

PDP11

07124K MEMORY EXERCISER
MD-11-DDQAB-A

EP DDQAB A DL A

OCT 1976

COPYRIGHT © 1976

digital

FICHE 1 OF 1

Made in U.S.A.

The image displays a grid of 100 small tables, arranged in 10 rows and 10 columns. Each table represents a memory exercise. The tables contain various data, including headers, column labels, and rows of numerical or alphanumeric values. The data is presented in a structured, tabular format, typical of a memory exerciser program. The tables are densely packed and cover most of the left and center portions of the page.

A small table located in the bottom right corner of the page. It contains a few rows of data, possibly a summary or a specific set of results. The table is small and less detailed than the others in the grid.

TABLE OF CONTENTS

ABSTRACT

CHAPTER 1 REQUIREMENTS

1.1 EQUIPMENT

1.2 STORAGE

1.3 PRELIMINARY PROGRAMS

CHAPTER 2 LOADING AND STARTING PROCEDURE

2.1 ACT11 OPERATION

CHAPTER 3 SWITCH SETTINGS

CHAPTER 4 SUBROUTINE ABSTRACTS

4.1 SCOPE

CHAPTER 5 ERRORS

5.1 PARITY ERROR

CHAPTER 6 RESTRICTIONS

6.1 STARTING RESTRICTION

6.2 OPERATIONAL RESTRICTION

CHAPTER 7 MISCELLANEOUS

7.1 STACK POINTER

7.2 PASS COUNT

7.3 ERROR COUNT

7.4 DISPLAY REGISTER

7.5 PROGRAM RELOCATION

7.6 POWER FAIL

7.7 EXECUTION TIME

TABLE OF CONTENTS (CONT'D)

CHAPTER 8 PROGRAM DESCRIPTION

- 8.1 PROGRAM 2 (USER SELECTIONS)
 - 8.1.1 PROGRAM 2 STARTING PROCEDURE
 - 8.1.2 PROGRAM 2 USER PARAMETERS
 - 8.1.3 PROGRAM 2 RESTARTING PROCEDURE
 - 8.1.4 PROGRAM 2 USE
- 8.2 PROGRAM 3
- 8.3 PROGRAM 4
- 8.4 PROGRAM 5
- 8.5 PROGRAM 6

CHAPTER 9 BRANCH GOBBLE MOS TEST

- 9.1 ABSTRACT
- 9.2 OPERATING PROCEDURE
- 9.3 ERRORS
- 9.4 PROGRAM DESCRIPTION

E01

TEST DDQAB-A 0-124K MEMORY EXERCISER MACY11 27(732) 10-SEP-76 10:35 PAGE 4
DDQABA.P11

PDP-11 0-124K MEMORY EXERCISER
ABSTRACT

PAGE 5

ABSTRACT

PROGRAM DDQAB 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.

THE PROGRAM DDQAB HAS BEEN CREATED BY MODIFYING THE

F01

TEST DQAB-A 0-124K MEMORY EXERCISER MACY11 27(732) 10-SEP-76 10:35 PAGE 5
DQABA.P11

DZOMB PROGRAM. THE DZOMB PROGRAM IS A 0 - 124K
MEMORY EXERCISER. BUT THE SEDM SYSTEM HAS MUCH LESS
MEMORY AND NO MOS MEMORY AND NO MEMORY MANAGEMENT OPTION.
SO, THERE ARE SOME HARMLESS EXTRA CODES (E.G. BRANCH GOBBLE
MOS MEMORY TEST) IN DQAB PROGRAM WHICH COULD HAVE BEEN
DELETED TO CREATE AN EXCLUSIVE SEDM MEMORY EXERCISER.
PLEASE IGNORE SUCH EXTRA CODES WHILE RUNNING
A SEDM SYSTEM.

GO1

TEST DDQAB-A 0-124K MEMORY EXERCISER MACY11 27(732) 10-SEP-76 10:35 PAGE 6
DDQABA.P11

PDP-11 0-124K MEMORY EXERCISER
REQUIREMENTS

PAGE 6

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).
SET SW15=1 OR UP---IF NO TTY WITH THE SYSTEM
PRESS START.

THERE WILL BE A 5 ON THE DISPLAY LIGHTS FOR A
FEW SECONDS AFTER EACH PASS. THEN PROGRAM WILL HALT
AT LOCATION 252, IF SW6 WAS UP.
IF PROGRAM HALTS AT 252, PRESS CONTINUE.

