PDP11

07124K MEMORY EXERCISER

MD-11-DZQMB-G

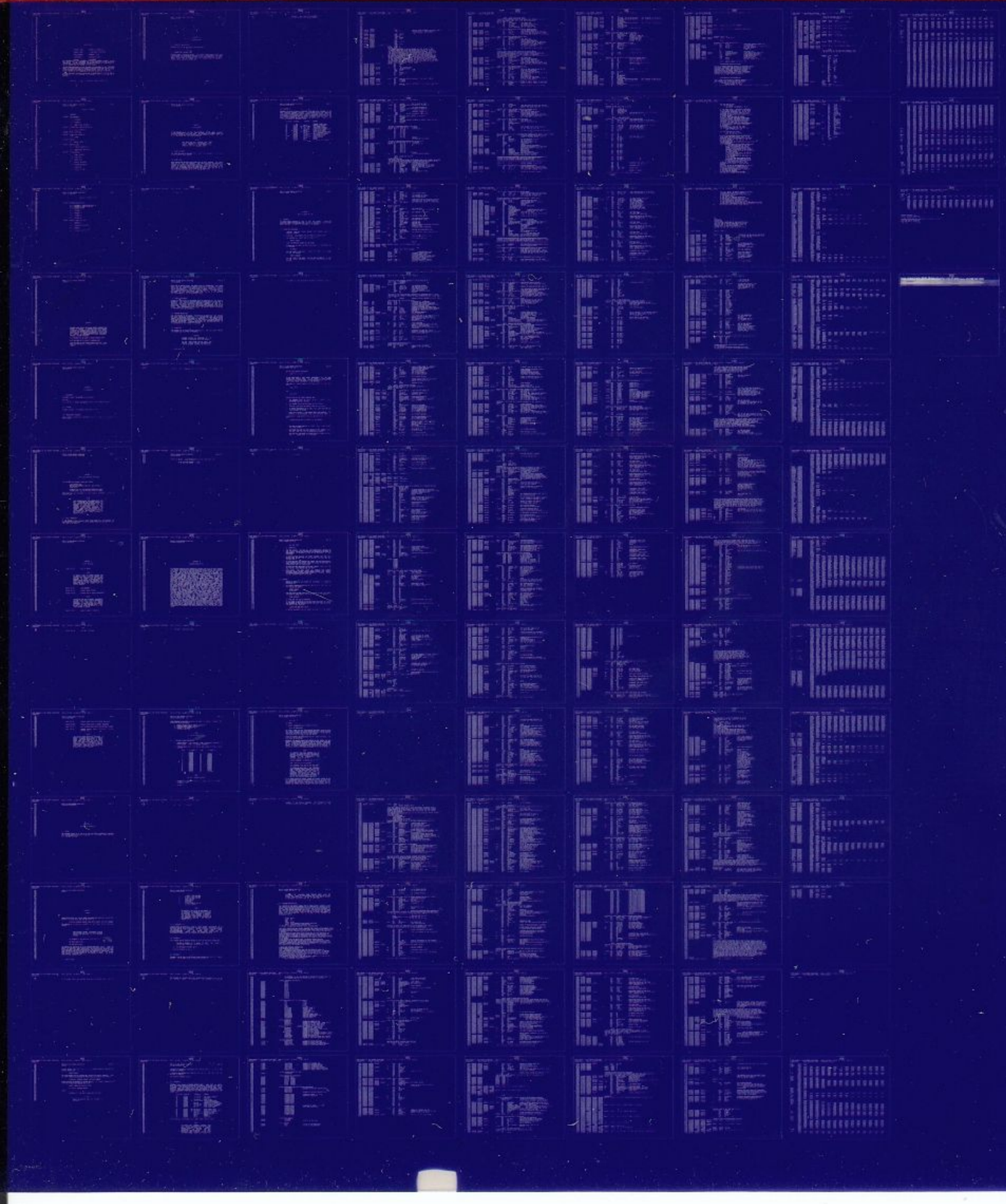
EP DZQMB G DL A

OCT. 1976

FICHE 1 OF 1

digital

Made In U.S.A.



```
                                .REM    %
```

IDENTIFICATION

PRODUCT CODE:	MAINDEC-11-DZQMB-G-D
PRODUCT NAME:	D-124K MEMORY EXERCISER
DATE CREATED:	FEBRUARY, 1974
MAINTAINER:	DIAGNOSTIC GROUP
AUTHOR:	JOHN ADAMS

PDP-11 0-124K MEMORY EXERCISER PAGE 3
TABLE OF CONTENTS

TABLE OF CONTENTS

PDP-11 0-124K MEMORY EXERCISER PAGE 4
TABLE OF CONTENTS

TABLE OF CONTENTS (CONT'D)

ABSTRACT

PROGRAM DZQMB TESTS CONTIGUOUS MEMORY ADDRESS FROM 000000 TO 757776. IT VERIFIES THAT EACH ADDRESS IS UNIQUE (AN ADDRESS TEST) AND THAT EACH MEMORY LOCATION CAN BE READ/WRITTEN RELIABLY (WORST CASE NOISE TESTS). IF MEMORY MANAGEMENT IS AVAILABLE, ALL TESTING IS PERFORMED WITH MEMORY MANAGEMENT ENABLED, (UNLESS DISABLED).

THIS PROGRAM MAY BE USED TO ADJUST/MARGIN MEMORY.

ALSO INCLUDED IS A TOGGLE IN ADDRESS TEST.

ALSO INCLUDED IS THE BRANCH GOBBLE MOS TEST. NOTE THAT ONLY SECTIONS 9.1 THROUGH 9.4 APPLY TO BRANCH GOBBLE.

CHAPTER 1
REQUIREMENTS

1.1 EQUIPMENT

THE PDP-11 FAMILY PROCESSOR WITH 8K MEMORY.

OPTIONAL...

KT11-C OR KT11-D MEMORY MANAGEMENT OPTION OR MF11 PARITY OPTION.

1.2 STORAGE

PROGRAM STORAGE - THE PROGRAM USES MEMORY 0-17777.

1.3 PRELIMINARY PROGRAMS

IPDP-11 FAMILY INSTRUCTION EXERCISER (DCQKC OR DZQKC) KT11-C/KT11-D LOGIC TESTS.

CHAPTER 2

LOADING AND STARTING PROCEDURE

LOAD PROGRAM INTO MEMORY USING ABS LOADER.

LOAD ADDRESS 200
SET SW12 IN DESIRED POSITION (SEE CHAPTER 3).
PRESS START.

ASTERISK "*" WILL BE PRINTED AFTER EACH PASS.
"DZQMB DONE!" WILL BE PRINTED AFTER 8 PASSES.

PASS COUNT MAY BE MONITORED IN THE DISPLAY REGISTER (11/45) OR LOCATION 756.

NOTE

THIS PROGRAM SAVES THE LOADERS BOOT AND ABS. TO RESTORE THE LOADERS, RESTART AT 162. BEFORE RESTARTING INSURE THAT THE PROGRAM IS NOT RELOCATED. IF THE PROGRAM IS RELOCATED, THE PC WILL INDICATE WHICH BANK CONTAINS THE PROGRAM. NEXT START THE PROGRAM AT *+ 12354, WHERE * = BITS 13-15 OF THE PC. THE PROGRAM WILL RELOCATE BACK TO 0-4K AND HALT AT 176. PRESS CONTINUE TO RESUME TESTING.

2.1 ACT11 OPERATION

IF THE PROGRAM IS RUN IN QUICK VERIFY MODE UNDER ACT11 THE PROGRAM IS DONE AFTER THE FIRST PASS. ALSO THE PROGRAM DOES NOT RELOCATE TO TEST THE LOWER 4K OF MEMORY.

CHAPTER 3
SWITCH SETTINGS

SW15=1 OR UP HALT ON ERROR

NOTE

IF SW15=1 WHEN AN ERROR OCCURS THE PROGRAM WILL HALT, AND THE CORRECT DATA WILL NOT BE LOADED INTO THE FAILING ADDRESS. IF SW15 IS RAISED AFTER THE ERROR TYPEOUT BEGINS, THE PROGRAM WILL HALT WHEN THE TYPEOUT COMPLETES AND THE CORRECT DATA WILL BE LOADED INTO THE FAILING ADDRESS.

SW14=1 OR UP LOOP SUBTEST

SW13=1 OR UP INHIBIT ERROR TYPEOUT

SW12=1 OR UP INHIBIT USE OF MEMORY MANAGEMENT

NOTE

INHIBITING THE USE OF MEMORY MANAGEMENT CAN BE DONE ONLY WHEN THE PROGRAM IS STARTED. IF THE USE OF MEMORY MANAGEMENT IS INHIBITED THE LAST ADDRESS AS TYPED BY THE PROGRAM WILL ONLY REFLECT THE AMOUNT OF MEMORY UP TO 28K (LAST ADDRESS = 160000).

SW11=1 OR UP INHIBIT SUBTEST ITERATION

```
382
383                                 SW10=1 OR UP        RING BELL ON ERROR
```

SWITCH SETTINGS

SW9=1 OR UP	DISPLAY ERROR COUNT IN DISPLAY REGISTER
SW9=0 OR DOWN	DISPLAY PASS COUNT IN DISPLAY REGISTER
SW8=1 OR UP	HALT PROGRAM UNRELOCATED AND RESTORE LOADERS
SW5=1 OR UP	INHIBIT PARITY ERROR DETECTION (INITIAL STARTUP ONLY)

NOTE

WITH PARITY ERROR DETECTION ENABLED, A MEMORY FAILURE WILL CAUSE A PARITY ERROR. THE ERROR PRINTOUT ON A PARITY ERROR DOES NOT TYPE THE GOOD DATA. THUS, A BIT DROP OR PICKUP WILL NOT BE TYPED AS SUCH. IT IS BEST TO RUN THE PROGRAM FOR 1 PASS (UNTIL AN * IS TYPED) WITH PARITY DISABLED, THEN RESTART THE PROGRAM WITH PARITY ENABLED.

CHAPTER 4

SUBROUTINE ABSTRACTS

4.1 SCOPE

THE PROGRAM STORES IN R1 THE PC OF THE LAST TEST SUCCESSFULLY EXECUTED AND MAY BE USED AS AN AID IN DEBUGGING IF THE PROGRAM 'BOMBS' BECAUSE OF A HARDWARE FAILURE.

CHAPTER 5

ERRORS

THESE TESTS PRINT OUT THE PC WHERE THE ERROR WAS DETECTED, THE FAILING ADDRESS, THE GOOD DATA, AND THE BAD DATA I.E.

PC=XXXXXX ADDRESS AAAAAA GOOD DATA GGGGGG BAD DATA BBBBBB

THE ADDRESS OF THE FAILING LOCATION IS THE TRUE 18 BIT PHYSICAL ADDRESS.

NOTE

WHEN TESTING MEMORY LOCATIONS 0-17776 THE PC TYPED WILL BE A MULTIPLE OF 20000 GREATER THAN REFLECTED IN THE PROGRAM LISTING.

THE ADDRESS OF THE BAD DATA IS IN (R2) -2

THE GOOD DATA IN R0

THE BAD DATA IN R3

THE ADDRESS OF GOOD DATA IS IN R4 (RANDOM DATA TEST ONLY). WHEN AN ERROR IS DETECTED WHEN EXERCISING THE MEMORY USING THE WORST CASE NOISE PATTERNS, THE USER SHOULD RESTART THE PROGRAM SELECTING PROGRAM (SEE CHAPTER 8 FOR DETAILS) SELECTING THE APPROPRIATE PARAMETERS. THE USER CAN USE THE PC AND ADDRESS OF THE FAILURE TO SELECT THE PROPER CORE BANK(S) AFFECTED AND ALSO THE SPECIFIC PATTERN. THIS ALLOWS MAXIMUM SCOPE CAPABILITIES.

6.1 PARITY ERROR

```
  492
  493                         IF THE MEMORY PARITY OPTIONS ARE INSTALLED THE PROGRAM RUNS  WITH  THE
```

ACTION ENABLE BIT SET (BIT 0). IF A PARITY ERROR IS DETECTED THE PROGRAM WILL TYPE:

PARITY ERROR

AND SCAN MEMORY FOR THE ADDRESS(ES) CAUSING THE PARITY ERROR(S). WHEN THE PARITY ERROR IS DETECTED AN ERROR WILL BE TYPED AS SHOWN BELOW:

PC=XXXXXX ADDRESS AAAAAA BAD DATA BBBBBB

PRESS CONTINUE OR RESTART TO RESUME TESTING. IF A PARITY ERROR IS NOT DETECTED ON SCAN THE PROGRAM WILL TYPE:

PARITY ERROR NOT FOUND ON SCAN

PC=XXXXXX ADDRESS=AAAAAA

WHERE:

AAAAAA=PC AT TIME PARITY ERROR WAS DETECTED.

NOTE

PARITY IS DISABLED WHEN THE PROGRAM IS RELOCATED.

CHAPTER 6
RESTRICTIONS

7.1 STARTING RESTRICTION

PROGRAM MUST NOT BE RELOCATED WHEN RESTARTING.

7.2 OPERATIONAL RESTRICTION

PROGRAM CHECKS CONTIGUOUS MEMORY. IF A PARITY ERROR TRAP OCCURS WHEN THE PROGRAM IS RELOCATED PROGRAM ACTION IS UNDEFINED. IF PARITY MEMORY IS AVAILABLE OR SELECTED THE 3 XOR 9 TEST PATTERN IS FOR PARITY MEMORY ONLY. DO NOT POWER FAIL THE PROGRAM WHEN THE PROGRAM IS RUNNING IN MOS MEMORY OR RELOCATED.

CHAPTER 7

MISCELLANEOUS

IF THE PROGRAM HALTS IN THE TRAP/INTERRUPT VECTOR AREA (0-1000), EXAMINE REGISTER 6 (THE STACK PTR). R6 CONTAINS THE ADDRESS WHERE THE PC OF THE INSTRUCTION THAT CAUSED THE TRAP ABORT IS STORED. SEE ALSO R1 (R1 SPECIFIES THE LAST TEST COMPLETED).

NOTE

THE PDP11/45 WILL DISPLAY THE TRAP VECTOR ADDRESS+4 IN THE ADDRESS LIGHTS. THUS, A TRAP TO 4 (BUS ERROR) WILL DISPLAY 10 IN THE ADDRESS LIGHTS.

7.1 STACK POINTER

THE STACK POINTER IS INITIALLY SET TO 500. AND IS RESET TO THIS VALUE AT THE START OF EACH SUBTEST.

7.2 PASS COUNT

SEVEN PASSES ARE REQUIRED FOR COMPLETION OF THIS PROGRAM; AT WHICH TIME AN "*" WILL BE PRINTED. THE PASS COUNT MAY BE OBSERVED BY TURNING THE SWITCH TO THE DISPLAY POSITION. (THE PASS COUNT IS ALSO STORED IN LOCATION !000.) THE PASS COUNT SHOULD BE MONITORED IN THE EVENT THAT THE PROGRAM ENTERS AN UNDEFINED LOOP. NOTE THAT BIT 15 OF THE DISPLAY REGISTER IS NOT PART OF THE PASS COUNT. BIT 15, IF ON, INDICATES THAT THE PROGRAM IS IN ITS RELOCATED CYCLE.

615 7.3 ERROR COUNT

EACH TIME AN ERROR OCCURS, THE ERROR COUNT IS INCREMENTED. THE ERROR COUNT CAN BE OBSERVED BY TURNING THE SWITCH TO THE DISPLAY POSITION AND SETTING SWITCH 9. (THE ERROR COUNT IS ALSO STORED IN LOCATION 1002.) THE PROGRAM WILL COUNT 17777(OCTAL) ERRORS; THE ERROR COUNT IS NOT INCREMENTED PAST THIS VALUE. NOTE THAT BIT 15, OF THE DISPLAY REGISTER, IS NOT PART OF THE ERROR COUNT. BIT 15, IF ON, INDICATES THAT THE PROGRAM IS IN ITS RELOCATED CYCLE.

7.4 DISPLAY REGISTER

EITHER THE PASS COUNT OR THE ERROR COUNT IS DISPLAYED IN THE DISPLAY REGISTER. THE COUNT TO BE DISPLAYED IS CONTROLLED BY THE SETTING OF SWITCH 9. BIT 15 OF THE DISPLAY REGISTER, HOWEVER, IS USED AS A RELOCATION INDICATOR AND IS NOT PART OF EITHER THE PASS COUNT OR THE ERROR COUNT. WHEN BIT 15 IS ON, THE PROGRAM IS PERFORMING A RELOCATED CYCLE. WHEN THE PROGRAM IS RELOCATED, THE SPECIAL RESTART PROCEDURES OF CHAPTER 2 MUST BE FOLLOWED.

7.5 PROGRAM RELOCATION

WHEN THE PROGRAM IS RELOCATED, VERIFICATION IS MADE THAT THE PROGRAM HAS BEEN RELOCATED CORRECTLY. IF THE PROGRAM CANNOT BE RELOCATED UPWARD, THE RELOCATED TEST PHASE IS BYPASSED. IF AN ERROR OCCURS WHILE RELOCATING THE PROGRAM BACK TO THE LOWER 4K, AN ERROR MESSAGE IS TYPED AND THE PROGRAM HALTS. CONTINUING THE PROGRAM RETRIES THE DOWNWARD RELOCATION. DOWNWARD RELOCATION WILL BE ATTEMPTED UNTIL IT IS SUCCESSFUL OR THE PROGRAM IS RELOADED.

7.6 POWER FAIL

THE PROGRAM MAY BE POWER FAILED WHEN RUNNING. WHEN THE POWER RETURNS THE PROGRAM WILL CONTINUE IN SEQUENCE.

CAUTION

PROGRAM ACTION IS UNDEFINED IF THE PROGRAM IS RELOCATED OR IN MOS MEMORY.

DO NOT TURN POWER OFF/ON UNTIL THE MESSAGE 'POWER FAILED' HAS BEEN TYPED. THIS IS BECAUSE THE STACK MAY OVERFLOW.

7.7 EXECUTION TIME

```
672
673                             EXECUTION TIME IS DEPENDENT ON TYPE OF PROCESSER, TYPE OF MEMORY,  AND
```

```
                        PDP-11 0-124K MEMORY EXERCISER                  PAGE 16
                        MISCELLANEOUS


                        AMOUNT OF MEMORY.  SOME REPRESENTATIVE TIMES (PER PASS) ARE:

                                 11/05 WITH 28K MEMORY - 1 MIN.
                                 11/45 WITH 96K MEMORY - 3 MIN.
```

CHAPTER 8

PROGRAM DESCRIPTION

THE PROGRAM VERIFIES EACH ADDRESS BY WRITING THE VALUE OF EACH ADDRESS INTO ITSELF STARTING AT LOCATION 20000 AND ENDING AT THE LAST LOCATION IN MEMORY. THE VALUE OF THE LAST LOCATION +2 IS TYPED ON THE TTY. NEXT THE VALUES WRITTEN ARE VERIFIED. TO COMPLETE THE ADDRESS TEST THE COMPLEMENT VALUE OF EACH MEMORY ADDRESS IS WRITTEN STARTING AT THE LAST MEMORY ADDRESS AND ENDING AT ADDRESS 20000. THE WRITTEN COMPLEMENT VALUES ARE THEN VERIFIED. THE NEXT PHASE OF TESTING INCLUDES READING, WRITING AND CHECKING MEMORY USING SEVERAL WORST CASE NOISE TEST PATTERNS (1 XOR 8, 3 XOR 9, AND 8 XOR 13). A SUBTEST IS DEDICATED TO CHECKING EACH PATTERN. THE TEST PROCEEDS BY EXERCISING EACH BANK OF MEMORY USING THE TEST PATTERNS NOTED ABOVE. NOTE THAT WITH THE MEMORY MANAGEMENT OPTION INSTALLED THAT ALL ADDRESSES ARE WRITTEN, READ AND CHECKED WITH THE MEMORY MANAGEMENT ENABLED. AFTER ALL MEMORY FROM 20000 TO THE LAST ADDRESS HAS BEEN TESTED, THE PROGRAM RELOCATES TO THE NEXT 4K MEMORY BANK AND TESTS LOCATIONS 0-17776 USING (1 XOR 8). THE PROGRAM THEN RELOCATES TO 40000 (100000 IF AVAILABLE) AND CHECKS MEMORY USING 3 XOR 9, AND 8 XOR 13 TEST PATTERN. THE PROGRAM THEN CHECKS MEMORY USING RANDOM DATA (RANTST). THIS ROUTINE MOVES THE PROGRAM CODE THROUGHOUT MEMORY STARTING AT LOCATION 20000, AND RELOCATES THE DATA BY A 32(DECIMAL) WORD OFFSET ON EACH SUBSEQUENT RELOCATION. I.E., FIRST RELOCATION IS TO 20000, NEXT IS TO 20100, THEN 20200, ETC. AFTER RELOCATION THE CODE MOVED IS CHECKED AGAINST THE ORIGINAL CODE (0-17776). WHEN THE RANDOM DATA TEST IS COMPLETE THE PROGRAM THEN SUCCESSIVELY ROTATES A '0' BIT (ROT0) AND A '1' BIT (ROT1) THROUGH ALL OF MEMORY. WHEN ALL TESTING IS COMPLETE THE PROGRAM RELOCATES TO ITS ORIGINAL POSITION, INCREMENTS THE PASS COUNT (LOCATION 1000) AND RESTARTS BEGINNING WITH THE WORST CASE NOISE TESTS. AN ASTERISK (*) WILL BE TYPED ON COMPLETION OF EACH PASS, AND WHEN 8 PASSES HAVE BEEN COMPLETED THE PROGRAM WILL TYPE 'DZQMB DONE' AND RESTART THE PROGRAM BEGINNING WITH THE MEMORY ADDRESS TESTS.

```
   738                                  8.1  PROGRAM 2 (USER SELECTIONS)
```

PROGRAM DESCRIPTION

THIS PROGRAM IS PROVIDED TO ALLOW THE USER TO SPECIFY CERTAIN TEST PARAMETERS AS SHOWN BELOW:

1. ENABLE/DISABLE PARITY ERROR INTERRUPTS
2. STARTING BANK NUMBER FOR TEST
3. NUMBER OF 4K BANKS TO TEST
4. PATTERN TO BE USED

NOTE

ALL INPUTS ARE IN OCTAL.

8.1.1 PROGRAM 2 USER PARAMETERS

1. ENABLE PARITY? 1/0 = YES/NO. TYPE 1 TO ENABLE INTERRUPT ON PARITY ERROR. TYPE 0 TO DISABLE INTERRUPT.

2. STARTING BANK #(8)? TYPE THE 4K BANK WHERE YOU WISH TO BEGIN TESTING.

TYPE	TO START AT	TYPE	TO START AT
0	000000		
1	020000	20	400000
2	040000	21	420000
3	060000	22	440000
4	100000	23	460000
5	120000	24	500000
6	140000	25	520000
7	160000	26	540000
10	200000	27	560000
11	220000	30	600000
12	240000	31	620000
13	260000	32	640000
14	300000	33	660000
15	320000	34	700000
16	340000	35	720000
17	360000	36	740000

NOTE

TYPE ONLY NUMBERS SHOWN!!!

3. INUMBER OF 4K BANKS TO TEST (8)? TYPE IN OCTAL THE NUMBER OF 4K BANKS TO TEST.

4. PATTERN #?

```
795
796                                          TYPE      TO SELECT
```

0	1 XOR 8 TEST PATTERN
1	3 XOR 9 TEST PATTERN
2	8 XOR 13 TEST PATTERN
3	USER CONSTANT
4	ROTATING 0
5	ROTATING 1
6	3 XOR 9 PARITY PATTERN
7	0,1,2,4,5 ABOVE

NOTE

PROGRAM WILL NOT ALLOW AN ODD NUMBER OF 4K BANKS TO BE TESTED IF PATTERN 2 OR 7 IS SELECTED. IF PATTERN #3 IS SELECTED THE PROGRAM WILL REQUEST A CONSTANT. TYPE A 6 DIGIT OCTAL NUMBER, TO ENTER A NEW CONSTANT TYPE AN 'A' AND WAIT FOR THE PROGRAM TO RESPOND. THE STARTING ADDRESS IS 214.

8.1.2 PROGRAM 2 USE

PROGRAM 2 CAN BE EFFECTIVELY USED TO MAKE PROPER ADJUSTMENTS TO A SPECIFIC MEMORY BANK AND ALSO TO 'MARGIN' MEMORY. THIS IS SO BECAUSE THE PROGRAM IS NOT RUNNING IN THE MEMORY BANK(S) BEING ADJUSTED/MARGINED. THUS ALL MEMORY FROM 0-124K MAY BE ADJUSTED/MARGINED. PARITY SHOULD BE DESELECTED WHEN MAKING ANY ADJUSTMENTS PARTICULARLY WHEN TESTING THE FIRST 4K BANK(S).

8.2 PROGRAM 3

THIS PROGRAM IS THE SAME AS PROGRAM 2 WITH THE FOLLOWING EXCEPTIONS:

1. INSTEAD OF NUMBER OF 4K BANKS TO TEST, TYPE NUMBER OF 256(DECIMAL),400(OCTAL) WORD BLOCKS TO TEST.

2. DO NOT SELECT PATTERN 2 OR 7.

E STARTING ADDRESS IS 220.

8.3 PROGRAM 4

PROGRAM 4 CAN BE USED TO WRITE/READ USER DEFINED DATA INTO ANY SINGLE ADDRESS. THE PROGRAM WRITES THE DATA AND CHECKS IT.

```
  853                     THE PROGRAM WILL REQUEST AN 18 BIT ADDRESS AND IF SWITCH 0 = 0,  A  16
  854                     BIT  CONSTANT  (DATA).   IF  SWITCH 0 = 1  THE  PROGRAM  WILL TYPE THE
```

CONTENTS OF SEQUENTIAL ADDRESSES UNTIL EITHER SWITCH 0 = 0 OR A NEW ADDRESS IS ENTERED.

TO ENTER A NEW ADDRESS AND CONSTANT TYPE AN 'A' AND WAIT FOR THE PROGRAM TO RESPOND.

THE STARTING ADDRESS IS 224.

8.4 PROGRAM 5

PROGRAM 5 IS A TOGGLE IN MEMORY ADDRESS TEST. THIS TEST IS USEFUL WHEN AN ADDRESS SELECTION FAILURE IS SUSPECTED INVOLVING THE FIRST 4K OF MEMORY. THIS PROGRAM WRITES THE VALUE OF EACH ADDRESS INTO ITSELF STARTING WITH THE LOWER LIMIT AND CONTINUING TO THE UPPER LIMIT. AFTER ALL ADDRESSES HAVE BEEN WRITTEN EACH ADDRESS IS CHECKED FOR THE CORRECT CONTENTS STARTING WITH THE UPPER LIMIT AND CONTINUING TO THE LOWER LIMIT.

```
        LOCATION  CONTENTS      MNEMONIC     COMMENT

          10      012700          MOV #50,R0   ;GET FIRST ADDRESS
        * 12      000050                       ;TO TEST
          14      010001          MOV R0,R1    ;SAVE IN R1
          16      020037    1$:   CMP R0,@#SWR ;CHECK UPPER LIMIT
          20      177570                       ;(IN SWITCH REGISTER)
          22      001403          BEQ 2$       ;BRANCH IF AT UPPER LIMIT
          24      010010          MOV R0,(R0)  ;LOAD VALUE INTO ADDRESS
          26      005720          TST (R0)+    ;STEP TO NEXT ADDRESS
          30      000772          BR 1$        ;LOOP UNTIL DONE

          32      010004    2$:   MOV R0,R4    ;SAVE UPPER LIMIT
          34      020001    3$:   CMP R0,R1    ;CHECK IF AT LOWER LIMIT
        * 36      001767          BEQ 1$       ;BRANCH IF DONE
        * 40      024000          CMP -(R0),R0 ;CHECK DATA WRITTEN
          42      001774          BEQ 3$       ;BRANCH IF OK
          44      000000          HALT         ;ERROR
          46      000772          BR 3$        ;LOOP BACK
```

AFTER TOGGLING THE PROGRAM LA=10 **SET UPPER LIMIT**, START.

NOTE

THE UPPER LIMIT ADDRESS OBTAINED FROM THE SWITCH REGISTER MAY BE CHANGED DURING PROGRAM OPERATION. HOWEVER, OCCASIONALLY THE PROGRAM MAY HALT BECAUSE OF 'SWITCH BOUNCE'. (THE BEST PROCEDURE WHEN CHANGING LIMITS IS TO STOP THE PROGRAM MAKE THE CHANGE AND

```
911                                     CONTINUE.)  THE LOWER LIMIT ADDRESS (12)
912                                     MAY BE PATCHED TO ANY DESIRED ADDRESS.
```

8.5 PROGRAM 6

PROGRAM 6 IS ALSO A TOGGLE IN PROGRAM TO BE USED WITH PROGRAM 5 FOR MORE COMPLETE ADDRESS TESTING. THIS PROGRAM WRITES THE COMPLEMENT VALUE OF EACH ADDRESS INTO ITSELF STARTING WITH THE UPPER LIMIT AND CONTINUING TO THE LOWER LIMIT. AFTER ALL ADDRESSES HAVE BEEN WRITTEN EACH ADDRESS IS CHECKED FOR THE CORRECT CONTENTS STARTING WITH THE LOWER LIMIT ADDRESS AND CONTINUING TO THE UPPER LIMIT. TOGGLE IN THE FOLLOWING PATCHES TO PROGRAM 5 ABOVE.

LOCATION	CONTENTS		MNEMONIC	COMMENT
12	100			;CHANGE LOWER LIMIT
36	001404		BEQ 4$	;BRANCH TO PROGRAM 6
50	010402	4$:	MOV R4,R2	;GET UPPER LIMIT
52	005142	5$:	COM -(R2)	;COMPLEMENT ADDRESS
54	020201		CMP R2,R1	;CHECK IF AT LOWER LIMIT
56	001375		BNE 5$	;LOOP UNTIL DONE
60	020204	6$:	CMP R2,R4	;CHECK IF AT UPPER LIMIT
62	001755		BEQ 1$	;GO TO PROGRAM 5 IF DONE
64	010203		MOV R2,R3	;GET VALUE OF ADDRESS
66	005103		COM R3	;COMPLEMENT VALUE
70	020322		CMP R3,(R2)+	;CHECK ADDRESS
72	001772		BEQ 6$	;BRANCH IF OK
74	000000		HALT	;ERROR
76	000770		BR 6$	;GO CHECK NEXT ADDRESS

CHAPTER 9

BRANCH GOBBLE MOS TEST

9.1 ABSTRACT

THE BRANCHGOBBLE PROGRAM IS USED TO TEST MOS MEMORY. CONTIGUOUS LOCATIONS ARE TESTED BETWEEN TWO LIMITS IN A MINIMUM 8K, MAXIMUM 124K MEMORY MACHINE. IF PARITY IS AVAILABLE IT IS ENABLED.

9.2 OPERATING PROCEDURE

1. LOADING: LOAD THE DZQMBG PROGRAM INTO MEMORY USING THE ABSOLUTE LOADERS.

2. STARTING: LOAD ADDRESS 270 AND PRESS THE START BUTTON.

3. THE PROGRAM WILL FIRST IDENTIFY ITSELF ON TTY:

   BRANCH GOBBLE

4. THEN THE ABSOLUTE LOADER WILL BE SAVED.

5. A CHECK WILL BE MADE FOR PARITY REGISTERS. IF NONE ARE FOUND THE MESSAGE:

   NO PARITY

   WILL BE TYPED TO THE USER. IF PARITY IS FOUND IT IS TURNED ON, AND THE MESSAGE,

   PARITY ENABLED

   WILL BE TYPED TO THE USER. THIS WILL BE FOLLOWED BY A LIST OF THE UNIBUS ADDRESSES OF THE PARITY REGISTERS FOUND AND ENABLED.

```
  999
 1000                    6.  THE USER WILL THEN BE ASKED IF HE WANTS THE PENDING  TEST  TO
```

BE RUN USING MEMORY MANAGEMENT.

USE KT11? (Y OR N)
>

IF THE USER TYPES Y THEN MEMORY MANAGEMENT WILL BE USED DURING THE PENDING TEST. IF HE TYPES N, THEN MEMORY MANAGEMENT WILL NOT BE USED. TYPING ANYTHING ELSE OTHER THAN Y OR N WILL CAUSE THE QUESTION TO BE REPEATED.

7. THE USER WILL THEN BE ASKED TO GIVE THE LIMITS OF THE TEST SPAN:

HIGH LIMIT?
>

AND:

LOW LIMIT?
>

RESTRICTIONS ON THE USER'S RESPONSE ARE:

A. THE NUMBERS MUST BE VALID (18-BIT) 6-DIGIT OCTAL ADDRESSES, REAL NOT VIRTUAL.

B. THE NUMBERS SHOULD BE MULTIPLES OF 100 (OCTAL).

C. THE HIGH LIMIT MUST BE GREATER THAN THE LOW LIMIT.

D. IF MEMORY MANAGEMENT IS NOT USED, HIGH LIMIT MUST BE LESS THAN OR EQUAL TO 160000.

E. HIGH LIMIT CAN BE 1 + THE HIGHEST REAL CORE ADDRESS. FOR EXAMPLE, IN AN 8K MACHINE, HIGH LIMIT CAN EQUAL 40000.

VIOLATIONS TO THESE RESTRICTIONS WILL BE DEALT WITH IN THIS WAY:

A. A QUESTION MARK AND THE PROMPT WILL BE ISSUED:

?
>

THE USER IS THEN EXPECTED TO INPUT THAT LIMIT AGAIN; THIS TIME CORRECTLY.

B. WHAT EVER THE LAST TWO OCTAL DIGITS OF THE NUMBER WHICH THE USER TYPED THEY WILL BE ASSEMBLED AS ZEROES.

C. THE USER WILL BE ASKED FOR OTHER LIMITS BY REPEATING THIS STEP (7). BEFORE STEP (7) IS REPEATED

```
1057
1058                                    NOT VALID!
```

WILL BE TYPED.

D. SAME AS 3.

8. THE TEST STARTS. THE TEST WILL LOOP INDEFINITELY BETWEEN THE TWO LIMITS UNLESS THE USER HALTS THE PROGRAM OR AN ERROR IS ENCOUNTERED. AN ASTERISK IS TYPED AT THE BEGINNING OF EACH PASS MODE.

9. TO STOP THE TEST RUNNING AND START ANOTHER HIT THE HALT SWITCH AND RETURN STEP 2. ANY TEST BUT THE FIRST WILL NOT INCLUDE STEP 4.

10. TO STOP THE TEST AND RESTORE THE LOADER, HIT THE HALT SWITCH, LOAD ADDRESS 162 AND START. WHEN THE LOADER IS RESTORED THE PROGRAM WILL HALT AT LOCATION 200.

11. TO STOP THE TEST AND START THE 0-124K MEMORY TEST HIT THE HALT SWITCH, LOAD ADDRESS 200, AND START.

12. DATA LIGHTS. THE DATA LIGHTS WILL DISPLAY THE CURRENT LOCATION BEING TESTED DURING A BRANCH GOBBLE TEST. WHEN A MOS MEMORY FAILURE OCCURS THESE LIGHTS WILL CONTAIN VIRTUAL (16-BIT) ADDRESS "NEAR" THE FAILURE.

9.3 ERRORS

1. ERRORS IN OPERATING THE PROGRAM ARE DESCRIBED IN OPERATING PROCEDURE 9.2.

2. IF A PARITY ERROR IS DETECTED THE USER IS TOLD THE PC+2 AT THET IME OF THE ERROR:

```
PARITY ERROR
PC=XXXXXX
```

THEN THE SCAN IS MADE THROUGH ALL OF MEMORY TO TRY TO FORCE THE ERROR TO ARISE AGAIN. IF IT IS NOT FOUND THE MESSAGE

```
SCAN COMPLETE
```

IS TYPED AND THE TEST IS RESTARTED.

IF THE ERROR IS DETECTED ON THE SCAN THEN THE USER IS GIVEN THE ADDRESS OF THE LOCATION CAUSING THE PARITY ERROR AND THE CONTENTS OF THAT LOCATION:

```
XXXXXX HAD BAD DATA XXXXXX
```

IF MEMORY MANAGEMENT WAS OFF DURING THE SCAN FOR THE ERROR

1115 THE ADDRESS GIVEN FOR THE ERROR IS REAL AND:

KT11 OFF

IS TYPED.

IF MEMORY MANAGEMENT WAS ON DURING THE SCAN:

KT11 ON PAR=XXXXXX

IS TYPED, WHERE THE PAR (PAGE ADDRESS REGISTER) GIVEN IS THAT PAR WHICH SHOULD BE USED IN RELOCATING THE VIRTUAL ADDRESS GIVEN FOR THE ERROR ONTO A REAL CORE ADDRESS. THE METHOD FOR THIS RELOCATION IS GIVEN IN THE NOTE BELOW.

AFTER ANY PARITY IS ENCOUNTERED AND THE USER NOTIFIED, THE TEST WILL BE RESTARTED.

3. WHENEVER THE BRANCH GOBBLE TEST BRINGS OUT AN ERROR IN MOS MEMORY IT MAY SURFACE AS A PARITY AND BE HANDLED AS DESCRIBED ABOVE. OTHERWISE THE ADDRESS (VIRTUAL IF MEMORY MANAGEMENT IS ON) IN THE DATA LIGHTS WILL DESIGNATE THE VICINITY OF THE ERROR. IF MEMORY MANAGEMENT IS ON, RELOCATE THE ADDRESS IN THE DATA LIGHTS IN THE MANNER DESCRIBED IN THE NOTE BELOW.

