PDP11

MOS/CORE MEMORY EXERCISER
MD-11-DZKMA-D

EP-DZKMA-D-DL-D

FICHE 1 OF 1

DEC 1977
digital
MADE IN USA

PEPT 0

IDENTIFICATION

PRODUCT CODE	MAINDEC-11-DZKMA-D-D
PRODUCT NAME	MOS/CORE MEMORY EXERCISER FOR 0 TO 124K WITH OR WITHOUT PARITY BITS
DATE CREATED	AUGUST 15, 1977
MAINTAINER	DIAGNOSTIC GROUP

CONTENTS
----

1 0 ABSTRACT

THIS DIAGNOSTIC WILL TEST 0 - 124K OF MOS OR CORE MEMORY
ON ANY PDP-11 FAMILY COMPUTER. SOME TESTS ARE WORST CASE
FOR MOS AND SOME FOR CORE, BUT ALL TESTS ARE ALWAYS RUN
THE TESTS OCCUPIES LESS THAN 2K OF MEMORY SO IT CAN BE
USED TO TEST A SYSTEM WITH ONLY 4K OF MEMORY IF ONLY 4K
EXISTS, HOWEVER, THE ABSOLUTE LOADER IS NOT SAVED.

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

1 1 GETTING STARTED

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

TO START

A SET SWITCH REGISTER = 00000
B START AT 200.
C THE MEMORY LIMITS WILL BE PRINTED
D SEE SECTION 4.4 FOR REST OF PRINTOUTS EXPECTED
E "END PASS #01" WILL BE TYPED LAST, AND THE TEST WILL
   RESTART.
F. TO HALT THE TEST, TYPE CONTROL-C, THIS WILL INSURE THE
   PROGRAM IS RELOCATED BACK TO LOWER MEMORY
      BE PATIENT, THE CONTROL-C IS ONLY RECOGNIZED AT THE END
      OF THE CURRENT SUBTEST
G IF AN UNEXPECTED HALT OCCURS SEE SECTION 6 0 IF AN
   ERROR # IS TYPED SEE SECTION 6 2

   'CAUTION' BEFORE "DIGGING" INTO THE LISTING READ
              SECTION 9

SWITCH SETTING SUMMARY (SEE SECTION 4 1 FOR DETAILS)

BIT15(100000) HALT ON ERROR
BIT14(040000) LOOP IN SUBTEST DEFINED BY BITS <3:0>
BIT13(020000) INHIBIT ERROR PRINTOUTS
BIT12(010000) ENABLE TESTING ABOVE 28K (MEMORY MANAGEMENT)
BIT11(004000) ENABLE PARITY TESTING
BIT10(002000) HALT AFTER EACH SUBTEST
BIT09(001000) INHIBIT PROGRAM RELOCATION
BIT08(000400) TYPE FIRST FAILING BIT ERROR PER 4K
BIT07(000200) ENABLE LONG GALLOPING TEST
BIT06(000100) INHIBIT MEMORY SIZING
BIT05(000040) INHIBIT "END PASS #XX" PRINTOUTS
BIT04(000020) INHIBIT PRINTOUTS
BIT03-BIT00   BEGINNING TEST NUMBER

```
        2.0     REQUIREMENTS

        2.1     EQUIPMENT

                STANDARD 11 FAMILY COMPUTER WITH A CONSOLE OUTPUT DEVICE
                AND FROM 4K TO 124K OF MEMORY


        2.2     STORAGE

                PROGRAM STORAGE - 0000 - 7744   PROGRAM EXPANDS FOR ERROR
                HISTORY AND TO SAVE ABSOLUTE LOADER OR XXDP CHAIN MONITOR
                (SEE SECTION 9  FOR DETAILS)


        3.0     LOADING PROCEDURE

                USE STANDARD PROCEDURE FOR PDP-11 ABSOLUTE BINARY FORMATTED TAPES


        4.0     STARTING PROCEDURE

        4.1     SWITCH SETTINGS

                SOFTWARE SWITCH REGISTER = LOCATION 176

                BIT15(100000)   HALT ON ERROR

                BIT14(040000)   LOOP ON TEST DEFINED BY SWITCH REGISTER BITS <3:0>
                BIT13(020000)   INHIBIT ERROR PRINTOUTS
                BIT12(010000)   ENABLE TESTING ABOVE 28K   (MEMORY MANAGEMENT)

                BIT11(004000)   ENABLE PARITY MODULES.
                                !"PARITY" WILL BE TYPED
                BIT10(002000)   HALT AFTER EACH SUBTEST
                                !PRESS CONTINUE TO DO NEXT SUBTEST
                BIT09(001000)   INHIBIT PROGRAM RELOCATION
                                !IF SET LOCATIONS 430-7776 WILL NOT BE
                                !TESTED

                BIT08(000400)   TYPE FIRST FAILING BIT IN EACH 4K BANK ONLY
                                !THE TOTAL ERROR COUNT (UP TO 377) WILL
                                !BE SAVED IN THE ERROR HISTORY
                BIT07(000200)   ENABLE LONG GALLOPING TEST
                                !"GLP" WILL BE TYPED
                                !CAUTION! INCREASES TEST TIME BY FACTOR OF 25.
                BIT06(000100)   INHIBIT MEMORY SIZING
                                !THE MEMORY LIMITS MUST BE SETUP IN THE FOLLOWING LOCATIONS
                                (VALUES TO TEST 0-8K ARE SHOWN)
                                (LOWTWO=LOCATION 322)

                                LOWTWO  0               ;STORE BITS 17 16 OF LOW TEST ADDRESS
                                LOWADD  0               ;STORE REST OF LOW TEST ADDRESS
```

```
                        HIGHTWO: 0              ;STORE BITS 17.16 OF HIGH TEST ADDRESS
                        HIGHADD: 37776          ;STORE REST OF HIGH TEST ADDRESS
                        NOTE. HIGHADD MUST BE SET TO A 4K BOUNDARY  (E.G.  37776)

            BIT05(000040)   INHIBIT "END PASS #XX" PRINTOUTS

            BIT04(000020)   A   INHIBIT ERROR HISTORY PRINTOUTS.  THE
                                ERROR HISTORY CAN STILL BE OBTAINED
                                BY TYPING CONTROL-C
                            B   INHIBIT PRINTOUTS "PARITY","GLP","TST13 BNK XX"

            BIT03-BIT00     NUMBER OF TEST (0-13) TO RUN FIRST
                            'NORMALLY USED WITH BIT14 (LOOP ON TEST)


    4 2     CONTROL-C OPTION

            CONTROL C   C   AFTER COMPLETION OF THE CURRENT TEST
                            THE ERROR HISTORY (SEE SEC  6 3) WILL BE
                            TYPED  THE PROGRAM WILL HALT IN LOWER MEMORY
                            PRESSING CONTINUE WILL RESTART THE DIAGNOSTIC

    4 3     STARTING ADDRESS= 200
            RESTART ADDRESS = 250 OR 200

            RESTART AT 200 CLEARS PASS COUNT ($PASS) AND PRINTS "DZKMA-D" TITLE


    4 4     PROGRAM AND/OR OPERATOR ACTION

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

            "XXXXX-YYYYY"           ,ADDRESSES OF TEST BOUNDARIES

            "PARITY"                ,IF PARITY OPTION SELECTED

            "GLP"                   ,IF LONG GALLOPING OPTION SELECTED
                                    ,PRINTED AS TST11 IS ENTERED

            "TST13 BNK 00"          ,ENTERING BANK 00 IN TEST 13
            "TST13 BNK 01"          ,AND BANK 1..
            ETC  .                  ,UNTIL ALL BANKS (UP TO 6) HAVE BEEN TESTED
            "RELOC"                 ,THE DIAGNOSTIC RELOCATES TO HIGHEST
                                    ,BANK UNDER TEST  AND RUNS TST0-TST13 AGAIN
            "TST13 BNK 00"          ,TESTING BANK 00 IN TEST 13 (RELOCATED STATE )
                                    ,NOTE-ONLY BANK 00 IS TESTED IN THE RELOCATED STATE

            "END PASS #XX"          ,WHERE "XX" IS THE PASS NO
```

ADDITIONAL PRINTOUTS
"NO PAR" ,PRINTED IF PARITY SELECTED BUT NOT AVAILABLE

"NO MNG" ,PRINTED IF GREATER THAN 28K AND NO MEMORY
,MANAGEMENT AVAILABLE

4 5 LONG GALLOP OPTION

NORMAL WORST CASE SR SETTING = 0000 FOR LONG GALLOP
SR = 200 LONG GALLOP OPTION SHOULD ONLY BE USED IF AN
MOS MEMORY PROBLEM IS SUSPECTED AND NO OTHER SUBTESTS
WILL FAIL. THE TEST TIME IS INCREASED 25 TIMES

5 0 PROGRAM HALTS (NORMAL+ ERROR)

THIS IS A LIST OF EXPECTED HALTS IF THE TEST HALTS
IN A LOCATION NOT IN THIS LIST AND ITS LESS THAN 776, IT
MAY BE DUE TO A DEVICE INTERRUPTING
NOTE THE HALT AT END OF SUBTEST AND HALT ON ERROR HALT LOCATIONS
MAY BE RELOCATED THE ACTUAL LOCATIONS THEY ARE IN CAN BE FOUND
BY SUBTRACTING 500 FROM 1664 SWHALT AND ADDING THIS DIFFERENCE TO THE
CONTENTS OF SAVR6 LOC 346 .

PC	REASON	RECOVERY
112	TRAP TO LOC 4	EXAMINE R6, IT CONTAINS THE POINTER TO THE PC WHERE THE TRAP OCCURRED
146	POWER FAIL	POWER UP WILL RECOVER IF IN CORE MEMORY.
1666	HALT AT END OF TEST SWITCH SET	PRESS CONTINUE TO GO TO NEXT SUBTEST
6132	HALT ON ERROR SWITCH SET	PRESS CONTINUE
6216	CONTROL-C TYPED OR FATAL ERROR OCCURRED	PRESS CONTINUE TO RE-START TEST

6 0 ERRORS

6 1 ERROR MESSAGE FORMAT

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

"LOCATION   GOOD   BAD PC   ERROR   PASFLG"

"ADR ERR" WILL BE PRINTED PRIOR IF AN ADDRESSING ERROR IS SUSPECTED
"PAR ERR" WILL BE PRINTED PRIOR IF A PARITY ERROR TRAP OCCURRED
'CAUTION' IF PARITY ERROR THE GOOD DATA PRINTOUT IS THE
PARITY MODULE UNIBUS ADDRESS THAT FAILED.

WHERE

LOCATION= FAILING MEMORY LOCATION
GOOD   = GOOD DATA DATA THAT WAS EXPECTED
BAD   = BAD DATA DATA THAT WAS FOUND
PC   = PROGRAM COUNTER AT ERROR CALL.
ERROR   = FAILING ERROR NO. (SEE SEC 6 2 - ERROR DICTIONARY)
PASFLG = CONTENTS OF LOCATION PASFLG THIS MAY NOT BE RELEVANT
(SEE SEC 6. 2-ERROR DICTIONARY)

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

'"NO PAR" WILL BE TYPED IF PARITY OPTION SELECTED
'AND NO PARITY MODULES WERE FOUND

(FATAL ERRORS)

"ERROR #XXXXXX" WILL BE TYPED WHERE "XXXXXX" IS
THE ERROR NUMBER THE DIAGNOSTIC WILL USUALLY HALT ON THIS TYPE
OF ERROR SEE SEC 6 2 -ERROR DICTIONARY - FOR DESCRIPTIONS
OF THE ERROR

(APT MODE ERRORS)

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

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

6 2 ERROR DICTIONARY

THIS IS A LIST OF ERROR NUMBERS PRINTED AND POSSIBLE
CAUSES FOR THE ERROR
THE ROUTINE NAME WHERE THE ERROR CALL ORIGINATED IS GIVEN IN
BRACKETS
NOTE- "BAKPAT" REFERS TO THE BACKGROUND PATTERN WRITTEN INTO MEMORY

```
                FOR VARIOUS TESTS  IF PARITY SELECTED IT HAS A VALUE = 376 ,ELSE=377
                "SWAPPED BAKPAT" = 77000 IF PARITY SELECTED, ELSE=77400

         ENDR


        ,ERROR # 0      , BUSER  BUS ERROR TRAP TO LOC  4 OCCURRED
                        ,       THIS ERROR IS NOT PRINTED AND IS FOR "APT" USE

        ,ERROR # 1      , TSTTRP FATAL DATA ERROR
                        ,LOCATIONS 0000-430 FAILED 1'S + 0'S TEST
                        ,R0 = GOOD DATA
                        ,R1 = 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      ; TSTO   TEST SEQUENCE ERROR
                        ;TSTO 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      ; TSTO  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      ; TSTO  ADDRESS AND DATA ERROR
                        ,IDENTICAL TO PREVIOUS ERROR EXCEPT THE DATA
                        ,WRITTEN INTO THE FAILING LOCATION WAS IN
                        ,ERROR ALSO.

        ,ERROR # 10     , TSTO  DATA EPROR
                        ;IF BAD DATA = 0C00 COULD BE AN ADDRESSING
                        ,ERROR , ELSE COMPARE GOOD AND BAD DATA FOR FAILING BITS

        ;ERROR # 11     ; TSTO  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
                        ;$TESTN  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
                        ,$TESTN  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
                        ,$TFSTN  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
                        ,$TESTN  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
                        ;$TESTN 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
                        ;$TESTN 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 CORPECTLY
                        ;             BUT LOST THE DATA

        ;ERROR # 34      ; TST10  TEST SEQUENCE ERROR
                        ;$TESTN 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
                        ;$TESTN SHOULD = 11
                        ;       THE DIAGNOSTIC HAS BEEN CORRUPTED

        ;ERROR # 41      ; TST12  TEST SEQUENCE ERROR
                        ;$TESTN 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
                        ,               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 FOUNDJUST 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 (R0)" (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
```

6 3 ERROR HISTORY

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

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

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

ERROR HISTORY FORMAT

ERROR	BANK	COUNT

WHERE

ERROR = BIT THAT FAILED NUMBER OF THE FAILING BIT IN DECIMAL I E 0-15 WILL BE TYPED OUT OR THE WORDS "ADR ERR" 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 PAR-
TICULAR 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.
1 THE PAR (PAGE ADDRESS REGISTER) CONTENTS ARE SHOWN IN A TEST THAT SELF SIZES
   IF THE LIMITS OF TESTING ARE SET BY THE OPERATOR AND
   IF THE BANK IS ABOVE 28K PAR NO. 1 WILL BE SET TO THE
   BEGINNING PAGE FOR EXAMPLE IF THE TESTING WAS TO
   BEGIN WITH BANK 8 PAR NO. 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

$PASS 406 = PASS COUNT - CLEARED BY START AT 200

$TESTN 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-D" 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 $EOP 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-D" 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 PXDP CHAIN MONITOR VIA LOCATION 42