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 AT LOC. 1430
- 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
- SW10=1 OR UP RING BELL ON ERROR

PDP-11 0-124K MEMORY EXERCISER
SWITCH SETTINGS

PAGE 9

| | |
|----------------------------|------------------------------------------------------------------------------------------------------------|
| 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 |
| SW6=1 OR UP | HALT ON END OF PASS |
| SW5=1 OR UP SW1=1 OR UP | INHIBIT PARITY ERROR DETECTION (INITIAL THERE IS NO TTY WITH THE SYSTEM; SO PATCH TAGS 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 'PASS (UNTIL AN * IS TYPED) WITH PARITY DISABLED, THEN RESTART THE PROGRAM WITH PARITY ENABLED.

L01

TEST DDQAB-A 0-124K MEMORY EXERCISER
DDQABA.P11

MACY11 27(732) 10-SEP-76 10:35 PAGE 11

PDP-11 0-124K MEMORY EXERCISER
SUBROUTINE ABSTRACTS

PAGE 10

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.

IF SW15 WAS UP, PROGRAM WILL HALT AT LOC. 1430 ON ERROR.

NOTE

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

THE FAILING ADDRESS (PC + 2) IS IN R1

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.

NO1

TEST DDQAB-A 0-124K MEMORY EXERCISER MACY11 27(732) 10-SEP-76 10:35 PAGE 13
DDQABA.P11

6.1 PARITY ERROR

IF THE MEMORY PARITY OPTIONS ARE INSTALLED THE PROGRAM RUNS WITH THE

PDP-11 0-124K MEMORY EXERCISER
ERRORS

PAGE 12

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

WHEN PROGRAM IS RELOCATED, DONOT RESTART AT ADDRESS 200
OR 214.

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 MCS 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 1000.) 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.

E02

TEST DDQAB-A 0-124K MEMORY EXERCISEP
DDQABA.P11

MACY11 27(732) 10-SEP-76 10:35 PAGE 17

7.3 ERROR COUNT

PDP-11 0-124K MEMORY EXERCISER
MISCELLANEOUS

PAGE 15

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

GO2

TEST DDQAB-A 0-124K MEMORY EXERCISER MACY11 27(732) 10-SEP-76 10:35 PAGE 19
DDQABA.P11

EXECUTION TIME IS DEPENDENT ON TYPE OF PROCESSER, TYPE OF MEMORY, AND

H02

TEST 000AB-A 0-124K MEMORY EXERCISER MACY11 27(732) 10-SEP-76 10:35 PAGE 20
000ABA.P11

PDP-11 0-124K MEMORY EXERCISER
MISCELLANEOUS

PAGE 16

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 (ROTD) 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 'DQAMB DONE' AND RESTART THE PROGRAM BEGINNING WITH THE MEMORY ADDRESS TESTS.

J02

8.1 PROGRAM 2 (USER SELECTIONS)

PDP-11 0-124K MEMORY EXERCISER
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 STARTING PROCEDURE:

1. LOAD ADDRESS 214
2. SET SWD1=1 (UP)---ONLY IF THERE IS NO TTY

3. PRESS START

IF SWD1 WAS UP I.E., IF THERE IS NO TTY, PROGRAM
WILL HALT AT LOC. 5012.
IF SO, PATCH THE FOLLOWING TAGS: (REFER TO 8.1.2)

.PARIT --- WITH 0 (DISABLE PARITY) OR 1 (ENABLE PAR.)
 .STBANK --- WITH 0 OR 1 OR 2 ETC (STARTING BANK #)
 .BANKS ---- WITH 1 IF THERE IS BK OR LESS OF MEMORY
 .PAT --- WITH PATTERN #

4. PRESS CONTINUE (IF PROGRAM HAD HALTED BECAUSE
OF SWD1 BEING UP)

8.1.2 PROGRAM 2 USER PARAMETERS

1. ENABLE PARITY? 1/0 = YES/NO. TYPE (OR PATCH .PARIT) 1 TO ENABLE INTERRUPT ON
PARITY ERROR. TYPE (OR PATCH) 0 TO DISABLE INTERRUPT.
2. STARTING BANK #(8)? TYPE (OR PATCH .STBANK) THE 4K BANK WHERE YOU WISH TO BEGIN
TESTING.

| TYPE (OR PATCH) | 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 (OR PATCH .BANKS, IN OCTAL THE NUMBER OF 4K BANKS TO TEST.
- 4. PATTERN #?
TYPE TO SELECT
(OR PATCH)

PDP-11 0-124K MEMORY EXERCISER
PROGRAM DESCRIPTION

PAGE 19

| | |
|---|------------------------|
| 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.