NOTE

TO COMPUTE THE REAL ADDRESS OF AN ADDRESS RELOCATED BY MEMORY MANAGEMENT, ADD THE LOW ORDER 13-BITS OF THE VIRTUAL ADDRESS TO THE CORRESPONDING PAR SHIFTED, 6 BITS TO THE LEFT:

VIRTUAL ADDRESS = 0 00X XXX XXX XXX XXX

PAR = YYY YYY YYY YYY 000 000

REAL ADDRESS = ZZZ ZZZ ZZZ ZZZ ZZZ ZZZ

TO DETERMINE WHICH PAR TO USE REMEMBER THAT ON KERNEL SPACE IS USED IN ANY TEST HERE. TAKE THE HIGH ORDER 3-BITS OF THE VIRTUAL ADDRESS AND USE THEM TO DESIGNATE THE KIPAR TO USE. FOR INSTANCE, IF THE VIRTUAL ADDRESS IS 031676 USE KIPAR1 BECAUSE THE UPPER 3 BITS OF THE VIRTUAL ADDRESS ARE 001=1.

4. IF AN ERROR CONDITION ARISES, EITHER AS A PARITY ERROR OR ONE NOT A APPARENT SUCH AS THE PROGRAM HALTS OR IT IS CLEAR FROM THE ADDRESS AND DISPLAY LIGHTS THAT THE PROGRAM IS NOT RUNNING ITS NORMAL COURSE, THE USER CAN ENTER CONSOLE MODE AN EXAMINE THE CONTENTS OF THE MEMORY LOCATIONS STARTING AT THE

```
1172                                    ADDRESS  IN  THE  DISPLAY  REGISTER.   THE  CONTENTS OF THESE
1173                                    LOCATIONS  SHOULD  BE  COMPARED  TO  THE  CONTENTS  (IN   THE
```

LISTINGS) OF LOCATIONS 15226 THROUGH 15304 (WITH THE EXCEPTIONS: 15232 IS UNDETERMINABLE AND 15304 SHOULD CONTAIN EITHER 15370 OR 35370). IN THIS WAY THE USER SHOULD BE ABLE TO DETERMINE WHICH BITS WERE LOST IN WHAT WORDS.