ACT11

OPERATION IS IDENTICAL TO STAND ALONE EXCEPT

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

APT

OPERATION IS SIMILAR TO STAND ALONE EXCEPT

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

APT MANAGER INFORMATION

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

THE EXAMPLE ASSUMES YOU ARE LOGGED INTO THE APT MONITOR

   READY

```
RUN APPLU
APT 11 PAPER TAPE PROGRAM LOAD UTILITY

THE FOLLOWING COMMANDS ARE VALID

    ED      EDIT A PROGRAM
    LI      LIST A PROGRAM


COMMAND   ED
PROGRAM NAME TO EDIT   EXAMPL
DO YOU WANT TO LOAD A NEW REV OF THE PROGRAM(Y/N)? N
FIRST PASS RUN TIME IN SECONDS <110>
LONGEST TEST TIME IN SECONDS <10>
ADDITIONAL RUN TIME IN SECONDS <0>
WHICH ETABLE DO YOU WISH TO EDIT? A
SOFTWARE ENVIRONMENT<000>   1
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
  982                                    LIST   ME,BIN,SEQ,LOC
  983                                    TITLE  DZKMA
  984                                   ,*COPYRIGHT (C) AUGUST 1977
  985                                   ,*DIGITAL EQUIPMENT CORP.
  986                                   ,*MAYNARD, MASS. 01754
  987                                   ,*
  988                                   ,*PROGRAM BY PERVEZ ZAKI
  989                                   ,*
  990                                   ,*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
  991                                   ,*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977
  992                                   ,*
  993           160000                  $SWR=160000     ,,HALT ON ERROR, LOOP ON TEST, INHIBIT ERROR TYPOUT
  994
  995
  996
  997
  998                                           ,,TRAP CATCHER OF  +2 AND HALT FOR 0-776 LOCATIONS
  999
 1000
 1001
 1002
 1003           000240                  SCOPE   =NOP
 1004
 1005           000042                           =42
 1006  000042   000000                           WORD   0        ,FOR ACT/XXDP
 1007
 1008                                   SBTTL  ACT11 HOOKS
 1009
 1010                                  ,,*****************************************************************
 1011                                  ,HOOKS REQUIRED BY ACT11
 1012           000044                          $SVPC=                  ,SAVE PC
 1013           000046                           =46
 1014  000046   000156                          $ENDAD                  ,,1)SET LOC 46 TO ADDRESS OF $ENDAD IN  $EOP
 1015           000052                           =52
 1016  000052   040000                           WORD   40000            ,,2)SET LOC 52 TO 40000
 1017           000044                           =$SVPC                  ,, RESTORE PC
 1018
 1019           000070                           =70
 1020  000070   012737  000136  000024  PWRDN   MOV     #PWRUP,@#24
 1021  000076   000000                          HALT
 1022
```

```
1023
1024
1025            000104                  .=104
1026                                    ; GET HERE IF AN ILLEGAL TRAP TO LOC  4 OCCURRED
1027   000104   005237   000400          BUSER   INC     @#$MSGTY        ;TELL APT FATAL ERROR#000
1028   000110   000000                          HALT                    ;*ERROR* TRAP TO LOC  4 OCCURRED
1029   000112   000000                          HALT                    ;IN CASE CONTINUE PRESSED
1030                                    ;114 AND 116 ARE RESERVED FOR PARITY TRAP VECTORS   SETUP IN
1031                                    ;ROUTINE "BEGIN"
1032            000120                          =120
1033
1034
1035
1036                                            ;* WRITE MEMORY BACKGROUND
1037                                            ;* ----------------------
1038                                            *
1039                                            ;*      THIS ROUTINE IS USED TO WRITE THE MEMORY BACKGROUND TO
1040                                            ;*      THE VALUE STORED AT LOCATION BAKPAT.  THE ROUTINE ASSUMES
1041                                            ;*      THAT R4 IS POINTING TO THE LOWEST LOCATION AND R5 TO THE
1042                                            ;*      HIGHEST LOCATION TO BE WRITTEN  THE PROGRAM LEAVES THE
1043                                            ;*      SUBROUTINE WITH R0 CONTAINING THE CONTENTS OF BAKPAT
1044                                            ;*
1045
1046   000120   010401          WRTMEM  MOV     R4,R1           ;SET R1 TO LOWEST LOCATION UNDER TEST
1047   000122   013700   000316          MOV     @#BAKPAT,R0     ;LOAD R0 WITH THE CONTENTS OF LOCATION BAKPAT
1048   000126   010021          2$      MOV     R0,(R1)+        ;STARTING FROM THE LOWEST LOCATION WRITE THE
1049   000130   020105                  CMP     R1,R5           ;MEMORY TO BACK GROUND PATTERN
1050   000132   103775                  BLO     2$
1051   000134   000207                  RTS     PC              ;RETURN FROM THE SUBROUTINE
1052
1053
1054   000136   013706   000350          PWRUP   MOV     @#SAVR6,SP      ;RESTORE STACK POINTER
1055   000142   012700   006072                  MOV     #PNTMES-BEGIN,R0
1056   000146   060600                          ADD     SP,R0           ;GET THE INDIRECT ADDRESS OF LOCATION TPCRLF
1057                                                                   ;RELATIVE TO LOCATION OF DIAGNOSTIC IN THE CORE
1058   000150   004710                          JSR     PC,(R0)         ;GO TO THE TYPE ROUTINE AND TYPE CR, LF AND A "P"
1059   000152   000120                           ASCIZ  /P/
1060                                            EVEN
1061
1062   000154   000411                          BR      START
1063
1064                                    ;* SERVICE XXDP/ACT11
1065   000156   004710          $ENDAD  JSR     PC,(R0)         ;RETURN TO ACT11/XXDP MONITOR
1066   000160   000240                  NOP                     ;IF QUICK VERIFY=RESET ELSE NOP
1067   000162   000240                  NOP                     ;IF QUICK VERIFY=CLR #-1 ELSE INC #0
1068   000164   000240                  NOP                     ;IF QUICK VERIFY=BR  -4 ELSE NOP
1069   000166   000430                  BR      RESTRT           REPEAT TEST UNDER ACT11/XXDP
1070
1071            000176                  =176
1072   000176   000000          SWREG   .WORD   0
1073
1074
1075                                    ;;*****************************************************************
1076                                     SBTTL  START AND RESTART ROUTINES
1077                                     *      RESTART AT 200 TO CLEAR APT TABLES
1078                                    ;;*****************************************************************
```

```
1079  000200  013706  000350          START   MOV     @#SAVR6,SP      ;SETUP STACK POINTER
1080  000204  012703  000412                  MOV     #$UNIT,R3       ;CLEAR THE APT MAILBOX FROM $MAIL TO $DEVCT
1081  000210  005043                  1$      CLR     -(R3)           ;CLEAR A MAILBOX LOCATION
1082  000212  022703  000400                  CMP     #$MAIL,R3       ;DONE?
1083  000216  001374                          BNE     1$              ;BRANCH IF NO
1084  000220  105737  000042                  TSTB    @#42            ;ACT11 MODE?
1085  000224  001011                          BNE     RESTRT          ;BRANCH IF YES
1086  000226  105737  000405                  TSTB    @#$TESTN+1              ;ARE WE RELOCATED?
1087  000232  100406                          BMI     RESTRT          ;BR IF YES- SINCE TPCRLF IS RELOCATED ALSO-
1088  000234  004767  006324                  JSR     PC,TPCRLF       ;PRINT TITLE                                    ;
1089  000240  055104  046513  026501           ASCIZ   /DZKMA-D/
1090  000246  000104
1091                                           EVEN
1092
1093  000250  012704  007744          RESTRT  MOV     #ENDPRG,R4      ;LOAD R4 WITH THE ADDRESS OF THE END OF THE PROGRAM
1094  000254  012703  000346                  MOV     #SAVR5,R3       ;CAUSE R3 TO POINT TO THE LOCATION SAVR5
1095  000260  012305                          MOV     (R3)+,R5        ;RESTORE R5
1096  000262  012306                          MOV     (R3)+,SP        ;AND RESTORE R6 JUST IN CASE IT IS A RESTART
1097  000264  010600                          MOV     SP,R0           ;PLACE THE STARTING ADDRESS OF THE TEST IN R0
1098  000266  012746  000340                  MOV     #340,-(SP)      ;SET HIGH PRIORITY FOR RTI
1099  000272  010046                          MOV     R0,-(SP)
1100  000274  000002                          RTI                     ;GO TO "START"-MAY BE RELOCATED
1101                                                                   ;IF RELOCATED SEE LOCATION SAVR6 FOR START
1102
1103
1104
1105
1106
1107
1108
1109
1110                                    SBTTL  APT PARAMETER BLOCK
1111
1112                                    ;;*****************************************************************
1113                                    ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
1114                                    ; *****************************************************************
1115          000276                            $X=     ;;SAVE CURRENT LOCATION
1116          000024                            =24     ;;SET POWER FAIL TO POINT TO START OF PROGRAM
1117  000024  000200                           200     ;;FOR APT START UP
1118          000044                            =44     ;;POINT TO APT INDIRECT ADDRESS PNTR
1119  000044  000276                           $APTHDR ;;POINT TO APT HEADER BLOCK
1120          000276                            = $X    ;;RESET LOCATION COUNTER
1121                                    ; *****************************************************************
1122                                    ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
1123                                    ;INTERFACE SPEC
1124
1125  000276                            $APTHD
1126  000276  000000                   $HIBTS  WORD    0       ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR
1127  000300  000400                   $MBADR  WORD    $MAIL   ;;ADDRESS OF APT MAILBOX (BITS 0-15)
1128  000302  000012                   $TSTM   .WORD   10      ;;RUN TIM OF LONGEST TEST
1129  000304  000156                   $PASTM  .WORD   110     ;;RUN TIME IN SECS  OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
1130  000306  000000                   $UNITM  WORD            ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
1131  000310  000024                           WORD    $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
1132
1133
1134          000405                    REL=$TESTN+1                  IT WILL BE 0 IF THE PROGRAM IS IN THE LOWER
```

```
                                                ;CORE  BIT 7 OF THE BYTE WILL BE SET IF THE
                                                ,PROGRAM IS IN A RELOCATED STATE AND BIT 5
                                                ,WILL BE SET IF PARITY BITS ARE BEING TESTED
            000276                  =$APTHD
000276      MMAVA                               ;THIS BYTE IS USED TO DETERMINE IF MEMORY
                                                ,MANAGEMENT IS AVAILABLE OR NOT
            000277                  =MMAVA+1
000277      TYPENB                              ,THIS BYTE IS USED TO DETERMINE IF THE
                                                ,TYPE OUT OF ERROR HAS BEEN ENABLED OR NOT
            000300                  =TYPENB+1
000300      $PRERR                              ,THIS BYTE DETERMINES IF THE PROGRAM HAS FOUND
                                                ,A PARITY ERROR
            000301                  =$PRERR+1
000301      $ADERR                              ,THIS BYTE IS USED TO DETERMINE IF THE
                                                ,PROGRAM HAS ENCOUNTERED ADDRESS ERROR
            000302                  =$ADERR+1
000302      STRTDI
            000304                  =STRTDI+2
000304      LOWBNK
            000306                  =LOWBNK+2
000306      PASFLG                              ,LOWER BYTE OF THIS WORD GIVES THE PASS NUMBER FOR
                                                ,THE SPECIFIC TEST WHEREAS THE UPPER BYTE
                                                ,HAS BEEN USED BY DIFFERENT TEST FOR DIFFERENT PURPOSES
            000310                  =PASFLG+2
000310      ENDSTK
            000312                  =ENDSTK+2
000312      PBNK                    ,HOLDS BANK UNDER TEST FOR "TST BNK XX" PRINTOUT
000312      DECWRD
            000314                  =DECWRD+2
000314  000  TYPCNT   BYTE   0                 ,THIS BYTE DETERMINES THE NUMBER OF WORDS
                                                ,TO BE TYPED
000315  000  SAVKBB   BYTE   0                 ,THIS LOCATION IS USED TO SAVE THE CHARACTER
                                                ,HIT BY THE OPERATOR
                                                ,ALSO IS USED AS TEMP IN ROUTINE $GTSIZ
                 EVEN

            177560              TKS=    177560
            177562              $KBB=   177562
            177564              $TPS=   177564
            177566              $TPB=   177566
            177572              SRO=    177572
000316  000377  BAKPAT  .WORD   377           ,BACKGROUND PATTERN WRITTEN TO MEMORY

000320  000000  SWAPAT  .WORD
000322  000430  RELBOT. BEGIN-50              ,HOLDS LOWEST TEST ADDRESS WHEN RELOCATED

                ,,**********************************************************************
                ,LOCATIONS TO BE MODIFIED IF LIMITS SET BY OPERATOR
000324  000000  LOWTWO  0                     ,HOLDS BITS 17 16 OF LOW TEST ADDRESS
000326  000000  LOWADD  0                     ,HOLDS BITS 15 0 OF LOW TEST ADDRESS
```

```
1191
1192  000330  000000          HIGHTWO:    0              ,HOLDS BITS 17 16 OF HIGH TEST ADDRESS
1193  000332  037776          HIGHADD.    37776          ;HOLDS BITS 15 0 OF HIGH TEST ADDRESS
1194                          ,,**************************************************************
1195
1196  000334  000000          $HIMAX: 0                  ,HOLDS BITS 17 16 OF MAXIMUM AVAILABLE MEMORY
1197  000336  017776          $MAXM   17776              ;HOLDS BITS 15 0 OF MAXIMUM AVAILABLE MEMORY
1198
1199  000340  000000          MAXMEM.   WORD   ,MAXIMUM CUPRENT VIRTUAL MEMORY UNDER TEST
1200
1201  000342  000000          SAVMAX.   WORD
1202  000344  000000          SAVR4     WORD
1203  000346  000000          SAVR5:    .WORD
1204
1205                          .* SAVR6 POINTS TO WHERE THE PROGRAM STARTS EVEN WHEN RELOCATED
1206  000350  000500          SAVR6     WORD   BEGIN          ;CONTAINS START ADDRESS WHEN RELOCATED ALSO
1207  000352  000000          PARMAP  0              ,MAP OF PARITY MODULES UNDER TEST
1208  000354  000000          SAVLOC  0              ,TEST 6 STORES ERROR INFO HERE
1209  000356  000000          PARSP   0       ,SAVE SP DURING PARITY ERROR TRAP
1210  000360  000000          PARPS   0              ,SAVE PSW DURING PARITY ERROR TRAP
1211                                                 ,NOTE-PARSP +PARPS ARE NEEDED SINCE THERE IS
1212                                                 ;IS NOT ENOUGH POOM ON THE STACK (500-452) AND
1213                                                 ,SO THE STACK MUST BE RESET IN THE PARERR ROUTINE
1214                                                 ,IN THIS CRUDE FASHION
1215
1216
1217                           *364-400 IS USED AS A STACK AREA BY ERRCHK ROUTINE FOR ERROR HISTORY PRINTOUT
```

```
 1218           000400                          =400
 1219                                  SBTTL  APT MAILBOX-ETABLE
 1220
 1221                                 ;;*****************************************************************
 1222                                  EVEN
 1223  000400                         $MAIL.                          ;;APT MAILBOX
 1224  000400  000000                 $MSGTY. .WORD   AMSGTY  ;;MESSAGE TYPE CODE
 1225  000402  000000                 $FATAL  WORD    AFATAL  ;;FATAL ERROR NUMBER
 1226  000404  000000                 $TESTN: WORD    ATESTN  ;;TEST NUMBER
 1227  000406  000000                 $PASS   .WORD   APASS   ;;PASS COUNT
 1228  000410  000000                 $DEVCT. .WORD   ADEVCT  ;;DEVICE COUNT
 1229  000412  000000                 $UNIT   WORD    AUNIT   ;;I/O UNIT NUMBER
 1230  000414  000000                 $MSGAD. WORD    AMSGAD  ;;MESSAGE ADDRESS
 1231  000416  000000                 $MSGLG  WORD    AMSGLG  ;;MESSAGE LENGTH
 1232  000420                         $ETABLE                         ;;APT ENVIRONMENT TABLE
 1233  000420     000                 $ENV    .BYTE   AENV    ;;ENVIRONMENT BYTE
 1234  000421     000                 $ENVM   BYTE    AENVM   ;;ENVIRONMENT MODE BITS
 1235  000422  000000                 $SWREG: WORD    ASWREG  ;;APT SWITCH REGISTER
 1236  000424  000000                 $USWR   WORD    AUSWR   ;;USER SWITCHES
 1237  000426  000000                 $CPUOP  WORD    ACPUOP  ;;CPU TYPE,OPTIONS
 1238                                 ;*                      BITS 15-11=CPU TYPE
 1239                                 ;*                              11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
 1240                                 ;*                              11/70=06,PDQ=07,Q=10
 1241                                 ;*                      BIT 10=REAL TIME CLOCK
 1242                                 ;*                      BIT  9=FLOATING POINT PROCESSOR
 1243                                 ;*                      BIT  8=MEMORY MANAGEMENT
 1244  000430     000                 $MAMS1  BYTE    AMAMS1  ;;HIGH ADDRESS,M S  BYTE
 1245  000431     000                 $MTYP1  BYTE    AMTYP1  ;;MEM  TYPE,BLK#1
 1246                                 ;*                      MEM TYPE BYTE   --  (HIGH BYTE)
 1247                                 ;*                              900 NSEC CORE=001
 1248                                 ;*                              300 NSEC BIPOLAR=002
 1249                                 ;*                              500 NSEC MOS=003
 1250  000432  000000                 $MADR1  WORD    AMADR1  ;;HIGH ADDRESS,BLK#1
 1251                                 ;*                        MEM LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
 1252  000434     000                 $MAMS2. BYTE    AMAMS2  ;;HIGH ADDRESS,M.S.  BYTE
 1253  000435     000                 $MTYP2  BYTE    AMTYP2  ;;MEM.TYPE,BLK#2
 1254  000436  000000                 $MADR2. .WORD   AMADR2  ;;MEM.LAST ADDRESS,BLK#2
 1255  000440     000                 $MAMS3: .BYTE   AMAMS3  ;;HIGH ADDRESS,M.S.BYTE
 1256  000441     000                 $MTYP3  .BYTE   AMTYP3  ;;MEM.TYPE,BLK#3
 1257  000442  000000                 $MADR3: .WORD   AMADR3  ;;MEM.LAST ADDRESS,BLK#3
 1258  000444     000                 $MAMS4: .BYTE   AMAMS4  ;;HIGH ADDRESS,M S.BYTE
 1259  000445     000                 $MTYP4: .BYTE   AMTYP4  ;;MEM.TYPE,BLK#4
 1260  000446  000000                 $MADR4. WORD    AMADR4  ;;MEM LAST ADDRESS,BLK#4
 1261  000450                         $ETEND:
 1262                                  MEXIT
 1263
 1264                                 ;;*****************************************************************
 1265                                  SBTTL  BEGIN OF AREA TESTED (+20) WHEN PROGRAM RELOCATES
```

```
1266                                    ;;*****************************************************************
1267  000450  177570                    SWR:    177570  ;CHANGES TO SWREG IF NO HARDWARE SWITCH REGISTER
1268
1269          000500                     =500
1270  000500  010706                    BEGIN   MOV     PC,SP           ;SET UP STACK POINTER TO EQUAL BEGIN ADDRESS
1271
1272  000502  005746                            TST     -(SP)
1273  000504  010637  000350                    MOV     SP,@#SAVR6      ;SAVE SP FOR FUTURE USE
1274  000510  012737  000070  000024            MOV     #PWRDN,@#24     ;PREPARE FOR ANY FUTURE POWER DOWN
1275  000516  005037  000300                    CLR     @#$PRERR
1276  000522  005037  000314                    CLR     @#TYPCNT
1277  000526  012700  000114                    MOV     #114,R0         ;PREPARE TO SETUP PARITY TRAP VECTOR
1278  000532  012710  005462                    MOV     #PARERR-.-6,(R0)
1279  000536  060720                            ADD     PC,(R0)+        ;TO PARERR
1280  000540  012710  000340                    MOV     #340,(R0)       ;AND PSW OF 340
1281  000544  105737  000405                    TSTB    @#REL           ;IS THIS CODE RELOCATED?
1282  000550  100002                            BPL     ONEPAS          ;BRANCH IF NO
1283  000552  000167  000546                    JMP     TSTREL          ;THIS CODE IS RELOCATED SO GET TEST SIZE
1284
1285  000556  005737  000406            ONEPAS. TST     @#$PASS         ;IS THIS THE FIRST PASS?
1286  000562  001402                            BEQ     TSTRP           ;BRANCH IF YES (TEST TRAP CATCHER ADDRESSES)
1287  000564  000167  000400                    JMP     SETSTK          ; GET THE TEST SIZE
1288  000570  012704  007744            TSTRP   MOV     #ENDPRG ,R4     ;LOAD R4 WITH THE ADDRESS OF THE END OF THE PROGRAM
1289  000574  012700  000377                    MOV     #377,R0
1290  000600  010037  000316                    MOV     R0,@#BAKPAT
1291  000604  005001                            CLR     R1
1292  000606  012124                    2$      MOV     (R1)+,(R4)+     ;SAVE FROM 0000 TO BEGIN-30 AT END OF PROGRAM FOR NOW
1293  000610  020127  000400                    CMP     R1,#$MAIL
1294  000614  103774                            BLO     2$
1295  000616  005741                    3$      TST     -(R1)           ;PREPARE TO TEST THE TRAP VECTORS
1296  000620  010011                    4$      MOV     R0,(R1)         ;CHECK THE TRAP VECTORS FOR THE CAPABILITY
1297                                            ;OF HOLDING 0'S & 1'S
1298  000622  020011                            CMP     R0,(R1)         ;IS THE DATA OK?
1299  000624  001403                            BEQ     6$              ;BRANCH IF YES
1300
1301  000626  004767  005310                    JSR     PC,FATERR       ;*ERROR* REPORT  ERROR MESSAGE AND HALT AT FATHLT
1302  000632  000001                            1                       ;******ERROR NUMBER 1******
1303
1304  000634  000300                    6$      SWAB    R0
1305  000636  001370                            BNE     4$
1306  000640  005701                            TST     R1              ;IF WE HAVE NOT REACHED THE LOWEST MEMORY LOCATION
1307  000642  001365                            BNE     3$              ;THEN REPEAT FROM 3$
1308  000644  012701  000400                    MOV     #$MAIL,R1
1309  000650  014441                    8$      MOV     -(R4),-(R1)     ;RESTORE TRAP CATCHER ETC
1310  000652  005701                            TST     R1
1311  000654  001375                            BNE     8$
1312  000656  012700  000006            SETSWR: MOV     #6,R0
1313  000662  012710  000340                    MOV     #340,(R0)       ;SET UP TIME OUT TRAP PSW
1314  000666  012740  000700                    MOV     #4$,-(R0)       ;AND THE RETURN ADDRESS
1315  000672  005777  177552            2$:     TST     @SWR            ;DOES THE SWITCH REGISTER POINTED BY SWR EXIST ?
1316  000676  000404                            BR      5$              ;BRANCH IF YES
1317  000700  022626                    4$:     CMP     (SP)+,(SP)+     ;RESTORE THE STACK POINTER
1318  000702  012737  000176  000450            MOV     #SWREG,@#SWR    ;AND PLACE THE ADDRESS OF THE SWITCH REGISTER
1319                                            ;DESIGNED FOR THE COMPUTERS NOT HAVING HARDWARE
1320                                            ;SWITCH REGISTER AND RUNNING STAND ALONE
1321  000710  105737  000420            5$      TSTB    @#$ENV          ;RUNNING UNDER APT?
```

```
1322  000714  001403                          BEQ     APTSIZ          ;BRANCH IF NO
1323  000716  012737  000422  000450          MOV     #$SWREG,@#SWR   ;SET SWR EQUAL TO APT SWITCH REGISTER
1324
1325
1326
1327
1328                                  ;APTSIZ- THIS ROUTINE WILL SEARCH THE APT MEMORY ETABLE AND WHEN
1329                                  ;A NON ZERO TYPE IS FOUND WILL SETUP TO TEST TO GIVEN HIGH ADDRESS.
1330                                  ;       IF APT DEFINES SIZE THE LOW TEST ADDRESS MUST=00000. (DUE TO ETABLE FORMAT)
1331                                  ;FLOW;
1332                                  ;       IF BLOCK 4 (OR 3,2,1) TYPE NON ZERO THEN GET APT HIGH ADDRESS AND EXIT
1333                                  ;       ELSE    SEND ERROR #3
1334                                  ;NOTE, THE MEMORY TYPE IS IGNORED SINCE ALL TESTS ARE RUN REGARDLESS OF MEMORY TYPE
1335
1336  000724  012703  000340          APTSIZ: MOV     #MAXMEM,R3      ;POINT R3 TO MAXMEM.
1337  000730  013737  000330  000334          MOV     @#HIGHTWO,@#$HIMAX      ;IN CASE NO SELF SIZING DONE
1338  000736  013737  000332  000336          MOV     @#HIGHADD,@#$MAXM       ;IN CASE NO SELF SIZING DONE
1339  000744  105737  000421                  TSTB    @#$ENVM         ;DOES APT ALLOW SELF SIZING?
1340  000750  100021                          BPL     TRYSR           ;BRANCH IF YES
1341
1342  000752  012701  000451                  MOV     #$MTYP4+4,R1    ;POINT R1 TO BLOCK TYPE 4(+4)
1343  000756  162701  000004          1$      SUB     #4,R1           ;POINT R1 TO NEXT BLOCK TYPE.
1344  000762  105711                          TSTB    (R1)            ;IS THE BLOCK TYPE NON ZERO?
1345  000764  001006                          BNE     2$              ;BRANCH IF YES (MEMORY EXISTS)
1346  000766  020127  000431                  CMP     R1,#$MTYP1      ;ALL APT BLOCK TYPES BEEN CHECKED?
1347  000772  101371                          BHI     1$              ;BRANCH IF NO
1348
1349  000774  004767  005142                  JSR     PC,FATERR       ;*ERROR* REPORT  ERROR MESSAGE AND HALT AT FATHLT
1350  001000  000002                          2                       ;******ERROR NUMBER 2******
1351
1352  001002  004767  006306          2$:     JSR     PC,GETADR       ;GO SET MAXIMUM APT ADDRESS INTO $MAXM + $HIMAX
1353  001006  004767  006302                  JSR     PC,GETADR       ;GO SET MAXIMUM APT ADDRESS INTO HIGHADD+HIGHTWO
1354  001012  000450          BRTPSZ: BR      TYPSIZ          ; TYPE THE SIZE OF MEMORY UNDER TEST
1355
1356  001014  032777  000100  177426  TRYSR:  BIT     #100,@SWR       ;USER DEFINED MEMORY TEST BOUNDARIES??
1357  001022  001044                          BNE     TYPSIZ          ;BRANCH IF YES (DON'T SIZE MEMORY)
1358
1359
1360
1361
1362
1363  001024  010401          SLFSIZ: MOV     R4,R1           ;SETUP R1 AND R4 TO THE LOWEST ADDRESS OF MEMORY
1364  001026  012710  001042                  MOV     #4$,(R0)        ;SET UP RETURN ADDRESS FROM TIME OUT TRAP TO 4$
1365  001032  011111          2$      MOV     (R1),(R1)       ;WRITE A MEMORY LOCATION INTO ITSELF AND TRAP IF NONEXIS
1366