8.1.3 RESTARTING PROCEDURE:
IF YOU HAD SELECTED BANK 0 TO START, RESTART AT 32460 THEN AT 214. OTHERWISE, YOU WILL HAVE TO START BY RELOADING ABS. LOADER + PROGRAM.

8.1.4 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.

THE 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.

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

PDP-11 0-124K MEMORY EXERCISER
PROGRAM DESCRIPTION

PAGE 20

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 | 15: CMP R0,2#SWR | ;CHECK UPPER LIMIT |
| 20 | 177570 | | ; (IN SWITCH REGISTER) |
| 22 | 001403 | BEQ 25 | ;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 15 | ;LOOP UNTIL DONE |
| 32 | 010004 | 25: MOV R0,R4 | ;SAVE UPPER LIMIT |
| 34 | 020001 | 35: CMP R0,R1 | ;CHECK IF AT LOWER LIMIT |
| * 36 | 001767 | BEQ 15 | ;BRANCH IF DONE |
| 40 | 024000 | CMP -(R0),R0 | ;CHECK DATA WRITTEN |
| 42 | 001774 | BEQ 35 | ;BRANCH IF OK |
| 44 | 000000 | HALT | ;ERROR |
| 46 | 000772 | BR 35 | ;LOOP BACK |

AFTER TOGGING 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

C03

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

POP-11 0-124K MEMORY EXERCISER
PROGRAM DESCRIPTION

PAGE 21

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 | BEG 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 | BEG 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 | BEG 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 BK, 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.

F03

TEST DDQAB-A 0-124K MEMORY EXERCISER MACY11 27(732) 10-SEP-76 10:35 PAGE 31
DDQABA.P11

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

PDP-11 0-124K MEMORY EXERCISER
BRANCH GOBBLE MOS TEST

PAGE 23

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 (16-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 BK 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

H03

TEST 000AB-A 0-124K MEMORY EXERCISER
000ABA.P11

MACY11 27(732) 10-SEP-76 10:35 PAGE 33

NOT VALID:

PDP-11 0-124K MEMORY EXERCISER
BRANCH GOBBLE MOS TEST

PAGE 24

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 THE TIME 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

J03

TEST DCGAB-A 0-124K MEMORY EXERCISER MACY11 27(732) 10-SEP-76 10:35 PAGE 35
DCGABA.P11

THE ADDRESS GIVEN FOR THE ERROR IS REAL AND:

POP-11 0-124K MEMORY EXERCISER
BRANCH GOBBLE MOS TEST

PAGE 25

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 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 AND EXAMINE THE CONTENTS OF THE MEMORY LOCATIONS STARTING AT THE

L03

TEST DDQAB-A 0-124K MEMORY EXERCISER MACY11 27(732) 10-SEP-76 10:35 PAGE 37
DDQABA.P11

ADDRESS IN THE DISPLAY REGISTER. THE CONTENTS OF THESE
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.

1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228

```
.NLIST MD,MC
.LIST ME
.ABS
.MCALL $TYPE
.TITLE TEST DDQAB-A 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 FAIL'D WHEN RUNNING. THE PROGRAM WILL PRINT
; A MESSAGE (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
```

1229
 1230
 1231
 1232
 1233
 1234
 1235
 1236
 1237
 1238
 1239
 1240
 1241
 1242
 1243
 1244
 1245
 1246
 1247
 1248
 1249
 1250
 1251
 1252
 1253
 1254
 1255
 1256
 1257
 1258
 1259
 1260
 1261
 1262
 1263
 1264
 1265
 1266
 1267
 1268
 1269
 1270
 1271
 1272
 1273
 1274
 1275
 1276
 1277
 1278
 1279
 1280
 1281
 1282
 1283
 1284

000000
 000001
 000002
 000003
 000004
 000005
 000006
 000007
 000000
 000001
 000002
 000003
 000004
 000005
 000001
 000002
 000004
 000010
 000020
 000340
 000200
 000000
 040000
 140000
 000000
 010000
 030000
 004000
 000004
 000010
 000014
 000014
 000014
 000014
 000020
 000024
 000030
 000034
 000060
 000064
 000240
 000244
 000250
 177776

; OF MEMORY, TO RESTORE THE LOADERS RESTART AT 162.
 ; STACK POINTER IS SET AT 500
 ; AN ASTERISK '*' WILL BE PRINTED ON COMPLETION OF EACH PASS, AND
 ; THE PROGRAM NAME WILL BE PRINTED WHEN TEST IS COMPLETE.