9.4 PROGRAM DESCRIPTION

THIS VERSION OF THE BRANCH GOBBLE TEST IS TAKEN ALMOST DIRECTLY FROM THE DZQKA-A INSTRUCTION EXERCISER WHICH CONTAINED THE ORIGINAL BRANCH GOBBLE. WHAT HAS BEEN DONE HERE IS TO GIVE THAT TEST AN INTERFACE TO THE USER AND MEMORY MANAGEMENT FACILITIES. THESE ADDITIONS HAVE BEEN DONE IN A WAY WHICH ALLOWS THE TEST TO RUN AS IT DID IN ITS ORIGINAL FORM. DATA IS COLLECTED FROM THE USER AND IF MEMORY MANAGEMENT IS NEEDED IT IS SET UP AND THAT TEST IS ALLOWED TO RUN BETWEEN THE DESIGNATED LIMITS.

```
        %
        .NLIST  MD,MC
        .LIST   ME
        .ABS
        .MCALL  $TYPE
        .TITLE  TEST DZQMB-G 0-124K MEMORY EXERCISER
        .SBTTL  STARTING INST & DEFINITIONS
;COPYRIGHT 1973 DIGITAL EQUIPMENT CORP., MAYNARD,MASS.

;THIS TEST CHECKS THAT ALL MEMORY ADDRESSES ARE UNIQUE USING ADDRESS TESTS
;AND CHECKS DATA RELIABILITY OF MEMORY USING WORST CASE NOISE TEST PATTERNS
;A RANDOM # PATTERN (PROGRAM CODE RELOCATED), A ROTATING 0 AND ROTATING
;1 PATTERN.
;ALSO INCLUDED ARE USER TESTS WHICH CAN BE USED TO TEST SPECIFIED SEG-
;MENTS OF MEMORY USING THE PATTERNS MENTIONED ABOVE. ADDITIONALLY A
;28 WORD TOGGLE IN PROGRAM IS DOCUMENTED (SEC 9.5 OF THE DOCUMENT) WHICH
;CAN BE USED IF AN ADDRESSING MALFUNCTION IS SUSPECTED INVOLVING THE FIRST
;4K OF MEMORY.
;A MOS MEMORY TEST HAS BEEN ADDED, THE BRANCH GOBBLE ROUTINE.
;THE PROGRAM MAY BE POWER FAILED WHEN RUNNING. THE PROGRAM WILL PRINT
;A MESSEGE (POWER FAILED) AND CONTINUE IN SEQUENCE WHEN THE POWER COMES
;BACK UP. **CAUTION** DO NOT POWER FAIL THE PROGRAM IF THE PROGRAM IS IN
;MOS MEMORY OR IF THE PROGRAM IS RELOCATED.

;LOADING AND STARING INSTRUCTIONS
;LOAD ADDRESS 200 AND START
;NOTE:  PROGRAM WILL RUN WORST CASE TEST PATTERNS IN LOWEST 4K
;THUS THE PROGRAM CANNOT BE RESTARTED AT 200 IF RELOCATED. TO PREVENT
;RELOCATION FROM OCCURING DEPOSIT 200 INTO LOCATION 42 (NOT NECESSARY
;IF LOADED VIA ACT11). THIS ACTION WILL PREVENT RELOCATION AND ALSO
;INHIBIT TESTING MEMORY IN LOWEST 4K.
;THIS PROGRAM ALSO RELOCATES THE ABS AND BOOT LOADERS TO ALLOW TESTING
;OF MEMORY, TO RESTORE THE LOADERS RESTART AT 162.
;       STACK POINTER IS SET AT 500
```

```
1230                            ;       AN ASTERISK '*' WILL BE PRINTED ON COMPLETION OF EACH PASS, AND
1231                            ;       THE PROGRAM NAME WILL BE PRINTED WHEN TEST IS COMPLETE.
1232
1233                            ;GENERAL REGISTER ASSIGNMENTS
1234    000000                          R0=%0
1235    000001                          R1=%1
1236    000002                          R2=%2
1237    000003                          R3=%3
1238    000004                          R4=%4
1239    000005                          R5=%5
1240    000006                          SP=%6
1241    000007                          PC=%7
1242    000000                          R10=%0
1243    000001                          R11=%1
1244    000002                          R12=%2
1245    000003                          R13=%3
1246    000004                          R14=%4
1247    000005                          R15=%5
1248
1249                            ;STATUS REGISTER (PSW) BIT ASSIGNMENTS
1250    000001                          C=1                     ;C BIT
1251    000002                          V=2                     ;V BIT
1252    000004                          Z=4                     ;Z BIT
1253    000010                          N=10                    ;N BIT
1254    000020                          T=20                    ;'T' BIT
1255    000340                          PRTY7=340               ;PRIORITY LEVEL 7
1256    000200                          PRTY4=200               ;PRIORITY LEVEL 4
1257    000000                          KM=000000               ;KERNEL MODE
1258    040000                          SM=040000               ;SUPERVISORY MODE
1259    140000                          UM=140000               ;USER MODE
1260    000000                          PKM=000000              ;PREVIOUS KERNEL MODE
1261    010000                          PSM=010000              ;PREVIOUS SUPERVISORY MODE
1262    030000                          PUM=030000              ;PREVIOUS USER MODE
1263    004000                          REG=004000              ;SELECT R10-R15
1264
1265                            ;VECTOR ADDRESSES
1266    000004                          ERRVEC=4                ;ADDRESS OF ERROR VECTOR
1267    000010                          RESVEC=10               ;ADDRESS OF RESERVED INST. TRAP VECTOR
1268    000014                          TBITVEC=14              ;ADDRESS OF 'T' BIT TRAP VECTOR
1269    000014                          TRTVEC=14               ;ADDRESS OF 'TRACE' TRAP VECTOR
1270    000014  BPTVEC=14                      ;ADDRESS OF 'BREAKPOINT' TRAP VECTOR
1271    000020                          IOTVEC=20               ;ADDRESS OF IOT TRAP VECTOR
1272    000024                          PFVEC=24                ;ADDRESS OF POWER FAIL TRAP VECTOR
1273    000030                          EMTVEC=30               ;ADDRESS OF EMT VECTOR
1274    000034                          TRAPVEC=34              ;ADDRESS OF TRAP VECTOR
1275    000060                          TKVEC=60                ;ADDRESS OF TTY KEYBOARD INTERRUPT VECTOR
1276    000064                          TPVEC=64                ;ADDRESS OF TTY PRINTER INTERRUPT VECTOR
1277    000240                          PIRVEC=240              ;ADDRESS OF PIRQ VECTOR
1278    000244                          FPEVEC=244              ;ADDRESS OF FLOATING POINT INT. VECTOR
1279    000250                          MMVEC=250               ;ADDRESS OF MEM MGMT ERROR TRAP VECTOR
1280
1281                            ;REGISTER ADDRESSES
1282    177776                          PSW=177776              ;ADDRESS OF STATUS REGISTER
1283    177774                          SLR=177774              ;ADDRESS OF STACK LIMIT REGISTER
1284    177772                          PIRQ=177772             ;ADDRESS OF PROGRAM INTERRUPT REQUEST
1285    177770                          UBREAK=177770           ;ADDRESS OF MICRO BREAK REGISTER
```

```
1286    177560                  TKS=177560              ;ADDRESS OF KEYBOARD CSR
1287    177562                  TKB=177562              ;ADDRESS OF KEYBOARD BUFFER
1288    177564                  TPS=177564              ;ADDRESS OF TELEPRINTER CSR
1289    177566                  TPB=177566              ;ADDRESS OF TELEPRINTER BUFFER
1290    177570                  SWR=177570              ;ADDRESS OF CONSOL SWITCH REGISTER
1291    177570                  DISPLAY=177570          ;ADDRESS OF CONSOL DISPLAY REGISTER
1292
1293                    ;INITIAL STACK POINTER SETTING
1294    000500                  STKPTR=500
1295
1296                    ;MISCELLANEOUS BIT ASSIGNMENTS
1297    000100                  BIT15=  100
1298    040000                  BIT14=  040000
1299    020000                  BIT13=  020000
1300    010000                  BIT12=  010000
1301    001000                  BIT9=   001000
1302    000400                  BIT8=   000400
1303    000100                  BIT6=   000100
1304
1305                    ;MEMORY MANAGEMENT REGISTER ADDRESS ASSIGNMENTS
1306    177572                  SR0=177572              ;ADDRESS OF MEM MGMT REGISTER SR0
1307    177574                  SR1=177574              ;    "    "       "          "   SR1
1308    177576                  SR2=177576              ;    "    "       "          "   SR2
1309    172516                  SR3=172516              ;ADDRESS OF MEM MGMT REGISTER SR3
1310
1311    172300                  KIPDR0=172300           ;ADDRESS OF KERNEL 'I' PAGE
1312    172302                  KIPDR1=172302           ;DESCRIPTOR REGISTERS
1313    172304                  KIPDR2=172304
1314    172306                  KIPDR3=172306
1315    172310                  KIPDR4=172310
1316    172312                  KIPDR5=172312
1317    172314                  KIPDR6=172314
1318    172316                  KIPDR7=172316
1319
1320    172340                  KIPAR0=172340           ;ADDRESSES OD KERNEL 'I' SPACE
1321    172342                  KIPAR1=172342           ;PAGE ADRESS REGISTERS
1322    172344                  KIPAR2=172344
1323    172346                  KIPAR3=172346
1324    172350                  KIPAR4=172350
1325    172352                  KIPAR5=172352
1326    172354                  KIPAR6=172354
1327    172356                  KIPAR7=172356
1328
1329
1330                    ;INSTRUCTION EQUATES
1331    104400                  HLT=TRAP
1332    104000                  SCOPE=EMT               ;SCOPE IS AN EMT TRAP
1333
1334                    ;MISC. EQUATES
1335    000006                  RW=6                    ;R/W BIT IN PDR REGISTERS
1336    000000                  UP=0                    ;UP BIT IN PDR REGISTERS
1337
1338
1339
1340
1341
```

```
                    000000                          .=0
000000  000000                                      .WORD   0               ;SPECIAL TRAP/INTERRUPT CATCHER IF PRO-
000002  000000                                      .WORD   0               ;GRAM HALTS AT 0 THEN ADDRESS WAS NOT
                                                                            ;LOADED PROPERLY FROM VECTOR.
000004  001116                                      .WORD   ERRTRP
000006  000002                                      .WORD   RTI
                    000034                          .=TRAPVEC
000034  001174                                      .WORD   ERROR
000036  000340                                      .WORD   PRTY7
                    000046                          .=46
000046  004614                                      .WORD   LOGICAL
                    000052                          .=52
000052  040000                                      .WORD   BIT14
                    000120                          .=120
000120  000000                              RELFL:  .WORD   0
000122  000000                              SAVPC2: .WORD   0
                                            ;THE SUBROUTINE WHERE IS CALLED BEFORE ANY TEST IS RUN TO SEE
                                            ;IF BRANCH GOBBLE RELOCATED THE ENTIRE FIRST FOUR K OF CORE INTO
                                            ;THE SECOND FOR K AND DIDN'T RELOCATE EVERYTHING BACK. IF THIS IS
                                            ;THE CASE THE WHERE WILL PUT THE PROGRAM BACK INTO THE FIRST FOUR K
                                            ;AND RETURN TO THE BEGINNING OF THE TEST THE USER DESIGNATED
                                            ;BY LOADING HIS STARTING ADDRESS. NOTE THST THIS ROUTINE WILL NOT
                                            ;RELOCATE THE PROGRAM IF IT HAS BEEN MOVED BY ANY OTHER SUBPROGRAM
                                            ;EXCEPT THE BRAMCH BOBBLE PROGRAM. THE RELOCATION OF THE PROGRAM
                                            ;TO THE FIRST FOUR K IS INACTED BY THE USER IN THE SAME WAY IT WAS
                                            ;IN THE DZQMBF AND DZQMBE VERSIONS OF THE D-124 TEST, IF IT HAS BEEN
                                            ;RELOCATED BY THE PART OF THIS TEST WHICH WAS TAKEN FROM DZQMB, THAT
                                            ;IS ALL THIS PROGRAM EXCEPT BRANCH GOBBLE.
000124  005737  000120                      WHERE:  TST     @#120
000130  100401                                     BMI     1$
000132  000207                                     RTS     PC
000134  011667  017762                      1$:    MOV     (SP),SAVPC2+20000
000140  004567  026102                             JSR     R5,RELOC+20000
000144  020000                                     .WORD   20000
000146  000000                                     .WORD   0
000150  016716  177746                             MOV     SAVPC2,(SP)
000154  005037  000120                             CLR     @#120
000160  000207                                     RTS     PC
                                            ;
                    000162                          .=162
000162  012706  000500                      PONE:   MOV     #500,SP ;STARTING ADDRESS TO RELOCATE LOADERS.
000166  004767  177732                             JSR     PC,WHERE
000172  004767  001736                             JSR     PC,$RLDR
000176  000000                                     HALT
000200  012706  000500                      PTWO:   MOV     #500,SP ;STARTING ADDRESS OF D-124K MEMORY EXERCISER.
000204  004767  177714                             JSR     PC,WHERE
```

```
 1398  000210  000137  002310                 JMP     @#START         ;GO TO START OF TEST
 1399  000214  012706  000500       PTHREE:   MOV     #500,SP ;STARTING ADDRESS OF PROGRAM #2.
 1400  000220  004767  177700                 JSR     PC,WHERE
 1401  000224  000137  004630                 JMP     @#PRG2          ;GO START PROGRAM #2
 1402  000230  012706  000500       PFOUR:    MOV     #500,SP ;STARTING ADRESS OF PROGRAM #3.
 1403  000234  004767  177664                 JSR     PC,WHERE
 1404  000240  000137  005706                 JMP     @#PRG3          ;GO START PROGRAM #3
 1405          000250                         .=250
 1406  000250  000000                         .WORD   0       ;MEMORY MANAGEMENT TRAP VECTOR.
 1407  000252  000000                         .WORD   0
 1408  000254  012706  000500       PFIVE:    MOV     #500,SP ;START ADDRESS OF PROGRAM #4.
 1409  000260  004767  177640                 JSR     PC,WHERE
 1410  000264  000137  005740                 JMP     @#PRG4
 1411  000270  012706  000500       PSIX:     MOV     #500,SP ;STARTING ADDRESS OF BRANCH GOBBLE MOS TEST.
 1412  000274  004767  177624                 JSR     PC,WHERE
 1413  000300  000167  012134                 JMP     BRANCH
 1414                                         ;
 1415
 1416                                         ;ROUTINE TO SAVE REGISTERS ON THE STACK
 1417                                         ;CALLED BY SAVE MACRO OR JSR    PC,$SAVR
 1418  000304  012667  000016       $SAVR:    MOV     (SP)+,1$        ;SAVE RETURN PC
 1419  000310  010546                         MOV     %5,-(SP)
 1420  000312  010446                         MOV     %4,-(SP)
 1421  000314  010346                         MOV     %3,-(SP)
 1422  000316  010246                         MOV     %2,-(SP)
 1423  000320  010146                         MOV     %1,-(SP)
 1424  000322  010046                         MOV     %0,-(SP)
 1425  000324  012707                         MOV     (PC)+,PC        ;RETURN
 1426  000326  000000               1$:       0                       ;CONTAINS RETURN ADDRESS
 1427
 1428                                         ;ROUTINE TO RESTORE REGISTERS SAVED ON THE STACK
 1429                                         ;CALLED BY RESTORE MACRO OR JSR  PC,$RESTR
 1430  000330  012667  000016       $RESTR:   MOV     (SP)+,1$        ;SAVE RETURN PC
 1431  000334  012600                         MOV     (SP)+,%0
 1432  000336  012601                         MOV     (SP)+,%1
 1433  000340  012602                         MOV     (SP)+,%2
 1434  000342  012603                         MOV     (SP)+,%3
 1435  000344  012604                         MOV     (SP)+,%4
 1436  000346  012605                         MOV     (SP)+,%5
 1437  000350  012707                         MOV     (PC)+,PC        ;RETURN
 1438  000352  000000               1$:       0                       ;CONTAINS RETURN ADDRESS
 1439
 1440                                         .SBTTL  POWER FAIL ROUTINE
 1441          000502                         .=502
 1442                                         ;POWER FAIL ROUTINE
 1443                                         ;THE POWER DOWN ROUTINE SAVES THE KEYBOARD STATUS,THE GENERAL REGISTERS
 1444                                         ;(R0-R5),AND MEM MGMT REGISTERS (KIPDR0-KIPDR7,KIPAR0-KIPAR7,SR2,SR0)
 1445                                         ;ON THE STACK AND SAVES THE STACK POINTER IN PFSTK BELOW.
 1446  000502  013746  177560       PDWN:     MOV     @#TKS,-(SP)     ;SAVE KEYBOARD STATUS
 1447  000506  004767  177572                 JSR     PC,$SAVR        ;GO SAVE REGISTERS ON THE STACK
 1448  000512  005737  000752                 TST     @#MMAVA         ;CHECK IF MEM MGMT IS AVAILABLE
 1449  000516  001417                         BEQ     3$              ;BRANCH IF NOT AVAILABLE
 1450  000520  013746  177572                 MOV     @#SR0,-(SP)     ;SAVE SR0
 1451  000524  013746  177576                 MOV     @#SR2,-(SP)     ;SAVE SR2
 1452  000530  012700  172300                 MOV     #KIPDR0,R0      ;GET ADDRESS OF KIPDR0
 1453  000534  012702  000010                 MOV     #8.,R2
```

```
1454  000540  010203                          MOV     R2,R3
1455  000542  012046                  1$:     MOV     (R0)+,-(SP)     ;SAVE KIPDR0-KIPDR7
1456  000544  077202                          SOB     R2,1$
1457  000546  012700  172340                  MOV     #KIPAR0,R0      ;GET ADDRESS OF KIPAR0
1458  000552  012046                  2$:     MOV     (R0)+,-(SP)     ;SAVE KIPAR0-KIPAR7
1459  000554  077302                          SOB     R3,2$
1460  000556  010627                  3$:     MOV     SP,(PC)+        ;SAVE STACK PTR IN FOLLOWING LOCATION
1461  000560  000000                  PFSTK:  .WORD   0               ;CONTAINS STACK PTR AFTER POWER FAIL
1462  000562  012737  000572  000024          MOV     #PUP,@#PFVEC    ;SET POWER FAIL VECTOR TO PUP ROUTINE
1463  000570  000000                          HALT
1464
1465                                  ;POWER UP ROUTINE.
1466  000572  000240                  PUP:    NOP
1467  000574  013706  000560                  MOV     @#PFSTK,SP      ;SET STACK PTR
1468  000600  005767  000146                  TST     MMAVA           ;CHECK IF MEM MGMT IS AVAILABLE
1469  000604  001417                          BEQ     4$
1470  000606  012700  172360                  MOV     #KIPAR7+2,R0    ;GET ADDRESS OF KIPAR7+2
1471  000612  012702  000010                  MOV     #8.,R2
1472  000616  010203                          MOV     R2,R3
1473  000620  012640                  1$:     MOV     (SP)+,-(R0)     ;RESTORE KIPAR7-KIPAR0
1474  000622  077302                          SOB     R3,1$
1475  000624  012700  172320                  MOV     #KIPDR7+2,R0    ;GET ADDRESS OF KIPDR7+2
1476  000630  012640                  2$:     MOV     (SP)+,-(R0)     ;RESTORE KIPDR7-KIPDR0
1477  000632  077202                          SOB     R2,2$
1478  000634  012637  177576                  MOV     (SP)+,@#SR2     ;RESTORE SR2
1479  000640  012637  177572                  MOV     (SP)+,@#SR0     ;RESTORE SR0
1480  000644  005767  006144          4$:     TST     PARAVA          ;CHECK IF PARITY REGISTERS ARE ENABLED
1481  000650  001402                          BEQ     5$              ;BRANCH IF NOT
1482  000652  004767  006066                  JSR     PC,.MAMF        ;GO ENABLE PARITY REGISTERS
1483  000656                          5$:
1484  000656  004767  177446                  JSR     PC,$RESTR       ;RESTORE REGISTERS FROM STACK
1485  000662  012637  177560                  MOV     (SP)+,@#TKS
1486  000666  012737  000502  000024          MOV     #PDWN,@#PFVEC   ;SET POWER FAIL TRAP TO PDWN ROUTINE
1487  000674  005027                          CLR     (PC)+
1488  000676  000000                  10$:    .WORD   0
1489  000700  005267  177772          11$:    INC     10$             ;DELAY WAITING FOR TTY MOTOR
1490  000704  100375                          BPL     11$
1491  000706  004567  000046                  JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
1492  000712  000720                          PWRFAIL
1493  000714  000240                  6$:     NOP
1494  000716  000002                          RTI                     ;RETURN
1495
1496  000720  005015  047520  042527  PWRFAIL:.ASCIZ  <15><12>'POWER FAILED'<15><12>
1497  000726  020122  040506  046111
1498  000734  042105  005015      000
1499
1500                                          .SBTTL  TAGS & PRINT ROUTINE
1501  000742                                  .EVEN
1502  000742  000000                  ICNT:   .WORD   0                       ;CONTAINS PASS COUNT
1503  000744  000000                  ICOUNT: .WORD   0               ;CONTAINS ITERATION PATTERN
1504  000746  000000                  ERCNT:  0                       ;CONTAINS ERROR COUNT
1505  000750  000000                  LDDISP: 0                       ;CONTAINS DISPLAY REGISTER IMAGE
1506  000752  000000                  MMAVA:  0                       ;MEM MGMT AVAILABLE INDICATOR
1507                                                                  ;0=NOT AVAIL,-1=AVAIL
1508  000754  000000                  RELOCF: .WORD   0               ;CONTAINS RELOCATION FACTOR
1509  000756  000000                  COUNT:  .WORD   0               ;TEMPORARY WORKING LOACATION
```

```
1510
1511                                    ;ROUTINE TO PASS MESSAGE ADDRESS TO TYPE ROUTINE BELOW
1512                                    ;CALL:  JSR     R5,$PRINT
1513                                    ;       MESSAGE ADDRESS
1514  000760  000240                    $PRINT: NOP
1515  000762  012567  000016                    MOV     (R5)+,1$        ;GET MESSAGE ADDRESS
1516  000766  066767  177762  000010           ADD     RELOCF,1$       ;ADD RELOCATION FACTOR
1517  000774  013746  177776                    MOV     @#PSW,-(SP)     ;PUSH PSW ON THE STACK
1518  001000  004767  000014                    JSR     PC,.TYPE        ;CALL TYPE ROUTINE
1519  001004  000000                    1$:     .WORD   0               ;CONTAINS MESSAGE ADDRESS
1520  001006  000205                            RTS     R5              ;RETURN
1521
1522                                    ;ROUTINE TO TYPE ASCII MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
1523                                    ;THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
1524                                    ;CALL:  TYPE
1525                                    ;               MESADR          ;MESADR IS FIRST ADDRESS OF ASCIZ STRING
1526
1527                                    ;TAGS USED BY THE TYPE ROUTINE BELOW
1528  001010      000                   $NULL:  .BYTE   0               ;CONTAINS NULL CHARACTER
1529  001011      002                   $FILL:  .BYTE   2               ;CONTAINS # OF FILLER CHARACTERS
1530  001012      000                   $TPFLG: .BYTE   0               ;CONTAINS TELEPRINTER AVAILABLE FLAG
1531                                                                    ;0/377 = AVAIL/NOT AVAIL
1532  001013      000                   $TKFLG: .BYTE   0               ;CONTAINS KEYBOARD AVAILABLE FLAG
1533  001014  177564                    $TPS:   .WORD   177564          ;ADDRESS OF TLEPRINTER STATUS REGISTER
1534  001016  177566                    $TPB:   .WORD   177566          ;ADDRESS OF TELEPRINTER DATA BUFFER
1535  001020  010046                    .TYPE:  MOV     R0,-(SP)        ;SAVE R0
1536  001022  017600  000002                    MOV     @2(SP),R0       ;GET MESSAGE ADDRESS
1537  001026  062766  000002  000002           ADD     #2,2(SP)        ;ADJUST RETURN PC
1538
1539  001034  112046                    1$:     MOVB    (R0)+,-(SP)     ;PUSH CHARACTER TO BE TYPED ONTO STACK
1540  001036  001003                            BNE     2$              ;BRANCH IF NOT THE TERMINATOR
1541  001040  005726                            TST     (SP)+           ;POP TERMINATOR CHAR OFF THE STACK
1542  001042  012600                            MOV     (SP)+,R0        ;RESTORE R0
1543  001044  000002                            RTI                     ;RETURN TO CALLER
1544
1545  001046  004767  000026            2$:     JSR     PC,5$           ;TYPE CHARACTER
1546  001052  122726  000012            3$:     CMPB    #12,(SP)+       ;CHECK IF CHARACTER WAS A LINE FEED
1547  001056  001366                            BNE     1$              ;BRANCH IF NOT LINE FEED
1548  001060  016746  177724                    MOV     $NULL,-(SP)     ;GET # OF FILLERS REQUIRED AND FILLER
1549                                                                    ;CHARACTER.
1550
1551  001064  105366  000001            4$:     DECB    1(SP)           ;DECREMENT FILLERS REQ. COUNT
1552  001070  002770                            BLT     3$              ;BRANCH IF NO MORE FILLERS ARE REQUIRED
1553  001072  004767  000002                    JSR     PC,5$           ;TYPE FILLER CHARACTER
1554  001076  000772                            BR      4$
1555
1556  001100  105777  177710            5$:     TSTB    @$TPS           ;WAIT FOR OUTPUT DEVICE
1557  001104  100375                            BPL     .-4
1558  001106  116677  000002  177702           MOVB    2(SP),@$TPB     ;OUTPUT CHARACTER
1559  001114  000207                            RTS     PC
1560
1561                                    ;XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
1562                                    ;ERROR TRAP SERVICE ROUTINE
1563  001116  005737  177570            ERRTRP: TST     @#SWR           ;CHECK IF HALT ON ERROR
1564  001122  100001                            BPL     .+4             ;BRANCH IF NO HALT ON ERROR
1565  001124  000000                            HALT                    ;HALT
```

```
1566  001126  005727                          TST     (PC)+           ;CHECK IF PREV TRAP TO 4 REPORTED
1567  001130  000000                  1$:     .WORD   0               ;CONTAINS ERROR REPORTED FLAG
1568  001132  001013                          BNE     2$              ;BRANCH IF NOT REPORTED
1569  001134  010667  177770                  MOV     SP,1$           ;SET 'NOT REPORTED'
1570  001140  011602                          MOV     (SP),R2         ;GET PC OFF STACK
1571  001142  004767  000356                  JSR     PC,$FORMO       ;GO TO FORMAT ROUTINE
1572  001146  004567  177606                  JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
1573  001152  001452                          TRAP4
1574  001154  004567  177600                  JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
1575  001160  002275                          DIGITS
1576  001162  000000                  2$:     HALT                    ;ERROR! SECOND TRAP TO 4 OCCURRED
1577                                                                   ;BEFORE FIRST WAS PRINTED
1578  001164  005067  177740                  CLR     1$
1579  001170  000137  000200                  JMP     @#200           ;RESTART AT 200
1580
1581                                           .SBTTL  ERROR SERVICE ROUTINE
1582                                  ;ERROR SERVICE CALLED BY JSR  PC,ERROR INSTRUCTION
1583                                  ;OR HLT (A TRAP INST)
1584  001174  000240                  ERROR:  NOP
1585  001176  022767  017777  177542          CMP     #17777,ERCNT    ;CHECK FOR MAX ERROR CNT
1586  001204  001403                          BEQ     4$
1587  001206  062767  000001  177532          ADD     #1,ERCNT        ;INCREMENT ERROR COUNT
1588  001214  032737  001000  177570  4$:     BIT     #BIT9,@#SWR     ;SWITCH 9 UP?
1589  001222  001411                          BEQ     5$
1590  001224  042767  017777  177516          BIC     #17777,LDDISP   ;SAVE RELOCATION BITS
1591  001232  056767  177510  177510          BIS     ERCNT,LDDISP    ;LOAD ERROR COUNT
1592  001240  016737  177504  177570          MOV     LDDISP,@#DISPLAY        ;LOAD DISPLAY REGISTER
1593  001246  005737  177570          5$:     TST     @#SWR           ;HALT ON ERROR
1594  001252  100002                          BPL     .+6
1595  001254  000000                          HALT
1596  001256  000471                          BR      3$
1597  001260  032737  020000  177570          BIT     #20000,@#SWR    ;PRINT OUT DESIRED?
1598  001266  001052                          BNE     1$              ;BRANCH IF NO PRINTOUT
1599  001270  004767  177010                  JSR     PC,$SAVR        ;GO SAVE REGISTERS ON THE STACK
1600  001274  016602  000014                  MOV     14(SP),R2       ;GET PC OF ERROR CALL
1601  001300  004767  000220                  JSR     PC,$FORMO       ;GO TO FORMAT ROUTINE
1602  001304  004567  177450                  JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
1603  001310  001467                          ERRPC
1604  001312  004567  177442                  JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
1605  001316  002275                          DIGITS
1606  001320  016602  000004                  MOV     4(SP),R2        ;GET FAILING ADDRESS (IN R2)
1607  001324  004767  000174                  JSR     PC,$FORMO       ;GO TO FORMAT ROUTINE
1608  001330  004567  177424                  JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
1609  001334  002253                          ADRESS
1610  001336  105767  005157                  TSTB    PENFLG          ;BRANCH IF PARITY ERROR DETECTED
1611  001342  001017                          BNE     11$             ;BUT NOT FOUND
1612  001344  105767  005150                  TSTB    PEFLG           ;BRANCH IF PARITY ERROR DETECTED
1613  001350  001006                          BNE     10$             ;BUT FOUND
1614  001352  004567  177402                  JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
1615  001356  001473                          XMTDAT
1616  001360  010046                          MOV     R0,-(SP)        ;PUSH VALUE TO TYPED ONTO STACK
1617  001362  004767  000360                  JSR     PC,02A          ;GO PRINT VALUE
1618  001366                           10$:
1619  001366  004567  177366                  JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
1620  001372  001506                          RECDAT
1621  001374  010346                          MOV     R3,-(SP)        ;PUSH VALUE TO BE TYPED ONTO STACK
```

```
1622  001376  004767  000344                  JSR     PC,02A
1623  001402                          11$:
1624  001402  004567  177352                  JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
1625  001406  015020                          $CRLF
1626  001410  004767  176714                  JSR     PC,$RESTR       ;RESTORE REGISTERS FROM STACK
1627  001414  032737  002000  177570  1$:     BIT     #2000,@#SWR     ;RING BELL ON ERROR
1628  001422  001403                          BEQ     2$
1629  001424  004567  177330                  JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
1630  001430  001521                          BELL
1631  001432  005737  177570          2$:     TST     @#SWR           ;HALT AFTER PRINT OUT
1632  001436  100001                          BPL     .+4
1633  001440  000000                          HALT
1634  001442  010042                  3$:     MOV     R0,-(R2)        ;RESTORE CORRECT DATA TO ADDRESS
1635  001444  062702  000002                  ADD     #2,R2
1636  001450  000002                          RTI
1637
1638  001452  051124  050101  042520  TRAP4:  .ASCII  'TRAPPED TO 4 '
1639  001460  020104  047524  032040
1640  001466     040
1641  001467     120  036503     000  ERRPC:  .ASCIZ  'PC='
1642  001473     107  047517  020104  XMTDAT: .ASCIZ  'GOOD DATA='
1643  001500  040504  040524  000075
1644  001506  041040  042101  042040  RECDAT: .ASCIZ  ' BAD DATA='
1645  001514  052101  036501     000
1646  001521     007     000          BELL:   .ASCIZ  <7>
1647          001524                          .EVEN
1648                                  ;ROUTINE TO PLACE ASCII VALUE OF AN ADDRESS IN TO ADDRESS MESSAGE
1649  001524  066767  177224  000014  $FORM0: ADD     RELOCF,11$+2
1650  001532  066767  177216  000134          ADD     RELOCF,41$+2
1651  001540  004767  176540                  JSR     PC,$SAVR        ;GO SAVE REGISTERS ON THE STACK
1652  001544  012704  002275          11$:    MOV     #DIGITS,R4      ;ADDRESS WHERE ASCII VALUES ARE STORED
1653  001550  005003                          CLR     R3              ;WORKING & INDEX REGISTER
1654  001552  162702  000002                  SUB     #2,R2           ;ADJUST ADDRESS
1655  001556  010205                          MOV     R2,R5           ;SAVE
1656  001560  010501                          MOV     R5,R1
1657  001562  005767  177164                  TST     MMAVA           ;CHECK IF MEM MGMT IS AVAILABLE
1658  001566  001426                          BEQ     1$              ;BRANCH IF NOT AVAILABLE
1659  001570  032737  000001  177572          BIT     #1,@#SR0        ;IS MEM MGMT ENABLED
1660  001576  001422                          BEQ     1$
1661  001600  042701  017777                  BIC     #17777,R1       ;SAVE PAR SELECTOR BITS
1662  001604  000301                          SWAB    R1              ;SWAP BYTES
1663  001606  006001                          ROR     R1
1664  001610  006001                          ROR     R1              ;FORM INDEX VALUE
1665  001612  006001                          ROR     R1
1666  001614  006001                          ROR     R1
1667  001616  017102  001726                  MOV     @PARTAB(1),R2   ;GET CONTENTS OF PAR
1668  001622  012700  000006                  MOV     #6,R0           ;SHIFT COUNT
1669  001626  006302                          ASL     R2              ;SHIFT KIPAR1 6 PLACES LEFT
1670  001630  006103                          ROL     R3              ;2 MSB'S GO INTO R3
1671  001632  077003                          SOB     R0,.-4
1672  001634  042705  160000                  BIC     #160000,R5      ;CLEAR PAR SELECTOR BITS
1673  001640  060502                          ADD     R5,R2           ;FORM 18 BIT ADDRESS
1674  001642  005503                          ADC     R3              ;IN R2 & R3
1675  001644  006302                  1$:     ASL     R2              ;FIRST DIGIT TO R3
1676  001646  006103                          ROL     R3
1677  001650  012700  000006                  MOV     #6,R0           ;DIGIT COUNT
```

```
1678  001654  000404                        BR      3$              ;PRINT FIRST DIGIT
1679  001656  006302                2$:     ASL     R2
1680  001660  006103                        ROL     R3
1681  001662  005305                        DEC     R5
1682  001664  001374                        BNE     2$
1683  001666  012705  000003        3$:     MOV     #3,R5           ;DIGIT SHIFT COUNT
1684  001672  116324  002236        41$:    MOVB    DIGTAB(3),(4)+  ;LOAD DIGIT INTO MESSAGE
1685  001676  005003                        CLR     R3              ;CLEAR INDEX
1686  001700  005300                        DEC     R0              ;DEC DIGIT COUNT
1687  001702  001365                        BNE     2$
1688  001704  004767  176420              JSR     PC,$RESTR       ;RESTORE REGISTERS FROM STACK
1689  001710  046767  177040  177630      BIC     RELOCF,11$+2
1690  001716  046767  177032  177750      BIC     RELOCF,41$+2
1691  001724  000207                        RTS     PC              ;RETURN
1692
1693  001726  172340                PARTAB: KIPAR0
1694  001730  172342                        KIPAR1
1695  001732  172344                        KIPAR2
1696  001734  172346                        KIPAR3
1697  001736  172350                        KIPAR4
1698  001740  172352                        KIPAR5
1699  001742  172354                        KIPAR6
1700  001744  172356                        KIPAR7
1701
1702                                ;ROUTINE TO TYPE OCTAL VALUE PUSHED ONTO STACK
1703                                ;CALL:  MOV     VALUE,-(SP)     ;PUSH VALUE ONTO STACK
1704                                        JSR     PC,02A          ;CALL ROUTINE
1705  001746                        02A:
1706  001746  004767  176332              JSR     PC,$SAVR        ;GO SAVE REGISTERS ON THE STACK
1707  001752  016600  000016              MOV     16(SP),R0       ;GET VALUE
1708  001756  012703  000006              MOV     #6,R3           ;COUNTER
1709  001762  005002                      CLR     R2              ;WORKING REGISTER
1710  001764  006100                      ROL     R0
1711  001766  006102                      ROL     R2
1712  001770  062702  000260      1$:     ADD     #260,R2         ;FORM ASCII VALUE
1713  001774  010267  000040              MOV     R2,2$           ;MOVE CHAR TO TYPE LOCATION
1714  002000  004567  176754              JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
1715  002004  002040                      2$
1716  002006  005002                      CLR     R2
1717  002010  006100                      ROL     R0
1718  002012  006102                      ROL     R2
1719  002014  006100                      ROL     R0
1720  002016  006102                      ROL     R2
1721  002020  006100                      ROL     R0
1722  002022  006102                      ROL     R2
1723  002024  005303                      DEC     R3
1724  002026  001360                      BNE     1$
1725  002030  004767  176274              JSR     PC,$RESTR       ;RESTORE REGISTERS FROM STACK
1726  002034  012616                      MOV     (SP)+,(SP)
1727  002036  000207                      RTS     PC
1728  002040  000000              2$:     .WORD   0               ;CONTAINS CHARACTER TO BE TYPED
1729
1730  002042  000000              LODFLO: .WORD   0
1731                                ;ROUTINE TO SAVE ABS LOADER
1732  002044  005767  177772      $LDR:   TST     LODFLO
1733  002050  001401                      BEQ     3$
```

```
1734  002052  000207                          RTS     PC
1735  002054  012700  017776            3$:   MOV     #17776,R0
1736  002060  012737  002072  000004          MOV     #2$,@#ERRVEC    ;SET TIME OUT TRAP VECTOR
1737  002066  005720                          TST     (R0)+
1738  002070  000776                          BR      .-2
1739  002072  022626                    2$:   CMP     (SP)+,(SP)+
1740  002074  162700  002202                SUB     #2202,R0        ;POINT R0 BACK TO LOADER
1741  002100  010067  000102                MOV     R0,$LDR1        ;SAVE FOR RESTORE ROUTINE
1742  002104  012702  001100                MOV     #1100,R2        ;WORD COUNT
1743  002110  012703  015454                MOV     #LODAR,R3       ;WHERE LOADER IS TO BE STORED
1744  002114  012023                    1$:   MOV     (R0)+,(R3)+     ;STORE LOADER
1745  002116  005302                          DEC     R2
1746  002120  001375                          BNE     1$
1747  002122  014367  000042                MOV     -(R3),LSTLOC    ;SAVE LAST WORD OF LOADERS
1748  002126  005367  177710                DEC     LODFLO
1749  002132  000207                          RTS     PC              ;RETURN
1750
1751                                    ;ROUTINE TO RESTORE LOADER
1752  002134  005767  177702           $RLDR: TST     LODFLO
1753  002140  001001                          BNE     2$
1754  002142  000207                          RTS     PC
1755  002144  016705  000036            2$:   MOV     $LDR1,R5        ;GET FIRST ADDRESS OF WHERE LOADER IS
1756                                                                  ;TO BE RESTORED
1757  002150  012704  015454                MOV     #LODAR,R4       ;ADDRESS WHERE LOADER IS STORED
1758  002154  012702  001100                MOV     #1100,R2        ;WORD COUNT
1759  002160  012425                    1$:   MOV     (R4)+,(R5)+     ;RESTORE
1760  002162  005302                          DEC     R2
1761  002164  001375                          BNE     1$
1762  002166  012745                          MOV     (PC)+,-(R5)     ;RESTORE LAST LOCATION (SAVED BY SAVE
1763  002170  000000                    LSTLOC: .WORD   0               ;LOADERS ROUTINE ABOVE)
1764  002172  004567  176562                JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
1765  002176  002210                          $LDRM
1766  002200  005067  177636                CLR     LODFLO
1767  002204  000207                          RTS     PC              ;RETURN TO CALLER
1768
1769  002206  000000                    $LDR1:  .WORD   0               ;FIRST ADDRESS WHERE LOADERS ARETO BE
1770                                                                  ;RESTORED TO
1771  002210  047514  042101  051105    $LDRM:  .ASCIZ  'LOADER IS RESTORED'<15><12>
1772  002216  044440  020123  042522
1773  002224  052123  051117  042105
1774  002232  005015     000
1775          002236                          .EVEN
1776                                    ;DIGIT TABLE
1777  002236  030460                    DIGTAB: "01
1778  002240  031462                            "23
1779  002242  032464                            "45
1780  002244  033466                            "67
1781
1782                                    ;MESSAGES
1783  002246  040514  052123     040   LST:    .ASCII  'LAST '
1784  002253     115  046505  051117   ADRESS: .ASCII  'MEMORY ADDRESS IS '
1785  002260  020131  042101  051104
1786  002266  051505  020123  051511
1787  002274     040
1788  002275     060  030060  030060   DIGITS: .ASCIZ  '000000 '
1789  002302  020060     000
```

```
 1790          002306                          .EVEN
```

```
                                        PLACE:  .WORD   0
                                                .SBTTL  MEMORY ADDRESS TESTS

                                        ;THIS TEST ADDRESS MEMORY UP TO 128K AND PROVES 'UNIQNESS' OF ALL
                                        ;MEMORY ADDRESS IN A 32K SEGMENT. THE TEST WRITES INTO EACH MEMORY
                                        ;ADDRESS THE VALUE OF THAT ADDRESS AND THEN CHECKS FOR THE CORRECT
                                        ;DATA IN EACH ADDRESS.
                                        ;THE TWELVE MOST SIGNIFICANT BITS OF THE LAST AVAILABLE MEMORY ADDRESS
                                        ;IS STORED IN R5.
                                        ;STARTING INTRUCTIONS
                                        ;       LOAD ADDRESS=200
                                        ;       PRESS START
                                        ;       STACK POINTER IS AT 500
                                        ;*************RESTART AT 162 TO RESTORE LOADER*******************
                                        ;MEMORY ADDRESS TEST
002310  012737  002352  000212  START:  MOV     #START1,@#212   ;CHANGE START ADDRESS
002316  012706  000500                  MOV     #STKPTR,SP      ;SET UP STACK PTR
002322  004767  177516                  JSR     PC,$LDR         ;GO SAVE MONITOR & LOADERS
002326  004567  176426                  JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
002332  011752                          RESLDR
002334  005037  000746                  CLR     @#ERCNT         ;CLEAR ERROR COUNT
002340  005037  000750                  CLR     @#LDDISP        ;CLEAR DISPLAY REGISTER STORAGE LOCN
002344  013737  000750  177570          MOV     @#LDDISP,@#DISPLAY      ;CLEAR DISPLAY REGISTER
002352  012706  000500          START1: MOV     #STKPTR,SP      ;SET STACK PTR
002356  005037  006520                  CLR     @#PEFLG         ;CLEAR PARITY ERROR INDICATORS
002362  012727  002352                  MOV     #START1,(PC)+   ;LOAD PARITY ERROR RESTART ADDRESS
002366  000000          PERSTRT:.WORD   0               ;CONTAINS RESTART ADDRESS AFTER PAR ERR
002370  005037  000742                  CLR     @#ICNT          ;CLEAR PASS COUNT
002374  005037  000754                  CLR     @#RELOCF        ;CLEAR RELOACATION FACTOR
002400  012737  000502  000024          MOV     #PDWN,@#PFVEC   ;SET POWER FAIL TRAP VECTOR
002406  005037  000026                  CLR     @#PFVEC+2

                                        ;CHECK IF MEMORY MANAGEMENT IS AVAILABLE
002412  005067  176334                  CLR     MMAVA           ;CLEAR MEM MGMT AVAILABLE INDICATOR
002416  032737  010000  177570          BIT     #BIT12,@#SWR    ;CHECK IF TO RUN WITH MEM MGMT
002424  001007                          BNE     1$              ;DO NOT USE MEM MGMT IF SW12 WAS SET
002426  012737  002444  000004          MOV     #1$,@#ERRVEC    ;SET TIME OUT TRAP
002434  005037  177572                  CLR     @#SR0           ;REFERENCE MEM MGMT
002440  005167  176306                  COM     MMAVA           ;SET INDICATOR TO -1 IF AVAILABLE
002444  004767  004274          1$:     JSR     PC,.MAMF        ;GO ENABLE PARITY ACTION


                                        ;ROUTINE TO WRITE VALUE OF MEMORY ADDRESS INTO MEMORY ADDRESS
                                        ;FOR EXAMPLE ROUTINE WRITES 20000 INTO LOCATION 20000
002450  012737  002510  000004  WRTUP:  MOV     #DONE0,@#ERRVEC ;SET TIME OUT TRAP VECTOR
002456  010701                          MOV     PC,R1           ;LOAD TRACE REGISTER
002460  004767  004360                  JSR     PC,LDMM0
002464  012737  007142  000250          MOV     #MMABT0,@#MMVEC ;SET MEM MGMT ABORT VECTOR
002472  012702  020000                  MOV     #20000,R2       ;FIRST ADDRESS
002476  010203                          MOV     R2,R3           ;LOAD CONSTANT
002500  010322                          MOV     R3,(R2)+        ;WRITE VALUE OF ADDRESS INTO ADDRESS
002502  062703  000002                  ADD     #2,R3           ;NEXT VALUE
002506  000774                          BR      .-6             ;WRITE UNTIL DONE

002510  012706  000500          DONE0:  MOV     #STKPTR,SP      ;SET STACK PTR
```

```
1847  002514  004767  177004                  JSR     PC,$FORMO       ;GO TO FORMAT ROUTINE
1848  002520  004567  176234                  JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
1849  002524  002246                          LST
1850  002526  004567  176226                  JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
1851  002532  015020                          $CRLF
1852
1853                                  ;ROUTINE TO CHECK THAT VALUE OF MEMORY ADDRESS WAS WRITTEN CORRECTLY
1854  002534  010701                          MOV     PC,R1           ;LOAD TRACE REGISTER
1855  002536  012702  020000                  MOV     #20000,R2       ;SET R2
1856  002542  012737  002600  000004          MOV     #DONE1,@#ERRVEC ;SET TIME OUT TRAP
1857  002550  010200                          MOV     R2,R0
1858  002552  162700  000002                  SUB     #2,R0           ;SUBTRACT 2
1859  002556  004767  004262                  JSR     PC,LDMMO
1860  002562  062700  000002          1$:     ADD     #2,R0
1861  002566  012203                          MOV     (R2)+,R3        ;GET WRITTEN VALUE
1862  002570  020003                          CMP     R0,R3           ;CHECK
1863  002572  001773                          BEQ     1$
1864  002574  104400                          HLT                     ;ERROR! TO DETERMINE WHICH ADDRESS WAS
1865                                  ;WRITTEN IMPROPERLY EXAMINE R2. NEXT EXAMINE MEM MGMT REGISTER KIPAR1
1866                                  ;(IF MEM MGMT IS AVAILABLE). ADD R2 AND KIPAR1 TOGETHER AS SHOWN BELOW
1867
1868                                  ;       R2-2                    0 00X XXX XXX XXX XXX
1869                                  ;       KIPAR1(772342)  0 000 YYY YYY YYY YYY
1870                                  ;       ADDRESS                 ZZZ ZZZ ZZZ ZZZ ZZZ ZZZ
1871
1872  002576  000771                          BR      1$
1873  002600  012706  000500          DONE1:  MOV     #STKPTR,SP      ;SET STACK PTR
1874  002604  010701                          MOV     PC,R1           ;LOAD TRACE REGISTER
1875
1876                                  ;ROUTINE TO WRITE 1'S COMPLEMENT VALUE OF ADDRESS INTO ADDRESS
1877                                  ;FOR EXAMPLE ROUTINE WRITES 157777 INTO ADDRESS 20000
1878
1879  002606  005767  176140                  TST     MMAVA           ;MEMORY MAGNAGEMENT AVAILABLE?
1880  002612  001420                          BEQ     3$
1881  002614  013703  172342                  MOV     @#KIPAR1,R3     ;FIND LAST ADDRESS IF MEM MANAGE USED
1882  002620  006303                          ASL     R3
1883  002622  006303                          ASL     R3
1884  002624  006303                          ASL     R3
1885  002626  006303                          ASL     R3
1886  002630  006303                          ASL     R3
1887  002632  006303                          ASL     R3
1888  002634  010246                          MOV     R2,-(SP)        ;DEVELOP COMPLEMENT OF LAST ADDRESS
1889  002636  042716  020000                  BIC     #20000,(SP)     ;SAVE BITS IF MEMORY IS NOT A MULTIPLE OF 4K
1890  002642  062603                          ADD     (SP)+,R3
1891  002644  012737  007174  000250          MOV     #MMABT1,@#MMVEC ;SET ABORT VECTOR
1892  002652  000403                          BR      2$
1893  002654  162702  000002          3$:     SUB     #2,R2           ;R2=LAST ADDRESS
1894  002660  010203                          MOV     R2,R3
1895  002662  005103                  2$:     COM     R3              ;COMPLEMENT VALUE IN R3
1896  002664  062703  000002          1$:     ADD     #2,R3
1897  002670  010342                          MOV     %3,-(R2)        ;WRITE COMPLIMENT VALUE INTO ADDRESS
1898  002672  102403                          BVS     DONE3
1899  002674  020227  017776                  CMP     R2,#17776
1900  002700  001371                          BNE     1$
1901
1902                                  ;SET UP TO CHECK COMPLEMENT DATA WRITTEN DOWN
```

```
1903  002702  000240                    DONE3:  NOP
1904  002704  010701                            MOV     PC,R1           ;LOAD TRACE REGISTER
1905  002706  005767  176040                    TST     MMAVA           ;CHECK IF MM IS AVAIL
1906  002712  001406                            BEQ     1$
1907  002714  012737  000200  172342            MOV     #200,@#KIPAR1   ;INIT KIPAR1
1908  002722  012737  007142  000250            MOV     #MMABTO,@#MMVEC ;SET ABORT VECTOR
1909  002730  012737  002770  000004  1$:       MOV     #DONE4,@#ERRVEC
1910  002736  012702  020000                    MOV     #20000,R2       ;FIRST ADDRESS
1911  002742  010200                            MOV     R2,R0
1912  002744  005100                            COM     R0              ;FIRST DATA (COM OF ADDRESS)
1913  002746  062700  000002                    ADD     #2,R0
1914  002752  162700  000002          2$:       SUB     #2,R0
1915  002756  012203                            MOV     (R2)+,R3        ;GET VALUE
1916  002760  020003                            CMP     R0,R3           ;CHECK
1917  002762  001773                            BEQ     2$
1918  002764  104400                            HLT
1919  002766  000771                            BR      2$
1920  002770  000240                    DONE4:  NOP
1921
1922                                    ;ROUTINE TO WRITE BANK # INTO ALL ADDRESSES IN A 4K BANK
1923  002772  012737  003040  000004            MOV     #DONE4A,@#ERRVEC;SET TIME OUT TRAP VECTOR
1924  003000  010701                            MOV     PC,R1
1925  003002  004767  004036                    JSR     PC,LDMMO
1926  003006  012737  007142  000250            MOV     #MMABTO,@#MMVEC
1927  003014  012702  020000                    MOV     #20000,R2
1928  003020  005000                            CLR     R0
1929  003022  005200                  1$:       INC     R0              ;R0 WILL BE DATA WRITTEN
1930  003024  012704  010000                    MOV     #4096.,R4       ;SET 4K COUNTER
1931  003030  010022                  2$:       MOV     R0,(R2)+        ;WRITE BANK # INTO ALL ADDRESSES
1932  003032  005304                            DEC     R4
1933  003034  001375                            BNE     2$
1934  003036  000771                            BR      1$
1935
1936  003040  022626                    DONE4A: CMP     (SP)+,(SP)+     ;ADJUST STACK PTR
1937
1938                                    ;CHECK THAT DATA WRITTEN ABOVE CAN BE READ
1939  003042  012737  003110  000004            MOV     #DONE4B,@#ERRVEC
1940  003050  010701                            MOV     PC,R1
1941  003052  004767  003766                    JSR     PC,LDMMO
1942  003056  012702  020000                    MOV     #20000,R2
1943  003062  005000                            CLR     R0
1944  003064  005200                  1$:       INC     R0
1945  003066  012704  010000                    MOV     #4096.,R4
1946  003072  012203                  2$:       MOV     (R2)+,R3
1947  003074  020003                            CMP     R0,R3
1948  003076  001401                            BEQ     .+4
1949  003100  104400                            HLT
1950  003102  005304                            DEC     R4
1951  003104  001372                            BNE     2$
1952  003106  000766                            BR      1$
1953  003110  022626                    DONE4B: CMP     (SP)+,(SP)+
1954
1955                                    ;ROUTINE TO WRITE CONSTANT DATA INTO 4K
1956                                    ;BANK STARTING WITH LAST MEMORY LOCATION
1957  003112  010701                            MOV     PC,R1
1958  003114  012737  007174  000250            MOV     #MMABT1,@#MMVEC
```

```
1959  003122  162702  000002              SUB     #2,R2
1960  003126  005000                      CLR     R0
1961  003130  005300              1$:     DEC     R0
1962  003132  012704  010000              MOV     #4096.,R4
1963  003136  010042              2$:     MOV     R0,-(R2)
1964  003140  102406                      BVS     DONE4C
1965  003142  020227  017776              CMP     R2,#17776       ;CHECK IF DONE
1966  003146  001403                      BEQ     DONE4C
1967  003150  005304                      DEC     R4
1968  003152  001371                      BNE     2$
1969  003154  000765                      BR      1$
1970
1971  003156  012737  003252  000004  DONE4C: MOV     #DONE4D,@#ERRVEC
1972  003164  010701                      MOV     PC,R1
1973  003166  004767  003652              JSR     PC,LDMM0
1974  003172  012737  007142  000250      MOV     #MMABT0,@#MMVEC ;SET ABORT VECTOR
1975  003200  012702  020000              MOV     #20000,R2
1976  003204  022704  010000      1$:     CMP     #4096.,R4       ;CHECK IF WRITE ABOVE STARTED ON
1977                                                              ;4K BOUNDARY
1978  003210  001406                      BEQ     2$
1979  003212  012203                      MOV     (R2)+,R3
1980  003214  020003                      CMP     R0,R3
1981  003216  001401                      BEQ     .+4
1982  003220  104400                      HLT
1983  003222  005204                      INC     R4
1984  003224  001367                      BNE     1$
1985  003226  005200              2$:     INC     R0
1986  003230  012704  010000              MOV     #4096.,R4
1987  003234  012203              3$:     MOV     (R2)+,R3
1988  003236  020003                      CMP     R0,R3
1989  003240  001401                      BEQ     .+4
1990  003242  104400                      HLT
1991  003244  005304                      DEC     R4
1992  003246  001372                      BNE     3$
1993  003250  000766                      BR      2$
1994
1995  003252  022626              DONE4D: CMP     (SP)+,(SP)+
1996  003254  005737  000042              TST     @#42            ;BRANCH IF PROGRAM WAS NOT
1997  003260  001406                      BEQ     BEGIN1          ;LOADED VIA ACT11 IN QV OR AA MODES
1998  003262  005767  001330              TST     LOGICAL+2       ;BRANCH IF NOT IN QV MODE
1999  003266  100003                      BPL     BEGIN1
2000  003270  012737  000006  000742      MOV     #6,@#ICNT       ;SET ICNT TO DO 1 PASS ONLY IN QV
2001
```

```
2002
2003                                        .SBTTL  WORST CASE NOISE TESTS
2004                                    ;THIS TEST WRITES MEMORY WORST CASE NOISE TEST PATTERNS THROUGHOUT
2005                                    ;MEMORY AND CHECKS THAT THEY CAN BE WRITTEN AND READ.
2006                                    ;SET UP TRAP VECTORS
2007  003276  012706  000500            BEGIN1: MOV     #STKPTR,SP      ;SET STACK PTR
2008  003302  004767  003436                    JSR     PC,..MAMF       ;GO ENABLE PARITY ACTION
2009  003306  004767  006336                    JSR     PC,CKSWR        ;GO CHECK SWITCHES
2010  003312  012737  003334  000004            MOV     #DONE5,@#ERRVEC ;SET UP TIME OUT TRAP
2011
2012                                    ;WRITE 1 XOR 8 TEST PATTERN STARTING AT ADDRESS 20000
2013  003320  010701                            MOV     PC,R1           ;UPDATE TRACE REGISTER
2014  003322  012746  000001                    MOV     #1,-(SP)        ;PUSH STARTING BANK # ON THE STACK
2015  003326  005046                            CLR     -(SP)           ;PUSH # OF 128. WORD BLOCKS TO WRITE
2016  003330  004767  004054                    JSR     PC,..1X8        ;GO TO ROUTINE TO WRITE 1 XOR 8 PATTERN
2017