1367  001034  062701  000002                  ADD     #2,R1           ;ADD 2 TO THE ADDRESS POINTER
1368  001040  000774                          BR      2$              ;KEEP ON SIZING UP THE MEMORY UNTIL
1369                                                                  ;NXM TRAP (TIME OUT TRAP) IS ENCOUNTERED
1370
1371  001042  022626          4$      CMP     (SP)+,(SP)+     ;RESTORE THE STACK POINTER
1372  001044  004767  005774                  JSR     PC,MEMMNG       ; SERVICE MEMORY MANAGEMENT IF IT IS AVAILABLE
1373                                                                  ;AND IF IT HAS TO BE TESTED
1374  001050  105737  000276                  TSTB    @#MMAVA         ;SEE IF MEMORY MANAGEMENT HAS TO BE TESTED
1375  001054  001416                          BEQ     12$             ;IF NO MEM. MANG.  THEN GO TO 12$
1376  001056  012710  001070          6$      MOV     #8$,(R0)        ;SET UP THE RETURN ADDRESS FROM TRAP TO 8$
1377  001062  012701  020000                  MOV     #20000,R1       ;BEGIN CHECKING MEMORY ABOVE 28K
```

```
1378  001066  000761                    BR      2$
1379  001070  022626          8$        CMP     (SP)+,(SP)+     ;RESTORE STACK POINTER
1380  001072  022701  160000            CMP     #160000,R1      ;IF R1 DID NOT READ ALL THE LOCATIONS POINTED BY
1381                                                            ;PAGE ADDRESS REGISTER 6 THEN IT HAS REACHED THE
1382                                                            ;MAXIMUM AVAILABLE MEMORY
1383  001076  001005                    BNE     12$             ,IN WHICH CASE GO TO 12$
1384  001100  013702  172352            MOV     @#172352,R2     ;PREPARE TO UPDATE MEMORY MANAGEMENT REGISTERS
1385  001104  004767  005740            JSR     PC,MMREG                ;OTHERWISE GO TO UPDATE MEM. MANG. REGISTERS
1386  001110  000762                    BR      6$
1387  001112  024341          12$       CMP     -(R3),-(R1)     ;CAUSE R3 TO POINT TO LOCATION $MAXM AND R1
1388                                                            ,TO THE MAXIMUM AVAILABLE MEMORY
1389  001114  004767  006104            JSR     PC,PUTADR       ;GO TO THE SUBROUTINE TO PLACE THE ADDRESS IN R1
1390                                                            ,AT LOCATIONS $MAXM AND $HIMAX
1391  001120  024343                    CMP     -(R3),-(R3)     ,MAKE R3 POINT TO HIGHADD
1392  001122  004767  006076            JSR     PC,PUTADR       ,PLACE THE ADDRESS IN R1 AT LOCATIONS HIGHADD
1393                                                            ,AND HIGHTWO
1394  001126  005743                    TST     -(R3)
1395  001130  005043                    CLR     -(R3)           ,CLEAR THE LOCATION LOWADD
1396  001132  005043                    CLR     -(R3)           ,AND LOWTWO
1397  001134  012720  000104  TYPSIZ    MOV     #BUSER,(R0)+    ;SET UP VECTOR FOR ANY FUTURE TRAP
1398  001140  010403                    MOV     R4,R3           ,SET R3 TO POINT TO THE LOWEST AVAILABLE MEMORY
1399                                                            ,LOCATION
1400  001142  012701  000324            MOV     #LOWTWO,R1
1401  001146  004767  005400            JSR     PC,PCRLF        ,TYPE CR/LF
1402  001152  004767  005546            JSR     PC,OCTTYP       ,TYPE LOW TEST ADDRESSE (LOWTWO+LOWADD)
1403  001156  004767  005302  TYPMEM:   JSR     PC,$TYPE        ,  TYPE "-"
1404  001162  000055                     ASCIZ  /-/
1405                                     EVEN
1406  001164  004767  005534            JSR     PC,OCTTYP       ,TYPE HIGHEST TEST ADDRESS (HIGHTWO+HIGHADD)
1407  001170  012703  000330  SETSTK    MOV     #HIGHTWO,R3     ,MAKE R3 POINT TO THE HIGH ORDER BITS OF TOP ADDRESS
1408  001174  004767  006130            JSR     PC,$GTSIZ       ,  GET THE BITS 13-17 OF THE TOP ADDRESS
1409                                                            ,PLACED IN BITS 0-4 OF R2
1410  001200  010401                    MOV     R4,R1           ,SET R1 TO LOWEST TEST ADDRESS
1411
1412  001202  062704  000022  4$        ADD     #18 ,R4         ;APPEND THE ERROR STACK FOR THE MEMORY UNDER
1413                                                            ,TEST TO THE END OF THE PROGRAM
1414  001206  005302                    DEC     R2
1415  001210  002374                    BGE     4$
1416  001212  010437  000310            MOV     R4,@#ENDSTK     ,SAVE THE ADDRESS OF THE END OF THE ERROR STACK
1417  001216  005021          6$        CLR     (R1)+           ;CLEAR THE ERROR STACK
1418  001220  020104                    CMP     R1,R4
1419  001222  101775                    BLOS    6$
1420  001224  012737  157776  000340    MOV     #157776,@#MAXMEM                ,SET MAXMEM TO MAXIMUM VIRTUAL ADDRESS
1421  001232  005723                    TST     (R3)+           ,TESTING MEMORY MANAGEMENT?
1422  001234  001004                    BNE     SAVLDR          ,BRANCH IF YES (GO SAVE LOADERS AT TOP OF VIRTUAL MEMORY
1423  001236  021300                    CMP     (R3),R0         ;IS THE VIRTUAL ADDRESS ABOVE 157776?
1424  001240  103002                    BHIS    SAVLDR          ,BRANCH IF YES (GO SAVE LOADERS)
1425  001242  011363  000002            MOV     (R3),2(R3)      ;OTHERWISE MAKE THE CONTENTS OF LOCATION MAXMEM
1426                                                            ;EQUAL TO THE MAXIMUM AVAILABLE MEMORY
1427                                                            ;AND FALL INTO SAVE LOADERS
1428
1429  001246  004767  006136  SAVLDR   JSR     PC,CLRMM        ,  DISABLE THE MEMORY MANAGEMENT UNIT
1430  001252  005723                    TST     (R3)+           ,MAKE R3 TO POINT TO THE LOCATION MAXMEM
1431  001254  011305                    MOV     (R3),R5         ,R5 CONTAINS THE ADDRESS OF MAXIMUM AVAILABLE MEM
1432
1433                          ,IF ONLY 4K BEING TESTED DON'T SAVE LOADERS
```

```
1434
1435  001256  020527  017776            CMP     R5,#17776       ;ONLY TESTING 4K MAX?
1436  001262  103416                    BLO     4$              ;BRANCH IF YES (DON'T SAVE LOADERS)
1437
1438  001264  162705  000276      3$    SUB     #276,R5         ;PREPARE TO SAVE 300 BYTES OF THE LOADERS
1439  001270  005737  000042            TST     @#42            ,IS THE PROGRAM RUNNING UNDER ACT OR XXDP ?
1440  001274  001406                    BEQ     2$              ;IF NOT THEN GO TO 2$
1441  001276  023737  000042  000046    CMP     @#42,@#46       ;ARE WE RUNNING UNDER XXDP CHAIN MODE?
1442  001304  001402                    BEQ     2$              ,BRANCH IF NO
1443  001306  162705  005674            SUB     #(1502 *2),R5   ;SAVE 1500. WORDS FOR XXDP CHAIN MODE
1444  001312  012524            2$      MOV     (R5)+,(R4)+     ;SAVE LOADER
1445  001314  020513                    CMP     R5,(R3)
1446  001316  101775                    BLOS    2$
1447  001320  012323            4$      MOV     (R3)+,(R3)+     ;SAVE THE CONTENTS OF LOCATION MAXMEM IN SAVMAX
1448  001322  010423                    MOV     R4,(R3)+        ,AND THE CONTENTS OF R4 AT SAVR4
1449
1450  001324  010537  000346    TSTREL: MOV     R5,@#SAVR5      ,SAVE HIGHEST VIRTUAL ADDRESS+2
1451  001330  004767  006054    TSTSIZ. JSR     PC,CLRMM        ;GO TO DISABLE MEMORY MANAGEMENT UNIT
1452  001334  005745                    TST     -(R5)           ,SET R5 BACK TO HIGHEST VIRTUAL ADDRESS
1453  001336  012703  000324      1$    MOV     #LOWTWO,R3      ,PREPARE TO LOAD R4 AND R5 WITH THE MEMORY BOUNDRIES
1454  001342  005723                    TST     (R3)+           ,IF THE BITS 16,17 OF THE LOWEST LOCATION UNDER
1455                                                            ,TEST ARE NON ZERO
1456  001344  001003                    BNE     2$              ,THEN GO TO 2$
1457  001346  021327  157776            CMP     (R3),#157776    ,IF THE LOWEST LOCATION UNDER TES  IS HIGHER THAN
1458                                                            ,157776 THEN GO TO TEST MEMORY MANAGEMENT
1459  001352  103411                    BLO     4$
1460  001354  032777  010000  177066  2$ BIT     #10000,@SWR     ;IS MEMORY MANAGEMENT SELECTED?
1461  001362  001003                    BNE     3$              ,YES ALL IS WELL
1462  001364  004767  004552            JSR     PC,FATERR       ,*ERROR* REPORT  ERROR MESSAGE AND HALT AT FATHLT
1463  001370  000003                    3                       ,******ERROR NUMBER 3******
1464
1465  001372  000167  003526      3$    JMP     TSTMM           ,GO TO TEST MEMORY MANAGEMENT
1466  001376  020423            4$:     CMP     R4,(R3)+
1467  001400  103002                    BHIS    6$
1468  001402  016304  177776            MOV     -2(R3),R4       ,ADJUST R4 TO POINT TO THE LOWEST LOCATION UNDER TEST
1469  001406  005723            6$      TST     (R3)+           ,IF BITS 16-17 OF HIGHEST LOCATION TO BE TESTED
1470  001410  001003                    BNE     8$              ,ARE NON ZERO THEN GO TO 8$
1471  001412  021305                    CMP     (R3),R5         ,OTHERWISE SEE IF THE HIGHEST LOCATION TO BE
1472                                                            ,TESTED IS HIGHER THAN 157776
1473  001414  101001                    BHI     8$              ;IF SO THEN GO TO 8$
1474  001416  011305                    MOV     (R3),R5         ,MODIFY R5
1475  001420  105737  000405      8$    TSTB    @#REL           ,ARE WE RELOCATED.?
1476  001424  100014                    BPL     10$             ;BRANCH IF NO
1477  001426  013704  000322            MOV     @#RELBOT,R4     ,SET BOTTOM TEST ADDRESS WHEN RELOCATED
1478  001432  020527  017776            CMP     R5,#17776       ,ARE WE RELOCATED IN BANK 0?
1479  001436  103402                    BLO     9$              ,BRANCH IF YES
1480  001440  012705  017776            MOV     #17776,R5       ,ELSE SET HIGH MEMORY UNDER TEST=4K
1481
1482  001444  020405            9$      CMP     R4,R5           ,IS LOW LIMIT LOWER THAN HIGH LIMIT?
1483  001446  103403                    BLO     10$             ;BRANCH IF YES
1484  001450  004767  004466            JSR     PC,FATERR       ,*ERROR* REPORT  ERROR MESSAGE AND HALT AT FATHLT
1485  001454  000004                    4                       ,******ERROR NUMBER 4******
1486
1487  001456  012703  000342    10$     MOV     #SAVMAX,R3
1488  001462  011343                    MOV     (R3),-(R3)      ,RESTORE THE CONTENTS OF MAXMEM
1489  001464  062713  000002    MEMTST  ADD     #2,(R3)         ,MAKE THE CONTENTS OF MAXMEM = MAXIMUM AVAILABLE
```

```
1490                                                    ;MEMORY +2
1491  001470  005725                    TST     (R5)+      ;AND SET R5=MAX MEMORY+2
1492
1493                              ,CLEAR MEMORY UNDER TEST
1494
1495  001472  010500              CLRMEM  MOV     R5,R0      ,MOVE HIGH ADDRESS TO R0
1496  001474  005040              2$      CLR     -(R0)      ,BEGIN CLEARING THE MEMORY FROM THE TOP
1497  001476  020004                      CMP     R0,R4      ;UNTIL THE BOTTOM IS REACHED
1498  001500  101375                      BHI     2$
1499  001502  012702  000001              MOV     #1,R2      ,SET R2 TO ENABLE PARITY MODULE CODE
1500  001506  004767  005776              JSR     PC,PARITY  ,ENABLE PARITY IF WANTED AND AVAILABLE
1501  001512  012702  000316              MOV     #BAKPAT,R2
1502  001516  012212                      MOV     (R2)+,(R2) ,WRITE SWAPPED BAKPAT IN LOCATION SWAPAT
1503  001520  000312                      SWAB    (R2)
1504  001522  017702  176722              MOV     @SWR,R2    ;LOAD R2 WITH THE OPTIONS STORED AT $SWREG
1505  001526  042702  177760              BIC     #177760,R2 ,ONLY LEAVE THE LOWER 4 BITS OF $SWREG IN R2 TO GO TO
1506                                                    ,THE TEST # SPECIFIED  DEFAULT IS TEST#0
1507
1508
1509
1510                              ,ENTER HERE FROM TSTSCP ROUT NE AT END OF SUBTEST
1511
1512  001532  005037  000306      CONT    CLR     @#PASFLG   ,INIT SUBTEST PASS FLAG
1513  001536  110237  000404              MOVB    R2,@#$TESTN ,SET UP $TESTN WITH THE TEST NUMBER GOING
1514                                                    ,TO BE EXECUTED
1515  001542  010401              LOOP    MOV     R4,R1      ,LOAD R1 WITH THE LOWEST LOCATION UNDER TEST
1516  001544  010246                      MOV     R2,-(SP)   ,SAVE R2 ON THE STACK
1517  001546  012703  000376              MOV     #376,R3    ,POINT R3 TO SCRATCH STACK
1518  001552  004767  005446              JSR     PC,PUTADR  ,GO TO GENERATE 18 BIT ADDRESS OUT OF THE ADDRESS
1519                                                    ,STORED IN R1 AND STORE IT IN LOCATIONS (R3)
1520                                                    ,AND (R3-2)
1521  001556  005743                      TST     -(R3)      ,CAUSE R3 TO POINT TO THE HIGH ORDER BITS OF THE
1522                                                    ,18 BIT ADDRESS
1523  001560  004767  005544              JSP     PC,$GTSIZ  ; PLACE BITS 13-17 OF THE ADDRESS IN BITS
1524                                                    ,0-4 OF R2
1525  001564  010400                      MOV     R4,R0      ,PLACE THE ADDRESS OF THE LOWEST LOCATION UNDER
1526                                                    ,TEST IN R0
1527  001566  010401                      MOV     R4,R1      ,IN R1
1528  001570  010403                      MOV     R4,R3      ,AND IN R3
1529  001572  012602                      MOV     (SP)+,R2   ,RESTORE R2
1530  001574  006302                      ASL     R2
1531  001576  060702                      ADD     PC,R2
1532  001600  066207  000004              ADD     TBL-(R2),PC        ,GO TO THE TEST #
1533                                                    ,STORED IN BITS 0-3 OF SWITCH REGISTER
1534
1535
1536  001604  000102              TBL     TST0-TBL           ,RELATIVE ADDRESS OF TEST # 0
1537  001606  000334                      TST1-TBL           ,RELATIVE ADDRESS OF TEST # 1
1538  001610  000434                      TST2-TBL           ,RELATIVE ADDRESS OF TEST # 2
1539  001612  000544                      TST3-TBL           ;RELATIVE ADDRESS OF TEST # 3
1540  001614  001012                      TST4-TBL           ,RELATIVE ADDRESS OF TEST # 4
1541  001616  001122                      TST5-TBL           ,RELATIVE ADDRESS OF TEST # 5
1542  001620  001270                      TST6-TBL           ,RELATIVE ADDRESS OF TEST # 6
1543  001622  001424                      TST7-TBL           ,RELATIVE ADDRESS OF TEST # 7
1544  001624  001646                      TST10-TBL          ,RELATIVE ADDRESS OF TEST # 10
1545  001626  002174                      TST11-TBL          ,RELATIVE ADDRESS OF TEST # 11
```

```
 1546  001630  002246                    TST12-TBL           ,RELATIVE ADDRESS OF TEST # 12
 1547  001632  002520                    TST13-TBL           ,RELATIVE ADDRESS OF TEST # 13
 1548  001634  003146                    RELOC-TBL           ,RELATIVE ADDRESS OF ROUTINE 'RELOC'
 1549
 1550
 1551
 1552                                    ,R5 IS POINTING TO THE TOP OF THE MEMORY TO BE TESTED+2
 1553                                    ,R4 & R0 ARE POINTING TO THE LOWEST ADDRESS OF MEMORY TO BE TESTED
```

```
1554                                    ;*      SCOPE ROUTINE
1555                                    ;*      -------------
1556                                    ;*
1557                                    ;*
1558                                    ;*              PROGRAM COMES TO THIS ROUTINE AFTER COMPLETION OF EACH TEST AND
1559                                    ;*              IF CNTRL-C TYPED GOTO ERROR HISTORY TYPE ROUTINE
1560                                    ;*              IF SR= 2000 (BIT10) THEN HALT
1561                                    ;*              IF SR= 40000 (BIT14) THEN LOOP ON TEST DEFINED BY SR BITS<3 0>
1562                                    ;*              ELSE CONTINUE TO NEXT TEST.
1563                                    ;*
1564
1565
1566  001636  105737  000420            TSTSCP  TSTB    @#$ENV          ;ARE WE RUNNING UNDER APT?
1567  001642  001002                            BNE     CNTSCP          ;IF SO THEN GO TO CNTSCP
1568  001644  004767  006020                    JSR     PC,CHECKC       ;TEST FOR CONTROL-C AND IF TYPED GO
1569                                                                    ;PRINT ERROR HISTORY AND HALT AT FATHLT.
1570  001650  113702  000404            CNTSCP  MOVB    @#$TESTN,R2     ;PLACE THE TEST NUMBER IN THE LOWER BYTE OF R2
1571                                                                    ;SINCE THERE ARE LESS THAN 377 TESTS UPPER BYTE
1572                                                                    ;OF R2 WILL BE 0
1573  001654  005237  000410                    INC     @#$DEVCT        ;TELL APT WE ARE STILL RUNNING OKAY
1574  001660  032777  002000  176562            BIT     #2000,@SWR      ;IS THE PROGRAM GOING TO HALT AFTER EACH TEST?
1575  001666  001401                            BEQ     TSTGO           ;IF NOT THEN GO TO 2$
1576  001670  000000            SWHALT  HALT                            ;HALT AT END OF TEST SWITCH SET
1577
1578  001672  032777  040000  176550  TSTGO   BIT     #40000,@SWR     ;IS THE PROGRAM GOING TO LOOP ON TEST
1579  001700  001320                            BNE     LOOP            ;IF SO THEN GO TO THE STARTING OF THE SAME TEST
1580  001702  105202                            INCB    R2
1581  001704  000712                            BR      CONT            ;GO TO CONT AND CONTINUE EXECUTING THE NEXT TEST
```

```
1582                                    ;,**********************************************************************
1583                                    ,*TEST 0        TEST FOR PROPER BANK SELECTION
1584                                            ,*(1)   THIS TEST ASSUMES THAT THE MEMORY IS IN A STATE
1585                                            ,*      OF ALL 0'S AND R0 HAS THE ADDRESS OF THE LOWEST
1586                                            ,*      LOCATION UNDER TEST
1587                                            ,*(2)   IT CHECKS FOR PROPER BANK SELECTION BY WRITING
1588                                            ,*      1'S IN A LOCATION AND CHECKING FOR 0'S IN THE SAME
1589                                            ,*      LOCATIONS OF OTHER 4K BANKS OF THE MEMORY
1590                                            ,*       I.E. LOCATIONS LIKE 7766 AND 27766 ETC
1591                                            ,*(3)   THIS TEST ALSO CHECKS TO SEE THAT NONE OF THE NON EXIST-
1592                                            ,*      ING BANK RESPOND WHEN THEY ARE ADDRESSED
1593                                    ,,**********************************************************************
1594  001706  105737  000404          TST0    TSTB    @#$TESTN        ;CHECK FOR PROPER TEST SEQUENCE
1595  001712  001403                          BEQ      +10
1596  001714  004767  004222                  JSR     PC,SEQERR       ;*ERROR* REPORT ERROR MESSAGE AND HALT AT FATHLT
1597  001720  000005                          5                       ,******ERROR NUMBER 5******
1598
1599  001722  012703  177777                  MOV     #177777,R3
1600  001726  010401                  1$      MOV     R4,R1           ,R1 = ADDRESS OF LOWEST LOCATION OF MEMORY UNDER TEST
1601  001730  010310                          MOV     R3,(R0)         ,SET ALL THE BITS AT (R0)
1602  001732  020001                  2$      CMP     R0,R1           ,IS R0 POINTING TO THE SAME MEMORY LOCATION AS R1
1603  001734  001417                          BEQ     4$              ,IN WHICH CASE CHECK FOR ALL 1'S AT (R1)
1604  001736  005711                          TST     (R1)            ,OTHERWISE CHECK (R1) FOR ALL 0'S
1605  001740  001430                          BEQ     5$
1606  001742  020311                          CMP     R3,(R1)         ,IF R1 IS NOT EQUAL TO R0 AND (R1)
1607                                                                  ,DOES NOT CONTAIN ALL 0'S THEN
1608                                                                  ,CHECK TO SEE IF (R0) = (R1)
1609  001744  001004                          BNE     3$
1610  001746  012767  000006  000042          MOV     #6,12$          ,*ERPOR* SETUP ERROR NO  IN 12$
1611                                                                  ,******ERROR NUMBER #6******
1612  001754  000403                          BR      10$
1613  001756                          3$
1614  001756  012767  000007  000032          MOV     #7,12$          ,*ERROR* SETUP ERROR NO  IN 12$
1615                                                                  ,******ERROR NUMBER #7******
1616  001764  010046                  10$     MOV     R0,-(SP)        ,SAVE R0 ON STACK
1617  001766  105237  000301                  INCB    @#$ADERR        ,AN ADDRESSING ERROR IS SUSPECTED
1618  001772  000407                          BR      11$
1619  001774  020311                  4$      CMP     R3,(R1)         ,CHECK (R1) FOR ALL 1'S
1620  001776  001411                          BEQ     5$
1621  002000  012767  000010  000010          MOV     #10,12$         ,*ERROR* SETUP ERROR NO  IN 12$
1622                                                                  ,******ERROR NUMBER #10******
1623  002006  010046                          MOV     R0,-(SP)        ,SAVE R0 ON STACK
1624  002010  010300                          MOV     R3,R0
1625  002012  004767  003566          11$     JSR     PC,ERROR        ,GO TO THE ERROR SUBROUTINE
1626  002016  000000                  12$      WORD                   ,ERROR NUMBER TO BE REPORTED WILL BE PLACED HERE
1627  002020  012600                          MOV     (SP)+,R0        ,RESTORE R0
1628
1629  002022  013706  000350          5$      MOV     @#SAVR6,SP      ,RESTORE THE STACK POINTER
1630  002026  062701  020000                  ADD     #20000,R1       ,CAUSE R1 TO POINT TO THE SAME CHIP
1631                                                                  ,LOCATION IN THE NEXT 4K BANK OF MEMORY
1632                                                                  ,BY ADDING 1 TO THE 14TH BIT OF ADDRESS IN R1
1633  002032  020105                          CMP     R1,R5           ,COMPARE R1 WITH THE HIGHEST MEMORY
1634                                                                  ,LOCATION WHICH IS STORED IN R5
1635  002034  103736                          BLO     2$              ,IF R1 LESS THAN R5 THEN REPEAT THE TEST FROM 2$
1636
1637  002036  105737  000421                  TSTB    @#$ENVM         ,HAS APT INHIBITED SIZING?
```

```
1638  002042  100430                          BMI     8$              ;BRANCH IF YES (DON'T TEST NON-EXISTENT MEMORY)
1639  002044  032777  000100  176376          BIT     #100,@SWR       ;HAS USER INHIBITED SIZING?
1640  002052  001024                          BNE     8$              ;BRANCH IF YES (DON'T TEST NON-EXISTENT MEMORY)
1641
1642  002054  020137  000340                  CMP     R1,@#MAXMEM     ;IS R1 LOWER THAN THE MAXIMUM AVAILABLE
1643                                                                  ;MEMORY ?
1644  002060  103760                          BLO     5$              ;IF SO THEN GO TO 5$
1645  002062  012702  000006                  MOV     #6,R2           ;MAKE R2 POINT TO TRAP VECTOR+2 FOR NXM
1646  002066  012712  000340                  MOV     #340,(R2)       ;SET PSW TO 340
1647  002072  012742  177722                  MOV     #5$- -6,-(R2)   ;SET UP RETURN ADDRESS FROM TRAP TO 5$
1648  002076  060712                          ADD     PC,(R2)
1649  002100  020127  157776                  CMP     R1,#157776      ;SEE IF R1 HAS CROSSED 28K BOUNDRY OF VIRTUAL ADDRESS
1650  002104  101004                          BHI     6$              ;IN WHICH CASE GO TO 6$
1651  002106  011111                          MOV     (R1),(R1)       ;TRY TO WRITE TO NON-EXISTENT MEMORY (SHOULD TRAP)
1652  002110  004767  004026                  JSR     PC,FATERR       ;*ERROR* REPORT  ERROR MESSAGE AND HALT AT FATHLT
1653  002114  000011                          11                      ;******ERROR NUMBER 11******
1654
1655
1656  002116  012722  000006          6$      MOV     #6,(R2)+        ;RESTORE TRAP VECTOR
1657  002122  005012                          CLR     (R2)
1658  002124  005010                  8$      CLR     (R0)
1659
1660  002126  062700  020000                  ADD     #20000,R0       ;CAUSE R0 TO POINT TO THE SAME CHIP
1661                                                                  ;LOCATION IN THE NEXT 4K MEMORY BANK
1662                                                                  ;BY ADDING 1 TO THE 14TH BIT OF ADDRESS IN R0
1663  002132  020005                          CMP     R0,R5           ; COMPARE R0 WITH THE HIGHEST MEMORY
1664                                                                  ; LOCATION WHICH IS STORED IN R5
1665  002134  103674                          BLO     1$              ;IF R0 LESS THEN PEPEAT THE TEST
1666  002136  000637                  END7    BR      TSTSCP
1667
1668
```

```
1669                                    ;;*****************************************************************
1670                                    ;*TEST 1         CHECK DI/DO LINES
1671                                    ;*(1)    THIS TEST CHECKS THE DATI/DATO LINES BY SHIFTING
1672                                    ;*       A 1 IN THE WORD DIRECTION
1673                                    ;;*****************************************************************
1674  002140  122737  000001  000404  TST1    CMPB    #1,@#$TESTN     ;CHECK FOR PROPER TEST SEQUENCE
1675
1676
1677  002146  001403                          BEQ      +10
1678  002150  004767  003766                  JSR     PC,SEQERR       ;*ERROR* REPORT ERROR MESSAGE AND HALT AT FATHLT
1679  002154  000012                          12                      ;******ERROR NUMBER 12******
1680
1681  002156  012700  000001          1$      MOV     #1,R0
1682  002162  010002                          MOV     R0,R2           ;SET R2=1
1683  002164  010011          2$      MOV     R0,(R1)         ;MOV 1 AT LOCATION (R1)
1684  002166  020011          3$      CMP     R0,(R1)         ;COMPARE R1 WITH THE CONTENTS OF LOCATION (R1)
1685  002170  001403                  BEQ     4$
1686  002172  004767  003406                  JSR     PC,ERROR        ;*ERROR* REPORT ERROR MESSAGE
1687  002176  000013                          13                      ;******ERROR NUMBER 13******
1688
1689
1690  002200  005702          4$      TST     R2              ;ARE WE SHIFTING A 0 IN DATA DIRECTION?
1691  002202  001406                  BEQ     5$              ;IF SO THEN GO TO 5$
1692  002204  006300                  ASL     R0              ;SHIFT THE 1 BROUGHT IN AT 1$ IN
1693                                                          ;DATA DIRECTION
1694  002206  103366                  BCC     2$              ;IF THE 1 HAS NOT BEEN SHIFTED THRU
1695                                                          ;THE 16 DATA BITS THEN REPEAT FROM 2$
1696  002210  005002                  CLR     R2              ;INITIATE SHIFTING OF 0 IN DATA DIRECTION