;GENERAL REGISTER ASSIGNMENTS

RD=%0
 R1=%1
 R2=%2
 R3=%3
 R4=%4
 R5=%5
 SP=%6
 PC=%7
 R10=%0
 R11=%1
 R12=%2
 R13=%3
 R14=%4
 R15=%5

;STATUS REGISTER (PSW) BIT ASSIGNMENTS

C=1 ; C BIT
 V=2 ; V BIT
 Z=4 ; Z BIT
 N=10 ; N BIT
 T=20 ; 'T' BIT
 PRTY7=340 ; PRIORITY LEVEL 7
 PRTY4=200 ; PRIORITY LEVEL 4
 KM=000000 ; KERNEL MODE
 SM=040000 ; SUPERVISORY MODE
 UM=140000 ; USER MODE
 PKM=000000 ; PREVIOUS KERNEL MODE
 PSM=010000 ; PREVIOUS SUPERVISORY MODE
 PUM=030000 ; PREVIOUS USER MODE
 REG=004000 ; SELECT R10-R15

;VECTOR ADDRESSES

ERRVEC=4 ; ADDRESS OF ERROR VECTOR
 RESVEC=10 ; ADDRESS OF RESERVED INST. TRAP VECTOR
 TBITVEC=14 ; ADDRESS OF 'T' BIT TRAP VECTOR
 TRTVEC=14 ; ADDRESS OF 'TRACE' TRAP VECTOR
 BPTVEC=14 ; ADDRESS OF 'BREAKPOINT' TRAP VECTOR
 IOTVEC=20 ; ADDRESS OF IOT TRAP VECTOR
 PFVEC=24 ; ADDRESS OF POWER FAIL TRAP VECTOR
 EMTVEC=30 ; ADDRESS OF EMT VECTOR
 TRAPVEC=34 ; ADDRESS OF TRAP VECTOR
 TKVEC=60 ; ADDRESS OF TTY KEYBOARD INTERRUPT VECTOR
 TPVEC=64 ; ADDRESS OF TTY PRINTER INTERRUPT VECTOR
 PIRVEC=240 ; ADDRESS OF PIRQ VECTOR
 FPEVEC=244 ; ADDRESS OF FLOATING POINT INT. VECTOR
 MMVEC=250 ; ADDRESS OF MEM MGMT ERROR TRAP VECTOR