2018                                    ;CHECK 1 XOR 8 TEST PATTERN WRITTEN ABOVE
2019  003334  012737  001116  000004    DONE5:  MOV     #ERRTRP,@#ERRVEC
2020  003342  012706  000500            1$:     MOV     #STKPTR,SP      ;SET STACK PTR
2021  003346  010701                            MOV     PC,R1           ;UPDATE TRACE REGISTER
2022  003350  012746  000001                    MOV     #1,-(SP)        ;PUSH STARTING BANK # ON THE STACK
2023  003354  005403                            NEG     R3              ;R3 CONTAINS # OF 128. WORD BLOCKS
2024  003356  010346                            MOV     R3,-(SP)        ;WRITTEN BY .1X8 ROUTINE ABOVE
2025  003360  004767  004136                    JSR     PC,..1X8        ;GO CHECK 1X8 PATTERN
2026
2027  003364  005027                            CLR     (PC)+           ;SET INDICATOR TO WRITE NORMAL 3X9 PAT
2028  003366  000000            PARPAT: .WORD   0
2029
2030                                    ;WRITE 3 XOR 9 TEST PATTERN STARTING AT ADDRESS 20000
2031                                    ;NOTE PATTERN IS NORMAL 3 XOR 9 IF NO PARITY MEMORY IS AVAILABLE,
2032                                    ;AND IS A MODIFIED PATTERN IF PARITY MEMORY IS AVAILABLE.
2033                                    ;THE CONTENTS OF PARPAT IF 0/NOT 0 INDICATE IF NORMAL/MODIFIED PATTERN
2034                                    ;IS BEING USED IN TESTS BELOW.
2035  003370  012706  000500            DONE6:  MOV     #STKPTR,SP      ;SET STACK PTR
2036  003374  010701                            MOV     PC,R1           ;UPDATE TRACE REGISTER
2037  003376  012737  003416  000004            MOV     #DONE7,@#ERRVEC ;SET TIME OUT TRAP VECTOR
2038  003404  012746  000001                    MOV     #1,-(SP)        ;PUSH STARTING BANK # ON STACK
2039  003410  005046                            CLR     -(SP)           ;PUSH # OF 256. WORD BLOCKS TO WRITE
2040  003412  004767  004340                    JSR     PC,..3X9        ;CALL ROUTINE TO WRITE 3XOR9 PATTERN
2041
2042                                    ;CHECK 3 XOR 9 TEST PATTERN WRITTEN ABOVE
2043  003416  012737  001116  000004    DONE7:  MOV     #ERRTRP,@#ERRVEC
2044  003424  016600  000006                    MOV     6(SP),R0        ;GET # OF 256. WORD BLOCKS WRITTEN
2045  003430  005400                            NEG     R0              ;FORM TWO'S COMPLEMENT
2046  003432  010027                            MOV     R0,(PC)+        ;SAVE # OF 256 WORD BLOCKS
2047  003434  000000            WDS.256:.WORD   0               ;CONTAINS # OF 256 WORD BLOCKS IN MEM.
2048  003436  012706  000500                    MOV     #STKPTR,SP      ;SET STACK PTR
2049  003442  010701                            MOV     PC,R1           ;SET SCOPE PTR
2050  003444  012746  000001                    MOV     #1,-(SP)        ;PUSH BANK # ON THE STACK
2051  003450  010046                            MOV     R0,-(SP)        ;PUSH # OF 256. WORD BLOCKS TO WRITE
2052  003452  004767  004520                    JSR     PC,..3X9        ;GO CHECK DATA WRITTEN
2053
2054                                    ;SETUP TO RUN MODIFIED 3 XOR 9 PATTERN IF PARITY MEMORY IS AVAILABLE
2055  003456  005737  007014                    TST     @#PARAVA        ;BRANCH IF PARITY MEMORY IS NOT AVAIL
2056  003462  001406                            BEQ     DONE8
2057  003464  005737  003366                    TST     @#PARPAT        ;BRANCH IF PARITY PAT JUST WRITTEN
```

```
2058  003470  001003                          BNE     DONE8
2059  003472  010637  003366                  MOV     SP,@#PARPAT     ;SET INDICATOR TO WRITE 3X9 PAR PAT
2060  003476  000734                          BR      DONE6           ;REPEAT TEST USING MODIFIED 3X9 PATTERN
2061
2062                                  ;WRITE 8 XOR 13 TEST PATTERN STARTING AT ADDRESS 40000
2063  003500  012706  000500          DONE8:  MOV     #STKPTR,SP      ;SET STACK PTR
2064  003504  012737  003526  000004          MOV     #DONE9,@#ERRVEC ;SET TIME OUT TRAP VECTOR
2065  003512  010701                          MOV     PC,R1           ;UPDATE TRACE REGISTER
2066  003514  012746  000002                  MOV     #2,-(SP)        ;PUSH STARTING BANK # ON THE STACK
2067  003520  005046                          CLR     -(SP)           ;PUSH # OF BANKS TO WRITE ON THE STACK
2068  003522  004767  005120                  JSR     PC,.8X13        ;GO TO ROUTINE TO WRITE DATA
2069
2070                                  ;CHECK 8 XOR 13 TEST PATTERN WRITTEN ABOVE
2071  003526  012706  000500          DONE9:  MOV     #STKPTR,SP      ;SET STACK PTR
2072  003532  010701                          MOV     PC,R1           ;UPDATE TRACE REGISTER
2073  003534  012737  001116  000004          MOV     #ERRTRP,@#ERRVEC
2074  003542  012746  000002                  MOV     #2,-(SP)
2075  003546  005404                          NEG     R4
2076  003550  042704  000001                  BIC     #1,R4           ;SET 4K BANK COUNT TO 8K INCREMENT
2077  003554  001403                          BEQ     DONE10          ;DO NOT CHECK IF ONLY 12K
2078  003556  010446                          MOV     R4,-(SP)
2079  003560  004767  005140                  JSR     PC,..8X13       ;GO CHECK 8 XOR 13 PATTERN WRITTEN ABOVE
2080
2081
2082                                  ;RELOCATE PROGRAM TO CHECK ADDRESSES FROM 000000-017776 USING 1 XOR 8 PATTERN
2083  003564  000005          DONE10: RESET                           ;DISABLE MEM MGMT AND PARITY ACTION
2084  003566  005737  000042          1$:     TST     @#42            ;CHECK IF PROGRAM LOADED VIA ACT11
2085  003572  001402                          BEQ     2$
2086  003574  000137  004256                  JMP     @#RANTST        ;DO NOT RELOCATE IF ACT11
2087  003600  012706  000500          2$:     MOV     #STKPTR,SP      ;SET STACK PTR
2088  003604  004567  002436                  JSR     5,RELOC         ;RELOCATE PROGRAM CODE
2089  003610  000000                          000000                  ;FROM 000000 TO
2090  003612  020000                          20000                   ;20000
2091  003614  005704                          TST     R4              ;WAS RELOCATION SUCCESSFUL?
2092  003616  001402                          BEQ     3$              ;BRANCH IF SUCCESSFUL I.E. WAS THERE
2093                                                                  ;SUFFICIENT MEMORY TO RELOCATE TO.
2094  003620  000137  004256                  JMP     @#RANTST        ;END OF TEST
2095  003624  062707  020000          3$:     ADD     #20000,PC       ;RELOCATE PC
2096  003630  062706  020000                  ADD     #20000,SP       ;SET NEW STACK PTR
2097  003634  052767  100000  175106          BIS     #100000,LDDISP  ;SET RELOCATION INDICATOR
2098  003642  016737  175102  177570          MOV     LDDISP,@#DISPLAY;LOAD DISPLAY REGISTER
2099
2100                                  ;**********************IMPORTANT NOTE**********************************
2101                                  ;PROGRAM IS NOW EXECUTING CODE FROM PC AS SHOWN BELOW +20000.
2102                                  ;CAUTION: DO NOT ATTEMPT TO RESTART PROGRAM AT 200
2103                                  ;**********************************************************************
2104
2105  003650  010701          DONE11: MOV     PC,R1                   ;RESTART ADDRESS TO LOOP TEST
2106
2107                                  ;WRITE 1 XOR 8 TEST PATTERN IN LOCATIONS 000000-017776
2108  003652  005046                          CLR     -(SP)           ;PUSH STARTING BANK # ON THE STACK
2109  003654  012746  000040                  MOV     #32.,-(SP)      ;PUSH # OF 128. WORD BLOCKS TO WRITE
2110  003660  004767  003524                  JSR     PC,.1X8         ;GO TO ROUTINE TO WRITE 1 XOR 8 DATA
2111
2112  003664  010701                          MOV     PC,R1           ;RESTART & LOOP TEST ADDRESS
2113
```

```
2114
2115  003666  005046                        ;CHECK 1 XOR 8 TEST PATTERN AS WRITTEN ABOVE
2116  003670  012746  000040                      CLR     -(SP)           ;PUSH STARTING BANK # ON THE STACK
2117  003674  004767  003622                      MOV     #32.,-(SP)      ;PUSH # OF 128. WORD BLOCKS TO WRITE
2118                                              JSR     PC,..1X8        ;GO TO ROUTINE TO CHECK DATA
2119
2120                                        ;RELOCATE PROGRAM TO CHECK ADDRESSES 000000 - 037776 USING
2121  003700  010701                       ;3 XOR 9 AND 8 XOR 13 PATTERNS
2122  003702  004567  002340               DONE12: MOV     PC,R1           ;UPDATE TRACE REGISTER
2123  003706  020000                              JSR     R5,RELOC        ;MOVE PROGRAM BACK  TO LOWEST 4K
2124  003710  000000                              20000                   ;FROM 20000 TO
2125  003712  000137  003716                      000000                  ;000000
2126  003716  012767  040000  000106              JMP     @#.+4           ;RETURN UNRELOCATED
2127  003724  012767  000040  000150              MOV     #40000,2$
2128  003732  042737  100000  000750              MOV     #32.,BLKCNT
2129  003740  016737  175004  177570              BIC     #100000,@#LDDISP        ;CLEAR RELOCATION INDICATOR
2130  003746  012737  004040  000004              MOV     LDDISP,@#DISPLAY        ;DISPLAY NOT RELOCATED
2131  003754  005737  057776                      MOV     #3$,@#ERRVEC    ;SET TIME OUT TRAP
2132  003760  012737  004002  000004              TST     @#057776        ;CHECK IF 12K OF MEMORY IS AVAILABLE
2133  003766  005737  117776                      MOV     #1$,@#ERRVEC
2134  003772  006367  000034                      TST     @#117776        ;CHECK IF 20K OF MEMORY
2135  003776  006367  000100                      ASL     2$
2136  004002  012706  000500                      ASL     BLKCNT
2137  004006  010701                       1$:    MOV     #STKPTR,SP      ;SET STACK POINTER
2138  004010  042737  100000  000750              MOV     PC,R1           ;UPDATE TRACE REGISTER
2139  004016  013737  000750  177570              BIC     #100000,@#LDDISP        ;CLEAR RELOCATION INDICATOR
2140  004024  004567  002216                      MOV     @#LDDISP,@#DISPLAY      ;LOAD DISPLAY REGISTER
2141  004030  000000                              JSR     R5,RELOC        ;RELOCATE PROGRAM
2142  004032  040000                              000000                  ;FROM 000000
2143  004034  005704                       2$:    40000                   ;TO 40000
2144  004036  001402                              TST     R4              ;RELOCATION SUCCESSFUL?
2145  004040  000137  004256                      BEQ     4$              ;YES
2146  004044  066707  177762               3$:    JMP     @#RANTST        ;GO TO RANDOM DATA TEST
2147  004050  066706  177756               4$:    ADD     2$,PC           ;RELOCATE PC
2148  004054  052767  100000  174666              ADD     2$,SP                   ;SET NEW STACK PTR
2149  004062  016737  174662  177570              BIS     #100000,LDDISP  ;SET RELOCATION INDICATOR
2150                                              MOV     LDDISP,@#DISPLAY;RELOAD DISPLAY REGISTER
2151
2152                                        ;*********************IMPORTANT NOTE*********************************
2153                                        ;PROGRAM IS NOW EXECUTING CODE FROM PC AS SHOWN BELOW +40000 OR +100000
2154                                        ;CAUTION: DO NOT ATTEMPT TO RESTART PROGRAM AT 200
2155                                        ;********************************************************************
2156  004070  005037  003366                      CLR     @#PARPAT        ;SET INDICATOR TO WRITE NORMAL 3X9 PAT
2157  004074  010701                       DONE13: MOV     PC,R1           ;UPDATE TRACE REGISTER
2158
2159                                        ;WRITE 3XOR9 TEST PATTERN IN LOCATIONS 000000-037776 OR 000000-077776
2160  004076  005046                              CLR     -(SP)           ;PUSH BANK # 0 ON THE STACK
2161  004100  012746                              MOV     (PC)+,-(SP)     ;PUSH 256. BLOCK WORD COUNT ON STACK
2162  004102  000000                       BLKCNT: .WORD   0               ;CONTAINS 256. BLOCK WORD COUNT LOADED
2163                                                                        ;BY ABOVE ROUTINE. 40/100 IF 8/16K
2164  004104  004767  003646                      JSR     PC,.3X9
2165
2166                                        ;CHECK PATTERN WRITTEN IN LOCATIONS 000000-037776 OR 000000-077776
2167  004110  010701                              MOV     PC,R1
2168  004112  005046                              CLR     -(SP)           ;PUSH STARTING BANK # ON THE STACK
2169  004114  016746  177762                      MOV     BLKCNT,-(SP)    ;PUSH 256. WORD BLOCK COUNT ON THE STACK
```

```
2170  004120  004767  004052                  JSR     PC,..3X9        ;CHECK
2171
2172                                  ;ROUTINE TO CHECK IF MODIFIED 3 XOR9 PATTERN SHOULD BE RUN
2173  004124  005737  007014                  TST     @#PARAVA        ;BRANCH IF PARITY MEMORY NOT AVAIL
2174  004130  001406                          BEQ     DONE14
2175  004132  005737  003366                  TST     @#PARPAT        ;BRANCH IF MODIFIED PATTERN JUST RUN
2176  004136  001003                          BNE     DONE14
2177  004140  010667  177222                  MOV     SP,PARPAT       ;SET INDICATOR TO WRITE MODIFIED PAT
2178  004144  000753                          BR      DONE13          ;LOOP TEST USING MODIFIED PATTERN
2179                                  ;WRITE 8 XOR 13 TEST PATTERN IN LOCATIONS 000000-037777
2180  004146  010701                DONE14:  MOV     PC,R1           ;UPDATE RESTART & LOOP ADDRESS
2181  004150  005046                          CLR     -(SP)           ;PUSH STARTING BANK # ON THE STACK
2182  004152  012746  000002                  MOV     #2,-(SP)        ;PUSH # OF BANKS TO WRITE ON THE STACK
2183  004156  004767  004464                  JSR     PC,.8X13        ;GO TO ROUTINE TO WRITE DATA
2184
2185                                  ;CHECK 8 XOR 13 PATTERN WRITTEN ABOVE
2186  004162  010701                          MOV     PC,R1           ;UPDATE RESTART & LOOP ADDRESS
2187  004164  005046                          CLR     -(SP)           ;PUSH BANK # ON THE STACK
2188  004166  012746  000002                  MOV     #2,-(SP)        ;AND # OF BANKS TO CHECK (2)
2189  004172  004767  004526                  JSR     PC,..8X13       ;GO TO CHECK ROUTINE
2190
2191                                  ;RELOCATE PROGRAM BACK TO LOWER MEMORY
2192  004176  012767  040000  000022 DONE15:  MOV     #40000,1$
2193  004204  022767  000040  177670          CMP     #40,BLKCNT
2194  004212  001402                          BEQ     .+6
2195  004214  006367  000006                  ASL     1$
2196  004220  010701                          MOV     PC,R1           ;UPDATE TRACE REGISTER
2197  004222  004567  002020                  JSR     R5,RELOC        ;MOVE PROGRAM BACK INTO LOWER
2198  004226  040000                  1$:     40000                   ;FROM 40000 OR 100000
2199  004230  000000                          000000                  ;TO 000000
2200  004232  000137  004236                  JMP     @#.+4           ;RETURN UNRELOCATED
2201  004236  012706  000500                  MOV     #STKPTR,SP      ;RESET STACK POINTER
2202  004242  042737  100000  000750          BIC     #100000,@#LDDISP        ;CLEAR RELOCATION INDICATOR
2203  004250  013737  000750  177570          MOV     @#LDDISP,@#DISPLAY      ;LOAD DISPLAY REGISTER
2204
2205                                          .SBTTL  RANDOM DATA,ROTATING 1/O TESTS
2206                                  ;RANDOM DATA TEST. THIS TEST MOVES THE PROGRAM CODE THROUGHOUT MEMORY
2207  004256  010701                RANTST:  MOV     PC,R1           ;SET TRACE POINTER
2208  004260  012737  004416  000004          MOV     #7$,@#ERRVEC    ;SET TIME OUT TRAP
2209  004266  005767  174460                  TST     MMAVA           ;CHECK IF MEM MGMT IS AVAILABLE
2210  004272  001412                          BEQ     1$              ;BRANCH IF NOT AVAILABLE
2211  004274  004767  002544                  JSR     PC,LDMMO        ;GO SET UP MEM MGMT
2212  004300  105237  172301                  INCB    @#KIPDR0+1      ;ALLOW 4K ADDRESSING IN FIRST 4K
2213  004304  012737  077406  172304          MOV     #200*256.-400+UP+RW,@#KIPDR2    ;SET KIPDR2=RW UP 200 BLOCKS
2214  004312  012737  000400  172344          MOV     #400,@#KIPAR2
2215  004320  012702  020000          1$:     MOV     #20000,R2       ;SET 'TO' ADDRESS POINTER
2216  004324  005004                          CLR     R4              ;SET 'FROM' ADDRESS POINTER
2217  004326  012705  004000          2$:     MOV     #2048.,R5       ;SET 4K WORD COUNT
2218  004332  012422                  3$:     MOV     (R4)+,(R2)+     ;MOVE CODE
2219  004334  012422                          MOV     (R4)+,(R2)+
2220  004336  005305                          DEC     R5              ;DECREMENT 4K WORD COUNTER
2221  004340  001374                          BNE     3$
2222
2223  004342  012705  005473                  MOV     #4096.-PLACE+1,R5       ;SET 4K WORD COUNTER
2224  004346  014400                  4$:     MOV     -(R4),R0        ;GET 'GOOD' DATA
2225  004350  014203                          MOV     -(R2),R3        ;GET 'BAD' DATA
```

```
2226  004352  020003                        CMP     R0,R3           ;COMPARE 'GOOD' & 'BAD' DATA
2227  004354  001403                        BEQ     5$
2228  004356  005722                        TST     (R2)+           ;STEP ADDRESS FOR ERROR ROUTINE
2229  004360  104400                        HLT                     ;REPORT ERROR
2230  004362  005742                        TST     -(R2)           ;RESTORE ADDRESS POINTER
2231  004364  005305                5$:     DEC     R5              ;DECREMENT 4K WORD COUNTER
2232  004366  001367                        BNE     4$              ;LOOP UNTIL 4K WORDS CHECKED
2233
2234  004370  005767  174356              TST     MMAVA           ;CHECK IF MEM MGMT IS AVAILABLE
2235  004374  001405                        BEQ     6$              ;BRANCH IF NOT AVAILABLE
2236  004376  005237  172342              INC     @#KIPAR1
2237  004402  005237  172344              INC     @#KIPAR2
2238  004406  000744                        BR      1$
2239  004410  062702  000100          6$:   ADD     #64.,R2         ;STEP ADDRESS
2240  004414  000744                        BR      2$
2241  004416  012706  000500          7$:   MOV     #STKPTR,SP      ;RESET STACK PTR
2242  004422  012737  001116  000004        MOV     #ERRTRP,@#ERRVEC;RESTORE ERROR TRAP VECTOR
2243
2244                                  ;ROTATING 0 TEST. THIS TEST ROTATES A SINGLE '0' THROUGH MEMORY
2245  004430  012767  177777  000744  ROT0:  MOV     #-1,.CONST      ;SET CONSTANT =177777
2246  004436  012746  000001                MOV     #1,-(SP)        ;SET BANK #1
2247  004442  016746  176766                MOV     WDS.256,-(SP)   ;GET # OF 256. WORD BLOCKS IN MEMORY
2248  004446  004767  004676                JSR     PC,.USER        ;GO WRITE 1'S THROUGHOUT MEMORY
2249  004452  010701                        MOV     PC,R1           ;SET SCOPE PTR
2250  004454  012746  000001                MOV     #1,-(SP)        ;SET STARTING BANK #
2251  004460  016746  176750                MOV     WDS.256,-(SP)   ;SET # OF 256. WORD BLOCKS TO CHECK
2252  004464  004767  004430                JSR     PC,.ROT0        ;GO TO ROTATE 0 ROUTINE
2253
2254                                  ;ROTATING 1 TEST THIS TEST ROTATES A SINGLE '1' BIT THROUGH ALL OF
2255                                  ;MEMORY
2256  004470  005067  000706  ROT1:  CLR     .CONST          ;CLEAR CONSTANT
2257  004474  012746  000001                MOV     #1,-(SP)        ;PUSH STARTING BANK ONTO STACK
2258  004500  016746  176730                MOV     WDS.256,-(SP)   ;AND # OF 256. WORD BLOCKS IN MEMORY
2259  004504  004767  004640                JSR     PC,.USER        ;GO WRITE 0'S THROUGHOUT MEMORY
2260  004510  010701                        MOV     PC,R1           ;SET SCOPE PTR
2261  004512  012746  000001                MOV     #1,-(SP)        ;SET STARTING BANK #
2262  004516  016746  176712                MOV     WDS.256,-(SP)   ;SET # OF 256. WORD BLOCKS TO CHECK
2263  004522  004767  004466                JSR     PC,.ROT1        ;GO ROTATE A '1' BIT THROUGHOUT MEMORY
2264
2265                                  ;END  OF CYCLE
2266  004526  000005                END:    RESET
2267  004530  010701                        MOV     PC,R1           ;UPDATE TRACE REGISTER
2268  004532  012706  000500                MOV     #STKPTR,SP      ;SET STACK PTR
2269  004536  005237  000742                INC     @#ICNT          ;INCREMENT PASS COUNT
2270  004542  022737  000007  000742        CMP     #7,@#ICNT       ;8 PASSES?
2271  004550  001405                        BEQ     DONE            ;BRANCH IF 8 PASSES COMPLETED
2272  004552  004567  174202                JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
2273  004556  012340                        ASTERISK
2274  004560  000137  003276                JMP     @#BEGIN1
2275  004564                          DONE:
2276  004564  004567  174170                JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
2277  004570  012342                        ENDMSG
2278  004572  105737  177564                TSTB    @#TPS           ;WAIT FOR BELL TO RING
2279  004576  100375                        BPL     .-4
2280  004600  013700  000042                MOV     @#42,R0         ;GET DECTAPE MONITOR RETURN ADDRESS
2281  004604  001407                        BEQ     FINISH
```

```
 2282  004606  004767  175322                  JSR     PC,$RLDR        ;RESTORE MONITOR & LOADERS
 2283  004612  000005                          RESET
 2284  004614  004710                LOGICAL:  JSR     PC,(R0)         ;GO TO DECTAPE MONITOR
 2285  004616  000240                          NOP
 2286  004620  000240                          NOP
 2287  004622  000240                          NOP
 2288  004624  000167  175522        FINISH:   JMP     START1
 2289
 2290                                          .SBTTL  USER TESTS
 2291                                          .SBTTL  PROGRAM # 2
 2292                                ;PROGRAM # 2
 2293                                ;THIS PROGRAM ALLOWS THE USER TO SELECT A STARTING 4K BANK #,# OF 4K
 2294                                ;BANKS TO TEST, AND A WORST CASE PATTERN TO WRITE AND CHECK.
 2295  004630  005067  001062        PRG2:     CLR     PRG3FLG         ;CLEAR PROGRAM # 3 FLAG
 2296                                ;NOTE:    PROGRAM 3 ENTERS PROGRAM # 2 HERE, WITH PRG3FLG = 1.
 2297  004634  012706  000500        PRG2A:    MOV     #STKPTR,SP      ;SET STACK PTR
 2298  004640  005037  006520                  CLR     @#PEFLG         ;CLEAR PARITY ERROR INDICATORS
 2299  004644  005027                          CLR     (PC)+           ;CLEAR RUNNING ALL PATTERNS FLAG
 2300  004646  000000        ALLFLG:   .WORD   0                       ;CONTAINS RUNNING ALL PATERNS FLAG
 2301                                                                  ;0/-1 = RUNNING SELECTED/ALL PATTERNS
 2302  004650  012737  000502  000024        MOV     #PDWN,@#PFVEC   ;SET POWER FAIL TRAP VECTOR
 2303  004656  012737  000006  000004        MOV     #ERRVEC+2,@#ERRVEC
 2304  004664  005067  174062                CLR     MMAVA
 2305  004670  032737  010000  177570        BIT     #BIT12,@#SWR    ;CHECK IF TEST IS TO BE RUN
 2306  004676  001006                        BNE     1$              ;WITH MEM MGMT ENABLED
 2307  004700  000261                        SEC
 2308  004702  005737  177572                TST     @#SR0           ;CHECK IF MEM MGMT IS AVAILABLE
 2309  004706  103402                        BCS     1$              ;BRANCH IF NOT AVAILABLE
 2310  004710  005167  174036                COM     MMAVA
 2311  004714  012737  001116  000004  1$:   MOV     #ERRTRP,@#ERRVEC
 2312  004722  005037  000742                CLR     @#ICNT
 2313  004726  005037  000750                CLR     @#LDDISP
 2314  004732  005037  003366                CLR     @#PARPAT        ;SET INDICATOR TO WRITE NORM 3X9 PAT
 2315  004736  004567  174016                JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
 2316  004742  012016                        PARITY
 2317  004744  004767  004616                JSR     PC,RECD         ;GO GET ANSWER TO THE PARITY QUESTION
 2318  004750  000000        .PARIT:   .WORD   0                       ;TYPE 1 IF PARITY DESIRED 0 IF NOT
 2319  004752  005767  177772                TST     .PARIT
 2320  004756  001402                        BEQ     1$              ;BRANCH IF PARITY NOT DESIRED
 2321  004760  004767  001760                JSR     PC,.MAMF        ;GO ENABLE PARITY ACTION
 2322  004764                          1$:
 2323  004764  004567  173770                JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
 2324  004770  012053                        STBANK                  ;ASK USER FOR STARTING BANK
 2325  004772  004767  004570                JSR     PC,RECD
 2326  004776  000000        .STBANK:  .WORD   0                       ;CONTAINS STARTING BANK #
 2327  005000  005767  177772                TST     .STBANK         ;CHECK IF STARTING AT BANK #0
 2328  005004  001002                        BNE     3$
 2329  005006  004767  001336                JSR     PC,RELOCP       ;GO RELOCATE THE PROGRAM TO TOP OF MEM
 2330  005012                          3$:
 2331  005012  004567  173742                JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
 2332  005016  012102                        BANKS                   ;ASK USER FOR # OF 4K BANKS TO TEST
 2333  005020  004767  004542                JSR     PC,RECD
 2334  005024  000000        .BANKS:   .WORD   0                       ;CONTAINS # OF BANKS TO CHECK
 2335  005026  004567  173726                JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
 2336  005032  012137                        PAT                     ;ASK USER WHICH PATTERN
 2337  005034  004767  004526                JSR     PC,RECD
```

```
2338  005040  000000                  .PAT:   .WORD   0               ;CONTAINS PATTERN #
2339  005042  012767  005042  175316  PRG2R:  MOV     #PRG2R,PERSTRT  ;SET PAR ERROR RESTART ADDRESS
2340  005050  004767  004574          .       JSR     PC,CKSWR        ;GO CHECK SWITCHES
2341  005054  016700  177760          .       MOV     .PAT,R0
2342  005060  006300                          ASL     R0              ;SHIFT PATTERN # TO FORM INDEX
2343  005062  066700  173666                  ADD     RELOCF,R0       ;ADD RELOCATION FACTOR
2344  005066  016000  005100                  MOV     WRTTAB(0),R0    ;GET UNRELOCATED PC OF ROUTINE
2345  005072  066700  173656                  ADD     RELOCF,R0       ;ADD RELOCATION FACTOR
2346  005076  010007                          MOV     R0,PC           ;GO TO APPROPRIATE ROUTINE
2347
2348                                  ;TABLE OF ROUTINES TO WRITE SELECTED PATTERNS
2349  005100  005120                  WRTTAB: .WORD   $1X8             ;1 XOR 8 ROUTINE          (0)
2350  005102  005210                          .WORD   $3X9             ;3 XOR 9 ROUTINE          (1)
2351  005104  005276                          .WORD   $8X13            ;8 XOR 13 ROUTINE         (2)
2352  005106  005370                          .WORD   $USER            ;USER ROUTINE             (3)
2353  005110  005476                          .WORD   $ROT0            ;ROTATING '0' ROUTINE     (4)
2354  005112  005572                          .WORD   $ROT1            ;ROTATING '1' ROUTINE     (5)
2355  005114  005666                          .WORD   $3X9P            ;PARITY 3 XOR 9 PATTERN   (6)
2356  005116  005676                          .WORD   $ALL             ;ALL EXCEPT USER          (7)
2357
2358                                  ;ROUTINES
2359  005120  016746  177652          $1X8:   MOV     .STBANK,-(SP)   ;GET STARTING BANK #
2360  005124  016746  177674                  MOV     .BANKS,-(SP)    ;GET # OF 4K BANKS
2361  005130  006316                          ASL     (SP)            ;MULTIPLY BY 32.
2362  005132  005767  000560                  TST     PRG3FLG         ;IF PROGRAM # 3 STOP WITH 256.
2363  005136  001004                          BNE     1$              ;WORD BLOCK COUNT
2364  005140  006316                          ASL     (SP)            ;TO FORM 128.
2365  005142  006316                          ASL     (SP)            ;WORD BLOCK
2366  005144  006316                          ASL     (SP)            ;COUNT
2367  005146  006316                          ASL     (SP)
2368  005150  011627                  1$:     MOV     (SP),(PC)+      ;SAVE
2369  005152  000000                  2$:     .WORD   0               ;CONTAINS 128. WORD BLOCK COUNT
2370  005154  004767  002230                  JSR     PC,.1X8         ;GO WRITE 1 XOR 8 TEST PATTERN
2371
2372  005160  016746  177612                  MOV     .STBANK,-(SP)   ;GET STARTING BANK #
2373  005164  016746  177762                  MOV     2$,-(SP)        ;GET 128. WORD BLOCK COUNT
2374  005170  004767  002326                  JSR     PC,..1X8        ;GO CHECK PATTERN
2375  005174  005267  173542                  INC     ICNT
2376  005200  005767  177442                  TST     ALLFLG          ;CHECK IF RUNNING ALL PATERNS
2377  005204  001001                          BNE     $3X9            ;GO START 3X9 PATTERN IF RUNNING ALL
2378  005206  000744                          BR      $1X8            ;OTHERWISE LOOP
2379
2380  005210  016746  177562          $3X9:   MOV     .STBANK,-(SP)   ;GET STARTING BANK #
2381  005214  016746  177604                  MOV     .BANKS,-(SP)    ;GET # OF BANKS
2382  005220  005767  000472                  TST     PRG3FLG         ;IF PROGRAM # 2 STOP WITH 256.
2383  005224  001004                          BNE     1$              ;WORD BLOCK COUNT
2384  005226  006316                          ASL     (SP)            ;MULTIPLY BY 16.
2385  005230  006316                          ASL     (SP)            ;TO FORM
2386  005232  006316                          ASL     (SP)            ;256. WORD
2387  005234  006316                          ASL     (SP)            ;BLOCK COUNT
2388  005236  011627                  1$:     MOV     (SP),(PC)+      ;SAVE
2389  005240  000000                  2$:     .WORD   0               ;CONTAINS 256. WORD BLOCK COUNT
2390  005242  004767  002510                  JSR     PC,.3X9         ;GO WRITE PATTERN
2391
2392                                  ;CHECK PATTERN WRITTEN ABOVE
2393  005246  016746  177524          3$:     MOV     .STBANK,-(SP)   ;GET STARTING BANK #
```

```
2394  005252  016746  177762          MOV     2$,-(SP)        ;GET # OF 256. WORD BLOCKS
2395  005256  004767  002714          JSR     PC,..3X9        ;GO CHECK PATTERN
2396  005262  005267  173454          INC     ICNT
2397  005266  005767  177354          TST     ALLFLG          ;CHECK IF RUNNING ALL PATTERNS
2398  005272  001001                  BNE     $8X13           ;GO START 8X13 PATTERN IF RUNNING ALL
2399  005274  000764                  BR      3$
2400
2401  005276  005737  005716  $8X13:  TST     @#PRG3FLG       ;CANNOT DO 8X13 PATTERN USING
2402  005302  001007                  BNE     1$              ;PROGRAM # 3
2403  005304  016746  177466          MOV     .STBANK,-(SP)   ;GET STARTING BANK #
2404  005310  016746  177510          MOV     .BANKS,-(SP)    ;AND # OF 4K BANKS
2405  005314  032716  000001          BIT     #1,(SP)         ;MUST BE AN EVEN # OF 4K BANKS
2406  005320  001405                  BEQ     2$
2407  005322                  1$:
2408  005322  004567  173432          JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
2409  005326  012155                  QUEST                   ;PRINT ?
2410  005330  000167  177274          JMP     PRG2            ;RESTART
2411
2412  005334  004767  003306  2$:     JSR     PC,.8X13        ;GO WRITE PATTERN
2413  005340  016746  177432          MOV     .STBANK,-(SP)
2414  005344  016746  177454          MOV     .BANKS,-(SP)
2415  005350  004767  003350          JSR     PC,..8X13       ;GO CHECK PATTERN
2416  005354  005267  173362          INC     ICNT
2417  005360  005767  177262          TST     ALLFLG          ;CHECK IF RUNNING ALL PATERNS
2418  005364  001044                  BNE     $ROTO           ;GO DO ROTO PATTERN IF ALL SELECTED
2419  005366  000743                  BR      $8X13
2420
2421
2422                                  ;ROUTINE TO WRITE & CHECK USER CONSTANT
2423  005370                  $USER:
2424  005370  004567  173364          JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
2425  005374  012161                  CONST                   ;ASK FOR USER CONSTANT
2426  005376  004767  004164          JSR     PC,RECD
2427  005402  000000          .CONST: .WORD   0               ;CONTAINS USER CONSTANT
2428  005404  016746  177366  1$:     MOV     .STBANK,-(SP)   ;GET STARTING BANK #
2429  005410  016746  177410          MOV     .BANKS,-(SP)    ;GET 4K COUNT
2430  005414  005767  000276          TST     PRG3FLG
2431  005420  001004                  BNE     2$
2432  005422  006316                  ASL     (SP)            ;MULTIPLY 4K BANK COUNT BY 16.
2433  005424  006316                  ASL     (SP)            ;TO FORM 256. WORD BLOCK COUNT
2434  005426  006316                  ASL     (SP)
2435  005430  006316                  ASL     (SP)
2436  005432  011627          2$:     MOV     (SP),(PC)+      ;SAVE
2437  005434  000000          3$:     .WORD   0
2438  005436  004767  003706          JSR     PC,.USER        ;GO WRITE USER CONSTANT
2439  005442  016746  177330          MOV     .STBANK,-(SP)   ;GET STARTING BANK #
2440  005446  016746  177762          MOV     3$,-(SP)        ;AND # OF 256. WORD BLOCKS
2441  005452  004767  003744          JSR     PC,..USER       ;GO TO USER CHECK ROUTINE
2442  005456  005267  173260          INC     ICNT
2443  005462  105737  177560          TSTB    @#TKS           ;CHECK IF USER HAS TYPED A CHARACTER
2444  005466  100346                  BPL     1$
2445  005470  005737  177562          TST     @#TKB           ;CLEAR FLAG
2446  005474  000735                  BR      $USER
2447
2448                                  ;ROTATING '0' ROUTINE
2449  005476  016746  177274  $ROTO:  MOV     .STBANK,-(SP)   ;GET STARTING BANK #
```

```
2450  005502  016746  177316              MOV     .BANKS,-(SP)    ;GET # OF BANKS
2451  005506  005767  000204              TST     PRG3FLG
2452  005512  001004                      BNE     2$
2453  005514  006316                      ASL     (SP)            ;MULTIPLY 4K BANK COUNT BY 16.
2454  005516  006316                      ASL     (SP)            ;TO FORM 256. WORD BLOCK COUNT
2455  005520  006316                      ASL     (SP)
2456  005522  006316                      ASL     (SP)
2457  005524  011627              2$:     MOV     (SP),(PC)+      ;SAVE
2458  005526  000000              3$:     .WORD   0               ;CONTAINS 256. WORD BLOCK COUNT
2459  005530  012767  177777  177644      MOV     #-1,.CONST      ;SET CONSTANT
2460  005536  004767  003606              JSR     PC,.USER        ;GO WRITE ALL 1'S THROUH MEMORY
2461  005542  016746  177230              MOV     .STBANK,-(SP)   ;GET STARTING BANK #
2462  005546  016746  177754              MOV     3$,-(SP)        ;GET # OF 256. WORD BLOCKS TO CHECK
2463  005552  004767  003342              JSR     PC,.ROT0        ;GO CHECK ROTATING 0 PATTERN
2464  005556  005267  173160              INC     ICNT            ;INCREMENT DISPLAY COUNT
2465  005562  005767  177060              TST     ALLFLG          ;CHECK IF RUNNING ALL PATERNS
2466  005566  001001                      BNE     $ROT1           ;GO TO $ROT1 IF RUNNING ALL PATTERNS
2467  005570  000742                      BR      $ROT0           ;LOOP
2468
2469  005572  016746  177200      $ROT1:  MOV     .STBANK,-(SP)   ;GET STARTING BANK #
2470  005576  016746  177222              MOV     .BANKS,-(SP)    ;GET # OF 4 K BANKS
2471  005602  005767  000110              TST     PRG3FLG         ;CHECK IF RUNNING PROGRAM 3
2472  005606  001004                      BNE     2$              ;BRANCH IF RUNNING PROGRAM 3
2473  005610  006316                      ASL     (SP)            ;SHIFT 4K BANK COUNT BY 16.
2474  005612  006316                      ASL     (SP)            ;TO FORM 256. WORD BLOCK COUNT
2475  005614  006316                      ASL     (SP)
2476  005616  006316                      ASL     (SP)
2477  005620  011627              2$:     MOV     (SP),(PC)+      ;SAVE
2478  005622  000000              3$:     .WORD   0               ;CONTAINS 256. WORD BLOCK COUNT
2479  005624  005067  177552              CLR     .CONST          ;SET CONSTANT
2480  005630  004767  003514              JSR     PC,.USER        ;GO WRITE 0'S THROUGHOUT
2481  005634  016746  177136              MOV     .STBANK,-(SP)   ;GET STARTING BANK #
2482  005640  016746  177756              MOV     3$,-(SP)        ;AND 256. WORD BLOCK COUNT
2483  005644  004767  003344              JSR     PC,.ROT1        ;GO CHECK ROTATING 1 PATTERN
2484  005650  005267  173066              INC     ICNT            ;INCREMENT PASS COUNT
2485  005654  005767  176766              TST     ALLFLG          ;CHECK IF RUNNING ALL PATTERNS
2486  005660  001744                      BEQ     $ROT1           ;LOOP IF NOT RUNNING ALL PATTERNS
2487  005662  000167  177232              JMP     $1X8            ;GO DO $1X8 PATTERN
2488
2489                              ;ROUTINE TO CHECK MEMORY USING 3 XOR 9 PARITY PATTERN
2490  005666  010667  175474      $3X9P:  MOV     SP,PARPAT       ;SET INDICATOR TO WRITE PARITY PATTERN
2491  005672  000167  177312              JMP     $3X9
2492
2493                              ;ALL PATTERNS
2494  005676  010667  176744      $ALL:   MOV     SP,ALLFLG       ;SET INDICATOR
2495  005702  000167  177212              JMP     $1X8            ;BEGIN WITH 1X8 TEST PATTERN
2496                              .SBTTL  PROGRAM # 3
2497                              ;THIS PROGRAM IS THE SAME AS PROGRAM # 2 ABOVE EXCEPT THAT 256. WORD
2498                              ;DATA BLOCKS MAY BE WRITTEN
2499  005706  012706  000500      PRG3:   MOV     #STKPTR,SP      ;SET STACK PTR
2500  005712  012727  000001              MOV     #1,(PC)+        ;SET PROGRAM 3 FLAG
2501  005716  000000              PRG3FLG:.WORD   0               ;CONTAINS PRG3 INDICATOR
2502  005720  004567  173034              JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
2503  005724  012201                      PRG3M
2504  005726  004567  173026              JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
2505  005732  012102                      BANKS
```

```
2506  005734  000167  176674                  JMP     PRG2A           ;GO TO PROGRAM 2
2507
2508                                          .SBTTL  PROGRAM # 4
2509                                  ;THIS PROGRAM MAY BE USED TO READ/WRITE A USER CONTANT INTO ANY
2510                                  ;MEMORY LOCATION
2511  005740  012706  000500          PRG4:   MOV     #STKPTR,SP      ;SET STACK PTR
2512  005744  005037  006520                  CLR     @#PEFLG         ;CLEAR PARITY ERROR INDICATORS
2513  005750  012737  001116  000004          MOV     #ERRTRP,@#ERRVEC
2514  005756  000005                          RESET
2515  005760  012707  005764                  MOV     #.+4,PC         ;RELOCATE BACK TO FIRST 4K
2516  005764  004567  172770                  JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
2517  005770  012262                          PRG4M
2518  005772  004767  003570                  JSR     PC,RECD
2519  005776  000000          $ADRS0: .WORD   0               ;CONTAINS ADDRESS BITS <15-0>
2520  006000  016767  003634  000236          MOV     .1617,.EXTAD    ;GET EXTENDED ADDRESS BITS <17-16>
2521  006006  006037  177570                  ROR     @#SWR           ;CHECK SWITCH 0
2522  006012  103406                          BCS     PRG4A           ;GO TO PRG4A IF SET
2523  006014  004567  172740                  JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
2524  006020  012161                          CONST
2525  006022  004767  003540                  JSR     PC,RECD
2526  006026  000000          $CONST: .WORD   0               ;CONTAINS USER CONSTANT
2527  006030  012767  006030  174330  PRG4A:  MOV     #PRG4A,PERSTRT  ;SET RESTART ADDRESS ON PAR ERROR
2528  006036  016703  177734                  MOV     $ADRS0,R3       ;GET ADDRESS BITS <15-0>
2529  006042  016704  000176                  MOV     .EXTAD,R4       ;GET ADDRESS BITS <17-16>
2530  006046  001020                          BNE     10$             ;BRANCH IF MEM MGMT REQUIRED
2531  006050  022703  020000                  CMP     #20000,R3       ;CHECK IF ADDRESS IS LESS THAN 20000
2532  006054  101412                          BLOS    1$              ;BRANCH IF IT IS NOT
2533  006056  006037  177570                  ROR     @#SWR           ;CHECK IF READING
2534  006062  103431                          BCS     3$              ;BRANCH IF READING
2535  006064  004767  172214                  JSR     PC,$SAVR        ;GO SAVE REGISTERS ON THE STACK
2536  006070  004767  000254                  JSR     PC,RELOCP       ;GO RELOCATE PROGRAM
2537  006074  004767  172230                  JSR     PC,$RESTR       ;RESTORE REGISTERS FROM STACK
2538  006100  000422                          BR      3$              ;GO TO 3$
2539  006102  022703  160000          1$:     CMP     #160000,R3      ;CHECK IF MEM MGMT WILL BE REQUIRED
2540  006106  101017                          BHI     3$              ;GO TO 3$ IF NOT REQUIRED
2541  006110  010302                  10$:    MOV     R3,R2           ;GET ADDRESS BITS <15-0>
2542  006112  012700  000006                  MOV     #6,R0           ;SET SHIFT COUNT
2543  006116  006204                  2$:     ASR     R4              ;SHIFT 18 BIT ADDRESS
2544  006120  006003                          ROR     R3              ;6 PLACES RIGHT
2545  006122  077003                          SOB     R0,2$
2546  006124  004767  000714                  JSR     PC,LDMM0        ;GO SETUP MEM MGMT
2547  006130  010337  172342                  MOV     R3,@#KIPAR1     ;SET KIPAR1
2548  006134  042702  177700                  BIC     #177700,R2      ;CLEAR ADDRESS BITS <15-6>
2549  006140  052702  020000                  BIS     #20000,R2       ;SET ADDRESS REGISTER
2550  006144  000401                          BR      4$              ;GO TO 4$
2551  006146  010302                  3$:     MOV     R3,R2           ;SET ADDRESS REGISTER
2552  006150  016700  177652          4$:     MOV     $CONST,R0       ;GET USER CONSTANT
2553  006154  012737  005740  000060          MOV     #PRG4,@#TKVEC   ;SET KEYBOARD INTERRUPT VECTOR
2554  006162  052737  000100  177560          BIS     #100,@#TKS      ;SET IE BIT IN KEYBOARD CSR
2555  006170  006037  177570                  ROR     @#SWR           ;CHECK SWITCH 0
2556  006174  103014                          BCC     5$              ;BRANCH IF NOT SET
2557  006176  011246                          MOV     (R2),-(SP)      ;PUSH DATA TO BE TYPED ONTO STACK
2558  006200  004767  173542                  JSR     PC,O2A          ;GO TYPE DATA
2559  006204  004567  172550                  JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
2560  006210  015020                          $CRLF
2561  006212  062767  000002  177556          ADD     #2,$ADRS0       ;STEP ADDRESS
```

```
2562  006220  005567  000020              ADC     .EXTAD
2563  006224  000701                      BR      PRG4A
2564  006226  010012              5$:     MOV     R0,(R2)         ;WRITE USER CONSTANT INTO ADDRESS
2565  006230  012203                      MOV     (R2)+,R3        ;GET DATA WRITTEN
2566  006232  020003                      CMP     R0,R3           ;CHECK DATA
2567  006234  001401                      BEQ     6$
2568  006236  104400                      HLT                     ;REPORT ERROR
2569  006240  005742              6$:     TST     -(R2)           ;RESTORE ADDRESS
2570  006242  000771                      BR      5$              ;LOOP BACK
2571  006244  000000              .EXTAD: .WORD   0               ;CONTAINS EXTENDED ADDRESS BITS
2572                                      .SBTTL  PROGRAM SUBROUTINES
2573                                      .SBTTL  RELOCATION ROUTINES
2574                               ;ROUTINE TO RELOCATE PROGRAM CODE
2575  006246  012500              RELOC:  MOV     (R5)+,R0        ;GET FROM ADDRESS
2576  006250  011502                      MOV     (R5),R2         ;GET TO ADDRESS
2577  006252  010203                      MOV     R2,R3
2578  006254  062703  017776              ADD     #17776,R3       ;MOVES 4K
2579  006260  012737  006330  000004      MOV     #4$,@#ERRVEC    ;SET TIME OUT TRAP
2580  006266  005004                      CLR     R4              ;CLEAR RELOCATION SUCCESSFUL INDICATOR
2581  006270  005723                      TST     (R3)+           ;CHECK IF MEMORY IS AVAILABLE
2582  006272  012022              1$:     MOV     (R0)+,(R2)+     ;RELOCATE
2583  006274  020203                      CMP     R2,R3           ;RELOCATION COMPLETE?
2584  006276  001375                      BNE     1$
2585  006300  011503                      MOV     (R5),R3
2586  006302  020203              2$:     CMP     R2,R3
2587  006304  001413                      BEQ     5$              ;BRANCH IF DONE
2588  006306  024042                      CMP     -(R0),-(R2)     ;CHECK THAT DATA WAS RELOCATED PROPERLY
2589  006310  001774                      BEQ     2$
2590  006312  005703                      TST     R3              ;CHECK IF RELOCATING BACK TO 000000
2591  006314  001403                      BEQ     3$
2592  006316  104400                      HLT                     ;ERROR! CANNOT RELOCATE PROGRAM CODE
2593                                                              ;TO UPPER MEMORY BANK PROPERLY
2594  006320  000000                      HALT
2595  006322  000767                      BR      2$              ;CONTINUE RELOCATING AT YOUR PERIL
2596  006324  000000              3$:     HALT                    ;ERROR! CANNOT RELOCATE CODE BACK TO
2597                                                              ;TO 000000 PROPERLY
2598  006326  000777                      BR      .
2599  006330  022626              4$:     CMP     (SP)+,(SP)+     ;RESTORE STACK PTR
2600  006332  005104                      COM     R4
2601  006334  000240              5$:     NOP
2602  006336  012702  000754              MOV     #RELOCF,R2      ;GET ADDRESS OF RELOCATION FACTOR
2603  006342  061502                      ADD     (R5),R2         ;ADD FACTOR
2604  006344  012512                      MOV     (R5)+,(R2)      ;RELOCATED RELOCF NOW CONTAINS RELOCATION
2605                                                              ;FACTOR
2606  006346  000205                      RTS     5               ;RETURN, R4=-1 IF NO RELOCATION
2607
2608
2609                               ;ROUTINE TO RELOCATE PROGRAM CODE FROM ORIGINAL POSITION (0-4K) TO
2610                               ;TOP OF MEMORY.
2611  006350  012700  020000      RELOCP: MOV     #20000,R0       ;SET UP TO SCAN FOR TOP OF MEMORY
2612  006354  012737  000006  000004      MOV     #ERRVEC+2,@#ERRVEC
2613  006362  062700  020000      1$:     ADD     #20000,R0       ;INCREMENT SCAN ADDRESS
2614  006366  000261                      SEC                     ;SET TIME OUT INDICATOR
2615  006370  005710                      TST     (R0)            ;CHECK FOR EXISTANT MEMORY
2616  006372  103373                      BCC     1$              ;'C' WILL BE CLEAR IF MEMORY EXISTS
2617  006374  012737  001116  000004      MOV     #ERRTRP,@#ERRVEC
```

```
2618  006402  162700  020000          SUB     #20000,R0       ;ADJUST TO LAST EXISTANT 4K
2619  006406  010067  000006          MOV     R0,2$           ;PASS RELOCATION ADDRESS TO RELOC ROUTINE
2620  006412  004567  177630          JSR     R5,RELOC        ;RELOCATE PROGRAM
2621  006416  000000                  000000                  ;FROM ADDRESS 000000
2622  006420  000000          2$:     .WORD   0               ;TO LAST 4K BANK
2623  006422  004567  172332          JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
2624  006426  012301                  RELOCM
2625  006430  016746  177764          MOV     2$,-(SP)        ;PASS TO 02A ROUTINE
2626  006434  062716  012356          ADD     #REL24K,(SP)    ;SET UP RESTART ADDRESS
2627  006440  004767  173302          JSR     PC,02A          ;TYPE RESTART ADDRESS
2628  006444  011667  000006          MOV     (SP),3$         ;SAVE RETURN ADDRESS IN 3$ BELOW
2629  006450  066706  177744          ADD     2$,SP           ;RESET STACK PTR
2630  006454  012716                  MOV     (PC)+,(SP)      ;GET RETURN ADDRESS
2631  006456  000000          3$:     .WORD   0               ;CONTAINS RETURN PC
2632  006460  066716  177734          ADD     2$,(SP)         ;ADJUST RETURN PC
2633  006464  000207                  RTS     PC
2634
2635                                  .SBTTL  PARITY ERROR SERVICE ROUTINE
2636                          ;WHEN A PARITY ERROR IS DETECTED THIS ROUTINE SCANS MEMORY FOR THE AD-
2637                          ;DRESS CAUSING THE PARITY ERROR. WHEN THE ADDRESS IS LOCATED THE ROUTINE
2638                          ;HALTS WITH THE ADDRESS+2 IN R0. TO CONTINUE AFTER THE ERROR PRESS CONTINUE.
2639  006466  010067  000170  .PARSRV:MOV     R0,SAVR0        ;SAVE R0 IN SAVR0
2640  006472  012700  006664          MOV     #SAVR0+2,R0
2641  006476  010120                  MOV     R1,(R0)+
2642  006500  010220                  MOV     R2,(R0)+
2643  006502  010320                  MOV     R3,(R0)+
2644  006504  010420                  MOV     R4,(R0)+
2645  006506  010520                  MOV     R5,(R0)+
2646  006510  004567  172244          JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
2647  006514  006676                  PARERR
2648  006516  005027                  CLR     (PC)+           ;CLEAR PARITY ERROR INDICATORS
2649  006520     000          PEFLG:  .BYTE   0               ;NOT 0/0 =PAR ERR/NO PAR ERR
2650  006521     000          PENFLG: .BYTE   0               ;NOT 0/0=PAR ERR DETECTED/NOT DETECTED ON SCAN
2651  006522  012737  006570  000114  MOV     #2$,@#PARVEC            ;SET PARITY ERROR TRAP
2652  006530  012737  006626  000004  MOV     #4$,@#ERRVEC            ;SET TIME OUT TRAP VECTOR
2653  006536  005002                  CLR     R2
2654  006540  005767  172206          TST     MMAVA           ;CHECK IF MEM MGMT IS AVAILABLE
2655  006544  001407                  BEQ     1$              ;BRANCH IF NOT AVAILABLE
2656  006546  004767  000272          JSR     PC,LDMMD        ;SET UP MEM MGMT
2657  006552  105237  172301          INCB    @#KIPDR0+1      ;ALLOW FULL 4K PAGE ADDRESSING
2658  006556  012737  007142  000250  MOV     #MMABTO,@#MMVEC ;SET MEM MGMT ABORT TRAP VECTOR
2659  006564  012200          1$:     MOV     (R2)+,R0        ;SCAN ALL ADDRESSES
2660  006566  000776                  BR      1$
2661  006570  110667  177724  2$:     MOVB    SP,PEFLG        ;SET PARITY ERROR FOUND INDICATOR
2662  006574  010003                  MOV     R0,R3
2663  006576  104400                  HLT                     ;PARITY ERROR! ADDRESS+2 IS IN R2 DATA
2664                                                          ;IS IN R0
2665  006600  000002                  RTI                     ;CONTINUE SCAN
2666  006602  000240          3$:     NOP                     ;INSERT HALT INST TO EXAMINE PARITY REGS
2667  006604  005067  177710          CLR     PEFLG           ;CLEAR PARITY ERROR INDICATORS
2668  006610  012706  000500          MOV     #STKPTR,SP      ;RESET STACK PTR
2669  006614  000005                  RESET
2670  006616  004767  000122          JSR     PC,.MAMF        ;GO ENABLE PARITY ERROR DETECTION
2671  006622  000177  173540          JMP     @PERSTRT        ;RESTART SELECTED PROGRAM
2672
2673                          ;SERVICE ROUTINE IF PARITY ERROR NOT DETECTED ON SCAN
```

```
2674  006626  105767  177666          4$:     TSTB    PEFLG           ;BRANCH IF PARITY ERROR WAS
2675  006632  001363                          BNE     3$              ;DETECTED ON SCAN
2676  006634  016602  000004                  MOV     4(SP),R2        ;GET PC AT TIME OF ERROR
2677  006640  162702  000002                  SUB     #2,R2           ;BACK IT UP
2678  006644  110667  177651                  MOVB    SP,PENFLG       ;SET IND = NO PAR ERROR DETECTED ON SCAN
2679  006650  004567  172104                  JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
2680  006654  006717                          NOFIND
2681  006656  104400                          HLT                     ;ERROR! PARITY ERROR NOT DETECTED ON SCAN
2682  006660  000750                          BR      3$
2683                                  ;THE BELOW 6 WORDS CONTAINS THE SAVED CONTENTS OF R0-R5 WHEN THE
2684                                  ;PARITY ERROR OCCURRED
2685  006662  000000                  SAVR0:  .WORD   0
2686  006664  000000                  SAVR1:  .WORD   0
2687  006666  000000                  SAVR2:  .WORD   0
2688  006670  000000                  SAVR3:  .WORD   0
2689  006672  000000                  SAVR4:  .WORD   0
2690  006674  000000                  SAVR5:  .WORD   0
2691
2692  006676  005015  040520  044522  PARERR: .ASCIZ  <15><12>'PARITY ERROR'<15><12>
2693  006704  054524  042440  051122
2694  006712  051117  005015     000
2695  006717     116  052117  043040  NOFIND: .ASCIZ  'NOT FOUND ON SCAN'<15><12>
2696  006724  052517  042116  047440
2697  006732  020116  041523  047101
2698  006740  005015     000
2699          006744                          .EVEN
2700
2701                                  ;ROUTINE TO ENABLE PARITY ERROR ACTION ON MA/MF PARITY MEMORIES
2702          172100                          PARCSR=172100           ;ADDRESS OF FIRST PARITY REGISTER
2703          000114                          PARVEC=114              ;PARITY ERROR INTERRUPT VECTOR ADDRESS
2704
2705  006744  032737  000040  177570  .MAMF:  BIT     #40,@#SWR       ;CHECK IF PARITY ERROR DETECTION IS TO
2706  006752  001033                          BNE     DISPAR          ;BE ENABLED. BRANCH IF NOT TO BE ENABLED
2707  006754  013746  000004                  MOV     @#ERRVEC,-(SP)  ;SAVE ERROR TRAP VECTOR
2708  006760  012737  000006  000004          MOV     #ERRVEC+2,@#ERRVEC      ;SET TIME OUT TRAP TO RETURN (VIA RTI)
2709  006766  012737  006466  000114          MOV     #.PARSRV,@#PARVEC       ;SET PARITY ERROR TRAP VECTOR
2710  006774  012737  000340  000116          MOV     #340,@#PARVEC+2 ;PRIORITY LEVEL 7 ON TRAP
2711  007002  012700  172100                  MOV     #PARCSR,R0      ;GET FIRST ADDRESS OF PARITY REGISTER
2712  007006  012702  000001                  MOV     #1,R2
2713  007012  005027                          CLR     (PC)+           ;CLEAR AVAILABILITY INDICATOR
2714  007014  000000                  PARAVA: .WORD   0               ;CONTAINS AVAILABILITY INDICATOR
2715
2716                                  ;ENABLE ALL AVAILABLE PARITY REGISTERS
2717  007016  000262                  1$:     SEV                     ;SET TIME OUT INDICATOR
2718  007020  012720  000001                  MOV     #1,(R0)+        ;SET ACTION ENABLE IF AVAILABLE
2719  007024  102402                          BVS     2$              ;BRANCH IF NO PARITY AVAILABLE
2720  007026  050267  177762                  BIS     R2,PARAVA       ;SET AVAILABLITY INDICATOR
2721  007032  006302                  2$:     ASL     R2              ;SHIFT INDICATOR
2722  007034  103370                          BCC     1$
2723  007036  012637  000004                  MOV     (SP)+,@#ERRVEC  ;RESTORE ERROR TRAP VECTOR
2724  007042  000207                  DISPAR: RTS     PC              ;RETURN
2725
2726                                          .SBTTL  MEM MGMT ROUTINES
2727                                  ;ROUTINE TO INITIALIZE MEMORY MANAGEMENT REGISTERS
2728  007044  000240                  LDMM0:  NOP
2729  007046  005767  171700                  TST     MMAVA
```

```
2730  007052  001432                        BEQ     1$
2731  007054  012737  077006  172300        MOV     #177*256.-400+UP+RW,@#KIPDR0    ;SET KIPDR0=RW UP 177 BLOCKS
2732  007062  012737  077406  172302        MOV     #200*256.-400+UP+RW,@#KIPDR1    ;SET KIPDR1=RW UP 200 BLOCKS
2733  007070  005037  172304                CLR     @#KIPDR2
2734  007074  012737  000000  172344        MOV     #0,@#KIPAR2