1697  002212  012700  177776                  MOV     #177776,R0
1698  002216  000762                  BR      2$
1699
1700  002220  000261          5$      SEC                     ;SET C BIT
1701  002222  006100                  ROL     R0              ;SHIFT A 0 16 TIMES IN DATA DIRECTION
1702  002224  103757                  BCS     2$              ;IF THE 0 HAS NOT BEEN SHIFTED THRU
1703                                                          ;THE 16 DATA BITS THEN REPEAT FROM 2$
1704  002226  062701  020000                  ADD     #20000,R1       ;OTHERWISE GO TO THE NEXT BANK OF
1705                                                          ;4K MEMORY AND REPEAT THE TEST
1706  002232  020105                  CMP     R1,R5
1707  002234  103750                  BLO     1$
1708  002236  000737          END1    BR      ENDO
1709
```

```
                                        ;;*****************************************************************
                                        ;*TEST 2         TEST MEMORY FOR HOLDING DATA AND BYTE SELECTION
                                        ;               THIS TEST CHECKS THE MEMORY FOR THE CAPABILITY
                                        ;               OF HOLDING 1'S AND 0'S BY WRITING A BACKGROUND
                                        ;               OF BAKPAT AND READING IT
                                        ;               MEMORY IS WRITTEN USING A BYTE AT A TIME
                                        ;               STEPS 1 & 2 ARE REPEATED WITH A SWAPPED BACKGROUND PATTERN
                                        ;;*****************************************************************
002240  122737  000002  000404  TST2    CMPB    #2,@#$TESTN     ;CHECK FOR PROPER TEST SEQUENCE

002246  001403                          BEQ      +10
002250  004767  003666                  JSR     PC,SEQERR       ;*ERROR* REPORT ERROR MESSAGE AND HALT AT FATHLT
002254  000014                          14                      ;******ERROR NUMBER 14*******

002256  013700  000316          1$      MOV     @#BAKPAT,R0
002262  110021                          MOVB    R0,(R1)+
002264  113721  00031?                  MOVB    @#BAKPAT+1,(R1)+,WRITE THE MEMORY WITH THE WORD STORED IN BAKPAT
002270  020105                          CMP     R1,R5
002272  103771                          BLO     1$

002274  020041                  2$      CMP     R0,-(R1)        ;TEST THE MEMORY TO SEE IF IT CONTAINS
                                                                ;THE WOPD STORED IN BAKPAT
002276  001416                          BEQ     8$
002300  062701  000002                  ADD     #2,R1
002304  123741  000317                  CMPB    @#BAKPAT+1,-(R1);CHECK FOR BYTE SELECTION PROBLEM
002310  001402                          BEQ     4$
002312  120041                          CMPB    R0,-(R1)        ;AGAIN CHECK FOR BYTE SELECTION PROBLEM
002314  001002                          BNE     6$
002316  105237  000301          4$      INCB    @#$ADERR        ;PREPARE TO INFORM THAT IT IS ADDRESSING ERROR
002322  042701  000001          6$      BIC     #1,R1           ;MAKE THE ADDRESS IN R1 EVEN
002326  004767  003252                  JSR     PC,ERROR        ;*ERROR* REPORT ERROR MESSAGE
002332  000015                          15                      ;******ERROR NUMBER 15******

002334  020104                  8$      CMP     R1,R4           ;KEEP ON TESTING THE MEMORY UNTIL
002336  101356                          BHI     2$              ;R1 EQUALS THE LOWEST ADDRESS
002340  000337  000316                  SWAB    @#BAKPAT        ;CHANGE THE DATA PATTERN
002344  001744                          BEQ     1$              ;IF THE DATA PATTERN DOES NOT HAVE LOW
                                                                ; BYTE =0 THEN FALL THRU
002346  000733                  END2:   BR      END1

                                        ;THE TEST LEAVES BAKPAT LOCATION THE SAME AS IT WAS IN THE BEGINNING
```

```
1752                                    ,,**********************************************************************
1753                                    ;*TEST 3        DUAL ADDRESS TEST A
1754
1755                                            ,*(1)   THIS TEST CHECKS FOR DUAL ADDRESSING PROBLEMS BY WRITING A
1756                                            ,*      BACK GROUND OF BAKPAT.
1757                                            ,*(2)   STARTING FROM THE LOWEST LOCATION IN THE BANK THE TEST WRITES A
1758                                            ;*      LOCATION WITH SWAPPED BAKPAT
1759                                            ,*(3)   READS THE MEMORY FOR PROPER CONTENTS
1760                                            ,*(4)   SHIFTS A 1 ALONG THE ADDRESS DIRECTION AND REPEATS STEPS 1-3
1761                                            ,*(5)   REPEATS STEP 1-4 FOR EACH 4K BANK
1762                                    ,,**********************************************************************
1763  002350  122737  000003  000404  TST3    CMPB    #3,@#$TESTN     ;CHECK FOR PROPER TEST SEQUENCE
1764  002356  001403                          BEQ      +10
1765  002360  004767  003556                  JSR     PC,SEQERR       ,*ERROR* REPORT ERROR MESSAGE AND HALT AT FATHLT
1766  002364  000016                          16                      ,******ERROR NUMBER 16******
1767
1768  002366  005003                          CLR     R3
1769  002370  004737  000120          2$      JSR     PC,@#WRTMEM     , WRITE MEMORY WITH THE BACKGROUND STORED
1770                                                                  ,AT LOCATION BAKPAT
1771  002374  005002                  4$      CLR     R2
1772  002376  050302                  6$      BIS     R3,R2           ,MAKE R2 POINT TO THE MEMORY BANK POINTED BY R3
1773  002400  020204                          CMP     R2,R4           ,IF R2 IS LESS THAN R4
1774  002402  103465                          BLO     16$             ,THEN DO NOTHING
1775  002404  020205                          CMP     R2,R5           ,IF R2 IS HIGHER THAN THE HIGHEST LOCATION TO BE
1776  002406  103077                          BHIS    20$             ,TESTED THEN EXIT THE TEST
1777  002410  000312                          SWAB    (R2)            ,OTHERWISE WRITE THE COMPLEMENT OF BAKPAT IN
1778                                                                  ,THE LOCATION POINTED BY R2
1779  002412  005001                          CLR     R1
1780  002414  050301                  7$      BIS     R3,R1
1781  002416  020104                          CMP     R1,R4           ,IF R1 IS POINTING TO A LOCATION LOWER THAN R4
1782  002420  103445                          BLO     12$             ,THEN GO TO 12$
1783  002422  020105                          CMP     R1,R5
1784  002424  103053                          BHIS    15$
1785  002426  020102                          CMP     R1,R2           ,CHECK THE MEMORY FOR CORRECT DATA
1786  002430  001431                          BEQ     10$
1787  002432  020011                          CMP     R0,(R1)         ,IF R1 IS NOT = TO R2 THEN (R1) SHOULD HAVE
1788                                                                  ,THE SAME WORD AS BAKPAT
1789  002434  001437                          BEQ     12$             ;IN WHICH CASE GO BACK TO 12$
1790  002436  012767  000017  000032          MOV     #17,22$         ,*ERROR* SETUP ERROR NO.  IN 22$
1791                                                                  ;******ERROR NUMBER #17******
1792  002444  010046                  8$      MOV     R0,-(SP)        ,PLACE R0 ON THE STACK
1793  002446  000316                          SWAB    (SP)
1794  002450  022611                          CMP     (SP)+,(R1)      ,IF (R1) IS NOT = R0 THEN SEE IF IT IS SAME
1795                                                                  ,AS A SWAPPED R0
1796  002452  001003                          BNE     9$              ,IF NOT THEN A SUSPECTED DUAL ADDRESSING PROBLEM
1797                                                                  ;FOR THE BITS THAT ARE DIFFERENT IN R0 AND (R1)
1798                                                                  ,OTHERWISE THERE IS DUAL ADDRESSING FOR THE
1799                                                                  ;ENTIRE WORD
1800  002454  012767  000020  000014          MOV     #20,22$         ,*ERROR* SETUP ERROR NO  IN 22$
1801                                                                  ,******ERROR NUMBER #20******
1802  002462  105237  000301          9$      INCB    @#$ADERR        ,ADDRESSING PROBLEM IS DETECTED
1803  002466  010046                          MOV     R0,-(SP)        ,SAVE R0
1804  002470  010200                          MOV     R2,R0           ,SET R0=GOOD ADDRESS FOR ERROR REPORT
1805  002472  004767  003106                  JSR     PC,ERROR        ,GO TO THE ERROR SUBROUTINE
1806  002476  000000                  22$      WORD                    ;ERROR NUMBER TO BE REPORTED WILL BE PLACED HERE
1807  002500  012600                          MOV     (SP)+,R0        ,RESTORE R0
```

```
1808  002502  010011                          MOV     R0,(R1)         ;RESTORE (R1)
1809  002504  020037  000316                  CMP     R0,@#BAKPAT     ;IF THE CONTROL CAME HERE FROM 15$-2 THEN
1810  002510  001411                          BEQ     12$
1811  002512  000407                          BR      11$             ;RETURN TO 11$
1812  002514  000300                  10$     SWAB    R0              ;MAKE R0 SAME AS SWAPPED BAKPAT
1813  002516  020011                          CMP     R0,(R1)         ;IF R1 = R2 THEN (R1) SHOULD CONTAIN A WORD
1814                                                                  ;EQUAL TO SWAPPED R0
1815  002520  001404                          BEQ     11$             ;IN WHICH CASE GO BACK TO 11$
1816  002522  012767  000021  177746          MOV     #21,22$         ;*ERROR* SETUP ERROR NO  IN 22$
1817                                                                  ;******ERROR NUMBER #21******
1818  002530  000745                          BR      8$              ;AND GO TO 8$
1819  002532  000300                  11$     SWAB    R0              ;RESTORE R0 TO BAKPAT
1820  002534  040301                  12$     BIC     R3,R1           ;TAKE OUT THE BANK ADDRESS FROM THE ADDRESS IN R1
1821  002536  005701                          TST     R1              ;IF R1 IS 0 THEN PLACE A 1 IN R1
1822  002540  001001                          BNE     13$             ;OTHERWISE GO TO 13$
1823  002542  005201                          INC     R1
1824  002544  006101                  13$     ROL     R1
1825  002546  020127  020000                  CMP     R1,#20000       ;IF R1 IS LESS THAN A 4K BOUNDRY
1826  002552  103720                          BLO     7$              ;THEN REPEAT FROM 7$
1827  002554  000312                  15$     SWAB    (R2)            ;RESTORE (R2) TO BAKPAT
1828  002556  040302                  16$     BIC     R3,R2           ;TAKE OUT THE BANK ADDRESS FROM THE ADDRESS
1829                                                                  ;STORED IN R2
1830  002560  005702                          TST     R2              ;IF R2 = 0 THEN MOVE A 1 TO R2
1831  002562  001001                          BNE     18$             ;OTHERWISE GO TO 18$
1832  002564  005202                          INC     R2
1833  002566  006102                  18$     ROL     R2              ;SHIFT A ONE IN THE ADDRESS WORD
1834  002570  020227  020000                  CMP     R2,#20000       ;IS THE ADDRESS IN R2 MORE THAN THE BOUNDRY
1835                                                                  ;OF 4K
1836  002574  103700                          BLO     6$              ;IF NOT THEN GO TO 6$
1837  002576  060203                          ADD     R2,R3           ;OTHERWISE MAKE R3 POINT TO THE NEXT 4K BANK
1838  002600  020337  000340                  CMP     R3,@#MAXMEM     ;IF R3 IS POINTING TO A BANK THAT IS LOWER
1839                                                                  ;THAN MAXMEM
1840  002604  103673                          BLO     4$              ;THEN REPEAT FROM 4$
1841  002606  000337  000316          20$.    SWAB    @#BAKPAT
1842  002612  001656                          BEQ     TST3            ;REPEAT THE TEST WITH SWAPPED BAKPAT ONLY IF
1843                                                                  ;THE LOWER BYTE OF BAKPAT IS 0
1844  002614  000654                  END3    BR      END2
```

```
1845                                    ;;*****************************************************************
1846                                    ;*TEST 4         DUAL ADDRESS TEST B
1847                                    ;*(1)    THIS TEST CHECKS FOR DUAL ADDRESSING BY WRITING
1848                                    ;*       AND READING THE ADDRESS IN THE LOCATION AND THEN
1849                                    ;*       WRITING AND READING ADDRESS COMPLEMENT
1850                                    ;;*****************************************************************
1851  002616  122737  000004  000404    TST4    CMPB    #4,@#$TESTN     ;CHECK FOR PROPER TEST SEQUENCE
1852  002624  001403                            BEQ     .+10
1853  002626  004767  003310                    JSR     PC,SEQERR       ;*ERROR* REPORT ERROR MESSAGE AND HALT AT FATHLT
1854  002632  000022                            22                      ;******ERROR NUMBER 22******
1855
1856  002634  005003                            CLR     R3
1857  002636  010100                    1$      MOV     R1,R0
1858  002640  005703                            TST     R3              ;IF R3 IS NOT 0 THEN STORE THE ADDRESS
1859  002642  001401                            BEQ     2$              ;IN THE LOCATION
1860  002644  005100                            COM     R0              ;OTHERWISE STORE COMPLEMENT
1861  002646  010021                    2$      MOV     R0,(R1)+        ;OF THE ADDRESS
1862  002650  020105                            CMP     R1,R5           ;UNTIL THE HIGHEST MEMORY LOCATION IS REACHED
1863  002652  103771                            BLO     1$
1864
1865  002654  020041                    3$      CMP     R0,-(R1)        ,CHECK THE LOCATION FOR THE CORRECT CONTENTS
1866  002656  001405                            BEQ     4$
1867  002660  105237  000301                    INCB    @#$ADERR        ;THIS IS PROBABLY ADDRESS PROBLEM RATHER THAN
1868                                            ;BIT PROBLEM
1869  002664  004767  002714                    JSR     PC,ERROR        ,*ERROR* REPORT ERROR MESSAGE
1870  002670  000023                            23                      ;******ERROR NUMBER 23******
1871
1872  002672  010100                    4$      MOV     R1,R0
1873  002674  162700  000002                    SUB     #2,R0           ,CHECK THAT THE ADDRESS IS STORED AT
1874  002700  005703                            TST     R3              ,LOCATION IF R3 IS NOT 0
1875  002702  001401                            BEQ     5$              ,OTHERWISE CHECK FOR
1876  002704  005100                            COM     R0              ,ADDRESS COMPLEMENT
1877  002706  020104                    5$      CMP     R1,R4
1878  002710  101361                            BHI     3$
1879  002712  112737  000001  000306            MOVB    #1,@#PASFLG     ,SET PASFLG FOR ERROR REPORT
1880  002720  005103                            COM     R3              ,COMPLEMENT THE CONTENTS OF R3
1881  002722  001345                            BNE     1$              ,REPEAT TST3 IF R3, IS NON 0, ENABLING ADDRESS
1882                                                                    ,COMPLEMENT TO BE WRITTEN AND READ, OTHERWISE FALL THRU
1883  002724  000733                    END4    BR      END3
1884
```

```
                                        ;,*****************************************************************
                                        ;*TEST 5         MARCHING 1'S AND 0'S
                                                ,*(1)   THIS TEST WRITES A BACK GROUND OF THE WORD STORED
                                                ,*      AT BAKPAT.
                                                ,*(2)   READS EVERY LOCATION FOR CORRECT DATA, SWAPS BYTES
                                                ,*      AT THE LOCATION AND PROCEEDS IN MAX. TO MIN
                                                ,*      DIRECTION OF MEMORY LOCATIONS.
                                                ,*(3)   READS EVERY LOCATION FOR SWAPPED BAKPAT PATTERN
                                                ,*      WRITES BAKPAT BACKGROUND IN THE LOCATION AND PROCEEDS
                                                ,*      IN  MIN. TO MAX. DIRECTION
                                                ,*(4)   REPEATS STEP 2 GOING IN MIN. TO MAX. DIRECTION
                                                ;*(5)   REPEATS STEP 3 GOING IN MAX. TO MIN. DIRECTION
                                        ,,*****************************************************************
002726  122737  000005  000404  TST5    CMPB    #5,@#$TESTN     ,CHECK FOR PROPER TEST SEQUENCE
002734  001403                  TST5    BEQ      +10
002736  004767  003200                  JSR     PC,SEQERR       ;*ERROR* REPORT ERROR MESSAGE AND HALT AT FATHLT
002742  000024                          24                      ;******ERROR NUMBER 24******
002744  004737  000120          1$      JSR     PC,@#WRTMEM     ;GO TO WRITE THE MEMORY WITH A BACKGROUND OF THE
                                                                ;WORD STORED IN BAKPAT
002750  020041          2$      CMP     R0,-(R1)        ;READ THE CONTENTS OF LOCATION POINTED BY R1
002752  001403                  BEQ     3$              ,TO SEE IF IT HAS THE SAME VALUE AS R0
002754  004767  002624                  JSR     PC,ERROR        ,*ERROR* REPORT ERROR MESSAGE
002760  000025                          25                      ,******ERROR NUMBER 25******
002762  000300          3$      SWAB    R0
002764  010011                  MOV     R0,(R1)         ,SWAP THE BYTES AT (R1)
002766  021100                  CMP     (R1),R0         ,READ (R1) FOR CORRECT VALUE
002770  001403                  BEQ     4$
002772  004767  002606                  JSR     PC,ERROR        ,*ERROR* REPORT ERROR MESSAGE
002776  000026                          26                      ,******ERROR NUMBER 26******


003000  000300          4$      SWAB    R0              ,SWAP THE BYTES OF THE REGISTER
                                                        ;CONTAINING BACKGROUND PATTERN
003002  001023                  BNE     9$              ,IF THE LOWER BYTE OF THE REGISTER
                                                        ,IS NOT 0 THEN THE PROGRAM IS READING
                                                        ,THE MEMORY TO CONTAIN A BACK GROUND OF

                                                        ,BAKPAT AND WRITING THE SWAPPED WORD

                                                        ,IN WHICH CASE GO TO 9$

003004  005703          5$      TST     R3              ,R3 WAS 0 WHEN THE PROGRAM ENTERED
                                                        ,THIS TEST, AND IT IS NOT ALTERED UNTIL PASFLG=3
                                                        ,IF R3 EQUAL 0 THEN THE PROGRAM IS
                                                        ;READING/WRITING MIN. TO MAX  OTHERWISE
                                                        ;IT IS GOING IN MAX  TO MIN  DIRECTION
003006  001023                  BNE     10$             ;IF R3 IS NOT CLEAR THEN GO TO 10$
003010  062701  000002  6$      ADD     #2,R1           ,OTHERWISE ADD 2 TO THE CONTENTS OF R1
003014  020105                  CMP     R1,R5           ,COMPARE R1 WITH THE MAX. MEMORY LOCATION TO
                                                        ,BE TESTED
003016  103006                  BHIS    8$              ,IF R1>R5 THEN GO TO 8$ OTHERWISE
003020  020011          7$      CMP     R0,(R1)         ,READ (R1) FOR THE CORRECT DATA
```

```
1941  003022  001757                    BEQ     3$              ,WRITE COMPLEMENT OF THE DATA FOUND AT (R1)
1942                                                            ;AND REPEAT UNTIL R1 > R5
1943  003024  004767  002554            JSR     PC,ERROR        ,*ERROR* REPORT ERROR MESSAGE
1944  003030  000027                    27                      ;******ERROR NUMBER 27******
1945
1946  003032  000753                    BR      3$
1947  003034  105237  000306    8$      INCB    @#PASFLG
1948  003040  000300                    SWAB    R0
1949  003042  001742                    BEQ     2$              ;IF THE LOWER BYTE OF R0 IS ALL 0'S
1950                                                            ,THEN BEGIN READING BAKPAT SWAPPED WRITING BAKPAT
1951                                                            ;AND READING BAKPAT GOING FROM MAX.  TO MIN.  PASFLG=4
1952  003044  005103                    COM     R3              ,OTHERWISE CLEAR R0
1953  003046  010401                    MOV     R4,R1           ;PUT THE LOWEST TESTING ADDRESS IN R1
1954  003050  000763                    BR      7$              ,AND BEGIN READING 0'S, WRITING 1'S AND
1955                                                            ,READING 1'S IN MIN.  TO MAX  DIRECTION  PASFLG=3
1956
1957  003052  005703            9$      TST     R3              ,IF R3 IS NON 0, I.E  PASFLG=3
1958  003054  001353                    BNE     5$              ,THEN READ BAKPAT, WRITE
1959                                                            ,SWAPPED BAKPAT AND READ SWAPPED BAKPAT
1960                                                            ,IN MIN  TO MAX.  DIRECTION
1961  003056  020104            10$     CMP     R1,R4           ,OTHERWISE TEST IS PROCEEDING IN MAX  TO
1962                                                            ,MIN  DIRECTION
1963  003060  101333                    BHI     2$              ,KEEP ON LOOPING UNTIL R1=R4
1964  003062  105237  000306            INCB    @#PASFLG
1965  003066  000300                    SWAB    R0
1966  003070  001753                    BEQ     7$              ,IF R0 SWAPPED HAS LOWER BYTE=0
1967                                                            ,THEN READ BAKPAT SWAPPED, WRITE BAKPAT,
1968                                                            ,AND READ BAKPAT GOING FROM MIN  TO MAX
1969  003072  000714            END5    BR      END4
1970
```

```
1971                                    ;;*****************************************************************
1972                                    ;*TEST 6         CELLS' VOLATILITY TEST
1973
1974                                            ;*(1)   THIS TEST WRITES THE MEMORY WITH A BACK GROUND OF BAKPAT
1975                                            ;*(2)   WITH PASFLG=0 THE TEST READS THE MEMORY FOR BAKPAT
1976                                            ;*      AND THEN INCREMENTS PASFLG
1977                                            ;*(3)   IT THEN READS/SWAPS BYTES/WRITES A LOCATION X FOR
1978                                            ;*      OVER 2 MSEC AND THEN READS THE MEMORY FOR BAKPAT
1979                                            ;*(4)   REPEATS STEP 3 WITH X=X+4K UNTIL END OF MEMORY IS ENCOUNTERED
1980                                            ;*(5)   IT THEN INCREMENTS PASFLG AND WRITES THE MEMORY TO
1981                                            ;*      BAKPAT AND WITH PASFLG=2 IT READS MEMORY FOR ALL
1982                                            ;*      SWAPPED BAKPAT AFTER WHICH PASFLG IS INCREMENTED TO 3
1983                                            ;*(6)   REPEATS STEPS 3 AND 4 READING THE MEMORY FOR SWAPPED
1984                                            ;*      BAKPAT INSTEAD OF BAKPAT.
1985
1986                                    ;;*****************************************************************
1987  003074  122737  000006  000404    TST6    CMPB    #6,@#$TESTN     ;CHECK FOR PROPER TEST SEQUENCE
1988
1989
1990  003102  001403                            BEQ      +10
1991  003104  004767  003032                   JSR     PC,SEQERR       ;*ERROR* REPORT ERROR MESSAGE AND HALT AT FATHLT
1992  003110  000030                            30                      ;******ERROR NUMBER 30******
1993
1994  003112  004737  000120            RPT6    JSR     PC,@#WRTMEM     ;GO TO WRITE THE MEMORY WITH A BACKGROUND OF THE
1995                                                                    ;WORD STORED AT LOCATION BAKPAT
1996  003116  005037  000306                    CLR     @#PASFLG
1997  003122  010403                    1$      MOV     R4,R3           ;SET R3
1998  003124  010401                    2$      MOV     R4,R1           ;AND R1 TO THE STARTING ADDRESS OF MEMORY UNDER TEST
1999  003126  020011                    3$      CMP     R0,(R1)         ;CHECK (R1) FOR CORRECT DATA
2000  003130  001403                            BEQ     4$
2001  003132  004767  002446                   JSR     PC,ERROR        ;*ERROR* REPORT ERROR MESSAGE
2002  003136  000031                            31                      ;******ERROR NUMBER 31******
2003
2004  003140  062701  000002            4$      ADD     #2,R1           ;INCREMENT R1 BY 2
2005  003144  020105                            CMP     R1,R5           ;SEE IF R1 HAS REACHED THE MAX  OF MEMORY
2006  003146  103767                            BLO     3$
2007  003150  132737  000001  000306            BITB    #1,@#PASFLG     ;CHECK TO SEE IF PASFLG=0 OR 2
2008  003156  001002                            BNE     5$
2009  003160  105237  000306                    INCB    @#PASFLG        ;IN WHICH CASE INCREMENT PASFLG COUNTER BY 1
2010
2011  003164  020305                    5$      CMP     R3,R5           ;SEE IF R3 HAS REACHED THE MAX  OF THE MEMORY
2012  003166  103012                            BHIS    7$
2013  003170  012702  037776                    MOV     #37776,R2       ;WRITE INTO 1 LOC FOR >2MS (ABOUT 100MS)
2014  003174  000313                    6$      SWAB    (R3)
2015  003176  005302                            DEC     R2
2016  003200  001375                            BNE     6$
2017  003202  010337  000354                    MOV     R3,@#SAVLOC     ;SAVE LOCATION WRITTEN FOR 2MS FOR ERROR REPORT
2018  003206  062703  020000                    ADD     #20000,R3       ;BY ADDING 1 TO THE 14TH ADDRESS BIT CAUSE
2019                                                                    ;R3 TO POINT TO A LOCATION IN THE NEXT
2020                                                                    ;4K BANK OF MEMORY
2021  003212  000744                            BR      2$
2022  003214  105237  000306            7$      INCB    @#PASFLG        ;MAKE PASFLG=2
2023  003220  000337  000316                    SWAB    @#BAKPAT        ;IF BAKPAT IS NOT BEING SWAPPED FOR THE 2ND
2024  003224  001732                            BEQ     RPT6            ;THEN GO BACK TO THE LOCATION RPT6
2025  003226  000721                    END6    BR      END5
2026
```