;REGISTER ADDRESSES

PSW=177776

;ADDRESS OF STATUS REGISTER

| | | | |
|------|--------|--------------------------------------------------|----------------------------------------|
| 1295 | 177774 | SLR=177774 | : ADDRESS OF STACK LIMIT REGISTER |
| 1296 | 177772 | PIRQ=177772 | : ADDRESS OF PROGRAM INTERRUPT REQUEST |
| 1297 | 177770 | UBREAK=177770 | : ADDRESS OF MICRO BREAK REGISTER |
| 1298 | 177560 | TKS=177560 | : ADDRESS OF KEYBOARD CSR |
| 1299 | 177562 | TKB=177562 | : ADDRESS OF KEYBOARD BUFFER |
| 1290 | 177564 | TPS=177564 | : ADDRESS OF TELEPRINTER CSR |
| 1291 | 177566 | TPB=177566 | : ADDRESS OF TELEPRINTER BUFFER |
| 1292 | 177570 | SWR=177570 | : ADDRESS OF CONSOL SWITCH REGISTER |
| 1293 | 177570 | DISPLAY=177570 | : ADDRESS OF CONSOL DISPLAY REGISTER |
| 1294 | | | |
| 1295 | | | |
| 1296 | 000500 | : INITIAL STACK POINTER SETTING | |
| 1297 | | STKPTR=500 | |
| 1298 | | | |
| 1299 | | : MISCELLANEOUS BIT ASSIGNMENTS | |
| 1300 | 000100 | BIT15= 100 | |
| 1301 | 040000 | BIT14= 040000 | |
| 1302 | 020000 | BIT13= 020000 | |
| 1303 | 010000 | BIT12= 010000 | |
| 1304 | 001000 | BIT9= 001000 | |
| 1305 | 000400 | BIT8= 000400 | |
| 1306 | 000100 | BIT6= 000100 | |
| 1307 | 000001 | SW00=000001 | |
| 1308 | 000002 | SW01=000002 | |
| 1309 | 000100 | SW06=000100 | |
| 1310 | | : MEMORY MANAGEMENT REGISTER ADDRESS ASSIGNMENTS | |
| 1311 | 177572 | SR0=177572 | : ADDRESS OF MEM MGMT REGISTER SR0 |
| 1312 | 177574 | SR1=177574 | : " " " " " " " " SR1 |
| 1313 | 177576 | SR2=177576 | : " " " " " " " " SR2 |
| 1314 | 172516 | SR3=172516 | : ADDRESS OF MEM MGMT REGISTER SR3 |
| 1315 | | | |
| 1316 | 172300 | KIPDR0=172300 | : ADDRESS OF KERNEL 'I' PAGE |
| 1317 | 172302 | KIPDR1=172302 | : DESCRIPTOR REGISTERS |
| 1318 | 172304 | KIPDR2=172304 | |
| 1319 | 172306 | KIPDR3=172306 | |
| 1320 | 172310 | KIPDR4=172310 | |
| 1321 | 172312 | KIPDR5=172312 | |
| 1322 | 172314 | KIPDR6=172314 | |
| 1323 | 172316 | KIPDR7=172316 | |
| 1324 | | | |
| 1325 | 172340 | KIPAR0=172340 | : ADDRESSES OF KERNEL 'I' SPACE |
| 1326 | 172342 | KIPAR1=172342 | : PAGE ADDRESS REGISTERS |
| 1327 | 172344 | KIPAR2=172344 | |
| 1328 | 172346 | KIPAR3=172346 | |
| 1329 | 172350 | KIPAR4=172350 | |
| 1330 | 172352 | KIPAR5=172352 | |
| 1331 | 172354 | KIPAR6=172354 | |
| 1332 | 172356 | KIPAR7=172356 | |
| 1333 | | | |
| 1334 | | | |
| 1335 | | : INSTRUCTION EQUATES | |
| 1336 | 104400 | HLT=TRAP | |
| 1337 | 104000 | SCOPE=EMT | : SCOPE IS AN EMT TRAP |
| 1338 | | | |
| 1339 | | : MISC. EQUATES | |
| 1340 | 000006 | RW=6 | : R/W BIT IN PDR REGISTERS |

```

1341      000000
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357      000000
1358      000000 000000
1359      000002 000000
1360
1361      000004 001116
1362      000006 000002
1363
1364      000034 001174
1365      000036 000340
1366
1367      000046 004654
1368
1369      000052 040000
1370
1371
1372      000250 000000
1373
1374
1375
1376      000252 000207
1377
1378
1379      000120 000000
1380      000122 000000
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393      000124 005737 000120
1394      000130 100401
1395      000132 000207
1396      000134 011557 017762

```

```

UP=0 ;UP BIT IN PDR REGISTERS

.=0
.WORD 0 ;SPECIAL TRAP/INTERRUPT CATCHER IF PDC-
.WORD 0 ;GRAM HALTS AT 0 THEN ADDRESS WAS NOT
;LOADED PROPERLY FROM VECTOR.

.WORD ERRTRP
.WORD RTI
.=TRAPVEC
.WORD ERROR
.WORD PRY7
.=46
.WORD LOGICAL
.=52
.WORD BIT14

.=250
EOPHLT: HALT ;THIS IS AN END OF PASS HALT;
;NOT AN ERROR HALT. YOU GET HERE ONLY
;IF SW06 IS UP. PRESS CONTINUE TO
;RESUME THE PROGRAM.

RTS PC

.=120
RELFL: .WORD 0
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 THAT THIS ROUTINE WILL NOT
;RELOCATE THE PROGRAM IF IT HAS BEEN MOVED BY ANY OTHER SUBPROGRAM
;EXCEPT THE BRANCH 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 DZOMB AND DZOMBE VERSIONS OF THE 0-124 TEST. IF IT HAS BEEN
;RELOCATED BY THE PART OF THIS TEST WHICH WAS TAKEN FROM DDQAB, THAT
;IS ALL THIS PROGRAM EXCEPT BRANCH GOBBLE.
WHERE: TST J#120
BMI IS
RTS PC
IS: MOV (SP),SAVPC2+20000