2735  007102  012737  077406  172316        MOV     #200*256.-400+UP+RW,@#KIPDR7    ;SET KIPDR7=RW UP 200 BLOCKS
2736  007110  005037  172340                CLR     @#KIPAR0
2737  007114  012737  000200  172342        MOV     #200,@#KIPAR1
2738  007122  012737  007600  172356        MOV     #7600,@#KIPAR7
2739  007130  012737  000001  177572        MOV     #1,@#SR0        ;ENABLE MEM MGMT
2740  007136  000240                        NOP
2741  007140  000207                1$:     RTS     PC
2742
2743                                        ;MEMORY MANAGEMENT ABORT ROUTINE FOR WRITE UP
2744  007142  012702  020000        MMABT0: MOV     #20000,R2       ;RESET R2
2745  007146  062737  000200  172342        ADD     #200,@#KIPAR1   ;ADVANCE TO NEXT 4K
2746  007154  013716  177576                MOV     @#SR2,(SP)      ;RETURN TO INSTRUCTION THAT
2747  007160  005037  177572                CLR     @#SR0           ;DISABLE MEM MGMT
2748  007164  012737  000001  177572        MOV     #1,@#SR0        ;ENABLE MEM MGMT
2749  007172  000002                        RTI                     ;CAUSED THE ABORT
2750
2751                                        ;MEM MGMT ABORT SERVICE FOR WRITE DOWN
2752  007174  012702  040000        MMABT1: MOV     #40000,R2       ;RESET R2
2753  007200  162737  000200  172342        SUB     #200,@#KIPAR1
2754  007206  001406                        BEQ     2$
2755  007210  013716  177576                MOV     @#SR2,(SP)
2756  007214  012737  000001  177572        MOV     #1,@#SR0        ;ENABLE MEM MGMT
2757  007222  000002                        RTI
2758  007224                        2$:
2759  007224  005037  177572                CLR     @#SR0           ;DISABLE MEM MGMT
2760  007230  052766  000002  000002        BIS     #V,2(SP)
2761  007236  000002                        RTI
2762
2763                                        ;ROUTINE TO SET UP MEMORY MANAGEMENT FOR PATTERN TESTS
2764  007240  005702                STMM2:  TST     R2              ;CHECK IF TESTING BANK # 0
2765  007242  001442                        BEQ     2$              ;EXIT IF BANK # 0
2766  007244  005767  171502                TST     MMAVA
2767  007250  001005                        BNE     1$              ;BRANCH IF MEM MGMT AVAILABLE
2768  007252  006002                        ROR     R2              ;ADJUST ADDRESS
2769  007254  006002                        ROR     R2
2770  007256  006002                        ROR     R2
2771  007260  006002                        ROR     R2
2772  007262  000207                        RTS     PC              ;RETURN
2773
2774  007264  004767  177554        1$:     JSR     PC,LDMM0        ;GO MAKE INITIAL SET UP
2775  007270  000302                        SWAB    R2
2776  007272  006002                        ROR     R2
2777  007274  010237  172344                MOV     R2,@#KIPAR2
2778  007300  062702  000200                ADD     #200,R2
2779  007304  010237  172346                MOV     R2,@#KIPAR3
2780  007310  012737  077406  172304        MOV     #200*256.-400+UP+RW,@#KIPDR2    ;SET KIPDR2=RW UP 200 BLOCKS
2781  007316  012737  077406  172306        MOV     #200*256.-400+UP+RW,@#KIPDR3    ;SET KIPDR3=RW UP 200 BLOCKS
2782  007324  005037  172310                CLR     @#KIPDR4
2783  007330  012702  040000                MOV     #40000,R2
2784  007334  012737  007352  000250        MOV     #MMABT2,@#MMVEC
2785  007342  012737  000001  177572        MOV     #1,@#SR0        ;ENABLE MEM MGMT
```

```
2786  007350  000207                    2$:     RTS     PC
2787
2788                                    ;ROUTINE TO SERVICE 8 XOR 13 ABORTS
2789  007352  000240                    MMABT2: NOP
2790  007354  012702  040000                    MOV     #40000,R2
2791  007360  062737  000400  172344            ADD     #400,@#KIPAR2
2792  007366  062737  000400  172346            ADD     #400,@#KIPAR3
2793  007374  013716  177576                    MOV     @#SR2,(SP)      ;SET RETURN TO INSTRUCTION THAT ABORTED
2794  007400  012737  000001  177572            MOV     #1,@#SR0        ;ENABLE MEM MGMT
2795  007406  000002                            RTI
2796
2797                                            .SBTTL  1 XOR 8 ROUTINES
2798                                    ;ROUTINE TO WRITE 1 XOR 8 WORST CASE NOISE PATTERN
2799                                    ;CALL:  MOV     BANK #,-(SP)    ;PUSH STARTING BANK # ON THE STACK
2800                                    ;       MOV     BLKCNT,-(SP)    ;PUSH 128. WORD BLOCK COUNT ON THE STACK
2801                                    ;       JSR     PC,..1X8
2802
2803  007410  016603  000002            .1X8:   MOV     2(SP),R3        ;GET # OF 128. WORD BLOCKS TO WRITE
2804  007414  016602  000004                    MOV     4(SP),R2        ;GET STARTING BANK #
2805  007420  004767  177614                    JSR     PC,STMM2        ;GO SET UP MEM MGMT
2806  007424  012700  177777            1$:     MOV     #-1,R0          ;SET UP DATA REGISTERS
2807  007430  010005                            MOV     R0,R5
2808  007432  005105                            COM     R5
2809
2810  007434  005100                    2$:     COM     R0
2811  007436  005105                            COM     R5
2612  007440  012704  000010                    MOV     #8.,R4          ;SET 128. WORD COUNTER
2813  007444  010022                    3$:     MOV     R0,(R2)+        ;WRITE 128. WORDS
2814  007446  010522                            MOV     R5,(R2)+
2815  007450  010022                            MOV     R0,(R2)+
2816  007452  010522                            MOV     R5,(R2)+
2817
2818  007454  010022                            MOV     R0,(R2)+
2819  007456  010522                            MOV     R5,(R2)+
2820  007460  010022                            MOV     R0,(R2)+
2821  007462  010522                            MOV     R5,(R2)+
2822
2823  007464  010022                            MOV     R0,(R2)+
2824  007466  010522                            MOV     R5,(R2)+
2825  007470  010022                            MOV     R0,(R2)+
2826  007472  010522                            MOV     R5,(R2)+
2827
2828  007474  010022                            MOV     R0,(R2)+
2829  007476  010522                            MOV     R5,(R2)+
2830  007500  010022                            MOV     R0,(R2)+
2831  007502  010522                            MOV     R5,(R2)+
2832
2833  007504  005304                            DEC     R4              ;DECREMENT 128. WORD COUNTER
2834  007506  001356                            BNE     3$
2835  007510  005303                            DEC     R3              ;DECREMENT BLOCK COUNT
2836  007512  001350                            BNE     2$
2837  007514  012616                            MOV     (SP)+,(SP)      ;ADJUST STACK
2838  007516  012616                            MOV     (SP)+,(SP)
2839  007520  000207                            RTS     PC              ;RETURN TO CALLER
2840
2841                                    ;ROUTINE TO CHECK 1 XOR 8 PATTERN WRITTEN ABOVE
```

```
                                        ;CALL:  MOV     BANK #,-(SP)    ;PUSH STARTING BANK # ON THE STACK
                                        ;       MOV     BLKCNT,-(SP)    ;PUSH 128. WORD BLOCK COUNT ON STACK
                                        ;       JSR     PC,..1X8
                                        ..1X8:  NOP
007522  000240                                  JSR     PC,CKSWR        ;GO CHECK SWITCH REGISTER
007524  004767  002120                          MOV     2(SP),COUNT     ;GET BLOCK COUNT
007530  016667  000002  171220          10$:    MOV     4(SP),R2        ;GET STARTING BANK #
007536  016602  000004                          JSR     PC,STMM2        ;GO SET UP MEM MGMT
007542  004767  177472                  1$:     CLR     R0              ;CLEAR TEST WORD
007546  005000                                  TST     ICOUNT          ;IF BIT 15 OF ICOUNT =1 THEN PATTERN
007550  005767  171170                          BMI     .+4             ;IS COMPLEMENTED
007554  100401                                  COM     R0              ;COMPLEMENT TEST WORD
007556  005100                          2$:     MOV     #32.,R5         ;SET 128. WORD COUNTER
007560  012705  000040
                                        3$:     COM     R0
007564  005100                                  MOV     (R2)+,R3        ;GET TEST DATA
007566  012203                                  CMP     R0,R3           ;COMPARE WITH CHECK WORD
007570  020003                                  BEQ     .+10
007572  001403                                  CLR     -(SP)           ;PUSH FAKE STATUS ON THE STACK
007574  005046                                  JSR     PC,ERROR        ;ERROR! MEM DATA (R3) NOT = TEST DATA
007576  004767  171372                                                  ;(R0), ADDRESS=(R2)-2

007602  005100                                  COM     R0
007604  012203                                  MOV     (R2)+,R3        ;GET TEST DATA
007606  020003                                  CMP     R0,R3           ;COMPARE WITH CHECK WORD
007610  001403                                  BEQ     .+10
007612  005046                                  CLR     -(SP)           ;PUSH FAKE STATUS ON THE STACK
007614  004767  171354                          JSR     PC,ERROR        ;ERROR! MEM DATA (R3) NOT = TEST DATA
                                                                        ;(R0), ADDRESS=(R2)-2

007620  005100                                  COM     R0
007622  012203                                  MOV     (R2)+,R3        ;GET TEST DATA
007624  020003                                  CMP     R0,R3           ;COMPARE WITH CHECK WORD
007626  001403                                  BEQ     .+10
007630  005046                                  CLR     -(SP)           ;PUSH FAKE STATUS ON THE STACK
007632  004767  171336                          JSR     PC,ERROR        ;ERROR! MEM DATA (R3) NOT = TEST DATA
                                                                        ;(R0), ADDRESS=(R2)-2

007636  005100                                  COM     R0
007640  012203                                  MOV     (R2)+,R3        ;GET TEST DATA
007642  020003                                  CMP     R0,R3           ;COMPARE WITH CHECK WORD
007644  001403                                  BEQ     .+10
007646  005046                                  CLR     -(SP)           ;PUSH FAKE STATUS ON THE STACK
007650  004767  171320                          JSR     PC,ERROR        ;ERROR! MEM DATA (R3) NOT = TEST DATA
                                                                        ;(R0), ADDRESS=(R2)-2

007654  005305                                  DEC     R5              ;DECREMENT 128. WORD COUNTER
007656  001342                                  BNE     3$
007660  005100                                  COM     R0              ;COMPLEMENT CHECK WORD
007662  005367  171070                          DEC     COUNT           ;DECREMENT BLOCK COUNT
007666  001334                                  BNE     2$
007670  016602  000004                          MOV     4(SP),R2        ;GET BANK #
007674  004767  177340                          JSR     PC,STMM2
007700  016667  000002  171050  4$:     MOV     2(SP),COUNT     ;GET # OF 128. WORD BLOCKS TO COMPLEMENT
007706  006367  171032                          ASL     ICOUNT
```

```
 2898  007712  102306                        BVC     10$
 2899  007714  012705  000040        50$:    MOV     #32.,R5
 2900  007720  005122                5$:     COM     (R2)+           ;COMPLEMENT PATTERN
 2901  007722  005122                        COM     (R2)+
 2902  007724  005122                        COM     (R2)+
 2903  007726  005122                        COM     (R2)+
 2904  007730  005305                        DEC     R5
 2905  007732  001372                        BNE     5$
 2906  007734  005367  171016              DEC     COUNT
 2907  007740  001365                        BNE     50$
 2908  007742  005767  170776              TST     ICOUNT
 2909  007746  001270                        BNE     10$
 2910  007750  012616                        MOV     (SP)+,(SP)
 2911  007752  012616                        MOV     (SP)+,(SP)
 2912  007754  000207                        RTS     PC
 2913
 2914                                  .SBTTL  3 XOR 9 ROUTINES
 2915                          ;ROUTINE TO WRITE 3XOR9 WORST CASE NOISE TEST PATTERN
 2916                          ;CALL:  MOV     BANK #,-(SP)    ;PUSH STARTING BANK # ON STACK
 2917                          ;       MOV     BLKCNT,-(SP)    ;PUSH 256. WORD BLOCK COUNT ON STACK
 2918                          ;       JSR     PC,.3X9         ;CALL ROUTINE
 2919
 2920  007756  016602  000004  .3X9:   MOV     4(SP),R2        ;GET STARTING BANK #
 2921  007762  004767  177252          JSR     PC,STMM2
 2922  007766  005000                  CLR     R0
 2923  007770  010003                  MOV     R0,R3
 2924  007772  005103                  COM     R3              ;R0 (0) AND R3 (-1) IS THE DATA WRITTEN
 2925  007774  005767  173366          TST     PARPAT          ;BRANCH IF PARITY MEMORY PATTERN IS
 2926  010000  001402                  BEQ     1$              ;NOT TO BE WRITTEN
 2927
 2928  010002  012700  000401          MOV     #401,R0         ;WRITE PARITY 3X9 PATTERN
 2929  010006  012704  000020  1$:     MOV     #16.,R4         ;EACH LOOP WRITES 256. WORDS
 2930
 2931  010012  010022          2$:     MOV     R0,(R2)+
 2932  010014  010022                  MOV     R0,(R2)+
 2933  010016  010022                  MOV     R0,(R2)+
 2934  010020  010022                  MOV     R0,(R2)+
 2935
 2936  010022  010322                  MOV     R3,(R2)+
 2937  010024  010322                  MOV     R3,(R2)+
 2938  010026  010322                  MOV     R3,(R2)+
 2939  010030  010322                  MOV     R3,(R2)+
 2940
 2941  010032  010022                  MOV     R0,(R2)+
 2942  010034  010022                  MOV     R0,(R2)+
 2943  010036  010022                  MOV     R0,(R2)+
 2944  010040  010022                  MOV     R0,(R2)+
 2945
 2946  010042  010322                  MOV     R3,(R2)+
 2947  010044  010322                  MOV     R3,(R2)+
 2948  010046  010322                  MOV     R3,(R2)+
 2949  010050  010322                  MOV     R3,(R2)+
 2950
 2951  010052  005304                  DEC     R4
 2952  010054  001356                  BNE     2$
 2953  010056  005100                  COM     R0
```

```
2954  010060  005103                        COM     R3
2955  010062  005767  173300                TST     PARPAT          ;BRANCH IF PARITY MEMORY PATTERN IS
2956  010066  001402                        BEQ     3$              ;NOT TO BE WRITTEN
2957
2958  010070  004767  000014                JSR     PC,..XOR39      ;GO GET CONSTANTS
2959  010074  005366  000002        3$:     DEC     2(SP)           ;DECREMENT 256. WORD BLOCK COUNT
2960  010100  001342                        BNE     1$
2961  010102  012616                        MOV     (SP)+,(SP)      ;ADJUST STACK
2962  010104  012616                        MOV     (SP)+,(SP)
2963  010106  000207                        RTS     PC
2964
2965                                ;ROUTINE TO SET CONSTANTS FOR WRITING/CHECKING 3 XOR PATTERN WITH
2966                                ;PARITY.
2967  010110  032702  000010        .XOR39: BIT     #10,R2          ;CHECK BIT 3
2968  010114  001404                        BEQ     .3IS0           ;BRANCH IF BIT 3 = 0
2969  010116  032702  001000        .3IS1:  BIT     #1000,R2        ;CHECK BIT 9
2970  010122  001404                        BEQ     .3NOT9          ;BRANCH IF BIT 9 =0
2971  010124  000407                        BR      .3IS9
2972  010126  032702  001000        .3IS0:  BIT     #1000,R2        ;CHECK BIT 9
2973  010132  001404                        BEQ     .3IS9           ;BRANCH IF 0
2974  010134  005767  170604        .3NOT9: TST     ICOUNT          ;CHECK IF NORMAL OR COMPLEMENT DATA
2975  010140  100004                        BPL     LDCOMP          ;GO LOAD COMPLEMENT CONSTANTS
2976  010142  100410                        BMI     LDNORM          ;GO LOAD NORMAL CONSTANTS
2977  010144  005767  170574        .3IS9:  TST     ICOUNT          ;CHECK IF NORMAL OR COMPLEMENT DATA
2978  010150  100005                        BPL     LDNORM          ;GO LOAD NORMAL CONSTANTS
2979  010152  012700  177777        LDCOMP: MOV     #-1,R0          ;SET COMPLEMENT CONSTANTS
2980  010156  012703  000401                MOV     #401,R3
2981  010162  000207                        RTS     PC              ;RETURN
2982  010164  012700  000401        LDNORM: MOV     #401,R0         ;LOAD NORMAL CONSTANTS
2983  010170  012703  177777                MOV     #-1,R3
2984  010174  000207                        RTS     PC
2985
2986                                ;ROUTINE TO CHECK 3 XOR 9 WORST CASE NOISE PATTERN
2987                                ;CALL:  MOV     BANK#,-(SP)     ;PUSH STARTING BANK # ONTO STACK
2988                                ;       MOV     BLKCNT,-(SP)    ;AND 256. WORD BLOACK COUNT
2989                                ;       JSR     PC,..3X9        ;CALL ROUTINE
2990
2991  010176  000240                ..3X9:  NOP
2992  010200  004767  001444                JSR     PC,CKSWR        ;GO CHECK SWITCH REGISTER
2993
2994                                ;CHECK WORST CASE PATTERN
2995  010204  016604  000002        1$:     MOV     2(SP),R4        ;GET 256. BLOCK WORD COUNT
2996  010210  016602  000004                MOV     4(SP),R2        ;GET FIRST BANK #
2997  010214  004767  177020                JSR     PC,STMM2        ;GO SET UP MEM MGMT
2998  010220  005000                        CLR     R0              ;SET CHECK WORD
2999  010222  005767  170516                TST     ICOUNT          ;IF ICOUNT IS NEG AM CHECKING COMP-
3000  010226  100001                        BPL     .+4             ;LEMENTED PATTERN
3001  010230  005100                        COM     R0              ;SO COMPLEMENT CHECK WORD
3002  010232  012705  000100        2$:     MOV     #64.,R5         ;SET 256. WORD COUNTER
3003
3004  010236  005767  173124        3$:     TST     PARPAT          ;BRANCH IF PARITY MEMORY PATTERN IS
3005  010242  001402                        BEQ     30$             ;NOT TO BE CHECKED
3006
3007  010244  004767  177640                JSR     PC,..XOR39      ;GO GET CONSTANT
3008  010250                        30$:
3009  010250  012203                        MOV     (R2)+,R3        ;GET TEST DATA
```

```
3010  010252  020003                    CMP     R0,R3           ;COMPARE WITH CHECK WORD
3011  010254  001403                    BEQ     .+10
3012  010256  005046                    CLR     -(SP)           ;PUSH FAKE STATUS ON THE STACK
3013  010260  004767  170710            JSR     PC,ERROR        ;ERROR! MEM DATA (R3) NOT = TEST DATA
3014                                                            ;(R0), ADDRESS=(R2)-2
3015
3016  010264  012203                    MOV     (R2)+,R3        ;GET TEST DATA
3017  010266  020003                    CMP     R0,R3           ;COMPARE WITH CHECK WORD
3018  010270  001403                    BEQ     .+10
3019  010272  005046                    CLR     -(SP)           ;PUSH FAKE STATUS ON THE STACK
3020  010274  004767  170674            JSR     PC,ERROR        ;ERROR! MEM DATA (R3) NOT = TEST DATA
3021                                                            ;(R0), ADDRESS=(R2)-2
3022
3023  010300  012203                    MOV     (R2)+,R3        ;GET TEST DATA
3024  010302  020003                    CMP     R0,R3           ;COMPARE WITH CHECK WORD
3025  010304  001403                    BEQ     .+10
3026  010306  005046                    CLR     -(SP)           ;PUSH FAKE STATUS ON THE STACK
3027  010310  004767  170660            JSR     PC,ERROR        ;ERROR! MEM DATA (R3) NOT = TEST DATA
3028                                                            ;(R0), ADDRESS=(R2)-2
3029
3030  010314  012203                    MOV     (R2)+,R3        ;GET TEST DATA
3031  010316  020003                    CMP     R0,R3           ;COMPARE WITH CHECK WORD
3032  010320  001403                    BEQ     .+10
3033  010322  005046                    CLR     -(SP)           ;PUSH FAKE STATUS ON THE STACK
3034  010324  004767  170644            JSR     PC,ERROR        ;ERROR! MEM DATA (R3) NOT = TEST DATA
3035                                                            ;(R0), ADDRESS=(R2)-2
3036
3037
3038  010330  005100                    COM     R0              ;COMPLEMENT CHECK WORD
3039  010332  005305                    DEC     R5              ;DECREMENT 256. WORD COUNTER
3040  010334  001340                    BNE     3$
3041  010336  005100                    COM     R0              ;COMPLEMENT CHECK WORD
3042  010340  005304                    DEC     R4              ;DECREMENT BLOCK COUNTER
3043  010342  001333                    BNE     2$
3044
3045  010344  032737  040000  177570    BIT     #40000,@#SWR    ;LOOP ON TEST?
3046  010352  001314                    BNE     1$              ;BRANCH IF LOOP ON TEST DESIRED
3047  010354  016667  000002  170374 40$: MOV    2(SP),COUNT     ;GET # OF 256. WORD BLOCKS TO CHECK
3048  010362  016602  000004            MOV     4(SP),R2        ;GET STARTING BANK #
3049  010366  004767  176646            JSR     PC,STMM2        ;GO SET UP MEM MGMT IF REQUIRD
3050
3051                                  ;CHECK WORST CASE BIT COMPLEMENT PATTERN
3052  010372  005000                    CLR     R0
3053  010374  005767  170344            TST     ICOUNT          ;CHECK IF COMPLEMENT PATERN
3054  010400  100001                    BPL     .+4
3055  010402  005100                    COM     R0              ;COMPLEMENT CHECK WORD
3056  010404  012704  000100      4$:   MOV     #64.,R4         ;SET 256. WORD COUNTER
3057  010410  012705  000004      5$:   MOV     #4,R5           ;SET 4 WORD COUNTER
3058  010414  005767  172746      6$:   TST     PARPAT          ;BRANCH IF PARITY MEMORY PATTERN IS
3059  010420  001402                    BEQ     60$             ;NOT TO BE CHECKED
3060  010422  004767  177462            JSR     PC,.XOR39
3061  010426  012203              60$:  MOV     (R2)+,R3        ;GET DATA
3062  010430  020003                    CMP     R0,R3           ;CHECK DATA
3063  010432  001403                    BEQ     .+10
3064  010434  005046                    CLR     -(SP)
3065  010436  004767  170532            JSR     PC,ERROR
```

```
3066  010442  005100                        COM     R0              ;COMPLEMENT CHECK WORD
3067  010444  005142                        COM     -(R2)           ;COMPLEMENT TEST DATA
3068  010446  012203                        MOV     (R2)+,R3        ;GET DATA
3069  010450  020003                        CMP     R0,R3           ;CHECK
3070  010452  001403                        BEQ     .+10
3071  010454  005046                        CLR     -(SP)           ;PUSH FAKE STATUS ON THE STACK
3072  010456  004767  170512                JSR     PC,ERROR
3073  010462  005100                        COM     R0              ;COMPLEMENT CHECK WORD
3074  010464  005162  177776                COM     -2(R2)          ;RESTORE DATA
3075  010470  005305                        DEC     R5              ;DECREMENT 4 WORD COUNTER
3076  010472  001350                        BNE     6$
3077  010474  005100                        COM     R0              ;COMPLEMENT CHECK WORD
3078  010476  005304                        DEC     R4              ;DECREMENT 256. WORD COUNTER
3079  010500  001343                        BNE     5$
3080  010502  005100                        COM     R0              ;COMPLEMENT CHECK WORD
3081  010504  005367  170246                DEC     COUNT           ;DECREMENT BLOCK COUNTER
3082  010510  001335                        BNE     4$
3083
3084  010512  016602  000004                MOV     4(SP),R2        ;GET BANK #
3085  010516  004767  176516                JSR     PC,STMM2
3086  010522  016603  000002                MOV     2(SP),R3        ;GET BLOCK COUNT
3087  010526  032737  040000  177570        BIT     #40000,@#SWR    ;LOOP ON TEST
3088  010534  001307                        BNE     40$             ;BRANCH IF LOOP ON TEST
3089  010536  006367  170202                ASL     ICOUNT
3090  010542  102220                        BVC     1$
3091  010544  012705  000040        7$:     MOV     #32.,R5         ;COMPLEMENT PATTERN
3092  010550  011200        10$:    MOV     (R2),R0         ;GET FIRST DATA WORD
3093  010552  016204  000010                MOV     10(R2),R4       ;GET FIFTH DATA WORD
3094  010556  110422                        MOVB    R4,(R2)+        ;SWAP WORDS 1-4
3095  010560  110422                        MOVB    R4,(R2)+        ;WITH 5-8
```

```
                                                    MOVB    R4,(R2)+
                                                    MOVB    R4,(R2)+
                                                    MOVB    R4,(R2)+
                                                    MOVB    R4,(R2)+
                                                    MOVB    R4,(R2)+
                                                    MOVB    R4,(R2)+
                                                    MOVB    R0,(R2)+        ;AND VICE VERSA
                                                    MOVB    R0,(R2)+
                                                    MOVB    R0,(R2)+
                                                    MOVB    R0,(R2)+
                                                    MOVB    R0,(R2)+
                                                    MOVB    R0,(R2)+
                                                    MOVB    R0,(R2)+
                                                    MOVB    R0,(R2)+
                                                    DEC     R5
                                                    BNE     10$
                                                    DEC     R3
                                                    BNE     7$

                                                    TST     ICOUNT
                                                    BEQ     11$
                                                    JMP     1$
                                            11$:    MOV     (SP)+,(SP)
                                                    MOV     (SP)+,(SP)
                                                    RTS     PC

                                            ;ROUTINE TO WRITE 8 XOR 13 WORST CASE NOISE TEST PATTERN
                                                    .SBTTL  8 XOR 13 ROUTINES
                                            ;CALL:  MOV     BANK #,-(SP)
                                            ;       MOV     #4KBANKS,-(SP)
                                            ;       JSR     PC,.8X13

                                            .8X13:  MOV     2(SP),R4        ;GET BANK COUNT
                                                    MOV     4(SP),R2        ;GET FIRST BANK #
                                                    JSR     PC,STMM2        ;GO SET MEM MGMT
                                            1$:     CLR     R0
                                            2$:     MOV     #32.,R5         ;EACH LOOP WRITES 4K WORDS
                                                    COM     R0
                                            3$:     MOV     #128.,R3        ;EACH SMALL LOOP WRITES 128 WORDS
                                                    COM     R0
                                            4$:     MOV     R0,(R2)+        ;WRITE INTO MEMORY ADDRESSES
                                                    DEC     R3              ;DECREMENT WORD COUNT
                                                    BNE     4$
                                                    DEC     R5              ;DECREMENT 128. WORD COUNT
                                                    BNE     3$
                                                    DEC     R4              ;DECREMENT 4K BANK COUNT
                                                    BNE     2$              ;LOOP UNTIL DONE
                                                    MOV     (SP)+,(SP)      ;ADJUST STACK
                                                    MOV     (SP)+,(SP)
                                                    RTS     PC

                                            ;ROUTINE TO CHECK 8 XOR 13 WORST CASE NOISE TEST PATTERN
                                            ;CALL:
                                            ;       MOV     BANK#,-(SP)     ;PUSH FIRST BANK # ON THE STACK
                                            ;       MOV     #BANKS,-(SP)    ;PUSH # OF 4K BANKS TO CHECK ON THE STACK
                                            ;       JSR     PC,..8X13       ;CALL ROUTINE
```

```
3152
3153  010724  000240                    ..8X13: NOP
3154  010726  004767  000716                    JSR     PC,CKSWR        ;CO CHECK SWITCH REGISTER
3155  010732  012700  177777                    MOV     #-1,R0          ;SET TEST DATA WORD
3156  010736  016602  000004              10$:  MOV     4(SP),R2        ;GET BANK #
3157  010742  004767  176272                    JSR     PC,STMM2        ;GO SET MEM MGMT IF REQUIRED
3158  010746  016667  000002  170002            MOV     2(SP),COUNT     ;GET # OF 4K BANKS TO CHECK
3159
3160  010754  012704  000040              1$:   MOV     #32.,R4         ;SET 4K WORD COUNTER
3161  010760  005100                      2$:   COM     R0              ;COMPLEMENT TEST WORD
3162  010762  012705  000100                    MOV     #64.,R5         ;SET 128 WORD COUNTER
3163
3164  010766                              3$:
3165  010766  012203                            MOV     (R2)+,R3        ;GET TEST DATA
3166  010770  020003                            CMP     R0,R3           ;COMPARE WITH CHECK WORD
3167  010772  001403                            BEQ     .+10
3168  010774  005046                            CLR     -(SP)           ;PUSH FAKE STATUS ON THE STACK
3169  010776  004767  170172                    JSR     PC,ERROR        ;ERROR! MEM DATA (R3) NOT = TEST DATA
3170                                                                    ;(R0), ADDRESS=(R2)-2
3171
3172  011002  012203                            MOV     (R2)+,R3        ;GET TEST DATA
3173  011004  020003                            CMP     R0,R3           ;COMPARE WITH CHECK WORD
3174  011006  001403                            BEQ     .+10
3175  011010  005046                            CLR     -(SP)           ;PUSH FAKE STATUS ON THE STACK
3176  011012  004767  170156                    JSR     PC,ERROR        ;ERROR! MEM DATA (R3) NOT = TEST DATA
3177                                                                    ;(R0), ADDRESS=(R2)-2
3178
3179  011016  005305                            DEC     R5              ;DECREMENT 128 WORD COUNTER
3180  011020  001362                            BNE     3$
3181  011022  005304                            DEC     R4              ;DECREMENT 4096. WORD COUNTER
3182  011024  001355                            BNE     2$
3183  011026  005100                            COM     R0
3184  011030  005367  167722                    DEC     COUNT           ;ALL 4K BANKS CHECKED?
3185  011034  001347                            BNE     1$
3186
3187  011036  016602  000004                    MOV     4(SP),R2        ;GET FIRST BANK ADDRESS
3188  011042  004767  176172                    JSR     PC,STMM2        ;GO SET UP MEM MGMT IF REQUIRED
3189  011046  016604  000002                    MOV     2(SP),R4        ;GET # OF 4K BANKS
3190  011052  006367  167666                    ASL     ICOUNT          ;SHIFT ITERATION PATTERN
3191  011056  001401                            BEQ     .+4
3192  011060  102326                            BVC     10$
3193  011062  012705  004000              40$:  MOV     #2048.,R5       ;SET 4096. WORD COUNTER
3194  011066  005122                      4$:   COM     (R2)+           ;COMPLEMENT TEST PATTERN
3195  011070  005122                            COM     (R2)+
3196  011072  005305                            DEC     R5
3197  011074  001374                            BNE     4$
3198  011076  005304                            DEC     R4
3199  011100  001370                            BNE     40$
3200  011102  005100                            COM     R0              ;COMPLEMENT TEST WORD
3201  011104  005767  167634                    TST     ICOUNT
3202  011110  001312                            BNE     10$
3203  011112  012616                            MOV     (SP)+,(SP)
3204  011114  012616                            MOV     (SP)+,(SP)
3205  011116  000207                            RTS     PC              ;RETURN
3206
3207                                      .SBTTL  ROTATING 1'S & 0'S ROUTINES
```

```
3208                                    ;ROUTINE TO CHECK ROTATING '0' BIT THROUGH FILD OF 1'S
3209                                    ;CALL:  MOV     BANK#,-(SP)     ;SET STARTING BANK #
3210                                    ;       MOV     BLKCNT,-(SP)    ;SET 256. WORD BLOCK COUNT
3211                                    ;       JSR     PC,.ROT0        ;CALL ROUITNE
3212
3213  011120  004767  000524            .ROT0:  JSR     PC,CKSWR        ;GO CHECK SWITCHES
3214  011124  016604  000002                    MOV     2(SP),R4        ;GET 256. WORD BLOCK COUNT
3215  011130  016602  000004                    MOV     4(SP),R2        ;GET FIRST BANK #
3216  011134  004767  176100                    JSR     PC,STMM2        ;GO SET UP MEM MGMT (IF AVAIL)
3217  011140  012700  177777                    MOV     #-1,R0          ;SET CHECK WORD
3218
3219  011144  012705  000400            1$:     MOV     #256.,R5        ;SET 256. WORD COUNT
3220  011150  000241                    2$:     CLC                     ;CLEAR CARRY BIT IN PSW
3221  011152  004767  000124                    JSR     PC,ROTATE
3222  011156  016203  177776                    MOV     -2(R2),R3       ;GET RESULT
3223  011162  103402                            BCS     3$              ;BRANCH IF 'C' BIT WAS SET
3224  011164  020003                            CMP     R0,R3           ;CHECK RESULT
3225  011166  001403                            BEQ     4$
3226  011170  005046                    3$:     CLR     -(SP)           ;ERROR! COULD NOT ROTATE '0' BIT
3227  011172  004767  167776                    JSR     PC,ERROR        ;THROUGH ADDRESS IN R2
3228  011176  005305                    4$:     DEC     R5              ;DECREMENT 256. WORD COUNT
3229  011200  001363                            BNE     2$              ;LOOP UNTIL DONE
3230  011202  005304                            DEC     R4              ;DECREMENT 256. WORD BLOCK COUNT
3231  011204  001357                            BNE     1$              ;LOOP UNTIL DONE
3232  011206  012616                            MOV     (SP)+,(SP)      ;POP CONSTANTS OFF THE STACK
3233  011210  012616                            MOV     (SP)+,(SP)
3234  011212  000207                            RTS     PC              ;RETURN TO CALLER
3235
3236                                    ;ROUTINE TO CHECK ROTATING '1' BIT THROUGH A FIELD OF 0'S
3237                                    ;CALL:  MOV     BANK#,-(SP)     ;SET STARTING BANK #
3238                                    ;       MOV     BLKCNT,-(SP)    ;SET # OF 256. WORD BLOCKS TO CHECK
3239                                    ;       JSR     PC,.ROT1        ;CALL ROUTINE
3240
3241  011214  004767  000430            .ROT1:  JSR     PC,CKSWR        ;GO CHECK SWITCHES
3242  011220  016604  000002                    MOV     2(SP),R4        ;GET # OF 256. WORD BLOCKS TO CHECK
3243  011224  016602  000004                    MOV     4(SP),R2        ;GET STARTING BANK #
3244  011230  004767  176004                    JSR     PC,STMM2        ;GO SET UP MEM MGMT (IF AVAIL)
3245  011234  005000                            CLR     R0              ;SET CHECK WORD
3246
3247  011236  012705  000400            1$:     MOV     #256.,R5        ;SET 256. WORD COUNTER
3248  011242  000261                    2$:     SEC                     ;SET 'C' BIT IN PSW
3249  011244  004767  000032                    JSR     PC,ROTATE       ;GO ROTATE '1' BIT
3250  011250  016203  177776                    MOV     -2(R2),R3       ;GET RESULT
3251  011254  103002                            BCC     3$              ;BRANCH IF 'C' IS CLEAR
3252  011256  020003                            CMP     R0,R3           ;CHECK RESULT
3253  011260  001401                            BEQ     .+4
3254  011262  104400                    3$:     HLT                     ;ERROR! COULD NOT ROTATE '1' BIT
3255                                                                    ;THROUGH ADDRESS IN R2
3256  011264  005305                            DEC     R5              ;DECREMENT 256. WORD COUNT
3257  011266  001365                            BNE     2$
3258  011270  005304                            DEC     R4              ;DECREMENT 256. WORD BLOCK COUNT
3259  011272  001361                            BNE     1$
3260  011274  012616                            MOV     (SP)+,(SP)      ;ADJUST RETURN ADDRESS
3261  011276  012616                            MOV     (SP)+,(SP)
3262  011300  000207                            RTS     PC              ;RETURN TO CALLER
3263
```

```
3264
3265  011302  106112                ;ROUTINE TO ROTATE "C" BIT THROUGH A MEMORY LOCATION.
3266  011304  106112                ROTATE: ROLB    (R2)            ;(R2)=177776 OR 000001
3267  011306  106112                        ROLB    (R2)            ;(R2)=177775 OR 000002
3268  011310  106112                        ROLB    (R2)            ;(R2)=177773 OR 000004
3269  011312  106112                        ROLB    (R2)            ;(R2)=177767 OR 000010
3270  011314  106112                        ROLB    (R2)            ;(R2)=177757 OR 000020
3271  011316  106112                        ROLB    (R2)            ;(R2)=177737 OR 000040
3272  011320  106112                        ROLB    (R2)            ;(R2)=177677 OR 000100
3273  011322  106122                        ROLB    (R2)            ;(R2)=177777 OR 000000
3274  011324  106112                        ROLB    (R2)+           ;(R2)=177577 OR 000200
3275  011326  106112                        ROLB    (R2)            ;(R2)=177377 OR 000400
3276  011330  106112                        ROLB    (R2)            ;(R2)=176777 OR 001000
3277  011332  106112                        ROLB    (R2)            ;(R2)=175777 OR 002000
3278  011334  106112                        ROLB    (R2)            ;(R2)=173777 OR 004000
3279  011336  106112                        ROLB    (R2)            ;(R2)=167777 OR 010000
3280  011340  106112                        ROLB    (R2)            ;(R2)=157777 OR 020000
3281  011342  106112                        ROLB    (R2)            ;(R2)=137777 OR 040000
3282  011344  106122                        ROLB    (R2)            ;(R2)=077777 OR 100000
3283  011346  000207                        RTS     PC              ;(R2)=177777 OR 000000
3284                                                                 ;RETURN
3285
3286                                ;ROUTINE TO WRITE USER SLECTED PATTERN INTO MEMORY
3287                                ;CALL:  MOV     BANK#,-(SP)     ;PUSH STARTING BANK # ONTO STACK
3288                                ;       MOV     BLKCNT,-(SP)    ;AND 128. WORD BLOCK COUNT
3289                                ;       JSR     PC,..USER       ;CALL ROUTINE
3290  011350  016604  000002        .USER:  MOV     2(SP),R4        ;GET BLOCK COUNT
3291  011354  016602  000004                MOV     4(SP),R2        ;GET STARTING BANK #
3292  011360  004767  175654                JSR     PC,STMM2        ;GO SET UP MEM MGMT
3293  011364  016700  174012                MOV     .CONST,R0       ;GET USER CONSTANT
3294  011370  012703  000100        1$:     MOV     #64.,R3         ;SET 256. WORD COUNTER
3295  011374  010022                2$:     MOV     R0,(R2)+        ;WRITE 256. WORDS
3296  011376  010022                        MOV     R0,(R2)+
3297  011400  010022                        MOV     R0,(R2)+
3298  011402  010022                        MOV     R0,(R2)+
3299  011404  005303                        DEC     R3              ;DECREMENT 256. WORD COUNTER
3300  011406  001372                        BNE     2$              ;LOOP UNTIL 256. WORDS HAVE BEEN WRITTEN
3301  011410  005304                        DEC     R4              ;DECREMENT BLOCK COUNT
3302  011412  001366                        BNE     1$
3303  011414  012616                        MOV     (SP)+,(SP)      ;ADJUST STACK
3304  011416  012616                        MOV     (SP)+,(SP)
3305  011420  000207                        RTS     PC
3306
3307                                        .SBTTL  USER PATTERN ROUTINE
3308                                ;ROUTINE TO CHECK USER SELECTED PATTERN
3309                                ;CALL:  MOV     BANK#,-(SP)     ;PUSH STARTING BANK # ONTO STACK
3310                                ;       MOV     BLKCNT,-(SP)    ;AND 256. WORD BLOCK COUNT
3311                                ;       JSR     PC,..USER       ;CALL ROUTINE
3312
3313  011422  004767  000222        ..USER: JSR     PC,CKSWR        ;GO CHECK SWITCH REGISTER
3314  011426  016700  173750                MOV     .CONST,R0       ;GET USER CONSTANT
3315  011432  016604  000002        1$:     MOV     2(SP),R4        ;GET # OF 256. WORD BLOCKS
3316  011436  016602  000004                MOV     4(SP),R2        ;GET STARTING BANK #
3317  011442  004767  175572                JSR     PC,STMM2        ;GO SET UP MEM MGMT IF REQUIRED
3318
3319  011446  012705  000100        2$:     MOV     #64.,R5         ;SET WORD COUNT
```

```
3320  011452                        3$:
3321  011452  012203                      MOV     (R2)+,R3        ;GET TEST DATA
3322  011454  020003                      CMP     R0,R3           ;COMPARE WITH CHECK WORD
3323  011456  001403                      BEQ     .+10
3324  011460  005046                      CLR     -(SP)           ;PUSH FAKE STATUS ON THE STACK
3325  011462  004767  167506              JSR     PC,ERROR        ;ERROR! MEM DATA (R3) NOT = TEST DATA
3326                                                              ;(R0), ADDRESS=(R2)-2
3327
3328  011466  012203                      MOV     (R2)+,R3        ;GET TEST DATA
3329  011470  020003                      CMP     R0,R3           ;COMPARE WITH CHECK WORD
3330  011472  001403                      BEQ     .+10
3331  011474  005046                      CLR     -(SP)           ;PUSH FAKE STATUS ON THE STACK
3332  011476  004767  167472              JSR     PC,ERROR        ;ERROR! MEM DATA (R3) NOT = TEST DATA
3333                                                              ;(R0), ADDRESS=(R2)-2
3334
3335  011502  012203                      MOV     (R2)+,R3        ;GET TEST DATA
3336  011504  020003                      CMP     R0,R3           ;COMPARE WITH CHECK WORD
3337  011506  001403                      BEQ     .+10
3338  011510  005046                      CLR     -(SP)           ;PUSH FAKE STATUS ON THE STACK
3339  011512  004767  167456              JSR     PC,ERROR        ;ERROR! MEM DATA (R3) NOT = TEST DATA
3340                                                              ;(R0), ADDRESS=(R2)-2
3341
3342  011516  012203                      MOV     (R2)+,R3        ;GET TEST DATA
3343  011520  020003                      CMP     R0,R3           ;COMPARE WITH CHECK WORD
3344  011522  001403                      BEQ     .+10
3345  011524  005046                      CLR     -(SP)           ;PUSH FAKE STATUS ON THE STACK
3346  011526  004767  167442              JSR     PC,ERROR        ;ERROR! MEM DATA (R3) NOT = TEST DATA
3347                                                              ;(R0), ADDRESS=(R2)-2
3348
3349  011532  005305                      DEC     R5              ;DECREMENT WORD COUNT
3350  011534  001346                      BNE     3$
3351  011536  005304                      DEC     R4              ;DECREMENT BLOCK COUNT
3352  011540  001342                      BNE     2$
3353
3354  011542  032737  040000  177570      BIT     #40000,@#SWR    ;CHECK LOOP SWITCH
3355  011550  001330                      BNE     1$              ;LOOP CHECKING THIS PATTERN
3356  011552  006367  167166              ASL     ICOUNT          ;SHIFT PATTERN INDICATOR
3357  011556  001325                      BNE     1$
3358  011560  012616                      MOV     (SP)+,(SP)      ;ADJUST STACK
3359  011562  012616                      MOV     (SP)+,(SP)
3360  011564  000207                      RTS     PC              ;RETURN TO CALLER
3361
3362                                      .SBTTL  GET TTY INPUT ROUTINE
3363                              ;ROUTINE TO GET ASCII INPUT FROM TTY,AND CONVERT TO OCTAL.
3364                              ;ROUTINE LEAVES THE FIRST 16 BITS IN ADDRESS FOLLOWINTG THE CALL
3365                              ;AND THE LAST 2 BITS IN .1617 BELOW.
3366                              ;CALL:  JSR  PC,RECD
3367  011566                      RECD:
3368  011566  004767  166512              JSR     PC,$SAVR        ;GO SAVE REGISTERS ON THE STACK
3369  011572  004767  002172              JSR     PC,INUM
3370  011576  010267  000042              MOV     R2,TEMP2
3371  011602  010367  000040              MOV     R3,TEMP3
3372  011606  004767  166516              JSR     PC,$RESTR       ;RESTORE REGISTERS FROM STACK
3373  011612  011667  000024              MOV     (SP),TEMP1
3374  011616  016777  000024  000016      MOV     TEMP3,@TEMP1
3375  011624  016767  000014  000006      MOV     TEMP2,.1617
```

```
3376  011632  062716  000002                    ADD     #2,(SP)
3377  011636  000207                            RTS     PC
3378  011640  000000                    .1617:  .WORD   0
3379  011642  000000                    TEMP1:  .WORD   0
3380  011644  000000                    TEMP2:  .WORD   0
3381  011646  000000                    TEMP3:  .WORD   0
3382
3383                                    ;ROUTINE TO CHECK THE SWITCH REGISTER
3384                                    ;CHECK SWITCH 9: IF SET, LOAD ERROR COUNT INTO THE DISPLAY REGISTER;
3385                                    ;IF NOT SET, LOAD PASS COUNT INTO THE DISPLAY REGISTER
3386  011650  042767  017777  167072  CKSWR:  BIC     #17777,LDDISP   ;SAVE RELOCATION BITS
3387  011656  032737  000400  177570          BIT     #BIT8,@#SWR     ;CHECK SWITCH 8
3388  011664  001402                          BEQ     10$             ;BRANCH IF SET
3389  011666  004767  000464                  JSR     PC,REL24K       ;GO RELOCATE PROGRAM BACK TO 4K AND STOP
3390  011672  032737  001000  177570  10$:    BIT     #BIT9,@#SWR     ;SWITCH 9 SET ?
3391  011700  001404                          BEQ     1$
3392  011702  056767  167040  167040          BIS     ERCNT,LDDISP    ;LOAD ERROR COUNT
3393  011710  000403                          BR      2$
3394  011712  056767  167024  167030  1$:     BIS     ICNT,LDDISP     ;LOAD PASS COUNT
3395  011720  016737  167024  177570  2$:     MOV     LDDISP,@#DISPLAY;LOAD THE DISPLAY REGISTER
3396  011726  012767  040177  167010          MOV     #040177,ICOUNT  ;LOAD ITERATION COUNT WORD
3397  011734  032737  004000  177570          BIT     #4000,@#SWR     ;CHECK SW11
3398  011742  001402                          BEQ     .+6
3399  011744  105067  166774                  CLRB    ICOUNT          ;ICOUNT =040000 IF SW11 =1
3400  011750  000207                          RTS     PC
3401
3402                                    ;MESSAGES
3403  011752  005015  047524  051040  RESLDR: .ASCIZ  <15><12>'TO RESTORE LOADERS START AT 162'<15><12>
3404  011760  051505  047524  042522
3405  011766  046040  040517  042504
3406  011774  051522  051440  040524
3407  012002  052122  040440  020124
3408  012010  033061  006462  000012
3409  012016  005015  047105  041101  PARITY: .ASCIZ  <15><12> 'ENABLE PARITY? 1/0=YES/NO '
3410  012024  042514  050040  051101
3411  012032  052111  037531  030440
3412  012040  030057  054475  051505
3413  012046  047057  020117     000
3414  012053     015  051412  040524  STBANK: .ASCIZ  <15><12>'STARTING BANK #(8)? '
3415  012060  052122  047111  020107
3416  012066  040502  045516  021440
3417  012074  034050  037451  000040
3418  012102  005015  020043  043117  BANKS:  .ASCIZ  <15><12>'# OF 4K BANKS TO TEST(8)? '
3419  012110  032040  020113  040502
3420  012116  045516  020123  047524
3421  012124  052040  051505  024124
3422  012132  024470  020077     000
3423  012137     015  050012  052101  PAT:    .ASCIZ  <15><12>'PATTERN #? '
3424  012144  042524  047122  021440
3425  012152  020077     000
3426  012155     015  037412     000  QUEST:  .ASCIZ  <15><12>'?'
3427  012161     015  052012  050131  CONST:  .ASCIZ  <15><12>'TYPE CONSTANT'
3428  012166  020105  047503  051516
3429  012174  040524  052116     000
3430  012201     015  044412  050116  PRG3M:  .ASCIZ  <15><12>'INPUT # OF 256. WORD BLOCKS TO TEST INSTEAD OF'
3431  012206  052125  021440  047440
```

```
 3432  012214  020106  032462  027066
 3433  012222  053440  051117  020104
 3434  012230  046102  041517  051513
 3435  012236  052040  020117  042524
 3436  012244  052123  044440  051516
 3437  012252  042524  042101  047440
 3438  012260  000106
 3439  012262  005015  054524  042520  PRG4M:  .ASCIZ  <15><12>'TYPE ADDRESS'
 3440  012270  040440  042104  042522
 3441  012276  051523     000
 3442  012301     015  052012  020117  RELOCM: .ASCIZ  <15><12>'TO RESTORE PROGRAM START AT '
 3443  012306  042522  052123  051117
 3444  012314  020105  051120  043517
 3445  012322  040522  020115  052123
 3446  012330  051101  020124  052101
 3447  012336  000040
 3448  012340  000052                  ASTERISK:.ASCIZ '*'
 3449  012342  055104  046521  020102  ENDMSG: .ASCIZ  'DZQMB DONE!'
 3450  012350  047504  042516  000041
 3451                                          .EVEN
 3452
 3453                                  ;ROUTINE TO RELOCATE PROGRAM BACK TO 0
 3454  012356  010700                  REL24K: MOV     PC,R0                   ;FORM BASE ADDRESS WHERE CODE
 3455  012360  042700  017777                  BIC     #17777,R0               ;IS RELOCATED
 3456  012364  010067  000004                  MOV     R0,1$                   ;PUT FROM ADDRESS INTO SUBROUTINE CALL
 3457  012370  004567  173652                  JSR     R5,RELOC                ;RELOCATE CODE TO
 3458  012374  000000                  1$:     0                               ;LOWEST 4K
 3459  012376  000000                          0
 3460  012400  012706  000500                  MOV     #STKPTR,SP              ;SET STACK PTR
 3461  012404  042737  100000  000750          BIC     #100000,@#LDDISP        ;CLEAR RELOCATION INDICATOR
 3462  012412  013737  000750  177570          MOV     @#LDDISP,@#DISPLAY      ;LOAD DISPLAY REGISTER
 3463  012420  005037  000754                  CLR     @#RELOCF                ;CLEAR RELOCATION FACTOR
 3464  012424  000005                          RESET                           ;DISABLE MEM MGMT
 3465  012426  005037  000176                  CLR     @#176                   ;PUT A HALT AT 176
 3466  012432  000137  000162                  JMP     @#162                   ;RESTORE LOADERS & HALT AT 176
 3467                                  ;
 3468                                          .SBTTL  BRANCH GOBBLE MOS MEMORY TEST
 3469                                  ;
 3470                                  ;       *****PROGRAM DESCRIPTION*****
 3471                                  ;
 3472                                  ;THIS IS A PSEUDO-MODIFIED VERSION OF THE BRANCH GOBBLE
 3473                                  ;MOS MEMORY EXERCISER. PSEUDO-MODIFIED BECAUSE THE
 3474                                  ;ORIGINAL CODE, TAKEN FROM THE DZQKA-A INSTRUCTION
 3475                                  ;EXERCISER,WHICH IS BRANCH GOBBLE IS INCLUDED
 3476                                  ;HERE IN ITS ORIGINAL FORM, BUT MEMORY MANAGEMENT
 3477                                  ;CABABILITIES HAVE BEEN ADDED TO GIVE IT OPERATING
 3478                                  ;ABLITIES IN A D-128K MEMORY ENVIRONMENT.
 3479                                  ;
 3480                                  ;
 3481                                  ;       *****OPERATING PROCEDURE*****
 3482                                  ;
 3483                                  ;WHEN LOADED THIS PROGRAM'S STARTING ADDRESS IS XXXXXX.
 3484                                  ;WHEN RUNNING THE FOLLOWING STEPS ARE TAKEN:
 3485                                  ;1.)    A PROGRAM ID IS TYPED ON THE TTY:
 3486                                  ;               BRANCH GOBBLE MOS TEST
 3487                                  ;1.5)   THE PROGRAM DETERMINES IF THERE IS MOS PARITY. IF YES IT IS ENABLED
```

```
                        ;       AND THE USER IS TOLD.
                        ;2.)    THE USER IS ASKED:
                        ;               USE MEMORY MANAGEMENT?(Y OR N)
                        ;               >
                        ;       IF THE USER TYPES Y, THE MEMORY MANAGEMENT
                        ;       WILL BE USED TO RUN THE TEST. IF HE TYPES
                        ;       N THEN MEMORY MANAGEMENT WILL NOT BE USED
                        ;       IN EITHER CASE THE ACTUAL TEST WILL BE
                        ;       PERFORMED BY THE UNMODIFIED ORIGINAL VERSION
                        ;       OF BRANCH GOBBLE.
                        ;       NOTE THAT WHEN THE TEST OF MEMORY
                        ;       LOCATED IN UNIBUS ADDRESSES HIGHER THAN
                        ;       177777 IS DESIRED, THE ENABLING OF MEMORY
                        ;       MANAGEMENT IN THIS PROGRAM IS MANDATORY.
                        ;       NOTE ALSO THAT WHEN THE TEST IS TO BE
                        ;       IN THE 0 TO 177777 RANGE OF UNIBUS ADDRESSES
                        ;       IT IS RECOMMENDED THAT MEMORY MANAGEMENT
                        ;       BE DISABLED.
                        ;3.)    THE PROGRAM WILL THEN ASK THE USER
                        ;       FOR THE HIGH ADDRESS LIMIT FOR THE TEST:
                        ;               WHAT IS THE HIGH LIMIT:
                        ;               >
                        ;       AND THEN FOR THE LOW LIMIT:
                        ;               WHAT IS THE LOW LIMIT:
                        ;               >
                        ;       BOTH OF THE ADDRESSES SHOULD BE SPECIFIED IN
                        ;       OCTAL USING THE FORM XXXX00, THAT IS THE
                        ;       ADDRESSES MUST BE THE BEGINNING OF 100 BYTE
                        ;       BLOCKS OF MEMORY. IF THEY ARE NOT THEN THEY
                        ;       WILL BE TRUNCATED!
                        ;       THE ADDRESSES WILL BE INTERPRETED AS FULL 18-BIT
                        ;       UNIBUS ADDRESSES.
                        ;       VALID ADDRESSES MEET THE FOLLOWING CONDITIONS:
                        ;               1. THE SPECIFIED SPAN OF THE TEST
                        ;               SHOULD NOT ENCOMPASS THE ACTUAL
                        ;               MEMORY LOCATIONS OCCUPIED BY THIS
                        ;               PROGRAM. THIS PROGRAM IS RELOCATABLE
                        ;               AT LOADING TIME, SO THAT NO LIMITATIONS
                        ;               ARE THUS IMPOSED.
                        ;               2. THE HIGH LIMIT SHOULD BE GREATER
                        ;               THAN THE LOW LIMIT.
                        ;               3. THE NUMBER MUST BE SPECIFIED BY
                        ;               OCTAL DIGITS.
                        ;               4. IF THE USER HAS DISABLED MEMORY
                        ;               MANAGEMENT THE ADDRESSES SHOULD BE
                        ;               IN LOW MEMORY.I.E. LESS THAN 177777.
                        ;       IF ANY OF THESE CONDITIONS IS NOT MET
                        ;       THE USER WILL BE ASKED TO INPUT ANOTHER
                        ;       ADDRESS.
                        ;       NOTE THAT IF ANY OF THE ADDRESSES IN THE TEST
                        ;       SPAN IS A NON-EXISTANT MEMORY LOCATION
                        ;       A TIME OUT ERROR WILL OCCUR FROM WHICH
                        ;       NO RECOVERY CAN BE MADE EXCEPT TO RESTART
                        ;       AT THE STARTING ADDRESS.
                        ;4.)    WHEN THE ABOVE INFORMATION HAS BEEN