2027

```
2028                                    ;,*****************************************************************
2029                                    ;*TEST 7         SHIFTING DIAGONAL
2030
2031                                    ,*(1)    THIS TEST WRITES THE MEMORY WITH A BACKGROUND OF BAKPAT
2032                                    ,*(2)    IT WRITES A DIAGONAL OF SWAPPED BAKPAT THROUGH EACH MEMORY BANK
2033                                    ;*(3)    READS THE MEMORY FOR CORRECT DATA
2034                                    ,*(4)    SHIFTS THE DIAGONAL AND REPEATS STEP 3 UNTIL THE
2035                                    ,*       DIAGONAL HAS BEEN SHIFTED 64 TIMES
2036                                    ,*(5)    WRITES A BACKGROUND OF SWAPPED BAKPAT, A DIAGONAL OF
2037                                    ,*       BAKPAT AND REPEATS FROM STEP 3
2038                                    ,,*****************************************************************
2039  003230  122737  000007  000404  TST7    CMPB    #7,@#$TESTN     ;CHECK FOR PROPER TEST SEQUENCE
2040
2041  003236  001403                    BEQ      +10
2042  003240  004767  002676            JSR     PC,SEQERR       ,*ERROR* REPORT ERROR MESSAGE AND HALT AT FATHLT
2043  003244  000032                    32                      ;******ERROR NUMBER 32******
2044
2045  003246  005037  000306    2$      CLR     @#PASFLG
2046  003252  010337  000304            MOV     R3,@#LOWBNK     ,LOWBNK CONTAINS ADDRESS OF THE LOWEST LOCATION
2047                                                            ;IN THE 4K BANK THAT CAN BE TESTED
2048  003256  010302                    MOV     R3,R2
2049  003260  052702  017776            BIS     #17776,R2       ,R2 CONTAINS THE ADDRESS OF THE TOP OF THE BANK
2050  003264  005722                    TST     (R2)+           ,ADD 2 TO R2
2051  003266  020502                    CMP     R5,R2
2052  003270  103001                    BHIS    4$              ;IF R2 IS GREATER THAN R5 THEN GO TO 4$
2053  003272  010502                    MOV     R5,R2           ,NOW R2 CONTAINS THE ADDRESS OF THE HIGHEST LOCATION
2054                                                            ;THAT CAN BE TESTED
2055  003274  010337  000302    4$      MOV     R3,@#STRTDI     ,LOAD STRTDI WITH THE STARTING ADDRESS OF THE
2056                                                            ,DIAGONAL
2057  003300  013701  000304            MOV     @#LOWBNK,R1     ,R1 IS NOW POINTING TO THE LOWEST LOCATION IN THE 4K
2058                                                            ,BANK
2059  003304  013700  000316    6$      MOV     @#BAKPAT,R0     ;STORE THE CONTENTS OF BAKPAT IN R0
2060  003310  020103                    CMP     R1,R3           ;IS R1 POINTING TO A LOCATION IN THE DIAGONAL ?
2061  003312  001010                    BNE     10$             ,IF NOT THEN GO TO 10$
2062  003314  062703  000002            ADD     #2,R3           ,THE FOLLOWING CODE IS USED TO PLACE THE
2063  003320  032703  000176            BIT     #176,R3         ,ADDRESS OF THE NEXT LOCATION IN THE DIAGONAL
2064  003324  001402                    BEQ     8$              ,IN R3
2065  003326  062703  000200            ADD     #200,R3         ,
2066  003332  000300                8$  SWAB    R0              ;DIAGONAL WILL CONTAIN SWAPPED BACKGROUND PATTERN
2067  003334  132737  000001  000306  10$     BITB    #1,@#PASFLG     ;CONTENTS OF LOCATION PASFLG WILL BE EVEN IF THE
2068                                                            ;MEMORY IS BEING WRITTEN AND IT WILL BE ODD
2069                                                            ,IF IT IS ONLY BEING READ
2070  003342  001001                    BNE     12$             ,IF IT IS BEING READ ONLY THEN GO TO 12$
2071  003344  010011                    MOV     R0,(R1)         ,OTHERWISE WRITE THE MEMORY WITH THE CONTENTS
2072                                                            ,OF R0
2073  003346  020011            12$     CMP     R0,(R1)         ,CHECK THE LOCATION POINTED BY R1 TO CONTAIN
2074                                                            ,PROPER DATA
2075  003350  001403                    BEQ     14$             ,IF IT IS OK THEN GO TO 14$
2076  003352  004767  002226            JSR     PC,ERROR        ;*ERROR* REPORT ERROR MESSAGE
2077  003356  000033                    33                      ,******ERROR NUMBER 33******
2078
2079  003360  062701  000002    14$     ADD     #2,R1           ,CAUSE R1 TO POINT TO THE NEXT MEMORY LOCATION
2080  003364  020102                    CMP     R1,R2           ;IS IT THE END OF THE BANK ?
2081  003366  103746                    BLO     6$              ,IF NOT THEN GO TO 6$
2082  003370  005237  000410    16$     INC     @#$DEVCT                ,TELL APT WE ARE STIL RUNNING OKAY
2083  003374  105237  000306            INCB    @#PASFLG
```

```
 2084  003400  013703  000302          MOV     @#STRTDI,R3     ;LOAD R3 WITH THE STARTING ADDRESS OF THE DIAGONAL
 2085  003404  132737  000001  000306  BITB    #1,@#PASFLG     ;HAS THE READ OF THE MEMORY BEEN DONE ?
 2086  003412  001330                  BNE     4$              ;IF NOT THEN GO TO 4$
 2087  003414  005723                  TST     (R3)+           ;ADD 2 TO THE STARTING ADDRESS OF THE DIAGONAL
 2088  003416  020302                  CMP     R3,R2           ,AND UNLESS THE END OF THE BANK IS REACHED
 2089  003420  103003                  BHIS    18$             ,
 2090  003422  105737  000306          TSTB    @#PASFLG        ,OR THE DIAGONAL HAS BEEN ROTATED 64 TIMES
 2091  003426  100322                  BPL     4$              ,REPEAT FROM 4$
 2092  003430  013703  000304  18$     MOV     @#LOWBNK,R3     ;MAKE R3 POINT TO THE LOWEST LOCATION IN THE
 2093                                                          ;IN THE BANK UNDER TEST
 2094  003434  000337  000316          SWAB    @#BAKPAT
 2095  003440  001715                  BEQ     4$              ,AND IF THE TEST HAS NOT BEEN PERFORMED WITH THE
 2096                                                          ,SWAPPED BACK GROUND PATTERN THEN GO TO 4$
 2097  003442  010203                  MOV     R2,R3           ,MAKE THE PRESENT HIGH BOUNDRY AS THE NEXT
 2098                                                          ,LOW BOUNDRY
 2099  003444  020205                  CMP     R2,R5           ,UNLESS THE PRESENT HIGH BOUNDRY  S ALSO THE
 2100                                                          ,HIGH BOUNDRY FOR THE MEMORY UNDER TEST
 2101  003446  103677                  BLO     2$
 2102  003450  000666          END7    BR      END6
```

```
2103                                    ;;*****************************************************************
2104                                    ;*TEST 10        READ RECOVERY GALLOPING TEST THROUGH EVERY 64TH CELL
2105
2106                                            ;*(1)   THIS TEST WRITES THE MEMORY WITH A BACK GROUND PATTERN
2107                                            ;*      STORED AT LOCATION BAKPAT
2108                                            ;*(2)   TEST BEGINS AT LOWEST LOCATION BEING TESTED
2109                                            ;*      (LETS NAME IT 'A')
2110                                            ;*(3)   LETS NAME THE 1ST LOCATION IN THE ROW/COLUMN UNDER TEST AS 'B'
2111                                            ;*(4)   SWAPS BYTES FOR LOCATION 'A'.
2112                                            ;*(5)   READS 'A', READS 'B'
2113                                            ;*(6)   'B' = 'B'+200 (MAKES 'B'=64TH CELL I E  200TH OCTAL
2114                                            ;*      LOCATION FROM THE PRESENT LOCATION OF 'B')
2115                                            ;*(7)   REPEATS STEPS 5 AND 6 UNTIL 'B' IS GREATER THAN THE
2116                                            ;*      END OF THE 4K BANK OF THE MEMORY IN WHICH 'A' IS RESIDING
2117                                            ;*(8)   A = A+2
2118                                            ;*(9)   REPEATS  STEPS 3-8 UNTILL 'A' REACHES THE END OF THE BANK
2119                                            ;*(10)  GOES TO THE NEXT 4K BANK OF MEMORY AND REPEATS STEPS
2120                                            ;*      3-9 UNTIL THE END OF THE MEMORY
2121                                            ;*(11)  AFTER EXECUTING THE TEST BYTES ARE SWAPPED AT
2122                                            ;*      LOCATION BAKPAT AND STEPS 1-10 ARE REPEATED
2123                                            ;*(12)  IN THIS TEST R0 IS POINTING TO LOCATION 'A', R1 TO
2124                                            ;*      LOCATION 'B', R2 TO THE END OF THE 4K BANK IN WHICH THE
2125                                            ;*      TEST IS TAKING PLACE AND R3 TO THE LOWEST LOCATION IN THE
2126                                            ;*      COLUMN/ROW CONTAINING 'A' AND 'B'
2127                                            ;*(13)  MOST OF THE CODE USED BY THIS TEST IS ALSO USED BY TEST 11
2128
2129                                    ;;*****************************************************************
2130  003452  122737  000010  000404  TST10   CMPB    #10,@#$TESTN    ;CHECK FOR PROPER TEST SEQUENCE
2131
2132  003460  001403                          BEQ      +10
2133  003462  004767  002454                 JSR     PC,SEQERR       ;*ERROR* REPORT ERROR MESSAGE AND HALT AT FATHLT
2134  003466  000034                          34                      ;******ERROR NUMBER 34******
2135
2136  003470  010402                          MOV     R4,R2           ;SET R2 TO THE LOWEST MEMORY UNDER TEST
2137  003472  052702  017776          RPT10   BIS     #17776,R2       ;MAKE R2 POINT TO THE HIGHEST LOCATION IN THE 4K
2138                                                                  ;BANK FOR WHICH GALLOPING WILL BE PERFOMED
2139  003476  062702  000002          GALLOP  ADD     #2,R2           ;INCREMENT R2 BY 2
2140  003502  020205                          CMP     R2,R5           ;IF THE HIGH BOUNDRY OF THE TEST IS HIGHER THAN
2141  003504  101401                          BLOS    2$
2142  003506  010502                          MOV     R5,R2           ;THE MAXIMUM ALLOWED ADDRESS THEN ADJUST R2
2143  003510  005046                  2$      CLR     -(SP)
2144  003512  010200                          MOV     R2,R0
2145  003514  013740  000316          4$      MOV     @#BAKPAT,-(R0)  ;WRITE THE MEMORY UNDER TEST WITH A BACKGROUND OF
2146                                                                  ;BAKPAT
2147  003520  020003                          CMP     R0,R3
2148  003522  101374                          BHI     4$
2149  003524  010301                  6$      MOV     R3,R1           ;R3 AND R1 ARE POINTING TO THE LOWEST LOCATION THAT
2150                                                                  ;CAN BE TESTED IN THIS BLOCK
2151  003526  023710  000316                 CMP     @#BAKPAT,(R0)   ;BEFORE STARTING THE GALLOPING TEST FOR LOCATION
2152                                                                  ;(R0) CHECK IT
2153  003532  001410                          BEQ     8$              ;CONTINUE IF OK
2154  003534  010001                          MOV     R0,R1           ;OTHERWISE PREPARE TO REPORT THE ERROR
2155  003536  013700  000316                  MOV     @#BAKPAT,R0
2156  003542  004767  002036                 JSR     PC,ERROR        ;*ERROR* REPORT ERROR MESSAGE
2157  003546  000035                          35                      ;******ERROR NUMBER 35******
2158
```

```
2159  003550  010011                        MOV     R0,(R1)         ,RESTORE THE CONTENTS OF (R1)
2160  003552  010100                        MOV     R1,R0           ,RESTORE R0
2161
2162  003554  000310                  8$    SWAB    (R0)
2163  003556  031011                  10$   BIT     (R0),(R1)       ;CHECK TO SEE THAT NONE OF THE BITS SET
2164                                                                ,IN (R0) ARE SET IN (R1) AND VICE VERSA
2165  003560  020001                        CMP     R0,R1           ,THE ONLY EXCEPTION TO THIS WILL BE WHEN R0=R1
2166
2167  003562  001412                        BEQ     12$
2168  003564  021137  000316                CMP     (R1),@#BAKPAT   ,CHECK THAT (R1) HAS BAKPAT IN IT
2169  003570  001407                        BEQ     12$
2170  003572  010046                        MOV     R0,-(SP)        ,SAVE R0 ON STACK
2171  003574  013700  000316                MOV     @#BAKPAT,R0     ,PLACE THE PATTERN WORD IN R0
2172  003600  004767  002000                JSR     PC,ERROR        ;*ERROR* REPORT ERROR MESSAGE
2173  003604  000036                        36                      ;******ERROR NUMBER 36******
2174
2175  003606  012600                        MOV     (SP)+,R0        ,RESTORE R0
2176  003610  021037  000320          12$   CMP     (R0),@#SWAPAT   ,CHECK THAT (R0) HAS SWAPPED BAKPAT IN IT
2177  003614  001412                        BEQ     14$
2178  003616  010146                        MOV     R1,-(SP)        ,SAVE R1 ON THE STACK
2179  003620  010001                        MOV     R0,R1           ,MAKE R1 POINT TO THE FAILING LOCATION
2180  003622  013700  000320                MOV     @#SWAPAT,R0     ,LOAD R0 WITH THE EXPECTED RESULT IN (R1)
2181  003626  004767  001752                JSR     PC,ERROR        ,*ERROR* REPORT ERROR MESSAGE
2182  003632  000037                        37                      ,******ERROR NUMBER 37******
2183
2184  003634  010011                        MOV     R0,(R1)         ,RECOVER (R1) FROM THE ERROR
2185  003636  010100                        MOV     R1,R0           ,RESTORE R0
2186  003640  012601                        MOV     (SP)+,R1        ,AND RESTORE R1
2187  003642  122737  000011  000404  14$   CMPB    #11,@#$TESTN    ,IS THE PROGRAM EXECUTING TEST # 11 ?
2188  003650  001402                        BEQ     16$             ,IF SO THEN GO TO 16$
2189  003652  062701  000176                ADD     #176,R1
2190  003656  062701  000002          16$   ADD     #2,R1           ,MAKE R1 POINT TO THE NEXT ADJACENT CELL
2191  003662  020102                        CMP     R1,R2           ,AND IF R1 HAS NOT REACHED THE END OF THE BOUNDRY
2192  003664  103734                        BLO     10$             ,THEN REPEAT FROM 10$
2193  003666  000320                        SWAB    (R0)+           ,RESTORE THE LOCATION FOR WHICH THE GALLOPING TEST
2194                                                                ,WAS BEING PERFORMED
2195  003670  122737  000011  000404        CMPB    #11,@#$TESTN    ,IS IT TEST 11 ?
2196  003676  001407                        BEQ     17$             ;IF SO THEN GO TO 17$
2197  003700  005723                        TST     (R3)+           ,OTHERWISE INCREMENT R3 BY 2
2198  003702  062716  000002                ADD     #2,(SP)         ,FOR EVERY ROW/COLUMN TESTED ADD 2
2199  003706  105716                        TSTB    (SP)
2200  003710  100002                        BPL     17$             ,UNTIL (SP) IS 200
2201  003712  161603                        SUB     (SP),R3         ,SUBTRACT 200 FROM R3
2202  003714  005016                        CLR     (SP)
2203  003716  032700  000177          17$   BIT     #177,R0         ,AT A 64TH CALL BOUNDRY?
2204  003722  001002                        BNE     18$             ,BRANCH IF NO
2205  003724  005237  000410                INC     @#$DEVCT        ,TELL APT WE ARE STILL RUNNING
2206  003730  020002                  18$   CMP     R0,R2           ,IF R0 HAS NOT REACHED THE END OF THE BOUNDRY
2207  003732  103674                        BLO     6$              ,THEN REPEAT FROM 6$
2208  003734  162603                        SUB     (SP)+,R3        ,RESTORE SP AND R3
2209  003736  000337  000320                SWAB    @#SWAPAT
2210  003742  000337  000316                SWAB    @#BAKPAT
2211  003746  001660                        BEQ     2$              ,IF THE LOWER BYTE OF BAKPAT IS 0 THEN REPEAT FROM 2$
2212  003750  010203                        MOV     R2,R3           ,OTHERWISE MAKE THE PRESENT HIGH BOUNDRY AS THE
2213                                                                ,NEXT LOW BOUNDRY
2214  003752  020205                        CMP     R2,R5
```

```
DZKMA   MACY11 30(1046)  15-AUG-77  12 37  PAGE 47
DZKMAD P11    15-AUG-77 12.36           T10     READ RECOVERY GALLOPING TEST THROUGH EVERY 64TH CELL                    SEQ 0047

  2215  003754  001410                          BEQ     END10           ,IF PREVIOUS HIGH BOUNDRY WAS THE END OF THE
  2216                                                                  ,TEST BOUNDRY THEN EXIT THE TEST
  2217  003756  032702  017776                  BIT     #17776,R2       ,WAS IT A 4K BOUNDRY ?
  2218  003762  001025                          BNE     RPT11           ;IF NOT THEN WE WERE PERFORMING TEST 11 WITH LONG
  2219                                                                  ,GALLOPING TEST DISABLED
  2220  003764  122737  000011  000404          CMPB    #11,@#$TESTN    ;IF IT IS TEST # 11 THEN GO TO REPEAT TEST 11
  2221  003772  001421                          BEQ     RPT11
  2222  003774  000636                          BR      RPT10           ,OTHERWISE REPEAT TEST 10
  2223  003776  000624                  END10   BR      END7
  2224
  2225
  2226
```

```
2227                                    ;,*****************************************************************
2228                                    ;*TEST 11        READ RECOVERY LONG GALLOPING/FAST GALLOPING TEST
2229
2230                                            ;*(1)   THIS TEST WRITES MEMORY WITH BAKPAT
2231                                            ,*(2)   THE TEST BEGINS AT THE LOWEST LOCATION BEING TESTED
2232                                            ,*      (LETS NAME IT 'B')
2233                                            ,*(3)   'A'_'B'  MOVE THE ADDRESS OF 'B' TO THE POINTER FOR LOCATION 'A'
2234                                            ,*(4)   SWAPS BYTES FOR LOCATION 'A'
2235                                            ,*(5)   READS 'A', READS 'B'
2236                                            ;*(6)   'B'='B'+2
2237                                            ,*(7)    IF GALLOPING OPTION BIT AT $SWREG IS HIGH THEN STEPS 4 AND 5
2238                                            ,*      ARE REPEATED UNTIL 'B' REACHES THE HIGHEST MEMORY LOCATION
2239                                            ;*      OF THE 4K BANK IN WHICH 'A' IS RESIDING, THEN 'A' IS
2240                                            ,*      DECREMENTED BY 2 AND AFTER MAKING 'B' TO POINT TO THE LOWEST
2241                                            ,*      LOCATION OF THE 4K MEMORY BANK CONTAINING 'A' STEPS 3,4,5 AND
2242                                            ,*      6 ARE REPEATED UNTIL 'A' EQUALS THE END OF THE ENTIRE MEMORY
2243                                            ,*(8)   IF GALLOPING OPTION BIT IS NOT HIGH THEN STEPS 4 AND 5 ARE
2244                                            ,*      REPEATED UNTIL 'B' IS POINTING TO A CELL IN THE NEXT COLUMN
2245                                            ,*      IF SEQUENTIAL CELLS LIE ALONG THE ROW, OR THE NEXT ROW
2246                                            ,*      IF SEQUENTIAL CELLS LIE ALONG THE COLUMN, AT WHICH TIME
2247                                            ,*      STEPS 2,3,4,5 AND 7 ARE REPEATED UNTIL THE END OF THE MEMORY
2248                                            ;*(9)   TEST IS REPEATED FOR THE OPPOSITE BACKGROUND DATA
2249                                            ,*(10)  IN THIS TEST R0 POINTS TO LOCATION 'A', R1 TO LOCATION
2250                                            ;*      'B', R2 TO THE HIGHEST LOCATION AND R3 TO THE LOWEST
2251                                            ,*      LOCATION IN A 64/4K CELL BOUNDRY
2252                                            ,*(11)  MOST OF THE CODE USED BY TEST 10 IS ALSO USED BY THIS TEST
2253
2254                                    ,,*****************************************************************
2255  004000  122737  000011  000404  TST11.  CMPB    #11,@#$TESTN    ,CHECK FOR PROPER TEST SEQUENCE
2256
2257  004006  001403                        BEQ      +10
2258  004010  004767  002126                JSR      PC,SEQERR       ;*ERROR* REPORT ERROR MESSAGE AND HALT AT FATHLT
2259  004014  000040                        40                       ,******ERROR NUMBER 40******
2260
2261  004016  010402                        MOV      R4,R2           ,MAKE R2 TO POINT TO THE LOWEST LOCATION
2262                                                                  ,UNDER TEST
2263  004020  105777  174424                TSTB     @SWR            ;LONG GALLOP ENABLED?
2264  004024  100004                        BPL      RPT11           ,BRANCH IF NO
2265  004026  004767  002540                JSR      PC,PNTMES       ,TYPE "GLP"
2266  004032  046107  000120                 ASCIZ   /GLP/
2267  004036  105777  174406          RPT11 TSTB     @SWR            ,LONG GALLOPING ENABLED?
2268  004042  100613                        BMI      RPT10           ,BRANCH IF YES
2269                                                                  ,TO RPT10
2270  004044  052702  000176                BIS      #176,R2         ,OTHERWISE SET THE LOW ORDER BITS OF THE ADDPESS
2271                                                                  ,TO GET THE HIGH BOUNDRY
2272
2273  004050  000612                        BR       GALLOP          , PERFORM GALLOPING TEST
```

```
2274                                    ;;*****************************************************************
2275                                    .*TEST 12       WORST CASE TESTING FOR CORE MEMORY
2276                                            .*(1)   STARTING FROM THE LOWEST LOCATION UNDER TEST THE MEMORY
2277                                            .*      IS WRITTEN WITH A BACKGROUND OF BAKPAT, HOWEVER LOCATIONS
2278                                            .*      HAVING ADDRESS SUCH THAT EXCLUSIVE OR OF ADDRESS BITS 1 &
2279                                            .*      8 = 1 ARE WRITTEN TO A VALUE OF SWAPPED BAKPAT
2280                                            .*(2)   STARTING FROM THE LOWEST LOCATION THE MEMORY IS CHECKED
2281                                            .*      TO CONTAIN THE CORRECT DATA AS EXPLAINED IN STEPS 3 & 4,
2282                                            .*      UNTILL THE HIGHEST LOCATION UNDER TEST IS REACHED
2283                                            .*(3)   READ EACH LOCATION FOR THE CORRECT CONTENT
2284                                            .*(4)   COMPLEMENT THE LOCATION AND READ IT; COMPLEMENT THE LOCATION
2285                                            .*      BACK TO ITS ORIGINAL VALUE AND READ IT AGAIN
2286                                            .*(5)   STARTING FROM THE HIGHEST LOCATION UNDER TEST REPEAT STEPS
2287                                            .*      3 & 4 UNTIL THE LOWEST LOCATION UNDER TEST IS REACHED
2288                                            .*(6)   REPEAT STEPS 1-5, HOWEVER THIS TIME LOCATIONS WITH XOR
2289                                            .*      OF ADDRESS BITS 8 & 13 =1 ARE WRITTEN TO SWAPPED BAKPAT
2290                                            .*(7)   REPEAT STEPS 1-5, HOWEVER THIS TIME LOCATIONS WITH XOR
2291                                            .*      OF ADDRESS BITS 3 & 9 =1 ARE WRITTEN TO SWAPPED BAKPAT
2292                                            .*(8)   REPEAT STEPS 1-7 WITH A BACKGROUND OF SWAPPED BAKPAT AND
2293                                            .*      THE LOCATIONS TO BE WRITTEN TO SWAPPED BAKPAT WRITTEN TO
2294                                            .*      BAKPAT.
2295                                    ,,*****************************************************************
2296  004052  122737  000012  000404  TST12   CMPB    #12,@#$TESTN    ;CHECK FOR PROPER TEST SEQUENCE
2297  004060  001403                   BEQ      +10
2298  004062  004767  002054           JSR     PC,SEQERR       ;*ERROR* REPORT ERROR MESSAGE AND HALT AT FATHLT
2299  004066  000041                   41                      ;******ERROR NUMBER 41******
2300
2301
2302  004070  012702  000002           MOV     #2,R2           ;PREPARE TO TAKE THE EXCLUSIVE OR OF ADDRESS BITS 1
2303  004074  012703  000400           MOV     #400,R3         ;AND 8
2304  004100  112737  000001  000306  1$      MOVB    #1,@#PASFLG     ;INITIALIZE THE COUNTER FOR THE SUBTEST
2305  004106  010401                   2$      MOV     R4,R1           ;PLACE THE STARTING ADDRESS OF MEMORY UNDER
2306                                                                   ;TEST IN R1
2307  004110  013700  000316           4$      MOV     @#BAKPAT,R0
2308  004114  030201                   BIT     R2,R1           ;CHECK TO SEE IF ADDRESS BIT STORED IN R2 IS SET
2309  004116  001004                   BNE     8$              ;IF IT IS SET THEN GO TO 8$
2310  004120  030301                   BIT     R3,R1           ;CHECK TO SEE IF ADDRESS BIT POINTED BY R3 IS SET
2311  004122  001404                   BEQ     12$             ;IF IT IS NOT SET THEN GO TO 12$
2312  004124  005100                   6$      COM     R0              ;COME HERE ONLY IF EXCLUSIVE OR OF ADDRESS BITS
2313                                                                   ;POINTED BY R2 & POINTED BY R3 = 1 IN WHICH
2314                                                                   ;CASE PREPARE TO WRITE THE LOCATION
2315                                                                   ;WITH A COMPLEMENT OF LOCATIONS NOT MEETING
2316                                                                   ;THIS CONDITION
2317  004126  000402                   BR      12$
2318  004130  030301                   8$      BIT     R3,R1           ;COME HERE IF ADDRESS BIT POINTED BY R2 IS 1 AND
2319                                                                   ;CHECK ADDRESS BIT POINTED BY R3
2320  004132  001774                   BEQ     6$              ;IF ADDRESS BIT POINTED BY R3 IS 0 THEN GO TO 6$
2321  004134  132737  000002  000306  12$     BITB    #2,@#PASFLG     ;IS IT 2ND OR 3RD PASS OF THE SUBTEST ?
2322  004142  001001                   BNE     14$             ;IF SO THEN READ THE MEMORY
```

```
2323  004144  010011                        MOV     R0,(R1)         ;OTHERWISE WRITE THE MEMORY BFORE READING IT
2324  004146  020011              14$:      CMP     R0,(R1)         ;READ THE MEMORY FOR CORRECT CONTENTS
2325  004150  001403                        BEQ     16$
2326  004152  004767  001426                JSR     PC,ERROR        ;*ERROR* REPORT ERROR MESSAGE
2327  004156  000042                        42                      ;******ERROR NUMBER 42******
2328
2329  004160  012746  000002        16$:    MOV     #2,-(SP)
2330  004164  005100              18$:      COM     R0
2331  004166  005111                        COM     (R1)
2332  004170  020011                        CMP     R0,(R1)         ;READ THE MEMORY AGAIN
2333  004172  001404                        BEQ     19$
2334  004174  004767  001404                JSR     PC,ERROR        ;*ERROR* REPORT ERROR MESSAGE
2335  004200  000043                        43                      ;******ERROR NUMBER 43******
2336
2337  004202  010011                        MOV     R0,(R1)         ;RESTORE THE LOCATION (R1)
2338  004204  005316              19$:      DEC     (SP)
2339  004206  001366                        BNE     18$             ;EXECUTE THE CODE FROM 18$ TWICE
2340  004210  005726                        TST     (SP)+           ;RESTORE THE STACK POINTER
2341  004212  122737  000003  000306        CMPB    #3,@#PASFLG     ;IS IT THE 3RD PASS OF THE SUBTEST ?
2342  004220  001412                        BEQ     20$             ;IF SO THEN GO TO 20$
2343  004222  062701  000002                ADD     #2,R1           ;IN FIRST 2 PASSES THE PROGRAM PROCEEDS IN
2344                                                                ;MIN  TO MAX. DIRECTION
2345  004226  020105                        CMP     R1,R5           ;HAVE WE REACHED THE MAX  ADDRESS UNDER TEST ?
2346  004230  103727                        BLO     4$              ;IF NOT THEN REPEAT FROM 4$
2347  004232  105237  000306                INCB    @#PASFLG
2348  004236  122737  000002  000306        CMPB    #2,@#PASFLG     ;IF IT IS THE 2ND PASS OF THE SUBTEST
2349  004244  001720                        BEQ     2$              ;THEN REPEAT FROM 2$
2350  004246  162701  000002        20$:    SUB     #2,R1           ;OTHERWISE EXECUTE THE TEST IN MAX  TO MIN
2351                                                                ;DIRECTION
2352  004252  020104                        CMP     R1,R4           ;HAVE WE REACHED THE MIN  ADDRESS UNDER TEST ?
2353  004254  103315                        BHIS    4$              ;IF NOT THEN REPEAT FROM 4$
2354  004256  012702  020000                MOV     #20000,R2       ;PREPARE TO CHECK THE MEMORY WITH THE XOR OF
2355                                                                ;ADDRESS BITS 8 AND 13
2356  004262  105237  000307                INCB    @#PASFLG+1      ;THE SUB TEST HAS CHECKED THE XOR ONE KIND
2357  004266  123727  000307  000002        CMPB    @#PASFLG+1,#2   ;HAS TWO XOR COMBINATIONS BEEN CHECKED ?
2358  004274  103701                        BLO     1$              ;IF NOT THEN GO TO 1$
2359  004276  101004                        BHI     22$             ;IF ALL THREE HAVE BEEN CHECKED THEN GO TO 22$
2360  004300  012702  000010                MOV     #10,R2          ;IF IT IS THE 2ND XOR COMBINATION THEN CHECK
2361  004304  006303                        ASL     R3              ;FOR ADDRESS BITS 3 & 8
2362  004306  000674                        BR      1$
2363  004310  005137  000316        22$:    COM     @#BAKPAT        ;IF THE TEST WAS NOT PERFORMED WITH THE SWAPPED
