORE USED: 36K (71 PAGES)