                        ;       SUCCESSFULLY GATHERED FROM THE USER, THE
```

```
                                        ;       TEST WILL BEGIN. IF BRANCH GOBBLE ENCOUNTERS
                                        ;       A MEMORY FAULT DURING THE TEST THE
                                        ;       PROCESSOR WILL BE HALTED AT THE "LOCATION"
                                        ;       OF THE MEMORY FAULT.
                                        ;5.)    IF THE TEST IS COMPLETE WITHOUT AN
                                        ;       ENCOUNTER WITH A MEMORY FAULT ANY WHERE
                                        ;       IN THE TESTED SPAN, THEN THE TEST
                                        ;       WILL BE REPEATED BY RETURNING TO STEP 4.
                                        ;6.)    IF THE USER WISHES TO STOP THE TEST IN PROGRESS AND START
                                        ;       ANOTHER WITH A DIFFERENT RANGE THEN HE SHOULD HIT THE HALT
                                        ;       SWITCH AND START THE TEST AGAIN AT 260.
                                        ;
                                        ;
                                        ;
                                        ;
                                        ;
                                        ;
                                        ;
                                        ;
                                        ;
                                        ;
                                        ;
012436                                  TOP:
                                        ;CONSTANTS:
        077406                          KPDR=077406
                                        ;THIS ROUTINE TAKES CARE OF THE IDENTIFICATION,
                                        ;ASK THE USER IF MEMORY MANAGEMENT SHOULD BE
                                        ;ENABLED, AND IF NOT DOES THE SET UP FOR THE
                                        ;ACTUAL TEST. IF MEMORY MANAGEMENT IS
                                        ;DESIRED THE ROUTINE YMMBGO IS GIVEN CONTROL.
                                        ;
012436  000000                          LODFLG: .WORD   0
                                        ;
                                        ;
012440  005067  165454                  BRANCH: CLR     RELFL           ;INITIALIZE THE RELOCATION INDICATOR.
012444  005067  166302                          CLR     MMAVA           ;INITIALIZE THE MEM. MANAGEMENT FLAG.
012450  004567  166304                          JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
012454  015206                                  .WORD   IDMESS
012456  004767  167362                          JSR     PC,$LDR
012462  004767  001566                  15$:    JSR     PC,MOSPAR
012466                                  1$:
012466  004567  166266                          JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
012472  015141                                  .WORD   MMMESS          ;MEMORY MANAGEMENT SHOULD
                                                                        ;BE ENABLED
012474  105737  177560                  2$:     TSTB    @#TKS           ;WAIT FOR A CHARACTER.
012500  100375                                  BPL     2$
012502  113746  177562                          MOVB    @#TKB,-(SP)     ;GET THE CHARACTER.
012506  042716  177600                          BIC     #177600,(SP)
012512  004767  001520                          JSR     PC,TYPIT        ;ECHO THE CHARACTER.
012516  005726                                  TST     (SP)+           ;RESET THE STACK.
012520  022766  000131  177776                  CMP     #'Y,-2(SP)      ;IF IT IS Y, THEN GO
012526  001513                                  BEQ     YMMBGO          ;TO YMMBGO TO ENABLE
                                                                        ;MEMORY MANAGEMENT
012530  022766  000116  177776                  CMP     #'N,-2(SP)      ;IS IT N?
012536  001406                                  BEQ     3$
012540  012746  000077                          MOV     #'?,-(SP)       ;IF IT WAS NIETHER
```

```
3600  012544  004767  001466                    JSR     PC,TYPIT        ;Y OR N THEN ASK
3601  012550  005726                            TST     (SP)+           ;THE USER AGAIN.
3602  012552  000745                            BR      1$
3603  012554  004767  001066            3$:     JSR     PC,LIMITS       ;GO GET THE LIMITING
3604                                                                      ;ADDRESSES FOR THE TEST.
3605                                                                      ;THEY WILL BE LEFT AS
3606                                                                      ;BLOCK NUMBERS IN LOLIM
3607                                                                      ;AND HILIM.
3608  012560  022767  001600  001172  SPOT:   CMP     #1600,HILIM
3609  012566  002010                            BGE     SPOT2
3610  012570  012700  177770                    MOV     #SPOT-.,R0
3611  012574  060700                            ADD     PC,R0
3612  012576  062700  177772                    ADD     #-6,R0
3613  012602  010046                            MOV     R0,-(SP)
3614  012604  000167  001140                    JMP     LIMERR
3615  012610  005046                    SPOT2:  CLR     -(SP)
3616  012612  012746  002502                    MOV     #FIRST1-.,-(SP)
3617  012616  060716                            ADD     PC,(SP)
3618  012620  062716  177772                    ADD     #-6,(SP)
3619  012624  004767  000774                    JSR     PC,LOSEG4
3620  012630  004767  000436                    JSR     PC,LOSEG1
3621  012634  012746  000052            REPET1: MOV     #'*,-(SP)
3622  012640  004767  001372                    JSR     PC,TYPIT
3623  012644  005726                            TST     (SP)+
3624  012646  022767  000200  001104            CMP     #200,HILIM
3625  012654  002026                            BGE     REPET3
3626  012656  016700  001076                    MOV     HILIM,R0
3627  012662  004767  000054            4$:     JSR     PC,ROT
3628  012666  010067  002556                    MOV     R0,HI           ;SET THE PARAMETERS
3629                                                                      ;IN THE ACTUAL TEST
3630                                                                      ;ROUTINE.
3631  012672  016700  001064                    MOV     LOLIM,R0        ;DO THE SAME FOR THE
3632                                                                      ;LOW ADDRESS LIMIT.
3633  012676  004767  000040                    JSR     PC,ROT
3634  012702  010067  002544                    MOV     R0,LO
3635  012706  005037  000036                    CLR     @#36            ;SET UP THE VECTORS
3636  012712  012704  000020                    MOV     #REPET3-.,R4    ;FOR AN INTERUPT
3637  012716  060704                            ADD     PC,R4           ;FROM A TRAP INSTRUCTION
3638  012720  062704  177772                    ADD     #-6,R4          ;WHICH WILL BE USED TO
3639  012724  010437  000034                    MOV     R4,@#34         ;RETURN FROM THE TEST
3640                                                                      ;ROUTINE WHEN IT IS DONE.
3641  012730  000002                            RTI                     ;START THE TEST.
3642  012732  004767  000404            REPET3: JSR     PC,LOSEG2
3643  012736  000167  177672                    JMP     REPET1
3644
3645                                    ;
                                        ;THIS ROUTINE IS CALLED TO SHIFT R0 TO THE LEFT SIX BITS.
3646  012742  012701  177772            ROT:    MOV     #-6,R1
3647  012746  006300                    1$:     ASL     R0
3648  012750  005201                            INC     R1
3649  012752  002775                            BLT     1$
3650  012754  000207                            RTS     PC
3651
3652                                    ;
                                        ;YMMBGO SETS UP FOR USING MEMORY MANAGEMENT TO
3653                                    ;DO THE MOS TEST.
3654                                    ;ALL THE MEMORY MANAGEMENT REGISTERS WHICH ARE
3655                                    ;TO REMAIN STATIC FOR THE TESTS DURATION ARE SET.
```

```
                                        ;THE LIMITS ARE THEN GOTTEN AND CHECKED FOR VALIDITY.
                                        ;THE TRAP INTERPUPT VECTORS ARE SET AND YMMBG1
                                        ;IS CALLED. YMMBG1 IS A ROUTINE THE SETS THOSE
                                        ;MEMORY MANAGEMENT REGISTERS WHICH NEED TO
                                        ;BE CHANGED DYNAMICALLY DURING THE TEST.
                                        ;
012756  012767  177777  165766  YMMBG0: MOV     #-1,MMAVA
012764  005037  177572                  CLR     @#SR0           ;SET ALL THE STATIC
012770  012700  172340                  MOV     #KIPAR0,R0      ;REGISTERS.
012774  012701  172300                  MOV     #KIPDR0,R1
013000  005003                          CLR     R3
013002  012704  177770                  MOV     #-10,R4
013006  010320                  1$:     MOV     R3,(R0)+
013010  062703  000200                  ADD     #200,R3
013014  012721  077406                  MOV     #KPDR,(R1)+
013020  005204                          INC     R4
013022  001371                          BNE     1$
013024  004767  000616                  JSR     PC,LIMITS       ;GET THE LIMITS FOR THE PENDING
                                                                ;TEST. THEY ARE LEFT IN BLOCK NUMBER
                                                                ;FORM IN HILIM AND LOLIM.
013030  004767  000570                  JSR     PC,LOSEG4       ;SEE IF PERMANENT RELOCATION IS
                                                                ;APPROPRIATE FOR THIS TEST SPAN.
013034  012737  007600  172356          MOV     #7600,@#KIPAR7  ;MAP THE UNIBUS DEVICE PAGE INTO
                                                                ;INTO HIGH VIRTUAL MEMORY.
013042  005046                  REPET2: CLR     -(SP)
013044  012746  002250                  MOV     #FIRST1-.,-(SP)
013050  060716                          ADD     PC,(SP)
013052  062716  177772                  ADD     #-6,(SP)        ;SET UP THE STACK
                                                                ;TO SIMULATE THE OCCURRENCE OF AN
                                                                ;AN INTERRUPT SO THAT THE TEST CAN
                                                                ;BE STARTED USING AN RTI.
013056  016767  000704  000674          MOV     HISAV,HILIM
013064  016767  000674  000670          MOV     LOSAV,LOLIM
013072  004767  000174                  JSR     PC,LOSEG1
013076  012746  000052                  MOV     #'*,-(SP)
013102  004767  001130                  JSR     PC,TYPIT
013106  005726                          TST     (SP)+
013110  012737  013122  000034          MOV     #YMMBG1,@#34
013116  005037  000036                  CLR     @#36            ;SET UP THE TRAP INTERRUPT VECTOR
                                                                ;WHICH WILL BE USED TO RETURN FROM THE
                                                                ;TESTING ROUTINE.
                                        ;
                                        ;YMMBG1 IS USED TO DYNAMICALLY ALLOCATE MEMORY UNDER MEMRY MANAGEMENT
                                        ;WHILE A TEST IS IN PROGRESS. WORKING UPWARDS FROM THE LOLIM
                                        ;MEMORY MANAGEMENT IR SET TO ENABLE BRGOB TO WORK THROUGH AS
                                        ;MUCH OF THE TEST SPAN AS POSSIBLE IN A SINGLE MANAGEMENT SET UP
                                        ;BEFORE HAVING TO RESET THE MANAGEMENT REGISTERS.
                                        ;BRGOB ALWAYS IS IN LOW VIRTUAL MEMORY AND UPPER VIRTUAL
                                        ;ADDRESSES ARE ALWAYS MAPPED INTO UNIBUS DEVICE ADDRESSES.
                                        ;
013122  005037  177572          YMMBG1: CLR     @#SR0
013126  026767  000626  000626          CMP     HILIM,LOLIM     ;IS THE TEST DONE?
013134  101451                          BLOS    DONIT           ;YES, THEN BRANCH.
013136  012700  172342                  MOV     #KIPAR1,R0      ;ELSE GET READY TO SET
013142  016705  000614                  MOV     LOLIM,R5        ;THE KERNAL PAGE ADDRESS REGISTERS.
013146  012701  177772                  MOV     #-6,R1
```

```
3712  013152  012767  020000  002272          MOV     #20000,LO
3713  013160  010520                  1$:     MOV     R5,(R0)+        ;RESET THE KIPAR'S
3714  013162  062705  000200          2$:     ADD     #200,R5
3715  013166  026705  000566          25$:    CMP     HILIM,R5        ;REACHED HILIM?
3716  013172  101407                          BLOS    3$              ;YES, GOTO 3$.
3717  013174  005201                          INC     R1              ;NO, INCREMENT R1 AND SEE IF ALL THE
3718                                                                  ;KIPAR'S HAVE BEEN SET.
3719  013176  002770                          BLT     1$              ;ALL THE KIPAR'S HAVE NOT BEEN SET SO
3720                                                                  ;LOOP TO GET THE NEXT ONE.
3721  013200  010567  002244                  MOV     R5,HI           ;DO THIS IS ALL THE TEST SPAN HAS NOT
3722                                                                  ;BEEN ALLOCATED TO SOME VIRTUAL ADDRSSES
3723                                                                  ;IS THE KERNAL INSTRUCTION SPACE.
3724  013204  162705  000002                  SUB     #2,R5
3725  013210  000403                          BR      4$
3726  013212  016767  000542  002230  3$:     MOV     HILIM,HI        ;DO THIS IF ALL THE TEST SPAN HAS BEEN
3727                                                                  ;ALLOCATED TO THE VIRTUAL KERNAL SPACE
3728                                                                  ;JUST ALLOCATED.
3729  013220  166767  000536  002222  4$:     SUB     LOLIM,HI        ;COMPUTE THE VIRTUAL LIMIT
3730                                                                  ;OF THE TEST SPAN.
3731  013226  016700  002216                  MOV     HI,R0
3732  013232  004767  177504                  JSR     PC,ROT
3733  013236  062700  020000                  ADD     #20000,R0
3734  013242  010067  002202                  MOV     R0,HI
3735  013246  010567  000510                  MOV     R5,LOLIM
3736  013252  005237  177572                  INC     @#SR0           ;TURN ON MEMORY MANAGEMENT.
3737  013256  000002                          RTI                     ;RETURN TO BRGOB TO
3738                                                                  ;PERFORM THE TEST IN
3739                                                                  ;THE SPAN INDICATED
3740                                                                  ;BY THE RESULT OF THE
3741                                                                  ;ABOVE
3742  013260  004767  000056          DONIT:  JSR     PC,LOSEG2
3743  013264  022626                          CMP     (SP)+,(SP)+
3744  013266  000167  177550                  JMP     REPET2          ;TEST COMPLETED, SO
3745                                                                  ;RESTART
3746                                  ;
3747                                  ;
3748                                  ;LOSEG1 IS CALLED TO DECIDE WHETHER OR NOT THE LIMITS ARE SUCH THAT
3749                                  ;THE FIRST 4K OF MEMORY WILL HAVE TO BE CHECKED IN THE TEST. THAT
3750                                  ;IS DO THE LIMITS INCLUDE ADDRESSES WHICH LIEM IN THE  FIRST 4K BLOCK
3751                                  ;OF MEMORY THUS REQUIRING THAT THE CONTENTS OF THIS FIRST BLOCK OF
3752                                  ;MEMORY BE MOVED INTO THE SECOND 4K BLOCK SO THAT THE TEST CAN BE
3753                                  ;RUN THROUGH THE FIRST 4K BLOCK. IF RELOCATION IS NECESSARY LOSEG1
3754                                  ;SETS LOSFL TO -1 AND RESETS THE LIMITS, LOLIM AND HILIM,
3755                                  ;APPROPRIATELY.
3756  013272  005067  000324          LOSEG1: CLR     LOSFL           ;INITIALIZE.
3757  013276  026727  000462  000200          CMP     LOSAV,#200      ;SEE IF THE LOW LIMIT OR THE HIGH LIMIT
3758  013304  103011                          BHIS    2$              ;LIE IN THE FIRST 4K BLOCK OF MEMORY.
3759  013306  005367  000310                  DEC     LOSFL
3760  013312  022767  000200  000446          CMP     #200,HISAV
3761  013320  103004                          BHIS    3$
3762  013322  012767  000200  000432  1$:     MOV     #200,LOLIM
3763  013330  000207                  2$:     RTS     PC
3764  013332  012767  000200  000420  3$:     MOV     #200,HILIM
3765  013340  000770                          BR      1$
3766                                  ;
3767                                  ;LOSEG2 IS CALLED TO DO THE ACTUAL RELOCATION OF THE FIRST 4K MEMORY
```

```
3768                                    ;BANK INTO THE SECOND 4K MEMORY BANK, THEN RUN THE TEST
3769                                    ;THROUGH THE DESIGNATED PARTS OF THE FIRST BANK AND THEN RESTORE
3770                                    ;THE CONTENTS OF THE FIRST BANK BY MOVING THE SECOND BANK'S
3771                                    ;CONTENTS BACK INTO THE FIRST BANK.
3772  013342  005767  000254            LOSEG2: TST     LOSFL           ;SEE IF RELOCATION IS NECESSARY.
3773  013346  100401                            BMI     1$              ;IF NOT RETURN.
3774  013350  000207                            RTS     PC
3775  013352  012667  000242            1$:     MOV     (SP)+,SAVPC
3776  013356  005737  000120                    TST     @#120           ;SEE IF THE PROGRAM IS ALREADY RELOCATED
3777  013362  100423                            BMI     14$
3778  013364  004567  172656                    JSR     R5,RELOC
3779  013370  000000                            .WORD   0
3780  013372  020000                            .WORD   20000
3781  013374  012737  177777  000120            MOV     #-1,@#120
3782  013402  012701  020000                    MOV     #20000,R1
3783  013406  060106                            ADD     R1,SP
3784  013410  060116                            ADD     R1,(SP)
3785  013412  060107                            ADD     R1,PC
3786  013414  060137  000114                    ADD     R1,@#114
3787  013420  060167  000174                    ADD     R1,SAVPC
3788  013424  012767  177777  164466          MOV     #-1,RELFL
3789  013432  016700  000326            14$:    MOV     LOSAV,R0        ;ESTABLISH VALID LIMITS FOR THE TEST
3790  013436  004767  177300                    JSR     PC,R0T          ;THROUGH THE FIRST-4K MEMORY BANK.
3791  013442  020027  000320                    CMP     R0,#320
3792  013446  002002                            BGE     15$
3793  013450  012700  000320                    MOV     #320,R0
3794  013454  010067  001772            15$:    MOV     R0,LO
3795  013460  016700  000302                    MOV     HISAV,R0
3796  013464  022700  000200                    CMP     #200,R0
3797  013470  002002                            BGE     2$
3798  013472  012700  000200                    MOV     #200,R0
3799  013476  020027  000004            2$:     CMP     R0,#4
3800  013502  003002                            BGT     3$
3801  013504  000167  000034                    JMP     LOSEG3
3802  013510  004767  177226            3$:     JSR     PC,R0T
3803  013514  010067  001730                    MOV     R0,HI
3804                                    ;
3805                                    ;
3806  013520  012701  000024                    MOV     #LOSEG3-.,R1
3807  013524  060701                            ADD     PC,R1
3808  013526  062701  177772                    ADD     #-6,R1
3809  013532  010137  000034                    MOV     R1,@#34
3810  013536  005037  000036                    CLR     @#36
3811  013542  000002                            RTI                     ;PERFORM THE TEST
3812                                    ;
3813                                    ;
3814                                    ;LOSEG3 RELOCATES BACK INTO THE FIRST 4K MEMORY BANK.
3815  013544  016746  000050            LOSEG3: MOV     SAVPC,-(SP)
3816  013550  005767  000042                    TST     PERRFL
3817  013554  100417                            BMI     1$
3818  013556  004567  172464                    JSR     R5,RELOC
3819  013562  020000                            .WORD   20000
3820  013564  000000                            .WORD   0
3821  013566  005037  000120                    CLR     @#120
3822  013572  012701  020000                    MOV     #20000,R1
3823  013576  160106                            SUB     R1,SP
```

```
3824  013600  160116                          SUB     R1,(SP)
3825  013602  160137  000114                  SUB     R1,@#114
3826  013606  160107                          SUB     R1,PC
3827  013610  160166  000002                  SUB     R1,2(SP)
3828  013614  000207                  1$:     RTS     PC
3829  013616  000000                  PERRFL: .WORD   0
3830  013620  000000                  SAVPC:  .WORD   0
3831  013622  000000                  LOSFL:  .WORD   0
3832                                  ;
3833                                  ;
3834  013624  005067  177766          LOSEG4: CLR     PERRFL
3835  013630  026727  000132  000200        CMP     HISAV,#200
3836  013636  101002                        BHI     1$
3837  013640  005367  177752                DEC     PERRFL
3838  013644  000207                  1$:     RTS     PC
3839                                  ;
3840                                  ;
3841                                  ;
3842                                  ;LIMITS IS CALLED TO ASK THE USER FOR BOTH
3843                                  ;THE HIGH AND LOW UNIBUS ADDRESS LI,ITS FOR
3844                                  ;THE IMPENDING TEST. THE TWO LIMITS ARE LEFT IN
3845                                  ;BLOCK NUMBER FORM AT LOCATIONS HILIM AND
3846                                  ;LOLIM (THEY ARE ALSO PUT IN LOSAV AND HISAV FOR
3847                                  ;LATER USE BY THE ROUTINE DONE). A VALIDITY CHECK
3848                                  ;IS MADE TO MAKE SURE THE INDICATED SPAN
3849                                  ;IS A VALID TEST SPAN.
3850                                  ;
3851  013646                          LIMITS:
3852  013646  004567  165106                JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
3853  013652  015104                        .WORD   HIMESS
3854  013654  004767  000110                JSR     PC,INUM ;ASSEMBLE THIS NUMBER.
3855  013660  004767  000334                JSR     PC,THRR
3856  013664  004767  000330                JSR     PC,THRR
3857  013670  010367  000064                MOV     R3,HILIM
3858  013674  004567  165060                JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
3859  013700  015124                        .WORD   LOMESS
3860  013702  004767  000062                JSR     PC,INUM         ;ASSEMBLE THIS NUMBER
3861  013706  004767  000306                JSR     PC,THRR
3862  013712  004767  000302                JSR     PC,THRR
3863  013716  010367  000040                MOV     R3,LOLIM
3864  013722  026767  000032  000032        CMP     HILIM,LOLIM     ;IF LOLIM IS GREATER
3865  013730  003407                        BLE     LIMERR          ;THAN HILIM THEN GOTO
3866                                                                ;LIMERR, ERROR
3867  013732  016767  000022  000026  2$:   MOV     HILIM,HISAV     ;STORE THE LIMITS IN
3868  013740  016767  000016  000016        MOV     LOLIM,LOSAV     ;SAVE REGISTERS.
3869  013746  000207                        RTS     PC              ;RETURN
3870  013750                          LIMERR:
3871  013750  004567  165004                JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
3872  013754  015171                        .WORD   ERRMESS         ;WRITE AN ERROR MESSAGE
3873  013756  000733                        BR      LIMITS          ;AND TRY AGAIN.
3874                                  ;
3875  013760  000000                  HILIM:  .WORD   0
3876  013762  000000                  LOLIM:  .WORD   0
3877                                  ;THESE ARE INTERMEDIATE STORGE REGISTERS:
3878  013764  000000                  LOSAV:  .WORD   0
3879  013766  000000                  HISAV:  .WORD   0
```

```
3880                                    ;
3881                                    ;
3882                                    ;THIS ROUTINE IS CALLED TO ASSEMBLE AN 18-BIT
3883                                    ;NUMBER FROM THE TTY AND TRUNCATE IT DOWN TO
3884                                    ;12-BITS
3885                                    ;A CALL IS MADE THUS:
3886                                    ;       JMP     PC,INUM
3887                                    ;RES:   .WORD   0
3888                                    ;THE NUMBER IS ASSEMBLED AND THE RESULTING 12-BIT
3889                                    ;TRUNCATED NUMBER IS LEFT IN RES. WHEN AND
3890                                    ;RTS RETURN IS MADE.
3891                                    ;NOTE THAT THE NUMBER SHOULD BE SPECIFIED
3892                                    ;IN OCTAL DIGITS. AN CHARACTERS WHICH DO NOT
3893                                    ;MEET THIS SPECIFICATION IN THE INPUT STRING
3894                                    ;WILL CAUSE AN ERROR WHICH WILL BE SIGNALED BY
3895                                    ;A ? ON THE TTY FOLLOWED BY A RETRY.
3896                                    ;
3897  013770  005046                    INUM:   CLR     -(SP)           ;PUT A ZERO MARKER ON
3898  013772  005002                            CLR     R2              ;SET UP THE TEMPORARY
3899  013774  005003                            CLR     R3              ;STORAGE AND COUNTER
3900  013776  012705  177771                    MOV     #-7,R5          ;REGISTERS.
3901  014002  005737  177562                    TST     @#TKB
3902  014006  012746  000076                    MOV     #'>,-(SP)
3903  014012  004767  000220                    JSR     PC,TYPIT
3904  014016  005726                            TST     (SP)+
3905  014020  105737  177560            1$:     TSTB    @#TKS           ;WAIT FOR A CHARACTER
3906  014024  100375                            BPL     1$
3907  014026  013746  177562                    MOV     @#TKB,-(SP)     ;GET IT ONTO THE
3908  014032  042716  177600                    BIC     #177600,(SP)    ;STACK
3909  014036  022716  000177                    CMP     #177,(SP)       ;IS IT RUBOUT?
3910  014042  001011                            BNE     2$              ;IF NOT GOTO 2$
3911  014044  005726                            TST     (SP)+           ;IF IT WAS A RUBOUT
3912  014046  005716                            TST     (SP)            ;FIRST SEE IF THEE
3913                                                                    ;IS A PREVIOUS
3914                                                                    ;CHARACTER ON THE STACK.
3915  014050  001763                            BEQ     1$              ;IF THERE WAS NO PREVIOUS
3916                                                                    ;CHARACTER TAKE NO
3917                                                                    ;RUBOUT ACTION AND
3918                                                                    ;GO WAIT AT 1$ FOR
3919                                                                    ;THE NEXT CHARACTER
3920  014052  012716  000134                    MOV     #'\,(SP)        ;IF THERE WAS
3921  014056  004767  000154                    JSR     PC,TYPIT        ;A PREVIOUS CHARACTER
3922  014062  005726                            TST     (SP)+           ;PRINT A SLASH
3923  014064  000755                            BR      1$
3924  014066  004767  000144            2$:     JSR     PC,TYPIT        ;IF THE LAST INPUT
3925                                                                    ;CHARACTER WAS NOT
3926                                                                    ;RUBOUT ECHO IT
3927  014072  022716  000015                    CMP     #15,(SP)        ;IS IT CR.
3928  014076  001350                            BNE     1$              ;NO, BRANCH TO 1$ FOR
3929                                                                    ;NEXT CHARACTER.
3930  014100  012716  000012                    MOV     #12,(SP)        ;YES, PRINT A LF
3931  014104  004767  000126                    JSR     PC,TYPIT
3932  014110  005726                            TST     (SP)+
3933  014112  005716            3$:     TST     (SP)            ;START TO ASSEMBLE
3934  014114  001415                            BEQ     4$              ;THE NUMBER. IF THE
3935                                                                    ;STACK IS AT THE
```

```
3936                                                            ;ZERO MARKER WE ARE
3937                                                            ;DONE, SO GOTO 4$
3938  014116  012604                        MOV     (SP)+,R4    ;CHECK TO SEE IF
3939  014120  062704  177710                ADD     #-70,R4     ;THE CHARACTER IS A
3940  014124  002022                        BGE     INERR       ;VALID OCTAL DIGIT.
3941  014126  062704  000010                ADD     #10,R4      ;IF NOT GOTO INERR.
3942  014132  002417                        BLT     INERR
3943  014134  005205                        INC     R5          ;ARE THERE TO MANY DIGITS,
3944  014136  001415                        BEQ     INERR       ;MORE THAN 6.
3945  014140  004767  000054                JSR     PC,THRR     ;ROTATE THE DOUBLE
3946                                                            ;LENGTH "WORD" MADE UP
3947                                                            ;OF THE DIGITS PROCESSED
3948                                                            ;THUS FAR
3949  014144  010402                        MOV     R4,R2       ;MAKE THIS NEW DIGIT
3950                                                            ;PART OF THE NUMBER.
3951  014146  000761                        BR      3$          ;LOOP TO GET THE REST OF
3952                                                            ;THE NUMBER
3953  014150  005205              4$:       INC     R5          ;MAKE SURE THE NUMBER
3954  014152  001403                        BEQ     5$          ;HAS BEEN RIGHT JUSTIFIED
3955  014154  004767  000040                JSR     PC,THRR     ;PROPERLY.
3956  014160  000773                        BR      4$
3957  014162  004767  000022        5$:     JSR     PC,ONER
3958  014166  005726                        TST     (SP)+       ;AND RETURN CONTROL.
3959  014170  000207                        RTS     PC
3960                              ;ERROR HANDLER FOR THE INUM ROUTINE.
3961                              ;RETURNS TO THE CALLING ROUTINE, LIMITS, TO ASK
3962                              ;FOR THE PARAMETER AGAIN.
3963  014172                      INERR:
3964  014172  004567  164562                JSR     R5,$PRINT   ;GO TO PRINT ROUTINE
3965  014176  015135                        .WORD   INRMES
3966  014200  005726              1$:       TST     (SP)+       ;CLEAR JUNK OFF THE
3967  014202  001376                        BNE     1$          ;STACK
3968  014204  000167  177560                JMP     INUM    ;THE ROUTINE LIMITS.
3969  014210  000241              ONER:     CLC                 ;ROUTINE WHICH ROTATES
3970  014212  006002                        ROR     R2          ;TEMP2 AND TEMP3 ONE
3971  014214  006003                        ROR     R3          ;BIT TO THE RIGHT
3972  014216  000207                        RTS     PC          ;TREATING THEM AS A 32-BIT
3973                                                            ;OPERAND.
3974  014220  004767  177764        THRR:   JSR     PC,ONER     ;ROUTINE WHICH CALL
3975  014224  004767  177760                JSR     PC,ONER     ;ONER 3 TIMES.
3976  014230  004767  177754                JSR     PC,ONER
3977  014234  000207                        RTS     PC
3978  014236  105737  177564        TYPIT:  TSTB    @#TPS       ;TYPIT TAKES THE
3979  014242  100375                        BPL     TYPIT       ;WORD 2 BYTES UP IN
3980  014244  116637  000002  177566        MOVB    2(SP),@#TPB ;THE STACK AND "PRINTS#
3981  014252  000207                        RTS     PC          ;IT ON THE TTY
3982                              ;MOSPAR IS CALLED TO CHECK OUT THE POSIBILITY OF TURNING ON MOS
3983                              ;PARITY DURING THE BRANCH GOBBLE TEST. FIRST MOSPAR SEES WHAT
3984                              ;MOS PARITY REGISTERS EXIST AND THEN IFOME ARE FOUND THEY
3985                              ;ARE ENABLED AND THE INTERRUPT VECTOR IS SET TO TRAP TO PARERR A ROUTINE
3986                              ;WHICH WILL NOTIFY THE USER OF THE ERROR AND ITS LOCATION.
3987                              ;
3988          172100              PAREGS=172100
3989  014254  004767  000454        MOSPAR: JSR     PC,MPVECT   ;SET THE PARITY ERROR TRAP VECTOR.
3990  014260  012700  172100                MOV     #PAREGS,R0  ;GET READY TO LOOK AT THE POSSIBLE
3991                                                            ;PARITY REGISTERS PRESENT.
```

```
 3992  014264  012701  177762                MOV     #-16,R1
 3993  014270  012737  014352  000004        MOV     #MP2,@#4
 3994  014276  005037  000006                CLR     @#6
 3995  014302  005067  000110                CLR     MPFL
 3996                                ;THE REAL REGISTERS ARE LOCATED USING A TIME OUT PLAN WHERE BY THE
 3997                                ;POSSIBILITIES ARE REFERENCED AND IF A TIME OUT OCCURS THEY ARE NONEXISTENT
 3998                                ;OR IF NO TIME OUT OCCURS THEN THE REGISTER IS REAL. WHEN A REAL
 3999                                ;REGISTER IS FOUND IT IS WRITTEN INTO AND READ TO DETERMINE
 4000                                ;WHAT KIND OF PARITY REGISTER IT IS, CORE OR MOS. IF IT IS MOS THEN THE
 4001                                ;REGISTER IS SET TO ENABLE PARITY.
 4002  014306  005710                MP1:    TST     (R0)            ;IF THIS INSTRUCTION TIMES OUT THEN
 4003                                                                ;THERE IS NO PARITY REGISTER AT THE
 4004                                                                ;ADDRESS IN R0.
 4005  014310  005767  000102                TST     MPFL            ;SEE IF THE TABLE HEADING HAS BEEN OUTPUT.
 4006  014314  100403                        BMI     1$
 4007  014316  004567  164436                JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
 4008  014322  015000                        .WORD   MPMES
 4009                                ;
 4010  014324  010046                1$:     MOV     R0,-(SP)                ;OUTPUT AN ENTRY INTO THAT TABLE.
 4011  014326  004767  165414                JSR     PC,O2A
 4012  014332  004567  164422                JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
 4013  014336  015020                        .WORD   $CRLF
 4014  014340  005367  000052                DEC     MPFL
 4015  014344  012710  000001                MOV     #1,(R0)
 4016  014350  000401                        BR      MP25
 4017                                ;
 4018                                ;IF A TIME OCCURS COME HERE. OR IF THE REGISTER WAS NOT MOS PARITY.
 4019  014352  022626                MP2:    CMP     (SP)+,(SP)+
 4020  014354  062700  000002        MP25:   ADD     #2,R0
 4021  014360  005201                        INC     R1
 4022  014362  002751                        BLT     MP1             ;BRANCH IF NOT DONE.
 4023  014364  005767  000026                TST     MPFL            ;SEE IF ANY MOS PARITY REGISTERS HAVE
 4024                                                                ;BEEN FOUND.
 4025  014370  100403                        BMI     MP3             ;IF NOT DON'T OUT PUT THE NO PARITY MESSAGE.
 4026  014372  004567  164362                JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
 4027  014376  015027                        .WORD   NOMPAR
 4028  014400  012737  001116  000004  MP3:  MOV     #ERRTRP,@#4
 4029  014406  012737  000002  000006        MOV     #2,@#6
 4030  014414  000207                        RTS     PC
 4031  014416  000000                MPFL:   .WORD   0
 4032                                ;
 4033                                ;IF A PARITY ERROR IS DETECTED THEN A TRAP IS MADE THROUGH LOCATION
 4034                                ;114 TO THIS ROUTINE. HERE THE USER IS TOLD OF THE PARITY ERROR AND THE
 4035                                ;LOCATION PLUS 2 OF THE INSTRUCTION WHICH CAUSED THE ERROR TO COME OUT.
 4036                                ;THEN A SCAN IS MADE THROUGH MEMORY. IF THE ERROR DOESN'T COME UP DURING
 4037                                ;THE SCAN THEN THE USER IS TOLD THAT THE ERROR WAS NOT FOUND ON THE SCAN.
 4038                                ;IF THE ERROR IS DETECTED ON THE SCAN THEN THE USER IS TOLD WHAT LOCATION.
 4039                                ;CAUSED THE ERROR AND WHETHER OR NOT MEMORY MANAGEMENT WAS ON DURING THE
 4040                                ;SCAN. IF IT WAS THEN THE PAR INVOLVED IN RELOCATING THE BAD LOCATION'S
 4041                                ;ADDRESS IS ALSO GIVEN TO THE USER SO HE CAN TAKE THE GIVEN
 4042                                ;ADDRESS AND RELOCATE IT USING THE PAR GIVEN TO FIND THE REAL MEMORY
 4043                                ;ADDRESS CAUSING THE PARITY ERROR. WHEN THIS HAS BEEN DONE THE TEST WHICH
 4044                                ;WAS IN PROGRESS WHEN THE ERROR WAS ENCOUNTERED IS RESTARTED, USING
 4045                                ;THE PARAMETERS WHICH THE USER LAST INPUT TO THE PROGRAM.
 4046  014420                        PARER2:
 4047  014420  004567  164334                JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
```

```
4048  014424  006676                          .WORD   PARERR          ;TELL THE USER A PARITY ERROR OCCURRED.
4049  014426  004567  164326                  JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
4050  014432  015023                          .WORD   PERMES  ;TELL THE USER WHAT INSTRUCT CAUSED THE ERROR.
4051  014434  004767  165306                  JSR     PC,02A
4052  014440  004567  164314                  JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
4053  014444  015020                          .WORD   $CRLF
4054                                  ;
4055  014446  012700  000052                  MOV     #PARR3-.,R0     ;SET THE TIME AND TRAP VECTORS SO
4056  014452  060700                          ADD     PC,R0           ;THAT THE SCAN CAN BE MADE
4057  014454  062700  177772                  ADD     #-6,R0
4058  014460  010037  000114                  MOV     R0,@#PARVEC
4059  014464  005037  000116                  CLR     @#PARVEC+2
4060  014470  012700  000176                  MOV     #PARR4-.,R0
4061  014474  060700                          ADD     PC,R0
4062  014476  062700  177772                  ADD     #-6,R0
4063  014502  010037  000004                  MOV     R0,@#ERRVEC
4064  014506  005037  000006                  CLR     @#ERRVEC+2
4065  014512  005003                          CLR     R3
4066                                  ;
4067  014514                          CONT:
4068  014514  012301                  1$:     MOV     (R3)+,R1        ;PERFORM THE SCAN. EITHER THIS INSTRUCTION.
4069  014516  000776                          BR      1$              ;WILL TIME OUT OR THE PARITY ERROR WILL
4070                                                                  ;ONCE AGAIN OCCUR SO THAT CONTROL WILL
4071                                                                  ;BE TRANSFERRED OUT OF THIS LOOP ONE WAY
4072                                                                  ;OR ANOTHER.
4073                                  ;
4074                                  ;IF THE PARITY ERROR IS AGAIN ENCOUNTERRED THEN THIS PROGRAM WILL
4075                                  ;RECIEVE CONTROL THROUGH A TRAP THROUGH VECTOR 114. HERE THE USER
4076                                  ;IS GIVEN THE ADDRESS OF THE LOCATION WHICH CAUSED THE PARITY ERROR
4077                                  ;DURING THE SCAN. THE USER IS ALSO TOLD WHETHER OR NOT MEMORY MANAGEMENT
4078                                  ;IS ENABLED OR NOT. IF IT IS ENABLED THEN THE PAR INVOLVED WITH
4079                                  ;THE ADDRESS RELOCATION IS ALSO GIVEN TO THE USER.
4080  014520  062703  177776          PARR3:  ADD     #-2,R3          ;R3 CONTAINS THE ADDRESS PLUS 2
4081  014524  010316                          MOV     R3,(SP)         ;OF THE LOCATION CAUSING THE ERROR.
4082  014526  062703  000002                  ADD     #2,R3
4083  014532  010367  000126                  MOV     R3,SAVPER
4084  014536  004767  165204                  JSR     PC,02A  ;GIVE THE USER THIS ADDRESS.
4085  014542  004567  164212                  JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
4086  014546  015045                          .WORD   LOCBAD          ;TELL THE USER THIS IS THE ADDRESS AND
4087  014550  004567  164204                  JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
4088  014554  001506                          .WORD   RECDAT
4089  014556  010116                          MOV     R1,(SP)         ;GIVE HIM THE BAD DATA.
4090  014560  004767  165162                  JSR     PC,02A
4091                                  ;
4092  014564  033727  177572  000001          BIT     @#SR0,#1                ;SEE IF MEMORY MANAGEMENT WAS USED
4093  014572  001423                          BEQ     2$              ;DURING THE SCAN THROUGH MEMORY.
4094  014574  004767  177420          1$:     JSR     PC,THRR         ;IF IT WAS THEN ESTABLISH WHICH KIPAR
4095  014600  004767  177414                  JSR     PC,THRR         ;IS INVOLVED WITH THE RELOCATION OF
4096  014604  004767  177410                  JSR     PC,THRR         ;THIS BAD ADDRESS.
4097  014610  004767  177404                  JSR     PC,THRR
4098  014614  042703  177761                  BIC     #177761,R3
4099  014620  062703  172340                  ADD     #KIPAR0,R3
4100  014624  011316                          MOV     (R3),(SP)
4101  014626  004567  164126                  JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
4102  014632  015052                          .WORD   KPARM
4103  014634  004767  165106                  JSR     PC,02A
```

```
4104  014640  000403                        BR      RETPR           ;GO RESTART THE TEST WHICH WAS IN PROGRESS.
4105
4106  014642                        2$:
4107  014642  004567  164112                JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
4108  014646  015071                        .WORD   NOKT11          ;THE SCAN SO TELL
4109  014650  004567  164104        RETPR:  JSR     R5,$PRINT
4110  014654  015020                        .WORD   $CRLF
4111  014656  016703  000002                MOV     SAVPER,R3
4112  014662  000714                        BR      CONT
4113                                                                ;THE ADDRESS GIVEN FOR THE ERROR. THEN
4114                                                                ;RESTART THE TEST WHICH WAS IN PROGRESS
4115                                                                ;WHEN THE ERROR OCCURRED.
4116  014664  000000        SAVPER: .WORD   0
4117
4118  014666                        PARR4:
4119  014666  004567  164066                JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
4120  014672  014760                        .WORD   NOTIND          ;TELL THE USER THAT IS THE CASE AND
4121                                                                ;RESTART THE SCAN WHICH WAS IN PROGRESS
4122                                                                        ;WHEN THE ERROR WAS ORIGINALLY
4123                                                                ;ENCOUNTERRED.
4124
4125  014674  012706  000500                MOV     #500,SP         ;RESTART THE TEST WHICH WAS INPROGRESS
4126  014700  000005                        RESET                   ;WHEN THE ORIGINAL PARITY ERROR WAS FIRST
4127  014702  004737  000124                JSR     PC,@#WHERE      ;ENCOUNTERRED. SEE IF RELOCATION INTO THE
4128  014706  012707  014712                MOV     #1$,PC          ;FISRT 4K BANK IS NECESSARY.
4129
4130  014712  004767  177336        1$:     JSR     PC,MOSPAR       ;GO RESET THE PARITY REGISTERS BEFORE
4131  014716  005767  164030                TST     MMAVA           ;STARTING EITHER A MEMORY MANAGED
4132  014722  001402                        BEQ     2$              ;OR NON MEMORY MANAGED TEST.
4133  014724  000167  176112                JMP     REPET2
4134  014730  000167  175654        2$:     JMP     SPOT2
4135
4136
4137
4138
4139
4140                                        ;ROUTINE USED TO SET THE TRAP VECTOR 114.
4141  014734  012700  177464        MPVECT: MOV     #PARER2-.,R0
4142  014740  060700                        ADD     PC,R0
4143  014742  062700  177772                ADD     #-6,R0
4144  014746  010037  000114                MOV     R0,@#114
4145  014752  005037  000116                CLR     @#116
4146  014756  000207                        RTS     PC
4147
4148                                        ;MESSAGES USED FOR THESE PARITY ROUTINES.
4149  014760  041523  047101  041440  NOTIND: .ASCIZ  'SCAN COMPLETE'<15><12>
4150  014766  046517  046120  052105
4151  014774  006505  000012
4152  015000  005015  040520  044522  MPMES:  .ASCII  <15><12>'PARITY ENABLED'
4153  015006  054524  042440  040516
4154  015014  046102  042105
4155  015020  005015     000          $CRLF:  .ASCIZ  <15><12>
4156  015023     120  036503     000  PERMES: .ASCIZ  'PC='
4157  015027     015  047012  020117  NOMPAR: .ASCIZ  <15><12>'NO PARITY'<15><12>
4158  015034  040520  044522  054524
4159  015042  005015     000
```

```
                                        ;THESE ARE MESSAGES USED FOR COMMUNICATIONS
                                        ;ON THE TTY BY THE PROGRAM.
                                        ;
015045      040  040510  000104  LOCBAD: .ASCIZ  ' HAD'
015052  005015  052113  030461  KPARM:  .ASCIZ  <15><12>'KT11 ON PAR='
015060  047440  020116  040520
015066  036522      000
015071      015  045412  030524  NOKT11: .ASCIZ  <15><12>'KT11 OFF'
015076  020061  043117  000106
015104  005015  044510  044107  HIMESS: .ASCIZ  <15><12>'HIGH LIMIT?'<15><12>
015112  046040  046511  052111
015120  006477  000012
015124  047514  020127  044514  LOMESS: .ASCII  'LOW LIMIT'
015132  044515      124
015135      077  005015      000  INRMES: .ASCIZ  '?'<15><12>
015141      015  052412  042523  MMMESS: .ASCII  <15><12>'USE KT11? (Y OR N)'<15>
015146  045440  030524  037461
015154  024040  020131  051117
015162  047040  006451
015166  037012      000         .ASCIZ  <12>'>'
015171      116  052117  053040  ERRMES: .ASCIZ  'NOT VALID!'<15><12>
015176  046101  042111  006441
015204  000012
015206  005015  051102  047101  IDMESS: .ASCIZ  <15><12>'BRANCH GOBBLE'<15><12>
015214  044103  043440  041117
015222  046102  006505  000012
                                        .EVEN
                                        ;THE FOLLOWING IS THE CODE TAKEN DIRECTLY FROM
                                        ;THE PDP-11 FAMILY INSTRUCTION EXERCISER DZQKA-A.
                                        ;
                                        ;
015230  125252                  MARKER: .WORD   125252
015232  000401                  FIRST:  BR      .+4
015234  000000                          .WORD   0
015236  010703                          MOV     PC,R3
015240  162703  000004                    SUB     #4,R3
015244  010304                          MOV     R3,R4
015246  005204                          INC     R4
015250  005013                          CLR     (R3)
015252  000261                  1$:     SEC
015254  105513                          ADCB    (R3)
015256  100402                          BMI     2$
015260  105214                          INCB    (R4)
015262  000773                          BR      1$
015264  102401                  2$:     BVS     .+4
015266  000000                          HALT
015270  000242                          CLV
015272  105214                          INCB    (R4)
015274  103402                          BCS     INCB1
015276  102001                          BVC     INCB1
015300  100401                          BMI     .+4
015302  000000                  INCB1:  HALT
015304  000137  015372          RTAD:   JMP     @#RELO
015310                          LAST:
015310  000000                  STRTAD: .WORD   0
015312  000027                  LENGTH: .WORD   1+LAST-FIRST/2
```

```
 4216  015314  016700  000130          FIRST1: MOV     HI,R0
 4217  015320  012701  177770                  MOV     #LAST-.,R1
 4218  015324  060701                          ADD     PC,R1
 4219  015326  062701  177772                  ADD     #-6,R1
 4220  015332  012703  000040                  MOV     #RELO-.,R3
 4221  015336  060703                          ADD     PC,R3
 4222  015340  062703  177772                  ADD     #-6,R3
 4223  015344  010367  177736                  MOV     R3,RTAD+2
 4224  015350  016702  177736                  MOV     LENGTH,R2
 4225  015354  014140                  ABC:    MOV     -(R1),-(R0)
 4226  015356  005302                          DEC     R2
 4227  015360  001375                          BNE     ABC
 4228  015362  010067  177722                  MOV     R0,STRTAD
 4229  015366  000177  177716                  JMP     @STRTAD
 4230  015372  016700  177712          RELO:   MOV     STRTAD,R0
 4231  015376  162767  000002  177704          SUB     #2,STRTAD
 4232  015404  026767  177700  000040          CMP     STRTAD,LO
 4233  015412  101414                          BLOS    RET
 4234  015414  016701  177672                  MOV     LENGTH,R1
 4235  015420  011060  177776          RELO1:  MOV     (R0),-2(R0)
 4236  015424  005720                          TST     (R0)+
 4237  015426  005301                          DEC     R1
 4238  015430  001373                          BNE     RELO1
 4239  015432  016737  177652  177570          MOV     STRTAD,@#177570
 4240  015440  000177  177644                  JMP     @STRTAD
 4241  015444  104400                  RET:    TRAP
 4242  015446  000722                          BR      FIRST1
 4243  015450  000000                  HI:     .WORD   0
 4244  015452  000000                  LO:     .WORD   0
 4245                                  ;
 4246                                  ;
 4247                                  ;
 4248  015454                          LODAR:
 4249          000001                          .END
```

```
ABC     015354          4225*   4227
ADRESS  002253          1609    1784*
ALLFLG  004646          2300*   2376    2397    2417    2465    2485    2494*
ASTERI  012340          2273    3448*
BANKS   012102          2332    2505    3418*
BEGIN1  003276          1997    1999    2007*   2274
BELL    001521          1630    1646*
BIT12 = 010000          1300*   1826    2305
BIT13 = 020000          1299*
BIT14 = 040000          1298*   1364
BIT15 = 000100          1297*
BIT6  = 000100          1303*
BIT8  = 000400          1302*   3387
BIT9  = 001000          1301*   1598    3390
BLKCNT  004102          2127*   2135*   2162*   2169    2193
BPTVEC= 000014          1270*
BRANCH  012440          1413    3578*
C       000001          1250*
CKSWR   011650          2009    2340    2847    2992    3154    3213    3241    3313    3386*
CONST   012161          2425    2524    3427*
CONT    014514          4067*   4112
COUNT   000756          1509*   2848*   2892*   2896*   2906*   3047*   3081*   3158*   3184*
DIGITS  002275          1575    1605    1652    1788*
DIGTAB  002236          1684    1777*
DISPAR  007042          2706    2724*
DISPLA= 177570          1291*   1592*   1814*   2098*   2129*   2139*   2149*   2203*   3395*   3462*
DONE    004564          2271    2275*
DONE0   002510          1836    1846*
DONE1   002600          1856    1873*
DONE10  003564          2077    2083*
DONE11  003650          2105*
DONE12  003700          2121*
DONE13  004074          2157*   2178
DONE14  004146          2174    2176    2180*
DONE15  004176          2192*
DONE3   002702          1898    1903*
DONE4   002770          1909    1920*
DONE4A  003040          1923    1936*
DONE4B  003110          1939    1953*
DONE4C  003156          1964    1966    1971*
DONE4D  003252          1971    1995*
DONE5   003334          2010    2019*
DONE6   003370          2035*   2060
DONE7   003416          2037    2043*
DONE8   003500          2056    2058    2063*
DONE9   003526          2064    2071*
DONIT   013260          3708    3742*
EMTVEC= 000030          1273*
END     004526          2266*
ENDMSG  012342          2277    3449*
ERCNT   000746          1504*   1585    1587*   1591    1812*   3392
ERRMES  015171          3872    4180*
ERROR   001174          1359    1584*   2862    2870    2878    2886    3013    3020    3027    3034    3065    3072    3169
                        3176    3227    3325    3332    3339    3346
ERRPC   001467          1603    1641*
ERRTRP  001116          1356    1563*   2019    2043    2073    2242    2311    2513    2617    4028
```

```
ERRVEC= 000004          1266# 1736* 1828* 1836* 1856* 1909* 1923* 1939* 1971* 2010* 2019* 2037* 2043*
                        2064* 2073* 2130* 2132* 2208* 2242* 2303* 2311* 2513* 2579* 2612* 2617* 2652*
                        2707  2708* 2723* 4063* 4064*
FINISH  004624          2281  2289#
FIRST   015232          4192# 4215
FIRST1  015314          3616  3681  4216# 4242
FPEVEC= 000244          1278#
HI      015450          3628* 3721* 3726* 3729* 3731  3734* 3803* 4216  4243#
HILIM   013760          3608  3624  3626  3687* 3707  3715  3726  3764* 3857*  3864  3867  3875#
HIMESS  015104          3853  4169#
HISAV   013766          3687  3760  3795  3835  3867* 3879#
HLT   = 104400          1331# 1864  1918  1949  1982  1990  2229  2568  2592  2663  2681  3254
ICNT    000742          1502# 1819* 2000* 2269* 2270  2312* 2375* 2396* 2416* 2442* 2464* 2484*  3394
ICOUNT  000744          1503# 2852  2897* 2908  2974  2977  2999  3053  3089*  3115  3190*  3201  3356*
                        3396* 3399*
IDMESS  015206          3581  4183#
INCB1   015302          4208  4209  4211#
INERR   014172          3940  3942  3944  3963#
INRMES  015135          3965  4174#
INUM    013770          3369  3854  3860  3897#  3968
IOTVEC= 000020          1271#
KIPAR0= 172340          1320# 1457  1693  2736* 3664  4099
KIPAR1= 172342          1321# 1694  1881  1307* 2236* 2547* 2737* 2745* 2753*  3709
KIPAR2= 172344          1322# 1695  2214* 2237* 2734* 2777* 2791*
KIPAR3= 172346          1323# 1696  2779* 2792*
KIPAR4= 172350          1324# 1697
KIPAR5= 172352          1325# 1698
KIPAR6= 172354          1326# 1699
KIPAR7= 172356          1327# 1470  1700  2738* 3678*
KIPDR0= 172300          1311# 1452  2212* 2657* 2731*  3665
KIPDR1= 172302          1312# 2732*
KIPDR2= 172304          1313# 2213* 2733* 2780*
KIPDR3= 172306          1314# 2781*
KIPDR4= 172310          1315# 2782*
KIPDR5= 172312          1316#
KIPDR6= 172314          1317#
KIPDR7= 172316          1318# 1475  2735*
KM    = 000000          1257#
KPARM   015052          4102  4164#
KPDR  = 077406          3567# 3670
LAST    015310          4213# 4215  4217
LDCOMP  010152          2975  2979#
LDDISP  000750          1505# 1590* 1591* 1592  1813* 1814  2097* 2098  2128* 2129  2138* 2139  2148*
                        2149  2202* 2203  2313* 3386* 3392* 3394* 3395  3461* 3462
LDMMO   007044          1838  1859  1925  1941  1973  2211  2546  2656  2728# 2774
LDNORM  010164          2976  2978  2982#
LENGTH  015312          4215# 4224  4234
LIMERR  013750          3614  3865  3870#
LIMITS  013646          3603  3673  3851# 3873
LO      015452          3634* 3712* 3794* 4232  4244#
LOCBAD  015045          4086  4163#
LODAR   015454          1743  1757  4248#
LODFLG  012436          3575#
LODFLO  002042          1730# 1732  1748* 1752  1766*
LOGICA  004614          1362  1998  2284#
LOLIM   013762          3631  3688* 3707  3710  3729  3735* 3762* 3863* 3864  3868  3876#
```