2364  004314  105737  000316                TSTB    @#BAKPAT
2365  004320  001654                        BEQ     TST12           ;BAKPAT THEN RE-EXECUTE THE TEST
2366  004322  000625              END12:    BR      END10
```

```
2367                                    ;,*****************************************************************
2368                                    ,*TEST 13       WRITE RECOVERY TEST
2369                                    ,*      THIS TEST DIFFERS FROM 0-12 IN THAT IT CONSISTS OF A SMALL TEST PROGRAM
2370                                    ;*      ACTUALLY RUNNING IN THE 4K BANK UNDER TEST
2371                                    ,*              THE PROGRAM IS SELF MODIFYING AND MAY BE DIFFICULT TO DEBUG.
2372                                    ;*      TO AID IN THE DEBUG, BEFORE A A BANK IS ENTERED "TST13 BANK XX"
2373                                    ;*      IS TYPED. THIS WILL ALLOW THE USER TO AT LEAST SEE WHICH MEMOR
2374                                    ;*      BANK FAILED.
2375                                    ;*      THE TEST CONSISTS OF 1/2 OF THE BANK STORED WITH "MOV R2,-(PC)"
2376                                    ,*      AND THE OTHER 1/2 CONTAINING "177667". "177667" IS THE COMPLEMENT
2377                                    ,*      OF "JMP (R0)" INSTRUCTION.
2378                                    ,*      R2 CONTAINS "COM -(R1)" INSTRUCTION ON ENTRY TO THE BANK AND R1 CONTAINS
2379                                    ,*      THE HIGHEST TEST ADDRESS IN THAT BANK. THE HIGHEST TEST ADDRESS IS
2380                                    ,*      USUALLY ON 4K BOUNDARIES. WHEN TESTING BANK 0 RELOCATED,HOWEVER
2381                                    ,*      R1 CONTAINS THE FIRST FREE TEST ADDRESS BELOW THE DIAGNOSTIC.
2382                                    ,*      IF YOU UNDERSTAND THIS SO FAR THE REST IS EASY.
2383                                    ,*              THE TEST EXECUTION IS AS FOLLOWS:
2384                                    ,*              1. THE "MOV R2,-(PC)" INSTRUCTION EXECUTES STORING
2385                                    ,*                 THE CONTENTS OF R2 IN THE ADDRESS IT VACATED (DUE TO -(PC).
2386                                    ,*              2. SINCE R2 CONTAINS A "COM -(R1)" INSTRUCTION IT COMPLEMENTS
2387                                    ,*                 THE HIGHEST ADDRESS UNDER TEST  THIS ADDRESS CONTAINED
2388                                    ,*                 "177667" SO AFTER THE COM -(R1) IT EQUALS 110
2389                                    ,*                 CLEVERLY THIS IS THE "JMP (R0)" INSTRUCTION.
2390                                    ,*              3  THIS SEQUENCE CONTINUES UNTIL THE "MOV R2,-(PC) INSTRUCTIONS
2391                                    ,*                 REACH THE MIDDLE OF THE TEST BANK.  THEN THE "JMP (R0)" INSTRUCTION IS
2392                                    ,*                 AND EXECUTED  R0 CONTAINED THE RETURN ADDRESS BACK
2393                                    ,*                 TO TEST 13
2394                                    ,*              4. THESE STEPS ARE REPEATED FOR EACH BANK UNDER TEST
2395                                    ;*
2396                                    ,,*****************************************************************
2397  004324  122737  000013  000404  TST13.  CMPB    #13,@#$TESTN    ;CHECK FOR PROPER TEST SEQUENCE
2398  004332  001403                    BEQ     .+10
2399  004334  004767  001602            JSR     PC,SEQERR       ;*ERROR* REPORT ERROR MESSAGE AND HALT AT FATHLT
2400  004340  000044                    44                      ,******ERROR NUMBER 44*******
2401
2402  004342  012702  010247            1$.     MOV     #10247,R2       ,PLACE THE OP CODE OF INSTRUCTION MOV R2,-(PC)
2403                                                                  ,IN R2.
2404  004346  012700  177667                    MOV     #177667,R0      ,PLACE THE COMPLEMENT OF THE INSTRUCTION
2405                                                                  ,JMP (R0) IN R0
2406                                    ,INSURE LOWEST TEST ADDRESS TO END OF 4K SEGMENT IS MULTIPLE OF 2
2407                                    ,SINCE THE TEST STORES "MOV R2,-(PC) IN 1/2 AND 177667 IN THE OTHER 1/2
2408
2409  004352  010546                    2$.     MOV     R5,-(SP)        ,SAVE R5
2410  004354  010446                            MOV     R4,-(SP)        ;STORE LOWEST ADDRESS ON STACK
2411  004356  000241                    29$     CLC
2412  004360  006005                            ROR     R5              ;MAKE POSITIVE BYTE COUNT OF HIGH ADDRESS
2413  004362  006004                            ROR     R4              ;DO SAME FOR LOWEST ADDRESS
2414  004364  160405                            SUB     R4,R5           ,GET DIFFERENCE OF LOWEST ADDRESS AND HIGHEST
2415  004366  006005                            ROR     R5              ;IF DIFFERENCE IS ODD THEN R4 IS AT LOWEST ADDRESS
2416  004370  103002                            BCC     30$             ;BRANCH IF R4 IS AT LOWEST TEST ADDRESS.
2417  004372  062716  000002                    ADD     #2,(SP)         ;INCREASE LOWEST TEST ADDRESS BY 2
2418  004376  012604                    30$.    MOV     (SP)+,R4        ;RESTORE R4 (POSSIBLY INCREASED BY 2 FROM ENTRY)
2419  004400  012605                            MOV     (SP)+,R5        ;RESTORE HIGHEST TEST ADDRESS
2420  004402  010403                            MOV     R4,R3           ,PLACE THE LOWEST LOCATION UNDER TEST
2421                                                                  ,IN R3
2422  004404  000405                            BR      28$             ,LEAVE LOW BITS OF R3 ALONE FIRST TIME IN CASE BANK 0
```

```
2423  004406  042703  017776          3$:     BIC     #17776,R3       ;CAUSE R3 TO POINT TO THE LOWEST LOCATION
2424                                                                    ;IN THE 4K BANK UNDER TEST
2425  004412  105737  000405                  TSTB    @#REL           ;ARE WE RELOCATED?
2426  004416  100504                          BMI     14$             ;BRANCH IF YES-TEST BANK0 ONLY-
2427  004420  020305                  28$:    CMP     R3,R5           ;IF R3 IS HIGHER THAN THE HIGHEST LOCATION
2428  004422  103102                          BHIS    14$             ;UNDER TEST THEN EXIT
2429                                  ;IF R5 LESS THAN 20000 THEN WE ARE TESTING BANK0 RELOCATED IN BANK0
2430  004424  020527  020000                  CMP     R5,#20000       ;IS HIGHEST TEST ADDRESS BELOW 4K?
2431  004430  103002                          BHIS    31$             ;BRANCH IF NO
2432  004432  010501                          MOV     R5,R1           ;SET R1 TO HIGHEST TEST ADDRESS IN BANK0
2433  004434  000405                          BR      32$
2434
2435  004436  010301                  31$:    MOV     R3,R1           ;SET R1 TO LOWEST CURRENT TEST ADDRESS
2436  004440  042701  017776                  BIC     #17776,R1       ;CLEAR LOW ORDER ADDRESS BITS
2437  004444  062701  020000                  ADD     #20000,R1       ;CAUSE R1 TO POINT TO THE HIGHEST LOCATION+2
2438                                                                    ;OF THE 4K BANK BEING POINTED BY R3
2439  004450  020137  000340          32$     CMP     R1,@#MAXMEM     ;IF R1 IS HIGHER THAN MAX. OF THE
2440  004454  101065                          BHI     14$             ;MEMORY+2 ALTHOUGH R3 IS LESS THAN R5
2441                                                                    ;THEN THE HIGHEST LOCATION UNDER
2442                                                                    ;TEST IS NOT IN A 4K BANK EXIT
2443
2444  004456  132737  000001  000306          BITB    #1,@#PASFLG     ;IS THE LOWEST BIT OF LOCATION PASFLG
2445  004464  001101                          BNE     16$             ;SET? IN WHICH CASE BACK GROUND HAS
2446                                                                    ;ALREADY BEEN WRITTEN AND WRITE RECOVERY
2447                                                                    ;TEST IS BEING PERFORMED
2448
2449  004466  020304                  4$      CMP     R3,R4           ;OTHERWISE WRITE THE BACKGROUND
2450  004470  103430                          BLO     8$              ;DEFINED AT STEP 3.
2451  004472  105737  000307                  TSTB    @#PASFLG+1      ;IS THE TEST JUST DOING READ, I E
2452  004476  001002                          BNE     6$              ;IS THE PASFLG+1 LOCATION NON ZERO? IF SO
2453                                                                    ;THEN GO TO 6$
2454  004500  012713  010247                  MOV     #10247,(R3)     ;WRITE THE LOCATION WITH THE OP CODE FOR MOV R2,-(PC)
2455  004504  020213                  6$      CMP     R2,(R3)         ;READ (R3) TO CONTAIN CORRECT DATA
2456  004506  001421                          BEQ     8$
2457  004510  010046                          MOV     R0,-(SP)        ;SAVE R0
2458  004512  010146                          MOV     R1,-(SP)        ;AND R1 ON THE STACK
2459  004514  010301                          MOV     R3,R1
2460  004516  010200                          MOV     R2,R0           ;SET R0= GOOD DATA FOR ERROR PRINTOUT
2461  004520  004767  001060                  JSR     PC,ERROR        ;*ERROR* REPORT ERROR MESSAGE
2462  004524  000045                          45                      ;******ERROR NUMBER 45******
2463
2464  004526  012601                          MOV     (SP)+,R1        ;RESTORE R1
2465  004530  012600                          MOV     (SP)+,R0        ;AND R0
2466  004532  105737  000306                  TSTB    @#PASFLG        ;IF PASFLG IS 0 AND THE MEMORY DOES NOT HAVE
2467                                                                    ;THE PROPER DATA THEN WE DON'T WANT TO GO AND
2468                                                                    ;EXECUTE THE INSTRUCTIONS STORED IN MEMORY UNDER
2469                                                                    ;TEST
2470  004536  001005                          BNE     8$              ;BRANCH IF PASFLG NOT =0
2471
2472  004540  010200                          MOV     R2,R0   ;SAVE FOR ERROR REPORT
2473  004542  004767  001036                  JSR     PC,ERROR        ;*ERROR* REPORT ERROR MESSAGE
2474  004546  000046                          46                      ;******ERROR NUMBER 46******
2475
2476  004550  000664                          BR      END12           ;ABORT TST 13
2477
2478  004552  062703  000002          8$      ADD     #2,R3           ;INCREMENT R3 BY 2
```

```
2479  004556  162701  000002                  SUB     #2,R1           ,DECREMENT R1 BY 2
2480  004562  020105                          CMP     R1,R5           ;WRITE THE BACKGROUND DEFINED AT STEP 4.
2481  004564  103014                          BHIS    12$
2482  004566  020103                          CMP     R1,R3           ,HAS STORING THE 177667 REACHED WHERE "MOV R2,-(PC) IS?
2483  004570  103405                          BLO     10$             ;BRANCH IF YES DON'T DESTROY THE MOV R2,-(PC) IS.
2484  004572  105737  000307                  TSTB    @#PASFLG+1      ;IS THE THE READ ONLY CHECK PASS?
2485  004576  001002                          BNE     10$             ,BRANCH IF YES
2486  004600  012711  177667                  MOV     #177667,(R1)    ;WRITE THE LOCATION WITH THE COMPLEMENT OF THE
2487                                                                  ,OP CODE JMP (R0)
2488  004604  020011                  10$     CMP     R0,(R1)         ,READ R1 TO CONTAIN CORRECT DATA
2489  004606  001403                          BEQ     12$
2490  004610  004767  000770                  JSR     PC,ERROR        ;*ERROR* REPORT ERROR MESSAGE
2491  004614  000047                          47                      ,******ERROR NUMBER 47******
2492
2493  004616  020301                  12$     CMP     R3,R1           ,IF WE HAVE NOT REACHED THE MIDDLE OF 4K BANK
2494  004620  103722                          BLO     4$              ;THEN REPEAT FROM 4$
2495
2496                                  ,RETURN HERE AFTER PROGRAM RUN IN BANK UNDER TEST
2497
2498  004622  062703  020000          13$     ADD     #20000,R3       ,OTHERWISE GO TO THE NEXT 4K BANK
2499  004626  000667                          BR      3$
2500
2501  004630  122737  000001  000306  14$     CMPB    #1,@#PASFLG     ,THE PROGRAM CONTROL COMES HERE AS FOLLOWS
2502                                                                  ,1-PASFLG=0, PROGRAM HAS JUST COMPLETED A
2503                                                                  ,  WRITE/READ CYCLE FOR THE BACK GROUND
2504                                                                  ,  AND WANTS TO BEGIN THE WRITE RECOVERY TEST
2505  004636  001440                          BEQ     24$             ;2-PASFLG=1, PROGRAM HAS JUST COMPLETED
2506                                                                  ,  THE WRITE RECOVERY TEST AND WANTS TO
2507                                                                  ,  READ MEMORY FOR CORRECT DATA
2508  004640  103630                          BLO     END12           ,3-PASFLG=2, PROGRAM HAS CORRECTLY READ THE
2509                                                                  ,  MEMORY AND WANTS TO GO THE NEXT TEST
2510
2511  004642  105137  000307                  COMB    @#PASFLG+1      ,ENTER HERE WITH PASFLG=0, ON THE FIRST ENTRY
2512                                                                  ,ENABLE READ ONLY FOR THE MEMORY AND ON THE SECOND
2513                                                                  ,ENTRY DISABLE READ ONLY
2514  004646  001241                          BNE     2$
2515  004650  012702  005141                  MOV     #5141,R2        ,PLACE THE OP CODE FOR INSTRUCTION COM -(R1)
2516                                                                  ,IN R2
2517  004654  012700  177740                  MOV     #13$- -6,R0     ,PLACE THE RETURN ADDRESS IN R0 AS 13$
2518  004660  060700                          ADD     PC,R0           ;THUS WHEN THE READ RECOVERY TEST REACHES
2519                                                                  ,THE MIDDLE OF THE 4K MEMORY THEN THE
2520                                                                  ,INSTRUCTION EXECUTED WILL BE JMP (PC)
2521                                                                  ,BRANCHING NG THE PROGRAM TO 13$
2522  004662  105237  000306          15$     INCB    @#PASFLG        ,INCREMENT PASFLG BY 1
2523  004666  000631                          BR      2$
2524
2525  004670  032777  000020  173552  16$     BIT     #20,@SWR        ,HAS THE PRINTOUTS BEEN SUPRESSED ?
2526  004676  001017                          BNE     18$             ,IF SO THEN GO TO 18$
2527  004700  105737  000042                  TSTB    @#42            ;IS THE PROGRAM RUNNING UNDER ACT?
2528  004704  001014                          BNE     18$             ,BRANCH IF YES
2529  004706  004767  001660                  JSR     PC,PNTMES       , TYPE THE BANK UNDER TEST
2530  004712  051524  030524  020063           ASCIZ   /TST13 BANK/
2531  004720  040502  045516     000
2532          004726                           EVEN
2533  004726  004767  002476                  JSR     PC,GETBNK       ,GET BANK NO  UNDER TEST INTO DECWRD FOR PRINT
2534  004732  004767  001662                  JSR     PC,$TPDEC       ,TYPE BANK NO  UNDER TEST
```

```
 2535
 2536  004736  000113              18$     JMP     (R3)            ,BEGIN EXECUTING MOV R2,-(PC) ,COM -(R1) SEQUENCE IN TES
 2537
 2538
 2539  004740  105137  000307      24$.    COMB    @#PASFLG+1
 2540  004744  012700  000110              MOV     #110,R0         ,PLACE THE OP CODE FOR JMP (R0) IN R0
 2541  004750  000744                      BR      15$             ; READ THE MEMORY FOR CORRECT DATA AFTER
 2542                                                              ;INCREMEMTING PASFLG TO 2
 2543
 2544                              ,TST13 EXITS VIA END12
 2545
```

```
2546  004752  012737  000377  000316  RELOC   MOV     #377,@#BAKPAT
2547  004760  105737  000276                  TSTB    @#MMAVA         ;IS THE MEMORY MANAGEMENT BEING TESTED ?
2548  004764  001065                          BNE     CONTMM          ;IF SO THEN GO TO CONTMM AND CONTINUE TESTING
2549                                                                  ;MEMORY MANAGEMENT
2550  004766  032777  001000  173454          BIT     #1000,@SWR      ,RELOCATION WANTED?
2551  004774  001046                          BNE     CKDONE          ;BRANCH IF NO
2552  004776  105737  000405                  TSTB    @#REL           ;IF THE PROGRAM HAS ALREADY BEEN RELOCATED THEN ALSO
2553  005002  100420                          BMI     RELOER          ; PLACE THE PROGRAM BACK IN LOWER CORE
2554  005004  112737  000200  000405          MOVB    #200,@#REL      ;OTHERWISE PREPARE TO RELOCATE
2555
2556                                  ,RELOCATE THE DIAGNOSTIC TO HIGHEST AVAILABLE MEMORY
2557
2558
2559  005012  004767  001554                  JSR     PC,PNTMES       ,TYPE "RELOC"
2560  005016  042522  047514  000103           ASCIZ /RELOC/
2561                                           EVEN
2562  005024  013705  000340                  MOV     @#MAXMEM,R5     ,PREPARE TO LOAD THE PROGRAM IN THE HIGHEST
2563                                                                  ,AVAILABLE MEMORY
2564  005030  014445                  2$      MOV     -(R4),-(R5)     ,RELOCATE THE PROGRAM
2565  005032  020427  000430                  CMP     R4,#BEGIN-50    ;NEITHER RELOCATE NOR TEST LOCATIONS LOWER THAN BEGIN-50
2566  005036  101374                          BHI     2$
2567  005040  000165  000050                  JMP     50(R5)
2568
2569                                  ,*RELOCATE THE DIAGNOSTIC BACK TO LOWER MEMORY
2570
2571
2572  005044  013705  000346          RELOER  MOV     @#SAVR5,R5      ,RESTORE R5
2573  005050  105737  000405                  TSTB    @#REL           ,IS DIAGNOSTIC IN RELOCATED STATE?
2574  005054  100016                          BPL     CKDONE          ,BRANCH IF NO
2575
2576  005056  012704  000430                  MOV     #BEGIN-50,R4    ,PREPARE TO RELOCATE THE PROGRAM TO LOWER CORE
2577  005062  012524                  2$      MOV     (R5)+,(R4)+
2578  005064  020537  000340                  CMP     R5,@#MAXMEM
2579  005070  103774                          BLO     2$
2580  005072  105037  000405                  CLRB    @#REL
2581  005076  010537  000346                  MOV     R5,@#SAVR5      ,SAVE R5
2582  005102  012706  000500                  MOV     #BEGIN,SP       ,RESET STACK TO LOWER MEMORY
2583  005106  010637  000350                  MOV     SP,@#SAVR6      ,"BEGIN" USES THIS TO RESET THE STACK
2584  005112  000137  005116          CKDONE  JMP     @#LOWER         ,TRANSFER THE PROGRAM CONTROL TO THE LOWER CORE
2585
2586
2587
2588  005116  105737  000315          LOWER   TSTB    @#SAVKBB        ,HERE DUE TO  C TYPED?
2589  005122  001073                          BNE     $TPSTK          ,BRANCH IF YES (TYPE ERROR STACK)
2590  005124  004767  001714          TSTMM   JSR     PC,MEMMNG       , SET THE REGISTERS 'F THE MEMORY MANAGEMENT
2591                                                                  ,IS AVAILABLE
2592  005130  105737  000276                  TSTB    @#MMAVA         ,IS MEM. MANAG  AVAILABLE ?
2593  005134  001462                          BEQ     ENDPAS          ,BRANCH IF NO
2594  005136  000402                          BR      $CNTMM          ,BEGIN TESTING ABOVE 28K
2595  005140  004767  002052          CONTMM  JSR     PC,UPMM         ,GO TO UFDATE MEM. MANAG  REGISTERS
2596  005144  012703  000324          $CNTMM: MOV     #LOWTWO,R3      ;MAKE R3 POINT TO THE LOCATION LOWTWO
2597  005150  004767  002160                  JSR     PC,GETSIZ       , LOAD BITS 6-10 OF R2 WITH THE BITS 13-17
2598                                                                  ,OF THE LOWEST ADDRESS UNDER TEST
2599  005154  012704  020000                  MOV     #20000,R4       ,MAKE R4 POINT TO THE LOWEST LOCATION IN THE BANK
2600                                                                  ,POINTED BY PAGE ADDRESS REGISTER 1 (PAR1)
2601  005160  020237  172342                  CMP     R2,@#172342     ,IS THE CONTENT OF R2 LOWER THAN THE CONTENT OF
```

```
2602
2603  005164  103405                  BLO     2$              ;IF SO THEN GO 2$
2604  005166  050104                  BIS     R1,R4           ;SUBROUTINE GETSIZ LOADED R1 WITH BITS 0-12
2605                                                          ;OF LOWADD WHICH HAVE NOW BEEN LOADED IN R4
2606  005170  162702  000200          SUB     #200,R2
2607  005174  004767  001650          JSR     PC,MMREG        ; SET MEM  MANAG.  REGISTERS
2608  005200  004767  002130  2$      JSR     PC,GETSIZ       ;PLACE BITS 13-17 OF HIGHEST LOCATION TO BE TESTED
2609                                                          ;IN BITS 6-10 OF R2, #160000 IN R0 AND BITS 0-12
2610                                                          ;OF LOCATION HIGHADD IN R1
2611  005204  004767  000020          JSR     PC,MAXADR       ; GET THE ADDRESS OF MAX  MEM.  UNDER TEST
2612  005210  010005                  MOV     R0,R5
2613  005212  004767  002116          JSR     PC,GETSIZ       ; PREPARE TO SET UP LOCATION MAXMEM
2614  005216  004767  000006          JSR     PC,MAXADR       ; GET THE MAXIMUM ADDRESS OF AVAILABLE MEMORY
2615  005222  010013                  MOV     R0,(R3)         ;AND STORE INTO "MAXMEM"
2616  005224  000167  174242          JMP     CLRMEM          ;GO TEST A 24K SLICE ABOVE 28K
2617
2618
2619                                  ;MAXADR - SUBROUTINE TO GE" CURRENT 24K SLICE OF MEMORY ADDRESSES ABOVE 28K
2620                                  ;REGISTERS,
2621                                  ;R0= ON ENTRY= #160000 AND ON EXIT=HIGHEST VIRTUAL ADDR  UNDER TEST
2622                                  ;R1= LOW ORDER 12 BITS OF VIRTUAL TEST ADDRESS
2623                                  ;R2= PAR BLOCK NO  CURRENTLY UNDER TEST
2624
2625  005230  010046          MAXADR  MOV     R0,-(SP)        ;PUT MAXIMUM AVAILABLE ADDRESS ON STACK
2626  005232  012700  172356          MOV     #172356,R0      ;R0=PAR7 UNIBUS ADDRESS
2627                                  ;**BEGIN LOOP TO FIND PAR ADDRESS UNDER TEST
2628  005236  162716  020000  2$      SUB     #20000,(SP)     ;DECREMENT VIRTUAL ADDRESS BY 4K
2629  005242  050116                  BIS     R1,(SP)         ;SET BITS 11 0 TO MAXIMUM VIRTUAL TEST ADDRESS
2630  005244  020240                  CMP     R2,-(R0)        ;DOES CURRENT PAR= TEST BLOCK NO ?
2631  005246  001410                  BEQ     3$              ;BRANCH IF YES
2632  005250  020027  172340          CMP     R0,#172340      ;ARE WE AT PAR0?
2633  005254  101370                  BHI     2$              ;NO KEEP TRYING
2634                                  ;**END LOOP TO FIND PAR ADDRESS UNDER TEST
2635  005256  005720                  TST     (R0)+           ;SET TO PAR CURRENT
2636  005260  021002                  CMP     (R0),R2 ;IS THE PAR BLOCK UNDER TEST GTR THAN ALLOWED?
2637  005262  003006                  BGT     4$              ;BRANCH IF YES (FALL INTO ENDPAS)
2638  005264  012716  157776          MOV     #157776,(SP)    ;EXIT WITH MAXADR= 28K SEGMENT TEST SIZE
2639  005270  012600          3$      MOV     (SP)+,R0        ;SET R0 TO MAXIMUM VIRTUAL TEST ADDRESS
2640  005272  062700  000002          ADD     #2,R0           ;MAKE MAXIMUM MEMORY+2
2641  005276  000207                  RTS     PC              ;AND EXIT MAXADR ROUTINE
2642
2643  005300  022626          4$      CMP     (SP)+,(SP)+     ;FIXUP STACK
2644                                                          ;AND FALL THRU TO ENDPAS
```

```
                                        ;* TYPE ROUTINE FOR ERROR STACK
                                        ;* --------------------------
                                        ;*
                                        ;*      THIS ROUTINE IS USED TO DETERMINE IF TYPE OUT OF THE ERROR STACK
                                        ;*      FOR ONLY THE FAILING BITS IS REQUIRED OR NOT


005302  032777  000020  173140  ENDPAS  BIT     #20,@SWR        ;ARE WE GOING TO TYPE THE ERROR STACK AND END OF PASS?
005310  001055                  BNE     $EOP            ;IF NOT THEN GO TO $EOP
005312  012746  177777          STPSTK  MOV     #-1,-(SP)       ;THE PROGRAM HAS REACHED THE END AND ERROR
                                                        ;STACK AND END OF PASS WILL BE TYPED OUT
005316  012701  007744                  MOV     #ENDPRG,R1      ;PLACE THE STARTING ADDRESS OF THE ERROR STACK
                                                        ;FOR 0 TO 4K MEMORY IN R1
005322  012703  000376          TYPSTK  MOV     #376,R3
005326  005216                  INC     (SP)            ;IF WE HAVE GONE THRU THE ENTIRE
005330  020137  000310                  CMP     R1,@#ENDSTK     ;HAS THE END OF THE ERROR STACK BEEN REACHED ?
005334  103043                  BHIS    $EOP            ;THEN GO TO TYPE END OF PASS
005336  112702  000022                  MOVB    #18 ,R2
005342  105302          RETSTK  DECB    R2      ;IF ALL 16 BITS OF THIS BANK HAVE BEEN CHECKED
005344  002766                  BLT     TYPSTK          ;BEEN CHECKED FOR ERROR THEN SEE IF THERE
                                                        ;IS ANY MORE 4K MEMORY BANK
005346  105721                  TSTB    (R1)+           ;OTHERWISE CHECK THE BYTE STORED AT (R1)
005350  001774                  BEQ     RETSTK          ;IF IT IS 0 WE WILL NOT TYPE IT
005352  020227  000020                  CMP     R2,#16          ;IS THE POINTER POINTING TO ERROR STACK BYTE
                                                        ;MEANT FOR COLLECTING ADDRESS FAILURES FOR
                                                        ;THE SPECIFIC MEMORY BANK
005356  103404                  BLO     2$              ;IF NOT THEN GO TO TYPE BIT NUMBER
005360  101026                  BHI     PARFL           ;IF IT IS POINTING TO THE STACK LOCATION INTENDED
                                                        ;TO COLLECT PARITY FAILURES THEN GO TO PARFL
005362  004767  001012                  JSR     PC,TPADER       ;OTHERWISE TYPE "ADDRESS ERROR"
005366  000404                  BR      FAILNM
005370  010237  000312          2$      MOV     R2,@#DECWRD     ;PREPARE TO TYPE THE NUMBER OF THE FAILING BIT
                                                        ;IN DECIMAL
005374  004767  001214                  JSR     PC,TYPDEC       ;GO TO TYPE THE BIT NUMBER IN DECIMAL
005400  011637  000312          FAILNM  MOV     (SP),@#DECWRD   ;PREPARE TO TYPE THE PAGE NUMBER
005404  004767  001210                  JSR     PC,$TPDEC       ;IN DECIMAL
005410  005043                  CLR     -(R3)
005412  114113                  MOVB    -(R1),(R3)      ;PREPARE TO PRINTOUT THE NUMBER OF TIMES THIS
                                                        ;FAILURE OCCURED
005414  105021                  CLRB    (R1)+           ;CLEAR THE ERROR STACK
005416  005043                  CLR     -(R3)
005420  105237  000314                  INCB    @#TYPCNT        ;ENABLE THE TYPE OUT OF 1 WORDS
005424  004767  001330                  JSR     PC,RPTOCT       ; TYPE THE 4K BANK AND THE NUMBER OF TIMES
                                                        ;THIS FAILURE WAS SEEN
005430  012703  000376                  MOV     #376,R3         ;RESET SCRATCH STACK FOR EACH BIT PRINTED
005434  000742                  BR      RETSTK
005436  004767  000762          PARFL   JSR     PC,TPPRER       ; TYPE "PAR ERR"
005442  000756                  BR      FAILNM
```

```
                                        ;* END OF PASS
                                        ;* ----------
                                        ;*
                                        ;*      TYPE "END PASS" AND DISABLE PARITY
                                        ;*      ALSO SERVICE ACT11
                                        ;*      AND EVERY CONSECUTIVE PASSES UNLESS BIT 4 OF $SWREG IS HIGH
                                        ;*
005444  005002                  $EOP    CLR     R2              ;SET R2= PARITY MODULE DISABLE CODE
005446  004767  002036                  JSR     PC,PARITY       ;GO DISABLE PARITY MODULES IF SELECTED
005452  105737  000315                  TSTB    @#SAVKBB        ;CONTROL-C TYPED?
005456  001046                          BNE     CTLC            ;BRANCH IF YES-RESTORE LOADERS AND HALT-
005460  005237  000406                  INC     @#$PASS         ;INCREMENT PASS COUNT
005464  032777  000040  172756          BIT     #40,@SWR        ;"END PASS #XX" PRINTOUT WANTED?
005472  001015                          BNE     ACT11           ;BRANCH IF NO
005474  004767  001064          TYPEOP  JSR     PC,TPCRLF       ; TYPE CR, LF, AND "END PASS #"
005500  047105  020104  040520          ASCIZ   /END PASS #/
005506  051523  021440     000
        005514                          EVEN
005514  013737  000406  000312          MOV     @#$PASS,@#DECWRD        ;GET PASS COUNT
005522  004767  001072                  JSR     PC,$TPDEC               ;TYPE IT
005526  013700  000042          ACT11   MOV     @#42,R0         ;GET THE MONITOR ADDRESS
005532  001405                          BEQ     $DOAGN          ;IF NONE
005534  004767  000012                  JSR     PC,RLODER       ;RESTORE XXDP MONITOR
005540  000005                          RESET                   ;RETURN TO ACT11 MONITOR

                                        ;* SERVICE XXDP/ACT11
005542  000137  000156                  JMP     @#$ENDAD        ;JUMP TO ACT SERVICE

005546  000137  000250          $DOAGN: JMP     @#RESTRT        ;REPEAT TEST IF NOT UNDER ACT11/XXDP

005552  004767  001632          RLODER: JSR     PC,CLRMM        ;STOP MEMORY MANAGEMENT SO CAN RESTORE LOADERS
005556  013704  000344                  MOV     @#SAVR4,R4      ;RESTORE R4 WITH SAVR4
005562  014445                  4$      MOV     -(R4),-(R5)     ;RESTORE LOADERS
005564  020437  000210                  CMP     R4,@#ENDSTK
005570  101374                          BHI     4$
005572  000207                          RTS     PC              ;RETURN FROM RLODER CALL

                                        ;CONTROL C HANDLER

005574  004767  177752          CTLC    JSR     PC,RLODER       ;RESTORE ABS LOADER
005600  000167  000402                  JMP     APTHLT          ;IF NOT APT HALT AT FATHLT
```

```
                                        ;* ERROR HANDLING ROUTINE
                                        ;* ---------------------
                                        ;*
                                        ;*      PROGRAM COMES HERE EACH TIME AN ERROR IS ENCOUNTERED THIS
                                        ;*      ROUTINE TYPES OUT THE ERROR MESSAGE IN THE FORMAT GIVEN EARLIER

005604  017637  000000  000402  ERROR   MOV     @(SP),@#$FATAL  ;LOAD THE LOCATION $FATAL WITH THE ERROR NUMBER
005612  010346                  1$      MOV     R3,-(SP)        ;SAVE R3
005614  010046                          MOV     R0,-(SP)        ;AND R0 ON THE STACK

                                ;SETUP BANK NO  IN FATAL FOR APT

005616  010103                          MOV     R1,R3           ;GET VIRTUAL ADDRESS UNDER TEST FOR GETBNK
005620  004767  001604                  JSR     PC,GETBNK       ;GET BANK NO  UNDER TEST INTO PBNK
005624  013703  000312                  MOV     @#PBNK,R3       ;GET BANK UNDER TEST
005630  110337  000403                  MOVB    R3,@#$FATAL+1   ;STORE FAILING BANK NO  FOR APT

                                ;

005634  010346                          MOV     R3,-(SP)        ;TEMPORARILY STORE R3
005636  012703  000376                  MOV     #376,R3         ;MAKE R3 AS THE STACK POINTER
005642  013743  000306                  MOV     @#PASFLG,-(R3)  ;OUTPUT THE WORD STORED AT
005646  005043                  2$      CLR     -(R3)
005650  113713  000402                  MOVB    @#$FATAL,(R3)   ;PUT ERROR NO.  ON ERROR STACK
005654  016643  000006                  MOV     6(SP),-(R3)     ;PLACE THE RETURN PC AT (R3)
005660  011143                          MOV     (R1),-(R3)      ;PLACE BAD DATA,
005662  010043                          MOV     R0,-(R3)        ;AND GOOD DATA ON THE STACK
005664  005043                          CLR     -(R3)
005666  016313  000004                  MOV     4(R3),(R3)      ;TAKE THE
005672  040013                          BIC     R0,(R3)         ;EXCLUSIVE OR OF GOOD AND BAD DATA
005674  046300  000004                  BIC     4(R3),R0        ;TO FIND THE BITS THAT FAILED
005700  050013                          BIS     R0,(R3)         ;AND PLACE IT ON THE STACK
005702  012700  002012                  MOV     #ENDPRG- -24  R0,THIS CODE BRINGS THE RELATIVE ADDRESS
005706  060700                          ADD     PC,R0           ;OF THE STARTING OF THE ERROR STACK
005710  062700  000022          6$      ADD     #18 ,R0         ;FOR THE SFECIFIC 4K BANK
005714  005316                          DEC     (SP)
005716  002374                          BGE     6$
005720  005726                          TST     (SP)+           ;RESTORE THE STACK POINTER

005722  105037  000277          EPRTYP  CLRB    @#TYPENB        ;DISABLE ANY TYPE OUT
005726  105737  000300          1$      TSTB    @#$PRERR        ;IF THIS IS PARITY PROBLEM
005732  001007                          BNE     3$              ;THEN GO TO 3$
005734  105720                          TSTB    (R0)+           ;OTHERWISE INCREMENT THE ERROR STACK POINTER BY 1
005736  105737  000301                  TSTB    @#$ADERR        ;IF THIS IS ADDRESSING PROBLEM
005742  001003                          BNE     3$              ;THEN GO TO 3$
005744  105720                          TSTB    (R0)+           ;INCREMENT THE POINTER R0 BY 1
005746  005713                  2$      TST     (R3)            ;IS BIT 15 OF (R3) SET?
005750  100015                          BPL     4$              ;IF NOT THEN GO TO 4$
005752  122710  000377          3$      CMPB    #377,(R0)       ;OTHERWISE SEE IF THIS ERROR HAS OCCURED 377 TIMES
005756  001401                          BEQ     5$              ;IF SO DON'T BUMP ERROR COUNT
005760  105210                          INCB    (R0)            ;INCREMENT THE ERROR COUNTER BY 1
005762  122710  000001          5$      CMPB    #1,(R0)         ;MORE THAN 1 ERROR OCCURRED ON THIS BIT?
005766  001404                          BEQ     7$              ;BRANCH IF NO
005770  032777  000400  172452          BIT     #400,@SWR       ;STOP ERROR PRINTOUT AFTER 1 WANTED?
005776  001002                          BNE     4$              ;BRANCH IF YES (DON'T TYPE ERROR)
```

```
2798  006000  105237  000277          7$      INCB    @#TYPENB        ;ENABLE THE TYPE OUT ROUTINE
2799  006004  105737  000300          4$      TSTB    @#$PRERR        ;PARITY ERROR?
2800  006010  001411                          BEQ     6$              ;BRANCH IF NO
2801  006012  004767  000406                  JSR     PC,TPPRER       ;ELSE TYPE "PAR ERR"
2802  006016  000411                          BR      8$              ;AND DON'T TEST INDIVIDUAL BIT FAILURES
2803  006020  105737  000301                  TSTB    @#$ADERR        ;ADDRESS ERROR?
2804  006024  001403                          BEQ     6$              ;BRANCH IF NO
2805  006026  004767  000346                  JSR     PC,TPADERR      ;PRINT "ADR ERR"
2806  006032  000403                          BR      8$
2807  006034  105720                  6$      TSTB    (R0)+           ;POINT TO NEXT ENTRY IN ERROR STACK
2808  006036  006313                          ASL     (R3)            ;IS THERE STILL AN ERROR BIT SET IN ERROR
2809  006040  001342                          BNE     2$              ;BR IF YES - KEEP FILLING ERROR STACK
2810  006042  112737  000006  000314  8$      MOVB    #6,@#TYPCNT     ;TELL TYPOCT TO TYPE 6 WORDS OF ERROR STACK
2811                                                                  ;THE STACK POINTED BY R3
2812  006050  004767  001150                  JSR     PC,PUTADR       ;GO TO THE SUBROUTINE TO PLACE THE ADDRESS IN R1
2813                                                                  ;AT LOCATIONS (R3) AND (R3-2)
2814  006054  004767  000622                  JSR     PC,TYPERR       ;TYPE ERROR STACK (7 WORDS)
2815
2816  006060  005037  000300          10$     CLR     @#$PRERR        ;CLEAR ADDRESS/PARITY ERROR FLAGS
2817  006064  012600                          MOV     (SP)+,R0        ;RESTORE R0
2818  006066  012603                          MOV     (SP)+,R3        ;AND R3
2819  006070  105737  000420          FNDERR  TSTB    @#$ENV          ;ARE WE RUNNING UNDER APT?
2820  006074  001404                          BEQ     2$              ;IF NOT THEN TEST FOR HALT
2821  006076  012737  000001  000400          MOV     #1,@#$MSGTY     ;OTHERWISE INFORM THE APT
2822  006104  000443                          BR      FATHLT          ;GOTO FATHLT AND WAIT FOR APT
2823
2824  006106  010246                  2$      MOV     R2,-(SP)        ;SAVE R2 TEMP
2825  006110  005777  172334                  TST     @SWR            ;DOES THE OPERATOR REQUIRE THE PROGRAM TO HALT
2826                                                                  ;ON ERROR
2827  006114  100405                          BMI     4$              ;IF SO THEN HALT ON ERROR
2828                                  CHECK FOR CONTROL-C KEY
2829
2830  006116  004767  001546                  JSR     PC,CHECKC       ;IF CONTROL-C TYPED THEN PRINT ERROR HISTORY
2831                                                                  ;AND HALT AT FATHLT
2832  006122  105737  000042          7$      TSTB    @#42            ;ARE WE RUNNING UNDER ACT?
2833  006126  001401                          BEQ     6$              ;BRANCH IF NO
2834
2835  006130  000000                  4$      HALT                    ;PROGRAM HAS HALTED ON ERROR, R1 IS POINTING
2836                                                                  ;TO A LOCATION WHICH SHOULD HAVE CONTAINED
2837                                                                  ;THE WORD STORED IN R0
2838  006132  012602                  6$      MOV     (SP)+,R2        ;RESTORE R2
2839  006134  062716  000002                  ADD     #2,(SP)         ;RESTORE THE RETURN ADDRESS
2840  006140  000207                          RTS     PC              ;RETURN FROM THE SUBROUTINE
2841
2842
2843
2844  006142                          FATERR
2845  006142  004767  000416          SEQERR  JSR     PC,TPCRLF       ;TYPE "ERROR #"
2846  006146  051105  047522  020122          .ASCIZ  /ERROR #/
2847  006154  000043
2848                                          .EVEN
2849
2850  006156  017637  000000  000402          MOV     @(SP),@#$FATAL  ;LOAD THE LOCATION $FATAL WITH THE ERROR NUMBER
2851  006164  105237  000314                  INCB    @#TYPCNT        ;TELL $TPNUM TO TYPE 1 WORD
2852  006170  012703  000376                  MOV     #376,R3         ;$TPNUM USES R3 AS STACK
2853  006174  013743  000402                  MOV     @#$FATAL,-(R3)  ;PUT ERROR NO. ON STACK
```

```
2854  006200  005743                  TST     -(R3)           ;$TPNUM REQUIRES THIS
2855  006202  004767  000562          JSR     PC,FATYP        ;TYPE ERROR NO
2856  006206  105737  000420  APTHLT  TSTB    @#$ENV  ;RUNNING UNDER APT?
2857  006212  001326                  BNE     FNDERR          ;BRANCH IF YES
2858  006214  000000  FATHLT  HALT                            ;FATAL ERROR OR  C HALT
2859  006216  000137  000250          JMP     @#RESTRT                ;RESTART TST BUT DON'T CLEAR PASS COUNT
2860                                                          ;IN CASE  C RESTART
2861
2862
2863                                  ;PARERR
2864                                  ;       PARITY TRAP HANDLER
2865                                  ;COME HERE FROM A TRAP TO 114
2866                                  ;       THIS ROUTINE SEARCHES THE AVAILABLE PARITY MODULES AND IF ONE
2867                                  ;HAS A PARITY ERROR BIT SET THE GET THE PARITY ERROR ADDRESS
2868                                  ;AND CALL THE "ERROR" ROUTINE TO PRINT ERROR MESSAGE
2869                                  ;IF NO PARITY ERROR BITS CAN BE FOUND A FATAL ERROR 'S DONE
2870                                  ;
2871                                  ;REGISTER US AGE
2872                                  ;R0= HOLDS PARITY MODULE ADDRESSES
2873                                  ;R1= GETS ERROR ADDRESS FOR "ERROR" CALL
2874
2875  006222  012637  000356  PAPERR  MOV     (SP)+,@#PARSP   ;SET PAPSP TO RETURN ADDRESS
2876  006226  011637  000360          MOV     (SP),@#PARPS    ;SAVE PSW FOR RETURN
2877  006232  013706  000350          MOV     @#SAVR6,SP      ;AND RESET THE SP SINCE NOT ENOUGH STACK ROOM
2878                                                          ;TO COMPLETE THE ERROR SERVICE ROUTINE
2879  006236  010067  000132          MOV     R0,SAVR0         SAVE R0 DURING PARITY SERVICE
2880  006242  010167  000130          MOV     R1,SAVR1        ;SAVE R1 DURING PARITY SERVICE
2881  006246  013701  000352          MOV     @#PARMAP,R1     ;GET PARITY AVAILABLE MAP
2882  006252  012700  172100          MOV     #172100,R0      ;P0= FIRST PARITY ADDRESS
2883
2884  006256  005701                  TST     R1               ANY PARITY MODULES AVAILABLE?
2885  006260  001442                  BEQ     4$              ;BR IF NO -FATAL ERROR-
2886  006262  000241                  CLC
2887  006264  006001          1$      ROR     R1              ;SHIFT PARITY MAP BIT INTO C BIT
2888  006266  103005                  BCC     2$              ;BRANCH IF THIS PARITY MODULE NOT AVAILABLE
2889  006270  005710                  TST     (R0)            ;PARITY MODULE ERROR BIT SET?
2890  006272  100406                  BMI     3$              ;BRANCH IF YES -CALL "ERROR" ROUTINE
2891  006274  020027  172136          CMP     R0,#172136      ;DONE ALL PARITY MODULES?
2892  006300  002032                  BGE     4$              ;BR IF YES- GO TO FATAL ERROR CALL-
2893  006302  062700  000002  2$      ADD     #2,R0   ;POINT TO NEXT PARITY ADDRESS
2894  006306  000766                  BR      1$              ;AND KEEP TRYING
2895  006310  042710  100000  3$      BIC     #100000,(R0)     CLEAR PARITY ERROR BIT
2896  006314  011001                  MOV     (R0),R1         ;GET PARITY MODULE CSR
2897  006316  006101                  ROL     R1               SHIFT ERROR ADDRESS BITS 11-5 INTO 15-9
2898  006320  006101                  ROL     R1
2899  006322  006101                  ROL     R1
2900  006324  006101                  ROL     R1
2901  006326  042701  000777          BIC     #777,R1  SAVE ERROR ADDRESS ONLY
2902  006332  105237  000300          INCB    @#$PRERR        ;TELL "ERROR" PARITY ERROR CALL
2903  006336  004767  177242          JSR     PC,ERROR        ;*ERROR* REPORT ERROR MESSAGE
2904  006342  000050                  50                      ;******ERROR NUMBER 50******
2905
2906  006344  016700  000024          MOV     SAVR0,R0        ;RESTORE R0
2907  006350  016701  000022          MOV     SAVR1,R1         RESTORE R1
2908  006354  013746  000360          MOV     @#PARPS,-(SP)    SET RETURN PSW ON STACK
2909  006360  013746  000356          MOV     @#PARSP,-(SP)    AND SET RETURN ADDRESS ON STACK
```

```
2910  006364  000002                  RTI                     ;RETURN TO TEST WHERE PARITY TRAP OCCURRED
2911
2912                                  ;COME HERE IF NO PARITY ERROR FLAG FOUND SET
2913  006366                          4$
2914  006366  004767  177550          JSR     PC,FATERR       ;*ERROR* REPORT  ERROR MESSAGE AND HALT AT FATHLT
2915  006372  000051                  51                      ;******ERROR NUMBER 51******
2916
2917
2918                                  ;R0+R1 ARE SAVED HERE FOR PARITY TRAP DUE TO INSUFFICIENT
2919                                  ;STACK SPACE BETWEEN 500-450
2920  006374  000000                  SAVR0   0               ;SAVE R0 DURING PARITY TRAP SERVICE
2921  006376  000000                  SAVR1   0               ;SAVE R1 DURING PARITY TRAP SERVICE
2922
2923
2924  006400  105737  000277          TPADER  TSTB    @#TYPENB        ;TYPE ERROR?
2925  006404  001406                  BEQ     1$              ;BRANCH IF NO
2926  006406  004767  000160          JSR     PC,PNTMES       ; TYPE CR, LF AND "ADR ER"
2927  006412  042101  020122  051105          .ASCIZ  /ADR ERR/
2928  006420  000122
2929                                          .EVEN
2930  006422  000207                  1$      RTS     PC
2931
2932  006424  105737  000277          TPPPER  TSTB    @#TYPENB        ;ERROR PRINTOUTS ALLOWED?
2933  006430  001406                  BEQ     1$              ;BRANCH IF NO
2934  006432  004767  000134          JSR     PC,PNTMES       ;GO TO TYPE CR, LF AND "PAR EPR"
2935  006436  040520  020122  051105          .ASCIZ  /PAR ERR/
2936  006444  000122
2937                                          .EVEN
2938  006446  000207                  1$      RTS     PC
```

```
2939
2940
2941                                    ;* TYPE OUT ROUTINE
2942                                    ;* ---------------
2943                                    ;*
2944                                    ;*      THIS ROUTINE IS USED BY THE PROGRAM TO TYPE OUT ANY CHARACTER
2945                                    ;*
2946
2947  006450  010146           NOTYP   MOV     R1,-(SP)
2948  006452  016601  000002           MOV     2(SP),R1
2949  006456  105721           4$      TSTB    (R1)+           ;IF THIS TYPE OUT HAS BEEN SUPRESSED THEN
2950  006460  001376                   BNE     4$              ;PREPARE TO RETURN
2951  006462  000412                   BR      RETTYP
2952  006464  010146           $TYPE   MOV     R1,-(SP)        ;SAVE R1
2953  006466  010046                   MOV     R0,-(SP)        ;AND R0 ON THE STACK
2954  006470  016601  000004           MOV     4(SP),R1        ;PLACE THE ADDRESS OF MESSAGE TO BE TYPED IN R1
2955  006474  112100           2$      MOVB    (R1)+,R0        ;PLACE THE BYTE TO BE TYPED IN R0
2956  006476  001403                   BEQ     4$              ;IF IT IS END OF MESSAGE THEN GO TO 4$
2957  006500  004767  000022           JSR     PC,$TPCHR       ;OTHERWISE GO TO TYPE THE CONTENTS OF R0
2958  006504  000773                   BR      2$
2959  006506  012600           4$      MOV     (SP)+,R0        ;RESTORE R0
2960  006510  005201           RETTYP  INC     R1              ;CAUSE P1 TO
2961  006512  042701  000001           BIC     #1,R1           ;POINT TO EVEN ADDRESS
2962  006516  010166  000002           MOV     R1,2(SP)        ;MODIFY THE RETURN ADDRESS
2963  006522  012601                   MOV     (SP)+,R1        ;RESTORE R1
2964  006524  000416                   BR      EXTYP           ;AND RETURN VIA RTS PC
2965
2966  006526  132737  000040  000421 $TPCHR  BITB    #40,@#$ENUM     ;HAVE TYPE OUTS BEEN DISABLED?
2967  006534  001005                   BNE     4$              ;IF SO THEN RETURN FROM THE SUBPOUTINE
2968  006536  105737  177564     2$      TSTB    @#$TPS          ;WAIT HERE
2969  006542  100375                   BPL     2$              ;UNTIL THE PRINTER IS READY
2970  006544  110037  177566           MOVB    R0,@#$TPB       ;LOAD DATA TO BE TYPED INTO DATA REG
2971  006550  000404           4$      BR      EXTYP           ;RETURN
2972
2973  006552  004767  177706     PCRLF   JSR     PC,$TYPE
2974  006556  005015  000               ASCIZ   <15><12>        ;CR/LF
2975          006562                     EVEN
2976  006562  000207           EXTYP   RTS     PC              ;RETURN
2977
2978  006564  004767  177762     TPCRLF  JSR     PC,PCRLF        ;TYPE CR/LF
2979  006570  000735                   BR      $TYPE           ;NOW GO TO TYPE THE REST OF THE MESSAGE
2980
2981
2982  006572  032777  000020  171650 PNTMES  BIT     #20,@SWR        ;PRINTOUTS ALLOWED?
2983  006600  001323                   BNE     NOTYP           ;BRANCH IF NO
2984  006602  123737  000042  000046       CMPB    @#42,@#46       ;RUNNING UNDER ACT 11?
2985  006610  001717                   BEQ     NOTYP           ;BRANCH IF YES -NOT PRINTOUT-
2986  006612  000764                   BR      TPCRLF          ;SEND CR/LF AND TYPE MESSAGE
```

```
2987
2988                                    ;* ROUTINE TO TYPE OUT A DECIMAL NUMBER
2989                                    ;* ----------------------------------
2990                                    ;*
2991                                    ;*      THIS ROUTINE IS USED TO CONVERT THE CONTENTS OF LOCATION
2992                                    ;*      DECWRD TO DECIMAL NUMBERS AND TYPE THEN FOLLOWING 3 SPACES
2993                                    ;*
2994
2995  006614  004767  177732       TYPDEC  JSR     PC,PCRLF        ;TYPE CR/LF
2996
2997  006620  005046                STPDEC: CLR     -(SP)
2998  006622  013746  000312               MOV     @#DECWRD,-(SP)  ;GET THE WORD THAT HAS TO BE CONVERTED TO A
2999                                                                ;DECIMAL NUMBER
3000  006626  162716  000012       2$      SUB     #10.,(SP)
3001  006632  002403                       BLT     4$              ;IF THE NUMBER IN (SP) WAS LESS THAN 10. THEN
3002                                                                ;GO TO 4$
3003  006634  005266  000002               INC     2(SP)           ;OTHERWISE ADD 1 TO THE LOCATION STORING 10'S DIGIT
3004  006640  000772                       BR      2$              ;AND RETURN TO 2$
3005  006642  062716  000012       4$      ADD     #10.,(SP)
3006  006646  052716  000060               BIS     #60,(SP)        ;MAKE THE CONTENTS OF (SP) A DECIMAL NUMBER
3007  006652  112667  000020               MOVB    (SP)+,6$-2      ;PLACE THE 1'S DIGIT TO BE TYPED
3008  006656  052716  000060               BIS     #60,(SP)        ;MAKE THE CONTENTS OF (SP) A DECIMAL NUMBER
3009  006662  112667  000007               MOVB    (SP)+,6$-3      ;PLACE THE 10'S DIGIT TO BE TYPED
3010  006666  004767  177572               JSR     PC,$TYPE        ;GO TO TYPE THE NUMBER IN DECIMAL FOLLOWED BY
3011                                                                ;3 SPACES
3012  006672  020040  030040  000060        .ASCIZ  /   00/
3013                                        .EVEN
3014  006700  000207               6$      RTS     PC              ;RETURN FROM THE SUBROUTINE
```

```
3015
3016
3017                                        ;* OCTAL TYPE OUT ROUTINE
3018                                        ;* ---------------------
3019                                        ;*
3020                                        ;*      THIS ROUTINE IS USED TO TYPE OUT THE OCTAL VALUES
3021                                        ;*      CONTROL SHOULD COME TO THIS ROUTINE WITH R3 POINTING TO
3022                                        ;*      THE LOW ORDER BITS (I.E. BITS 0-15) OF THE ADDRESS TO
3023                                        ;*      BE TYPED WHERE AS R3-2 SHOULD CONTAIN THE HIGH ORDER BITS
3024                                        ;*      (I.E. BITS 16 & 17)  CONTENTS OF LOCATION R3-1 AND R0 ARE
3025                                        ;*      DESTROYED BY THIS SUBROUTINE
3026                                        ;*      BYTE TYPCNT SHOULD BE SET TO THE NUMBER OF WORDS THAT HAVE
3027                                        ;*      TO BE TYPED.
3028                                        ;*
3029
3030  006702  032777  020000  171540  TYPERR  BIT     #20000,@SWR     ;ERROR PRINTOUT WANTED?
3031  006710  001054                          BNE     OCTXT           ;BRANCH IF NO
3032  006712  004767  177634                  JSR     PC,PCRLF        ;TYPE CR/LF
3033  006716  004767  000012                  JSR     PC,TYPOCT       ;TYPE OCTAL NO.
3034  006722  000447                          BR      OCTXT           ;RETURN VIA RTS PC
3035  006724  012123                  OCTTYP: MOV     (R1)+,(R3)+     ;PLACE THE HIGH ORDER BITS AT LOCATION POINTED
3036                                                                  ;BY R3
3037  006726  012113                          MOV     (R1)+,(R3)      ;AND NOW PLACE THE LOW ORDER BITS
3038  006730  105237  000314                  INCB    @#TYPCNT        ;ENABLE THE TYPE OUT OF ONE OCTAL WORD
3039  006734  052743  000004          TYPOCT  BIS     #4,-(R3)
3040  006740  106113                  2$:     ROLB    (R3)
3041  006742  103376                          BCC     2$
3042  006744  005000                          CLR     R0
3043  006746  106113                          ROLB    (R3)            ;GET BITS 17 & 16 INTO R0
3044  006750  006100                          ROL     R0
3045  006752  106113                          ROLB    (R3)
3046  006754  006100                          ROL     R0
3047  006756  000405                          BR      $TPNUM
3048  006760  004767  177500          RPTOCT: JSR     PC,$TYPE        ; TYPE 3 SPACES
3049  006764  020040  000040                   ASCIZ   /   /
3050                                           EVEN
3051  006770  005000                  FATYP:  CLR     R0
3052  006772  012723  000006          $TPNUM  MOV     #6,(R3)+        ;ENABLE THE TYPE OUT OF 6 OCTAL DIGITS
3053  006776  000241                  4$:     CLC
3054  007000  006113                          ROL     (R3)
3055  007002  006100                          ROL     R0              ;PLACE THE CARRY FROM (R3) IN R0
3056  007004  052700  000060                  BIS     #60,R0          ;OR THE CONTENTS OF R0 WITH AN ASCII 0
3057  007010  004767  177512                  JSR     PC,$TPCHR       ; TYPE THE OCTAL NUMBER STORED IN R0
3058  007014  005000                          CLR     R0
3059  007016  006113                          ROL     (R3)
3060  007020  006100                          ROL     R0              ;PLACE THE CARRY FROM (R3) IN R0
3061  007022  006113                          ROL     (R3)
3062  007024  006100                          ROL     R0              ;PLACE THE CARRY FROM (R3) IN R0
3063  007026  105363  177776                  DECB    -2(R3)          ;IF WE HAVEN'T TYPED THE 6 OCTAL DIGITS
3064  007032  001361                          BNE     4$              ;THEN REPEAT FROM 4$
3065  007034  105337  000314                  DECB    @#TYPCNT        ;IF ALL THE WORDS REQUIRED HAVE NOT BEEN
3066  007040  001347                          BNE     RPTOCT          ;TYPED THEN REPEAT FROM RPTOCT
3067  007042  000207                  OCTXT   RTS     PC
```

```
3068
3069                                    ;* ROUTINE TO SET UP MEMORY MANAGEMENT REGISTERS
3070                                    ,* ----------------------------------------
3071                                    ,*
3072                                    ,*      PROGRAM CONTROL COMES HERE TO DETERMINE IF THE MEMORY MANAGEMENT
3073                                    ;*      IS AVAILABLE OR NOT, AND IF IT IS AVAILABLE THEN WHETHER
3074                                    ,*      THE MEMORY ABOVE 28K IS REQUIRED TO BE TESTED OR NOT.
3075                                    ,*
3076
3077  007044  012702  001400        MEMMNG. MOV     #1400,R2
3078  007050  105037  000276        MMREG   CLRB    @#MMAVA         ;CLEAR THE BYTE THAT IS SUPPOSED TO INDICATE
3079                                                                  ;THAT MEM. MANAG. IS AVAILABLE FOR TESTING
3080  007054  032777  010000  171366        BIT     #10000,@SWR     ;HAS THE OPERATOR ASKED TO CHECK MEMORY MANAG ?
3081  007062  001441                        BEQ     RETMM           ;IF NOT THEN RETURN FROM THE SUBROUTINE
3082  007064  012700  000004                MOV     #4,R0           ;PREPARE TO SETUP TIME OUT VECTOR
3083  007070  012720  007170                MOV     #NOMM,(R0)+     ;RETURN ADDRESS TO NOMM
3084  007074  012710  000340                MOV     #340,(R0)       ,AND WITH A PSW OF 340
3085  007100  005037  177572                CLR     @#SR0           ;TRY TO REACH MEM. MANAG. SR0
3086  007104  105237  000276                INCB    @#MMAVA         ,IF IT IS AVAILABLE THEN SET MEM MANAG AVAILABLE
3087                                                                  ;BYTE
3088  007110  012701  172340                MOV     #172340,R1      ,R1 IS POINTING TO PAR0
3089  007114  005021                        CLR     (R1)+           ,PAR0 WILL POINT TO BANK 0
3090  007116  062702  000200        2$      ADD     #200,R2
3091  007122  010221                        MOV     R2,(R1)+        ,SETUP PAR1-PAR6
3092  007124  020127  172356                CMP     R1,#172356
3093  007130  103772                        BLO     2$
3094  007132  012711  007600                MOV     #7600,(R1)      ;PAR7 IS POINTING TO THE I/O PAGE
3095  007136  012701  172300                MOV     #172300,R1
3096  007142  012721  077406        4$      MOV     #77406,(R1)+    ,SETUP PDR0-PDR7
3097  007146  020127  172316                CMP     R1,#172316
3098  007152  101773                        BLOS    4$
3099  007154  005237  177572                INC     @#SR0           ;ENABLE MEM MANAG
3100  007160  005010                $RETMM: CLR     (R0)            ,RESTORE TIME OUT TRAP VECTOR FOR ANY FUTURE TRAP
3101  007162  012740  000104                MOV     #BUSER,-(R0)
3102  007166  000207                RETMM   RTS     PC
3103
3104  007170  022626                NOMM    CMP     (SP)+,(SP)+     ,RESTORE STACK POINTER
3105  007172  004767  177366                JSR     PC,TPCRLF       ,TYPE "NO MEMORY MANAGEMENT MESSAGE
3106  007176  047516  046440  043516        ASCIZ   /NO MNG/
3107  007204     000
3108          007206                        EVEN
3109  007206  004767  176730                JSR     PC,FATERR       ,*ERROR* REPORT ERROR MESSAGE AND HALT AT FATHLT
3110  007212  000052                        52                      ,******ERROR NUMBER 52******
3111
3112  007214  000761                        BR      $RETMM          , RESTORE TIME OUT TRAP VECTOR
3113
3114  007216  013702  172354        UPMM    MOV     @#172354,R2     ,PREPARE TO UPDATE MEMORY MANAG REGISTERS
3115  007222  000712                        BR      MMREG
```

```
3116
3117                                    ;* 18 BIT ADDRESS GENERATOR
3118                                    ;* ------------------------
3119                                    ;*
3120                                    ;*      THIS SUBROUTINE IS USED TO PLACE THE ADDRESS STORED IN R1
3121                                    ;*      IN THE LOCATION POINTED BY R3.  THE ADDRESS IN R1 IS CONVERTED
3122                                    ;*      TO AN 18 BIT ADDRESS ONLY IF MEM  MANAG  IS AVILABLE IN WHICH
3123                                    ;*      CASE THE HIGH ORDER BITS OF THE ADDRESS ARE PLACED IN LOCATION
3124                                    ;*      POINTED BY R3-2
3125                                    ;*
3126
3127  007224  005063  177776    PUTADR  CLR     -2(R3)
3128  007230  010113                    MOV     R1,(R3)         ;PLACE THE ADDRESS STORED IN R1 IN LOCATION (R3)
3129  007232  105737  000276            TSTB    @#MMAVA         ;IS THE MEM. MANAG. AVAILABLE ?
3130  007236  001425                    BEQ     6$              ;IF NOT THEN RETURN FROM THE SUBROUTINE
3131  007240  010146                    MOV     R1,-(SP)        ;SAVE R1
3132  007242  042701  017777            BIC     #17777,R1       ;CLEAR BITS 0-12 OF THE ADDRESS IN R1
3133  007246  040113                    BIC     R1,(R3)         ;LEAVE BITS 0-12 OF THE ADDRESS IN (R3)
3134  007250  052701  004000            BIS     #4000,R1        ;PREPARE TO SHIFT R1 BY 12 PLACES
3135  007254  006001            2$      ROR     R1
3136  007256  103376                    BCC     2$              ;GET THE NUMBER OF PAR IN R1