DZQMBG.P11 CROSS REFERENCE TABLE -- USER SYMBOLS

```
LOMESS  015124          3859    4172#
LOSAV   013764          3688    3757    3789    3868*   3878#
LOSEG1  013272          3620    3689    3756#
LOSEG2  013342          3642    3742    3772#
LOSEG3  013544          3801    3806    3815#
LOSEG4  013624          3619    3676    3834#
LOSFL   013622          3756*   3759*   3772    3831#
LST     002246          1783#   1849
LSTLOC  002170          1747*   1763#
MARKER  015230          4191#
MMABT0  007142          1839    1908    1926    1974    2658    2744#
MMABT1  007174          1891    1958    2752#
MMABT2  007352          2784    2789#
MMAVA   000752          1448    1468    1506#   1657    1825*   1830*   1879    1905    2209    2234    2304*   2310*   2654
                        2729    2766    3579*   3662*   4131
MMMESS  015141          3586    4175#
MMVEC =  000250         1279#   1839*   1891*   1908*   1926*   1958*   1974*   2658*   2784*
MOSPAR  014254          3583    3989#   4130
MPFL    014416          3995*   4005    4014*   4023    4031#
MPMES   015000          4008    4152#
MPVECT  014734          3989    4141#
MP1     014306          4002#   4022
MP2     014352          3993    4019#
MP25    014354          4016    4020#
MP3     014400          4025    4028#
N       =  000010       1253#
NOFIND  006717          2680    2695#
NOKT11  015071          4108    4167#
NOMPAR  015027          4027    4157#
NOTIND  014760          4120    4149#
ONER    014210          3957    3969#   3974    3975    3976
O2A     001746          1617    1622    1705#   2558    2627    4011    4051    4084    4090    4103
PARAVA  007014          1480    2055    2173    2714#   2720*
PARCSR=  172100         2702#   2711
PAREGS=  172100         3988#   3990
PARERR  006676          2647    2692#   4048
PARER2  014420          4046#   4141
PARITY  012016          2316    3409#
PARPAT  003366          2028#   2057    2059*   2156*   2175    2177*   2314*   2490*   2925    2955    3004    3058
PARR3   014520          4055    4080#
PARR4   014666          4060    4118#
PARTAB  001726          1667    1693#
PARVEC=  000114         2651*   2703#   2709*   2710*   4058*   4059*
PAT     012137          2336    3423#
PC      =%000007        1241#   1382*   1389*   1393*   1394*   1397*   1400*   1403*   1409*   1412*   1425*   1437*   1447*
                        1460*   1482*   1484*   1487*   1518*   1545*   1553*   1559*   1566    1571*   1599*   1601*   1607*
                        1617*   1622*   1626*   1651*   1688*   1691*   1706*   1725*   1727*   1734*   1749*   1754*   1762
                        1767*   1809*   1817*   1831*   1837    1838*   1847*   1854    1859*   1874    1904    1924    1925*
                        1940    1941*   1957    1972    1973*   2008*   2009*   2013    2016*   2021    2025*   2027*   2036
                        2040*   2046*   2049    2052*   2065    2068*   2072    2079*   2095*   2105    2110*   2112    2117*
                        2121    2137    2146*   2157    2161    2164*   2167    2170*   2180    2183*   2186    2189*   2196
                        2207    2211*   2248*   2249    2252*   2259*   2260    2263*   2267    2282*   2284*   2299*   2317*
                        2321*   2325*   2329*   2333*   2337*   2340*   2346*   2368*   2370*   2374*   2388*   2390*   2395*
                        2412*   2415*   2426*   2436*   2438*   2441*   2457*   2460*   2463*   2477*   2480*   2483*   2500*
                        2515*   2518*   2525*   2535*   2536*   2537*   2546*   2558*   2627*   2630    2633*   2648*   2656*
                        2670*   2713*   2724*   2741*   2772*   2774*   2786*   2805*   2839*   2847*   2850*   2862*   2870*
```

```
                    2878*  2886*  2895*  2912*  2921*  2958*  2963*  2981*  2984*  2992*  2997*  3007*  3013*
                    3020*  3027*  3034*  3049*  3060*  3065*  3072*  3085*  3120*  3130*  3145*  3154*  3157*
                    3169*  3176*  3188*  3205*  3213*  3216*  3221*  3227*  3234*  3241*  3244*  3249*  3262*
                    3283*  3292*  3305*  3313*  3317*  3325*  3332*  3339*  3346*  3360*  3368*  3369*  3372*
                    3377*  3389*  3400*  3454   3582*  3583*  3592*  3600*  3603*  3611   3617   3619*  3620*
                    3622*  3627*  3633*  3637   3642*  3650*  3673*  3676*  3682   3689*  3691*  3732*  3742*
                    3763*  3774*  3785*  3790*  3802*  3807   3826*  3828*  3838*  3854*  3855*  3856*  3860*
                    3861*  3862*  3869*  3903*  3921*  3924*  3931*  3945*  3955*  3957*  3959*  3972*  3974*
                    3975*  3976*  3977*  3981*  3989*  4011*  4030*  4051*  4056   4061   4084*  4090*  4094*
                    4095*  4096*  4097*  4103*  4127*  4128*  4130*  4142   4146*  4194   4218   4221
PDWN     000502     1446*  1486   1821   2302
PEFLG    006520     1612   1816*  2298*  2512*  2649#  2661*  2667*  2674
PENFLG   006521     1610   2650#  2678*
PERMES   015023     4050   4156#
PERRFL   013616     3916   3829#  3834*  3837*
PERSTR   002366     1818#  2339*  2527*  2671
PFIVE    000254     1408#
PFOUR    000230     1402#
PFSTK    000560     1461#  1467
PFVEC =  000024     1272#  1462*  1486*  1821*  1822*  2302*
PIRQ  =  177772     1284#
PIRVEC=  000240     1277#
PKM   =  000000     1260#
PLACE    002306     1792#  2223
PONE     000162     1392#
PRG2     004630     1401   2295#  2410
PRG2A    004634     2297#  2506
PRG2R    005042     2339#
PRG3     005706     1404   2499#
PRG3FL   005716     2295*  2362   2382   2401   2430   2451   2471   2501#
PRG3M    012201     2503   3430#
PRG4     005740     1410   2511#  2553
PRG4A    006030     2522   2527#  2563
PRG4M    012262     2517   3439#
PRTY4 =  000200     1256#
PRTY7 =  000340     1255#  1360
PSIX     000270     1411#
PSM   =  010000     1261#
PSW   =  177776     1282#  1517
PTHREE   000214     1399#
PTWO     000200     1396#
PUM   =  030000     1262#
PUP      000572     1462   1466#
PWRFAI   000720     1492   1496#
QUEST    012155     2409   3426#
RANTST   004256     2086   2094   2145   2207#
RECD     011566     2317   2325   2333   2337   2426   2518   2525   3367#
RECDAT   001506     1620   1644#  4088
REG   =  004000     1263#
RELFL    000120     1366#  3578*  3788*
RELO     015372     4212   4220   4230#
RELOC    006246     1384   2088   2122   2140   2197   2575*  2620   3457   3778   3818
RELOCF   000754     1508#  1516   1649   1650   1689   1690   1820*  2343   2345   2602   3463*
RELOCM   012301     2624   3442#
RELOCP   006350     2329   2536   2611#
RELOI    015420     4235#  4238
```

```
REL24K  012356          2626    3389    3454#
REPET1  012634          3621#   3643
REPET2  013042          3680#   3744    4133
REPET3  012732          3625    3636    3642#
RESLDR  011752          1811    3403#
RESVEC= 000010          1267#
RET     015444          4233    4241#
RETPR   014650          4104    4109#
ROTATE  011302          3221    3249    3265#
ROTO    004430          2245#
ROT1    004470          2256#
RTAD    015304          4212#   4223*
RW    = 000006          1335#   2213    2731    2732    2735    2780    2781
RO    =%000000          1234#   1452*   1455    1457*   1458    1470*   1473*   1475*   1476*   1535    1536*   1539    1542*
                        1616    1634    1668*   1671*   1677*   1686*   1707*   1710*   1717*   1719*   1721*   1735*   1737
                        1740*   1741    1744    1857*   1858*   1860*   1862    1911*   1912*   1913*   1914*   1916    1928*
                        1929*   1931    1943*   1944*   1947    1960*   1961*   1963    1980    1985*   1988    2044*   2045*
                        2046    2051    2224*   2226    2280*   2284    2341*   2342*   2343*   2344*   2345*   2346    2542*
                        2545*   2552*   2564    2566    2575*   2582    2588    2611*   2613*   2615    2618*   2619    2639
                        2640*   2641*   2642*   2643*   2644*   2645*   2659*   2662    2711*   2718*   2806*   2807    2810*
                        2813    2815    2818    2820    2823    2825    2828    2830    2851*   2854*   2857*   2859    2865*
                        2867    2873*   2875    2881*   2883    2891*   2922*   2923    2928*   2931    2932    2933    2934
                        2941    2942    2943    2944    2953*   2979*   2982*   2998*   3001*   3010    3017    3024    3031
                        3038*   3041*   3052*   3055*   3062    3066*   3069    3073*   3077*   3080*   3092*   3102    3103
                        3104    3105    3106    3107    3108    3109    3131*   3133*   3135*   3136    3155*   3161*   3166
                        3173    3183*   3200*   3217*   3224    3245*   3252    3293*   3295    3296    3297    3298    3314*
                        3322    3329    3336    3343    3454*   3455*   3456    3610*   3611*   3612*   3613    3626*   3628
                        3631*   3634    3647*   3664*   3668*   3709*   3713*   3731*   3733*   3734    3789*   3791    3793*
                        3794    3795*   3796    3798*   3799    3803    3990*   4002    4010    4015*   4020*   4055*   4056*
                        4057*   4058    4060*   4061*   4062*   4063    4141*   4142*   4143*   4144    4216*   4225*   4229
                        4230*   4235*   4236

ROT       012742        3627    3633    3646#   3732    3790    3802
R1      =%000001        1235#   1656*   1661*   1662*   1663*   1664*   1665*   1666*   1837*   1854*   1874*   1904*   1924*
                        1940*   1957*   1972*   2013*   2021*   2036*   2049*   2065*   2072*   2105*   2112*   2121*   2137*
                        2157*   2167*   2180*   2186*   2196*   2207*   2249*   2260*   2267*   2641    3646*   3648*   3665*
                        3670*   3711*   3717*   3782*   3783    3784    3785    3786    3787    3806*   3807*   3808*   3809
                        3822*   3823    3824    3825    3826    3827    3992*   4021*   4068*   4089    4217*   4218*   4219*
                        4225    4234*   4237*

R10     =%000000        1242#
R11     =%000001        1243#
R12     =%000002        1244#
R13     =%000003        1245#
R14     =%000004        1246#
R15     =%000005        1247#
R2      =%000002        1236#   1453*   1454    1456*   1471*   1472    1477*   1570*   1600*   1606*   1634*   1635*   1654*
                        1655    1667*   1669*   1673*   1675*   1679*   1709*   1711*   1712*   1713    1716*   1718*   1720*
                        1722*   1742*   1745*   1758*   1760*   1840*   1841    1842*   1855*   1857    1861    1888    1893*
                        1894    1897*   1899    1910*   1911    1915    1927*   1931*   1942*   1946    1959*   1963*   1965
                        1975*   1979    1987    2215*   2218*   2219*   2225    2228    2230    2239*   2541*   2548*   2549*
                        2551*   2557    2564*   2565    2569    2576*   2577    2582*   2583    2586    2588    2602*   2603*
                        2604*   2642    2653*   2659    2676*   2677*   2712*   2720    2721*   2744*   2752*   2764    2768*
                        2769*   2770*   2771*   2775*   2776*   2777    2778*   2779    2783*   2790*   2804*   2813*   2814*
                        2815*   2816*   2818*   2819*   2820*   2821*   2823*   2824*   2825*   2826*   2828*   2829*   2830*
                        2831*   2849*   2858    2866    2874    2882    2894*   2900*   2901*   2902*   2903*   2920*   2931*
                        2932*   2933*   2934*   2936*   2937*   2938*   2939*   2941*   2942*   2943*   2944*   2946*   2947*
                        2948*   2949*   2967    2969    2972    2996*   3009    3016    3023    3030    3048*   3061    3067*
```

```
                    3068    3074*   3084*   3092    3093    3094*   3095*   3096*   3097*   3098*   3099*   3100*   3101*
                    3102*   3103*   3104*   3105*   3106*   3107*   3108*   3109*   3129*   3136*   3156*   3165    3172
                    3187*   3194*   3195*   3215*   3222    3243*   3250    3265*   3266*   3267*   3268*   3269*   3270*
                    3271*   3272*   3273*   3274*   3275*   3276*   3277*   3278*   3279*   3280*   3281*   3282*   3291*
                    3295*   3296*   3297*   3298*   3316*   3321    3328    3335    3342    3370    3898*   3949*   3970*
                    4224*   4226*
R3      =%000003    1237#   1454*   1459*   1472*   1474*   1621    1653*   1670*   1674*   1676*   1680*   1685*   1708*
                    1723*   1743*   1744*   1747    1841*   1842    1843*   1861*   1862    1881*   1882*   1883*   1884*
                    1885*   1886*   1887*   1890*   1894*   1895*   1896*   1915*   1916    1946*   1947    1979*   1980
                    1987*   1988    2023*   2024    2225*   2226    2528*   2531    2539    2541    2544*   2547    2551
                    2565*   2566    2577*   2578*   2581    2583    2585*   2586    2590    2643    2662*   2803*   2935*
                    2858*   2859    2866*   2867    2874*   2875    2882*   2883    2923*   2924*   2936    2937    2938
                    2939    2946    2947    2948    2949    2954*   2980*   2983*   3009*   3010    3016*   3017    3023*
                    3024    3030*   3031    3061*   3062    3068*   3069    3086*   3112*   3134*   3137*   3165*   3166
                    3172*   3173    3222*   3224    3250*   3252    3294*   3299*   3321*   3322    3328*   3329    3335*
                    3336    3342*   3343    3371    3666*   3668    3669*   3857    3863    3899*   3971*   4065*   4068
                    4080*   4081    4082*   4083    4098*   4099*   4100    4111*   4194*   4195*   4196    4198*   4200*
                    4220*   4221*   4222*   4223
R4      =%000004    1238#   1652*   1757*   1759    1930*   1932*   1945*   1950*   1962*   1967*   1976    1983*   1986*
                    1991*   2075*   2076*   2078    2091    2143    2216*   2218    2219    2224    2529*   2543*   2580*
                    2600*   2644    2812*   2833*   2929*   2951*   2995*   3042*   3056*   3078*   3093*   3094    3095
                    3096    3097    3098    3099    3100    3101    3128*   3141*   3160*   3181*   3189*   3198*   3214*
                    3230*   3242*   3258*   3290*   3301*   3315*   3351*   3636*   3637*   3638*   3639    3667*   3671*
                    3938*   3939*   3941*   3949    4196*   4197*   4202*   4207*
R5      =%000005    1239#   1384*   1491*   1515    1520*   1572*   1574*   1602*   1604*   1608*   1614*   1619*   1624*
                    1629*   1655*   1656    1672*   1673    1681*   1683*   1714*   1755*   1759*   1762*   1764*   1810*
                    1848*   1850*   2122*   2140*   2197*   2217*   2220*   2223*   2231*   2272*   2276*   2315*   2323*
                    2331*   2335*   2408*   2424*   2502*   2504*   2516*   2523*   2559*   2575    2576    2585    2603
                    2604    2620*   2623*   2645    2646*   2679*   2807*   2808*   2811*   2814    2816    2819    2821
                    2824    2826    2829    2831    2855*   2889*   2899*   2904*   3002*   3039*   3057*   3075*   3091*
                    3110*   3132*   3139*   3162*   3179*   3193*   3196*   3219*   3228*   3247*   3256*   3319*   3349*
                    3457*   3580*   3585*   3710*   3713    3714*   3715    3721    3724*   3735    3778*   3818*   3852*
                    3858*   3871*   3900*   3943*   3953*   3964*   4007*   4012*   4026*   4047*   4049*   4052*   4085*
                    4087*   4101*   4107*   4109*   4119*
SAVPC   013620      3775*   3787*   3815    3830#
SAVPC2  000122      1367#   1383*   1387
SAVPER  014664      4083*   4111    4116#
SAVR0   006662      2639*   2640    2685#
SAVR1   006664      2686#
SAVR2   006666      2687#
SAVR3   006670      2688#
SAVR4   006672      2689#
SAVR5   006674      2690#
SCOPE = 104000      1332#
SLR   = 177774      1283#
SM    = 040000      1258#
SP      =%000006    1240#   1383    1387*   1392*   1396*   1399*   1402*   1408*   1411*   1418    1419*   1420*   1421*
                    1422*   1423*   1424*   1430    1431    1432    1433    1434    1435    1436    1446*   1450*   1451*
                    1455*   1458*   1460    1467*   1473    1476    1478    1479    1485    1517*   1535*   1536    1537*
                    1539*   1541    1542    1546    1548*   1551*   1558    1569    1570    1600    1606    1616*   1621*
                    1707    1726*   1739    1808*   1815*   1846*   1873*   1888*   1889*   1890    1936    1953    1995
                    2007*   2014*   2015*   2020*   2022*   2024*   2035*   2038*   2039*   2044    2048*   2050*   2051*
                    2059    2063*   2066*   2067*   2071*   2074*   2078*   2087*   2096*   2108*   2109*   2115*   2116*
                    2136*   2147*   2160*   2161*   2168*   2169*   2177    2181*   2182*   2187*   2188*   2201*   2241*
                    2246*   2247*   2250*   2251*   2257*   2258*   2261*   2262*   2268*   2297*   2359*   2360*   2361*
                    2364*   2365*   2366*   2367*   2368    2372*   2373*   2380*   2381*   2384*   2385*   2386*   2387*
```

```
                    2388    2393*   2394*   2403*   2404*   2405    2413*   2414*   2428*   2429*   2432*   2433*   2434*
                    2435*   2436    2439*   2440*   2449*   2450*   2453*   2454*   2455*   2456*   2457    2461*   2462*
                    2469*   2470*   2473*   2474*   2475*   2476*   2477    2481*   2482*   2490    2494    2499*   2511*
                    2557*   2599    2625*   2626*   2628    2629*   2630*   2632*   2661    2668*   2676    2678    2707*
                    2723    2746*   2755*   2760*   2793*   2803    2804    2837*   2838*   2848    2849    2861*   2869*
                    2877*   2885*   2894    2896    2910*   2911*   2920    2959*   2961*   2962*   2995    2996    3012*
                    3019*   3026*   3033*   3047    3048    3064*   3071*   3084    3086    3118*   3119*   3128    3129
                    3143*   3144*   3156    3158    3168*   3175*   3187    3189    3203*   3204*   3214    3215    3226*
                    3232*   3233*   3242    3243    3260*   3261*   3290    3291    3303*   3304*   3315    3316    3324*
                    3331*   3338*   3345*   3358*   3359*   3373    3376*   3460*   3590*   3591*   3593    3594    3597
                    3599*   3601    3613*   3615*   3616*   3617*   3618*   3621*   3623    3680*   3681*   3682*   3683*
                    3690*   3692    3743    3775    3783*   3784*   3815*   3823*   3824*   3827*   3897*   3902*   3904
                    3907*   3908*   3909    3911    3912    3920*   3922    3927    3930*   3932    3933    3938    3958
                    3966    3980    4010*   4019    4081*   4089*   4100*   4125*
SPOT      012560    3608#   3610
SPOT2     012610    3609    3615#   4134
SR0     = 177572    1306#   1450    1479*   1659    1829*   2308    2739*   2747*   2748*   2756*   2759*   2785*   2794*
                    3663*   3706*   3736*   4092
SR1     = 177574    1307#
SR2     = 177576    1308#   1451    1478*   2746    2755    2793
SR3     = 172516    1309#
START     002310    1398    1807#
START1    002352    1807    1815#   1817    2288
STBANK    012053    2324    3414#
STKPTR= 000500      1294#   1808    1815    1846    1873    2007    2020    2035    2048    2063    2071    2087    2136
                    2201    2241    2268    2297    2499    2511    2668    3460
STMM2     007240    2764#   2805    2850    2895    2921    2997    3049    3085    3130    3157    3188    3216    3244
                    3292    3317
STRTAD    015310    4214#   4228*   4229    4230    4231*   4232    4239    4240
SWR     = 177570    1290#   1563    1588    1593    1597    1627    1631    1826    2305    2521*   2533*   2555*   2705
                    3045    3087    3354    3387    3390    3397
T       = 000020    1254#
TBITVE= 000014      1268#
TEMP1     011642    3373*   3374*   3379#
TEMP2     011644    3370*   3375    3380#
TEMP3     011646    3371*   3374    3381#
THRR      014220    3855    3856    3861    3862    3945    3955    3974#   4094    4095    4096    4097
TKB     = 177562    1287#   2445    3590    3901    3907
TKS     = 177560    1286#   1446    1485*   2443    2554*   3588    3905
TKVEC   = 000060    1275#   2553*
TOP       012436    3565#
TPB     = 177566    1289#   3980*
TPS     = 177564    1288#   2278    3978
TPVEC   = 000064    1276#
TRAPVE= 000034      1274#   1358
TRAP4     001452    1573    1638#
TRTVEC= 000014      1269#
TYPIT     014236    3592    3600    3622    3691    3903    3921    3924    3931    3978#   3979
UBREAK= 177770      1285#
UM      = 140000    1259#
UP      = 000000    1336#   2213    2731    2732    2735    2780    2781
V       = 000002    1251#   2760
WDS.25    003434    2047#   2247    2251    2258    2262
WHERE     000124    1380#   1393    1397    1400    1403    1409    1412    4127
WRTTAB    005100    2344    2349#
WRTUP     002450    1836#
```

```
XMTDAT  001473          1615    1642#
YMMBGO  012756          3595    3662#
YMMBG1  013122          3693    3706#
Z     = 000004          1252#
$ADRSO  005776          2519#   2528    2561*
$ALL    005676          2356    2494#
$CONST  006026          2526#   2552
$CRLF   015020          1625    1851    2560    4013    4053    4110    4155#
$FILL   001011          1529#
$FORMO  001524          1571    1601    1607    1649#   1847
$LDR    002044          1732#   1809    3582
$LDRM   002210          1765    1771#
$LDR1   002206          1741*   1755    1769#
$NULL   001010          1528#   1548
$PRINT  000760          1491    1514#   1572    1574    1602    1604    1608    1614    1619    1624    1629    1714    1764
                        1810    1848    1850    2272    2276    2315    2323    2331    2335    2408    2424    2502    2504
                        2516    2523    2559    2623    2646    2679    3580    3585    3852    3858    3871    3964    4007
                        4012    4026    4047    4049    4052    4085    4087    4101    4107    4109    4119
$RESTR  000330          1430#   1484    1626    1688    1725    2537    3372
$RLDR   002134          1394    1752#   2282
$ROTO   005476          2353    2418    2449#   2467
$ROT1   005572          2354    2466    2469#   2486
$SAVR   000304          1418#   1447    1599    1651    1706    2535    3368
$TKFLG  001013          1532#
$TPB    001016          1534#   1558*
$TPFLG  001012          1530#
$TPS    001014          1533#   1556
$USER   005370          2352    2423#   2446
$1X8    005120          2349    2359#   2378    2487    2495
$3X9    005210          2350    2377    2380#   2491
$3X9P   005666          2355    2490#
$8X13   005276          2351    2398    2401#   2419
.     = 015454          1352#   1358#   1361#   1363#   1365#   1391#   1405#   1441#   1501#   1557    1564    1594    1632
                        1647#   1671    1738    1775#   1790#   1844    1948    1981    1989    2125    2194    2200    2279
                        2515    2598    2699#   2853    2860    2868    2976    2984    3000    3011    3018    3025    3032
                        3054    3063    3070    3167    3174    3191    3253    3323    3330    3337    3344    3398    3610
                        3616    3636    3681    3806    4055    4060    4141    4192    4204    4210    4217    4220
.BANKS  005024          2334#   2360    2381    2404    2414    2429    2450    2470
.CONST  005402          2245*   2256*   2427#   2459*   2479*   3293    3314
.EXTAD  006244          2520*   2529    2562*   2571#
.MAMF   006744          1482    1831    2008    2321    2670    2705#
.PARIT  004750          2318#   2319
.PARSR  006466          2639#   2709
.PAT    005040          2338#   2341
.ROTO   011120          2252    2463    3213#
.ROT1   011214          2263    2483    3241#
.STBAN  004776          2326#   2327    2359    2372    2380    2393    2403    2413    2428    2439    2449    2461    2469
                        2481
.TYPE   001020          1518    1535#
.USER   011350          2248    2259    2438    2460    2480    3290#
.XOR39  010110          2958    2967#   3007    3060
..USER  011422          2441    3313#
..1X8   007522          2025    2117    2374    2846#
..3X9   010176          2052    2170    2395    2991#
..8X13  010724          2079    2189    2415    3153#
.1X8    007410          2016    2110    2370    2803#
```

DZQMBG.P11 CROSS REFERENCE TABLE -- USER SYMBOLS

.1617	011640	2520	3375*	3378#	
.3ISO	010126	2968	2972#		
.3IS1	010116	2969#			
.3IS9	010144	2971	2973	2977#	
.3NOT9	010134	2970	2974#		
.3X9	007756	2040	2164	2390	2920#
.8X13	010646	2068	2183	2412	3128#

$TYPE 1200#

DZQMBG.P11 CROSS REFERENCE TABLE -- PERMANENT SYMBOLS

```
        3031  3062  3069  3166  3173  3224  3252  3322  3329  3336  3343  3594  3597  3608  3624
        3707  3715  3743  3757  3760  3791  3796  3799  3835  3864  3909  3927  4019  4232
CMPB    1546
COM     1830  1895  1912  2310  2600  2808  2810  2811  2854  2857  2865  2873  2881  2891  2900
        2901  2902  2903  2924  2953  2954  3001  3038  3041  3055  3066  3067  3073  3074  3077
        3080  3133  3135  3161  3183  3194  3195  3200
DEC     1681  1686  1723  1745  1748  1760  1932  1950  1961  1967  1991  2220  2231  2833  2835
        2889  2892  2904  2906  2951  2959  3039  3042  3075  3078  3081  3110  3112  3137  3139
        3141  3179  3181  3184  3196  3198  3226  3230  3256  3258  3299  3301  3349  3351  3759
        3837  4014  4226  4237
DECB    1551
EMT     1332
HALT    1395  1463  1565  1576  1595  1633  2594  2596  4205  4211
INC     1489  1929  1944  1983  1985  2236  2237  2269  2375  2396  2416  2442  2464  2484  3648
        3671  3717  3736  3943  3953  4021  4197
INCB    2212  2657  4202  4207
JMP     1398  1401  1404  1410  1413  1579  2086  2094  2125  2145  2200  2274  2288  2410  2487
        2491  2495  2506  2671  3117  3466  3614  3643  3744  3801  3968  4133  4134  4212  4229
        4240
JSR     1384  1393  1394  1397  1400  1403  1409  1412  1447  1482  1484  1491  1518  1545  1553
        1571  1572  1574  1599  1601  1602  1604  1607  1608  1614  1617  1619  1622  1624  1626
        1629  1651  1688  1706  1714  1725  1764  1809  1810  1831  1838  1847  1848  1850  1859
        1925  1941  1973  2008  2009  2016  2025  2040  2052  2068  2079  2088  2110  2117  2122
        2140  2164  2170  2183  2189  2197  2211  2248  2252  2259  2263  2272  2276  2282  2284
        2315  2317  2321  2323  2325  2329  2331  2333  2335  2337  2340  2370  2374  2390  2395
        2408  2412  2415  2424  2426  2438  2441  2460  2463  2480  2483  2502  2504  2516  2518
        2523  2525  2535  2536  2537  2546  2558  2559  2620  2623  2627  2646  2656  2670  2679
        2774  2805  2847  2850  2862  2870  2878  2886  2895  2921  2958  2992  2997  3007  3013
        3020  3027  3034  3049  3060  3065  3072  3085  3130  3154  3157  3169  3176  3188  3213
        3216  3221  3227  3241  3244  3249  3292  3313  3317  3325  3332  3339  3346  3368  3369
        3372  3389  3457  3580  3582  3583  3585  3592  3600  3603  3619  3620  3622  3627  3633
        3642  3673  3676  3689  3691  3732  3742  3778  3790  3802  3818  3852  3854  3855  3856
        3858  3860  3861  3862  3871  3903  3921  3924  3931  3945  3955  3957  3964  3974  3975
        3976  3989  4007  4011  4012  4026  4047  4049  4051  4052  4084  4085  4087  4090  4094
        4095  4096  4097  4101  4103  4107  4109  4119  4127  4130
MOV     1383  1387  1392  1396  1399  1402  1408  1411  1418  1419  1420  1421  1422  1423  1424
        1425  1430  1431  1432  1433  1434  1435  1436  1437  1446  1450  1451  1452  1453  1454
        1455  1457  1458  1460  1462  1467  1470  1471  1472  1473  1475  1476  1478  1479  1485
        1486  1515  1517  1535  1536  1542  1548  1569  1570  1592  1600  1606  1616  1621  1634
        1652  1655  1656  1667  1668  1677  1683  1707  1708  1713  1726  1735  1736  1741  1742
        1743  1744  1747  1755  1757  1758  1759  1762  1807  1808  1814  1815  1817  1821  1829
        1836  1837  1839  1840  1841  1842  1846  1854  1855  1856  1857  1861  1873  1874  1881
        1888  1891  1894  1897  1904  1907  1908  1909  1910  1911  1915  1923  1924  1926  1927
        1930  1931  1939  1940  1942  1945  1946  1957  1958  1962  1963  1971  1972  1974  1975
        1979  1986  1987  2000  2007  2010  2013  2014  2019  2020  2021  2022  2024  2035  2036
        2037  2038  2043  2044  2046  2048  2049  2050  2051  2059  2063  2064  2065  2066  2071
        2072  2073  2074  2078  2087  2098  2105  2109  2112  2116  2121  2126  2127  2129  2130
        2132  2136  2137  2139  2149  2157  2161  2167  2169  2177  2180  2182  2186  2188  2192
        2196  2201  2203  2207  2208  2213  2214  2215  2217  2218  2219  2223  2224  2225  2241
        2242  2245  2246  2247  2249  2250  2251  2257  2258  2260  2261  2262  2267  2268  2280
        2297  2302  2303  2311  2339  2341  2344  2346  2359  2360  2368  2372  2373  2380  2381
        2388  2393  2394  2403  2404  2413  2414  2428  2429  2436  2439  2440  2449  2450  2457
        2459  2461  2462  2469  2470  2477  2481  2482  2490  2494  2499  2500  2511  2513  2515
        2520  2527  2528  2529  2541  2542  2547  2551  2552  2553  2557  2564  2565  2575  2576
        2577  2579  2582  2585  2602  2604  2611  2612  2617  2619  2625  2628  2630  2639  2640
        2641  2642  2643  2644  2645  2651  2652  2658  2659  2662  2668  2676  2707  2708  2709
```

```
            2710  2711  2712  2718  2723  2731  2732  2734  2735  2737  2738  2739  2744  2746  2748
            2752  2755  2756  2777  2779  2780  2781  2783  2784  2785  2790  2793  2794  2803  2804
            2806  2807  2812  2813  2814  2815  2816  2818  2819  2820  2821  2823  2824  2825  2826
            2828  2829  2830  2831  2837  2838  2848  2849  2855  2858  2866  2874  2882  2894  2896
            2899  2910  2911  2920  2923  2928  2929  2931  2932  2933  2934  2936  2937  2938  2939
            2941  2942  2943  2944  2946  2947  2948  2949  2961  2962  2979  2980  2982  2983  2995
            2996  3002  3009  3016  3023  3030  3047  3048  3056  3057  3061  3068  3084  3086  3091
            3092  3093  3118  3119  3128  3129  3132  3134  3136  3143  3144  3155  3156  3158  3160
            3162  3165  3172  3187  3189  3193  3203  3204  3214  3215  3217  3219  3222  3232  3233
            3242  3243  3247  3250  3260  3261  3290  3291  3293  3294  3295  3296  3297  3298  3303
            3304  3314  3315  3316  3319  3321  3328  3335  3342  3358  3359  3370  3371  3373  3374
            3375  3395  3396  3454  3456  3460  3462  3599  3610  3613  3616  3621  3626  3628  3631
            3634  3636  3639  3646  3662  3664  3665  3667  3668  3670  3678  3681  3687  3688  3690
            3693  3709  3710  3711  3712  3713  3721  3726  3731  3734  3735  3762  3764  3775  3781
            3782  3788  3789  3793  3794  3795  3798  3803  3806  3809  3815  3822  3857  3863  3867
            3868  3900  3902  3907  3920  3930  3938  3949  3990  3992  3993  4010  4015  4028  4029
            4055  4058  4060  4063  4068  4081  4083  4089  4100  4111  4125  4128  4141  4144  4194
            4196  4216  4217  4220  4223  4224  4225  4228  4230  4234  4235  4239
MOVB        1539  1558  1684  2661  2678  3094  3095  3096  3097  3098  3099  3100  3101  3102  3103
            3104  3105  3106  3107  3108  3109  3590  3980
NEG         2023  2045  2075
NOP         1466  1493  1514  1584  1903  1920  2285  22[illegible]  2287  2601  2666  2728  2740  2789  2846
            2991  3153
RESET       2083  2266  2283  2514  2669  3464  4126
ROL         1670  1676  1680  1710  1711  1717  1718  1719  1720  1721  1722
ROLB        3265  3266  3267  3268  3269  3270  3271  3272  3273  3274  3275  3276  3277  3278  3279
            3280  3281  3282
ROR         1663  1664  1665  1666  2521  2533  2544  2555  2768  2769  2770  2771  2776  3970  3971
RTI         1357  1494  1543  1636  2665  2749  2757  2761  2795  3641  3737  3811
RTS         1382  1389  1520  1559  1691  1727  1734  1749  1754  1767  2606  2633  2724  2741  2772
            2786  2839  2912  2963  2981  2984  3120  3145  3205  3234  3262  3283  3305  3360  3377
            3400  3650  3763  3774  3828  3838  3869  3959  3972  3977  3981  4030  4146
SEC         2307  2614  3248  4199
SEV         2717
SOB         1456  1459  1474  1477  1671  2545
SUB         1654  1740  1858  1893  1914  1959  2618  2677  2753  3724  3729  3823  3824  3825  3826
            3827  4195  4231
SWAB        1662  2775
TRAP        1331  4241
TST         1380  1448  1468  1480  1541  1563  1566  1593  1631  1657  1732  1737  1752  1879  1905
            1996  1998  2055  2057  2084  2091  2131  2133  2143  2173  2175  2209  2228  2230  2234
            2308  2319  2327  2362  2376  2382  2397  2401  2417  2430  2445  2451  2465  2471  2485
            2569  2581  2590  2615  2654  2729  2764  2766  2852  2908  2925  2955  2974  2977  2999
            3004  3053  3058  3115  3201  3593  3601  3623  3692  3772  3776  3816  3901  3904  3911
            3912  3922  3932  3933  3958  3966  4002  4005  4023  4131  4236
TSTB        1556  1610  1612  2278  2443  2674  3588  3905  3978
.ABS        1199
.ASCII      1638  1783  1784  4152  4172  4175
.ASCIZ      1496  1641  1642  1644  1646  1771  1788  2692  2695  3403  3409  3414  3418  3423  3426
            3427  3430  3439  3442  3448  3449  4149  4155  4156  4157  4163  4164  4167  4169  4174
            4179  4180  4183
.BYTE       1528  1529  1530  1532  2649  2650
.END        4249
.EVEN       1501  1647  1775  1790  2699  3451  4186
.LIST       1198
.MCALL      1200
```

```
TEST DZQMB-G 0-124K MEMORY EXERCISER    MACY11 27(732)  10-SEP-76  11:59  PAGE 111
DZQMBG.P11      CROSS REFERENCE TABLE -- PERMANENT SYMBOLS

.NLIST   1197
.REM        1
.SBTTL   1202    1440    1500    1581    1793    2003    2205    2290    2291    2496    2508    2572    2573    2635    2726
         2797    2914    3123    3207    3307    3362    3468
.TITLE   1201
.WORD    1353    1354    1356    1357    1359    1360    1362    1364    1366    1367    1385    1396    1406    1407    1461
         1488    1502    1503    1508    1509    1519    1533    1534    1567    1728    1730    1763    1769    1792    1818
         2028    2047    2162    2300    2318    2326    2334    2338    2349    2350    2351    2352    2353    2354    2355
         2356    2369    2389    2427    2437    2458    2478    2501    2519    2526    2571    2622    2631    2685    2686
         2687    2688    2689    2690    2714    3378    3379    3380    3381    3575    3581    3586    3779    3780    3819
         3820    3829    3830    3831    3853    3859    3872    3875    3876    3878    3879    3965    4008    4013    4027
         4031    4048    4050    4053    4086    4088    4102    4108    4110    4116    4120    4191    4193    4214    4215
         4243    4244


ERRORS DETECTED:  0
DEFAULT GLOBALS GENERATED:  0

*DZQMBG,DZQMBG.SEQ/SOL/CRF/DS:ERFZ/EN:ABS=DSKM:DZQMBG.P11
RUN-TIME: 20 25 5 SECONDS
RUN-TIME RATIO: 83/51=1.6
CORE USED:  10K  (19 PAGES)
```

E09

Spooler runtime 14 Seconds, 425 KCS, 398 disk reads, 3 disk writes, 107 pages

[redacted]