3137  007260  062701  172340            ADD     #172340,R1      ;GET THE ADDRESS OF PAR IN R1
3138  007264  011101                    MOV     (R1),R1         ;LOAD R1 WITH THE CONTENTS OF PAR
3139  007266  052701  010000            BIS     #10000,R1
3140  007272  006101            4$      ROL     R1
3141  007274  103376                    BCC     4$              ;PLACE THE ADDRESS BITS 13-17 IN BITS 11-15 OF R1
3142  007276  006101                    ROL     R1
3143  007300  006143                    ROL     -(R3)           ;PLACE BIT 17 IN LOCATION POINTED BY R3-2
3144  007302  006101                    ROL     R1
3145  007304  006123                    ROL     (R3)+           ;PLACE BIT 16 OF THE ADDRESS
3146  007306  050113                    BIS     R1,(R3)         ;PLACE BITS 13-15 OF THE ADDRESS IN LOCATION (R3)
3147  007310  012601                    MOV     (SP)+,R1        ;RESTORE R1
3148  007312  000207            6$      RTS     PC              ;RETURN FROM THE SUBROUTINE
3149
3150                                    ;* GET ADDRESS FROM THE APT MAILBOX
3151                                    ;* --------------------------------
3152                                    ;*
3153                                    ;*      THIS SUBROUTINE IS USED TO GET ADDRESS FROM APT MAILBOX AND
3154                                    ;*      PLACE IT IN THE LOCATION USED BY THE PROGRAM TO  DEFINE THE
3155                                    ;*      MEMORY BOUNDRIES
3156                                    ;*      PROGRAM CONTROL SHOULD COME TO THIS SUBROUTINE WITH R1 POINT-
3157                                    ;*      ING TO THE MEMORY TYPE IN THE APT MAILBOX AND R3 POINTING TO
3158                                    ;*      THE LOCATION+2 WHERE THE LOW ORDER BITS OF THE ADDRESS HAVE
3159                                    ;*      TO BE PLACED
3160                                    ;*
3161
3162  007314  016143  000001    GETADR  MOV     1(R1),-(R3)     ;PLACE THE LOW ORDER BITS OF THE ADDRESS
3163  007320  005043                    CLR     -(R3)           ;CLEAR THE LOCATION WHERE THE HIGH ORDER BITS
3164                                                            ;HAVE TO BE PLACED
3165  007322  116113  177777            MOVB    -1(R1),(R3)     ;PLACE BITS 16 & 17
3166  007326  000207            2$      RTS     PC              ;RETURN FROM THE SUBROUTINE
```

```
                                        ;* CONVERT 18 BIT ADDRESS TO THE PAR FORM
                                        ;* ---------------------------------------
                                        ;*
                                        ;*      THIS SUBROUTINE IS USED TO CONVERT 18 BIT ADDRESS STORED IN
                                        ;*      LOCATIONS POINTED BY R3 AND R3+2 TO THE FORM IT WILL BE STORED
                                        ;*      IN A PAR. THE RESULT IS LEFT IN R2  R1 IS LOADED WITH BITS
                                        ;*      0-12 OF THE ADDRESS AND R0 WITH 160000
                                        ;*
007330  105237  000315          SGTSIZ  INCB    @#SAVKBB        ;PREPARE TO PLACE ADDRESS BITS 13-17 IN BITS
                                                                ;0-4 OF R2
007334  012301                  GETSIZ  MOV     (R3)+,R1
007336  011302                          MOV     (R3),R2         ;LOAD R2 WITH THE LOW ORDER BITS OF THE ADDRESS
007340  042702  017777                  BIC     #17777,R2       ;CLEAR ADDRESS BITS 0-12
007344  052702  000040          2$      BIS     #40,R2
007350  006001                  4$:     ROR     R1
007352  006002                          ROR     R2              ;ROTATE R1 AND R2 7 TIMES
007354  103375                          BCC     4$
007356  105737  000315                  TSTB    @#SAVKBB
007362  001405                          BEQ     6$
007364  105037  000315                  CLRB    @#SAVKBB
007370  052702  000100                  BIS     #100,R2
007374  000765                          BR      4$
007376  012301                  6$      MOV     (R3)+,R1        ;PLACE THE LOW ORDER ADDRESS BITS IN R1
007400  012700  160000                  MOV     #160000,R0
007404  040001                          BIC     R0,R1           ;LEAVE BITS 0-12 OF THE ADDRESS IN R1
007406  000207                          RTS     PC              ;RETURN FROM THE SUBRORNE


                                        ;* SUBROUTINE TO DISABLE MEMORY MANAGEMENT
                                        ;* ---------------------------------------
                                        ;*
                                        ;*      THIS SUBROUTINE IS CALLED TO DISABLE THE MEMORY MANAGEMENT
                                        ;*      UNIT

007410  105737  000276          CLRMM   TSTB    @#MMAVA         ;WAS THE MEMORY MANAGEMENT ENABLED ?
007414  001404                          BEQ     1$              ;IF NOT THEN GO TO 1$
007416  005037  177572                  CLR     @#SR0           ;DISABLE THE MEMORY MANAGEMENT
007422  105037  000276                  CLRB    @#MMAVA         ;AND DO NOT ATTEMPT TO TEST MEM  MANAG
007426  000207                  1$      RTS     PC              ;RETURN FROM THE SUBROUTINE


                                        ;* GET BANK NO  UNDER TEST
                                        ;       CALLED BY ERRTYP AND TST13 TO GET BANK NO  UNDER TEST INTO PBNK
                                        ;REGISTERS
                                        ;R0=POINTER TO PAR UNDER TEST
                                        ;R3=VIRTUAL ADDRESS ON ENTRY
                                        ;R0+R3 ARE RESTORED ON EXIT

007430  010046                  GETBNK: MOV     R0,-(SP)        ;SAVE R0
007432  010346                          MOV     R3,-(SP)        ;SAVE R3
007434  042703  017777                  BIC     #17777,R3       ;SAVE ONLY VIRTUAL BANK BITS
007440  052703  010000                  BIS     #10000,R3               ;SETUP R3 SHIFT BIT
```

```
3223  007444  000241                          CLC
3224  007446  006003                  1$      ROR     R3              ;SHIFT A BANK BIT
3225  007450  103376                          BCC     1$              ;UNTIL IN BITS <2.0> OF R3
3226  007452  105737  000276                  TSTB    @#MMAVA         ;MEMORY MANAGEMENT UNDER TEST?
3227  007456  001407                          BEQ     2$              ;NO EXIT
3228
3229                                  ,GET PAR ADDRESS AND PHYSICAL BANK NO.
3230  007460  006303                          ASL     R3              ,MAKE R3 PAR ADDRESS OFFSET
3231  007462  062703  172340                  ADD     #172340,R3      ,MAKE FULL PAR ADDRESS
3232  007466  011300                          MOV     (R3),R0         ;GET PAR CONTENTS
3233  007470  006300                          ASL     R0
3234  007472  000300                          SWAB    R0              ,SHIFT BANK BITS TO BITS <7 0>
3235  007474  110003                          MOVB    R0,R3           ,SET R3 TO PHYSICAL BANK NO
3236  007476  010337  000312          2$      MOV     R3,@#PBNK       ,STORE PHYSICAL BANK NO
3237  007502  012603                          MOV     (SP)+,R3        ,RESTORE R3
3238  007504  012600                          MOV     (SP)+,R0        ,RESTORE R0
3239  007506  000207                          RTS     PC              ,RETURN TO CALLER
3240
3241
3242
3243                                  , PARITY ENABLE/DISABLE ROUTINE
3244                                  ,
3245                                  ,       THIS ROUTINE ENABLES OR DISABLES PARITY MODULES AND PRINTS ASSOCIATED MEASSAGES
3246                                  ,       IF PARITY AVAILABLE THEN BIT13 OF "REL" IS SET AND "PAR"ITY IS PRINTED
3247                                  ,       ALSO THE BACKGROUND TEST PATTERN (LOC  BAKPAT) IS SET=376
3248                                  ,
3249                                  ,REGISTER USAGE.
3250                                  ,R0= POINTS TO BUS TIMEOUT TRAP VECTOR (LOC  4)
3251                                  ,R1= HOLDS PARITY MODULE UNIBUS ADDRESS
3252                                  ,R2= ON ENTRY HOLDS ENABLE/DISABLE CODE
3253                                  ,       IF R2=0 THEN DISABLE
3254                                  ,       IF R2=1 THEN ENABLE
3255                                  ,R3= SCRATCH TO SETUP LOC  PARMAP WITH A MAP OF PARITY MODULES PRESENT
3256
3257                                  ,CALL IS
3258                                  ,       MOV     #1,R2   ,ENABLE CODE
3259                                  ,       JSR     PC,PARITY
3260
3261
3262  007510  032777  004000  170732  PARITY  BIT     #4000,@SWR      ,PARITY TEST WANTED?
3263  007516  001460                          BEQ     6$                      ,BRANCH IF NO
3264
3265  007520  012700  000004                  MOV     #4,R0           ,POINT R0 TO BUS TIMEOUT ADDRESS
3266  007524  012710  000122                  MOV     #5$- -6,(R0)    ,SET RETURN FROM TIMEOUT TRAP TO 5$
3267  007530  060710                          ADD     PC,(R0)         ,IN THE CURRENT BANK
3268  007532  005037  000352          1$      CLR     @#PARMAP        ,CLEAR PARITY MAP HOLDER
3269  007536  012701  172140                  MOV     #172140,R1      ;SET R1 TO LAST PARITY MODULE ADDRESS+2
3270  007542  012703  100000                  MOV     #100000,R3      ,SET R3 TO PARMAP AVAILABLE CODE BEGIN
3271  007546  010241                  2$      MOV     R2,-(R1)        ,ENABLE A PARITY MODULE+TRAP IF NOT AVAILABLE
3272  007550  050337  000352                  BIS     R3,@#PARMAP     ,NO TRAP TO 5$, SO SET PARITY AVAILABLE
3273  007554  000241                          CLC
3274  007556  006003                  3$      ROR     R3              ,SETUP NEXT PARMAP BIT
3275  007560  103372                          BCC     2$              ,BRANCH IF NOT DONE ALL PARITY ADDRESSES
3276  007562  012710  000104                  MOV     #BUSER,(R0)     ,RESET BUS TIMEOUT TRAP VECTOR
3277  007566  005702                          TST     R2              ,IS THIS A DISABLE CALL?
3278  007570  001433                          BEQ     6$              ,BRANCH IF YES (EXIT)
```

```
3279  007572  005737  000352              TST     @#PARMAP        ;WERE ANY PARITY MODULES FOUND?
3280  007576  001011                      BNE     4$              ;BRANCH IF YES
3281  007600  004767  176760              JSR     PC,TPCRLF       ,PRINT "NO PAR"
3282  007604  047516  050040  051101       ASCIZ  /NO PAR/
3283  007612     000
3284          007614                       EVEN
3285  007614  004767  176322              JSR     PC,FATERR       ;*ERROR* REPORT  ERROR MESSAGE AND HALT AT FATHLT
3286  007620  000053                      53                      ,******ERROR NUMBER 53******
3287
3288
3289  007622  152737  000040  000405  4$: BISB    #40,@#REL       ;SET PARITY UNDER TEST FLAG
3290  007630  012737  000376  000316      MOV     #376,@#BAKPAT   ;SET BACKGROUND PATTERN TO
3291                                                              ,WORST CASE PARITY CODE
3292  007636  004767  176722              JSR     PC,TPCRLF       ,PRINT "TST PARITY"
3293  007642  040520  044522  054524       ASCIZ  /PARITY/
3294  007650     000
3295          007652                       EVEN
3296  007652  000405                      BR      EXITC           ,AND EXIT VIA RTS PC
3297
3298                          ,GET HERE IF PARITY ADDRESS TIMED OUT TO LOC  4
3299
```

```
3300  007654  022626                  5$      CMP     (SP)+,(SP)+     ,RESET STACK FROM TRAP
3301  007656  000737                          BR      3$              ;KEEP TRYING PARITY ADDRESES
3302
3303  007660  142737  000040  000405  6$      BICB    #40,@#REL       ,CLEAR PARITY TESTING FLAG
3304  007666                          EXITC
3305  007666  000207                  7$      RTS     PC              ,RETURN TO CALLER
3306
3307
3308
3309
3310                                  ,CHECKC
3311                                  ,       THIS ROUTINE CHECKS IF CONTROL-C WAS TYPED AT THE END OF EACH
3312                                  ,TEST OR IN THE ERROR TYPE ROUTINE
3313                                  .IF CONTROL-C TYPED THE PROGRAM IS RETURNED TO LOWER MEMORY IF IT WAS
3314                                  ,RELOCATED AND THE ERROR HISTORY IS PRINTED OUT
3315                                  ,FINALLY IT HALTS AT FATHLT
3316
3317  007670  105037  000315          CHECKC: CLRB    @#SAVKBB        ;INIT CONTROL-C FLAG
3318  007674  105737  177560                  TSTB    @#TKS           ,ANY CHAR  TYPED?
3319  007700  100372                          BPL     EXITC           ,BR IF NO-EXIT VIA RTS PC-
3320  007702  113702  177562                  MOVB    @#$KBB,R2       ,GET THE CHAR TYPED
3321  007706  042702  000200                  BIC     #200,R2 ,CLEAR THE PARITY BIT
3322  007712  122702  000003                  CMPB    #3,R2           ,IS IT CONTROL-C?
3323  007716  001363                          BNE     EXITC           ,BRANCH IF NO -EXIT VIA RTS PC-
3324  007720  110237  000315                  MOVB    R2,@#SAVKBB     ,ELSE STORE THE CHAR  FOR USE AS A FLAG
3325  007724  004767  176634                  JSR     PC,TPCRLF       .PRINT " C"
3326  007730  041536    U00                    ASCIZ   / C/
3327          007734                          EVEN
3328  007734  000167  175104                  JMP     RELOER          ,GO PETURN PROGRAM TO LOWER CORE IF RELOCATED.
3329
3330          007744                  =7744
3331  007744  000000                  ENDPRG  0               ,THIS BEGINS THE STORAGE FOR THE ERROR HISTORY
3332                                          ,STACK FOR EACH 4K BANK 18  BYTES ARE SAVED
3333                                          ,ALSO THE ABSOLUTE LOADER AND XXDP CODE IS SAVED
3334                                          ,AFTER THE ERROR STACK
3335                                          ,FOR 4K MEMORY SIZE THEN PROGRAM=7744+22=7776
3336          000001                  END
```

```
ABASE = 000000           1222
ACDW1 = 000000           1222
ACDW2 = 000000           1222
ACPUOP= 000000           1222    1237
ACT11   005526           2711    2718#
ADDW0 = 000000           1222
ADDW1 = 000000           1222
ADDW10= 000000           1222
ADDW11= 000000           1222
ADDW12= 000000           1222
ADDW13= 000000           1222
ADDW14= 000000           1222
ADDW15= 000000           1222
ADDW2 = 000000           1222
ADDW3 = 000000           1222
ADDW4 = 000000           1222
ADDW5 = 000000           1222
ADDW6 = 000000           1222
ADDW7 = 000000           1222
ADDW8 = 000000           1222
ADDW9 = 000000           1222
ADEVCT= 000000           1222    1228
ADEVM = 000000           1222
AENV  = 000000           1222    1233
AENVM = 000000           1222    1234
AFATAL= 000000           1222    1225
AMADR1= 000000           1222    1250
AMADR2= 000000           1222    1254
AMADR3= 000000           1222    1257
AMADR4= 000000           1222    1260
AMAMS1= 000000           1222    1244
AMAMS2= 000000           1222    1252
AMAMS3= 000000           1222    1255
AMAMS4= 000000           1222    1258
AMSGAD= 000000           1222    1230
AMSGLG= 000000           1222    1231
AMSGTY= 000000           1222    1224
AMTYP1= 000000           1222    1245
AMTYP2= 000000           1222    1253
AMTYP3= 000000           1222    1256
AMTYP4= 000000           1222    1259
APASS = 000000           1222    1227
APRIOR= 000000           1222
APTHLT  006206           2739    2856#
APTSIZ  000724           1322    1336#
ASWREG= 000000           1222    1235
ATESTN= 000000           1222    1226
AUNIT = 000000           1222    1229
AUSWR = 000000           1222    123e
AVECT1= 000000           1222
AVECT2= 000000           1222
BAKPAT  000316           1047    1182#   1290*   1501    1724    1726    1734    1745*   1809    1841*   2023*   2059    2094*
                         2145    2151    2155    2168    2171    2210*   2307    2363*   2364    2546*   3290*
BEGIN   000500           1055    1185    1206    1270#   2565    2576    2582
BPTPSZ  001012           1354#
BUSER   000104           1027#   1397    3101    3276
```

```
CHECKC  007670       1568   2830   3317#
CKDONE  005112       2551   2574   2584#
CLRMEM  001472       1495#  2616
CLRMM   007410       1429   1451   2729   3205#
CNTSCP  001650       1567   1570#
CONT    001532       1512#  1581
CONTMM  005140       2548   2595#
CTLC    005574       2708   2738#
DECWRD  000312       1167#  1168   2677*  2680*  2716*  2998
ENDPAS  005302       2593   2653#
ENDPRG  007744       1093   1288   2657   2775   3331#
ENDSTK  000310       1164#  1165   1416*  2661   2732
END0    002136       1666#  1708
END1    002236       1708#  1748
END10   003776       2215   2223#  2366
END12   004322       2366#  2476   2508
END2    002346       1748#  1844
END3    002614       1844#  1883
END4    002724       1883#  1969
END5    003072       1969#  2025
END6    003226       2025#  2102
END7    003450       2102#  2223
ERROR   005604       1625   1686   1740   1805   1869   1909   1916   1943   2001   2076   2156   2172   2181
                     2326   2334   2461   2473   2490   2749#  2903
ERRTYP  005722       2782#
EXITC   007666       3296   3304#  3319   3323
EXTYP   006562       2964   2971   2976#
FAILNM  005400       2676   2680#  2693
FATERR  006142       1301   1349   1462   1484   1652   2844#  2914   3109   3285
FATHLT  006214       2822   2858#
FATYP   006770       2855   3051#
FNDERR  006070       2819#  2857
GALLOP  003476       2139#  2273
GETADR  007314       1352   1353   3162#
GETBNK  007430       2533   2756   3219#
GETSIZ  007334       2597   2608   2613   3150#
HIGHAD  000332       1193#  1338
HIGHTW  000330       1192#  1337   1407
LOOP    001542       1515#  1579
LOWADD  000326       1190#
LOWBNK  000304       1157#  1158   2046*  2057   2092
LOWER   005116       2584   2588#
LOWTWO  000324       1189#  1400   1453   2596
M     = 000200       1002#  2490
MAXADR  005230       2611   2614   2625#
MAXMEM  000340       1199#  1336   1420*  1642   1838   2439   2562   2578
MEMMNG  007044       1372   2590   3077#
MEMTST  001464       1489#
MMAVA   000276       1139#  1142   1374   2547   2592   3078*  3086*  3129   3205   3208*  3226
MMREG   007050       1385   2607   3078#  3115
N     = 000054       1002#  1301   1304#  1349   1352#  1462   1465#  1484   1487#  1596   1599#  1610   1612#
                     1613   1616#  1621   1623#  1652   1655#  1678   1681#  1686   1689#  1721   1724#  1740
                     1743#  1765   1768#  1790   1792#  1800   1802#  1816   1818#  1853   1856#  1869   1872#
                     1902   1905#  1909   1912#  1916   1919#  1943   1946#  1991   1994#  2001   2004#  2042
                     2045#  2076   2079#  2133   2136#  2156   2159#  2172   2175#  2181   2184#  2258   2261#
                     2298   2301#  2326   2329#  2334   2337#  2399   2402#  2461   2464#  2473   2476#  2490
```

```
                    2493#   2903    2906#   2913    2917#   3109    3112#   3285    3288#
NOMM    007170      3083    3104#
NOTYP   006450      2947#   2983    2985
OCTTYP  006724      1402    1406    3035#
OCTXT   007042      3031    3034    3067#
ONEPAS  000556      1282    1285#
PARERR  006222      1278    2875#
PARFL   005436      2673    2692#
PARITY  007510      1500    2706    3262#
PARMAP  000352      1207#   2881    3268*   3272*   3279
PARPS   000360      1210#   2876*   2908
PARSP   000356      1209#   2875*   2909
PASFLG  000306      1159#   1163    1512*   1879*   1947*   1964*   1996*   2007    2009*   2022*   2045*   2067    2083*
                    2085    2090    2304*   2321    2341    2347*   2348    2356*   2357    2444    2451    2466    2484
                    2501    2511*   2522*   2539*   2764
PBNK    000312      1166#   2757    3236*
PCRLF   006552      1401    2973#   2978    2995    3032
PNTMES  006572      1055    2265    2529    2559    2926    2934    2982#
PUTADR  007224      1389    1392    1518    2812    3127#
PWRDN   000070      1020#   1274
PWRUP   000136      1020    1054#
REL     = 000405    1134#   1281    1475    2425    2552    2554*   2573    2580*   3289*   3303*
RELBOT  000322      1185#   1477
RELOC   004752      1548    2546#
RELOER  005044      2553    2572#   3328
RESTRT  000250      1069    1085    1087    1093#   2727    2859
RETMM   007166      3081    3102#
RETSTK  005342      2664#   2668    2691
RETTYP  006510      2951    2960#
RLODER  005552      2720    2729#   2738
RPTOCT  006760      2688    3048#   3066
RPT10   003472      2137#   2222    2268
RPT11   004036      2218    2221    2264    2267#
RPT6    003112      1994#   2024
SAVKBB  000315      1171#   2588    2707    3177*   3187    3189*   3317*   3324*
SAVLOR  001246      1422    1424    1429#
SAVLOC  000354      1208#   2017*
SAVMAX  000342      1201#   1487
SAVR0   006374      2879*   2906    2920#
SAVR1   006376      2880*   2907    2921#
SAVR4   000344      1202#   2730
SAVR5   000346      1094    1203#   1450*   2572    2581*
SAVR6   000350      1054    1079    1206#   1273*   1629    2583*   2877
SCOPE   = 000240    1003#
SEQERR  006142      1596    1678    1721    1765    1853    1902    1991    2042    2133    2258    2298    2399    2845#
SETSTK  001170      1287    1407#
SETSWR  000656      1312#
SLFSIZ  001024      1363#
SR0     = 177572    1181#   3086*   3099*   3207*
START   000200      1062    1079#
STRTD1  000302      1155#   1156    2055*   2084
SWAPAT  000320      1184#   2176    2180    2209*
SWHALT  001670      1576#
SWR     000450      1267#   1315    1318*   1323*   1356    1460    1504    1574    1578    1639    2263    2267    2525
                    2550    2653    2710    2796    2825    2982    3030    3080    3262
SWREG   000176      1072#   1318
```

```
SW11   = 004000          1002#
TBL      001604          1532    1536#   1537    1538    1539    1540    1541    1542    1543    1544    1545    1546    1547
                         1548
TPS    = 177560          1177#   3318
TPADER   006400          2675    2805    2924#
TPCRLF   006564          1088    2712    2845    2978#   2986    3105    3281    3292    3325
TPPRER   006424          2692    2801    2932#
TRYSR    001014          1340    1356#
TSTGO    001672          1575    1578#
TSTMM    005124          1465    2590#
TSTREL   001324          1283    1450#
TSTRP    000570          1286    1288#
TSTSCP   001636          1566#   1666
TSTSIZ   001330          1451#
TST0     001706          1536    1594#
TST1     002140          1537    1674#
TST10    003452          1544    2130#
TST11    004000          1545    2255#
TST12    004052          1546    2296#   2365
TST13    004324          1547    2397#
TST2     002240          1538    1718#
TST3     002350          1539    1763#   1842
TST4     002616          1540    1851#
TST5     002726          1541    1899#
TST6     003074          1542    1987#
TST7     003230          1543    2039#
TYPCNT   000314          1169#   1276*   2687*   2810*   2851*   3038*   3065*
TYPDEC   006614          2679    2995#
TYPENB   000277          1143#   1146    2782*   2798*   2924    2932
TYPEOP   005474          2712#
TYPERR   006702          2814    3030#
TYPMEM   001156          1403#
TYPOCT   006734          3033    3039#
TYPSIZ   001134          1354    1357    1397#
TYPSTK   005322          2659#   2665
UPMM     007216          2595    3114#
WRTMEM   000120          1046#   1769    1905    1994
$A     = 000001           390#    392     393#    395     396#    400     401#    405     406#    410     411#    416     417#
                          423     424#    429     430#    434     435#    438     439#    442     443#    446     447#    450
                          451#    454     455#    458     459#    462     463#    466     467#    471     472#    475     476#
                          479     480#    483     484#    487     488#    489     490#    497     498#    501     502#    505
                          506#    516     517#    520     521#    529     530#    533     534#    536     537#    542     543#
                          546     547#    550     551#    554     555#    562     563#    566     567#    570     571#    575
                          576#    580     581#    587     588#    593     594#    597     598#    602     603#    606#
$ADERR   000301          1151#   1154    1617*   1738*   1802*   1867*   2786    2803
$APTHD   000276          1119    1125#   1138
$CNTMM   005144          2594    2596#
$CPUOP   000426          1237#
$DEVCT   000410          1228#   1573*   2082*   2205*
$DOAGN   005546          2719    2727#
$ENDAD   000156          1014    1065#   2725
$ENV     000420          1233#   1321    1566    2819    2856
$ENVM    000421          1234#   1339    1637    2966
$EOP     005444          2654    2662    2705#
$ETABL   000420          1232#
$ETEND   000450          1131    1261#
```

```
$ATAL   000402          1225#   2749*   2758*   2766    2850*   2853
$GTSIZ  007330          1408    1523    3177#
$HD   = 000002           993
$HIBTS  000276          1126#
$HIMAX  000334          1196#   1337*
$KBB  = 177562          1178#   3320
$MADR1  000432          1250#
$MADR2  000436          1254#
$MADR3  000442          1257#
$MADR4  000446          1260#
$MAIL   000400          1082    1127    1131    1223#   1293    1308
$MAMS1  000430          1244#
$MAMS2  000434          1252#
$MAMS3  000440          1255#
$MAMS4  000444          1258#
$MAXM   000336          1197#   1338*
$MBADR  000300          1127#
$MSGAD  000414          1230#
$MSGLG  000416          1231#
$MSGTY  000400          1027*   1224#   2821*
$MTYP1  000431          1245#   1346
$MTYP2  000435          1253#
$MTYP3  000441          1256#
$MTYP4  000445          1259#   1342
$NWTST= 000001          1582#   1584    1669#   1671    1710#   1712    1752#   1754    1845#   1847    1885#   1887    1971#
                        1973    2028#   2030    2103#   2105    2227#   2229    2274#   2276    2367#   2369
$PASS   000406          1227#   1285    2709*   2716
$PASTM  000304          1129#
$PRERR  000300          1147#   1150    1275*   2783    2799    2816*   2902*
$RETMM  007160          3100#   3112
$SVPC = 000044          1012#   1017
$SWR  = 000000           993#   1002#   1595    1675    1719    1764    1852    1900    1988    2040    2131    2256    2297
                        2398
$SWREG  000422          1235#   1323
$TESTN  000404          1086    1134    1226#   1513*   1570    1594    1674    1718    1763    1851    1899    1987    2039
                        2130    2187    2195    2220    2255    2296    2397
$TN   = 000014           983#    993    1582    1595#   1669    1675#   1710    1719#   1752    1764#   1845    1852#   1885
                        1900#   1971    1988#   2028    2040#   2103    2131#   2227    2256#   2274    2297#   2367    2398#
$TPB  = 177566          1180#   2970*
$TPCHR  006526          2957    2966#   3057
$TPDEC  006620          2534    2681    2717    2997#
$TPNUM  006772          3047    3052#
$TPS  = 177564          1179#   2968
$TPSTK  005312          2589    2655#
$TSTM   000302          1128#
$TYPE   006464          1403    2952#   2973    2979    3010    3048
$UNIT   000412          1080    1229#
$UNITM  000306          1130#
$USWR   000424          1236#
$Z    = 000362          1215#
$ZZ   = 007740          3330#
$$M   = 000200          2490#
      = 007746          1000#   1005#   1012    1013#   1015#   1017#   1019#   1025#   1032#   1071#   1115    1116#   1118#
                        1120#   1138#   1142#   1146#   1150#   1154#   1156#   1158#   1163#   1165#   1168#   1215    1218#
                        1269#   1278    1532    1595    1647    1677    1720    1764    1852    1901    1990    2041    2132
                        2257    2297    2398    2517    2532#   2715#   2775    2975#   3108#   3266    3284#   3295#   3327#
```

```
                        3330#
 $$   = 000276          1115#   1120
```

```
ERRLST   390#   392    395    400    405    410    416    423    429    434    438    442    446    450    454
         458    462    466    471    475    479    483    487    489    497    501    505    516    520    529
         533    536    542    546    550    554    562    566    570    575    580    587    593    597    602
MSG     1582#  1584   1669#  1671   1710#  1712   1752#  1754   1845#  1847   1885#  1887   1971#  1973   2028#
        2030   2103#  2105   2227#  2229   2274#  2276   2367#  2369
NEWTST   983#  1582   1669   1710   1752   1845   1885   1971   2028   2103   2227   2274   2367
PLCERR   998#  1610   1613   1621   1790   1800   1816
STARS    983#  1010   1075   1078   1112   1114   1121   1187   1194   1221   1264   1266   1582   1593   1669
        1673   1710   1717   1752   1762   1845   1850   1885   1898   1971   1986   2028   2038   2103   2129
        2227   2254   2274   2295   2367   2396
$ERRNM   998#  1686   1740   1869   1909   1916   1943   2001   2076   2156   2172   2181   2326   2334   2461
        2473   2490   2903
$FATAL   391#   392    395    400    405    410    416    423    429    434    438    442    446    450    454
         458    462    466    471    475    479    483    487    489    497    501    505    516    520    529
         533    536    542    546    550    554    562    566    570    575    580    587    593    597    602
$FTERR   998#  1301   1349   1462   1484   1652   2913   3109   3285
$SQERR   998#  1596   1678   1721   1765   1853   1902   1991   2042   2133   2258   2298   2399
$$NEWT   983#  1582   1669   1710   1752   1845   1885   1971   2028   2103   2227   2274   2367
.HEADE   983#
 $ACT1   983#  1008
.$APTB   983#  1219
.$APTH   983#  1110


.  ABS   007746     000


 ERRORS DETECTED:  0

DZKMAD BIN,DZKMAD.LST/CRF/SOL/NL.TOC=DZKMAD P11
RUN-TIME.  8 8 .5 SECONDS
RUN-TIME RATIO:  271/17=15 6
CORE USED:  11K  (21 PAGES)
```