```

```

1397 000140 004567 026204 JSR R5,RELOC+20000
1398 000144 020000 .WORD 20000
1399 000146 000000 .WORD 0
1400 000150 016716 177746 MOV SAVPC2,(SP)
1401 000154 005037 000120 CLR @#120
1402 000160 000207 RTS PC
1403 ;
1404 ;
1405 000162 012706 000500 PONE: MOV @#162
1406 000166 004767 177732 JSR @#500,SP ;STARTING ADDRESS TO RELOCATE LOADERS.
1407 000172 004767 001732 JSR PC,WHERE
1408 000176 000000 JSR PC,$RLDR
1409 000200 012706 000500 PTWO: MOV @#500,SP ;STARTING ADDRESS OF 0-124K MEMORY EXERCISER.
1410 000204 004767 177714 JSR PC,WHERE
1411 000210 000137 002304 JMP @#START ;GO TO START OF TEST
1412 000214 012706 000500 PTHREE: MOV @#500,SP ;STARTING ADDRESS OF PROGRAM #2.
1413 000220 004767 177700 JSR PC,WHERE
1414 000224 000137 004670 JMP @#PRG2 ;GO START PROGRAM #2
1415 000230 012706 000500 PFOUR: MOV @#500,SP ;STARTING ADDRESS OF PROGRAM #3.
1416 000234 004767 177664 JSR PC,WHERE
1417 000240 000137 006010 JMP @#PRG3 ;GO START PROGRAM #3
1418 000250 000250 .=250
1419 000250 000000 .WORD 0 ;MEMORY MANAGEMENT TRAP VECTOR.
1420 000252 000000 .WORD 0
1421 000254 012706 000500 PFIVE: MOV @#500,SP ;START ADDRESS OF PROGRAM #4.
1422 000260 004767 177640 JSR PC,WHERE
1423 000264 000137 006042 JMP @#PRG4
1424 000270 012706 000500 PSIX: MOV @#500,SP ;STARTING ADDRESS OF BRANCH GOBBLE MCS TEST.
1425 000274 004767 177624 JSR PC,WHERE
1426 000300 000167 012236 JMP BRANCH
1427 ;
1428 ;
1429 ;ROUTINE TO SAVE REGISTERS ON THE STACK
1430 ;CALLED BY SAVE MACRO OR JSR PC,$SAVR
1431 000304 012667 000016 $SAVR: MOV (SP)+,1$ ;SAVE RETURN PC
1432 000310 010546 MOV %5,-(SP)
1433 000312 010446 MOV %4,-(SP)
1434 000314 010346 MOV %3,-(SP)
1435 000316 010246 MOV %2,-(SP)
1436 000320 010146 MOV %1,-(SP)
1437 000322 010046 MOV %0,-(SP)
1438 000324 012707 MOV (PC)+,PC ;RETURN
1439 000326 000000 1$: 0 ;CONTAINS RETURN ADDRESS
1440 ;
1441 ;ROUTINE TO RESTORE REGISTERS SAVED ON THE STACK
1442 ;CALLED BY RESTORE MACRO OR JSR PC,$RESTR
1443 000330 012667 000016 $RESTR: MOV (SP)+,1$ ;SAVE RETURN PC
1444 000334 012600 MOV (SP)+,%0
1445 000336 012601 MOV (SP)+,%1
1446 000340 012602 MOV (SP)+,%2
1447 000342 012603 MOV (SP)+,%3
1448 000344 012604 MOV (SP)+,%4
1449 000346 012605 MOV (SP)+,%5
1450 000350 012707 MOV (PC)+,PC ;RETURN
1451 000352 000000 1$: 0 ;CONTAINS RETURN ADDRESS
1452 ;

```

```

1453          .SBTTL POWER FAIL ROUTINE
1454          =502
1455          ;POWER FAIL ROUTINE
1456          ;THE POWER DOWN ROUTINE SAVES THE KEYBOARD STATUS,THE GENERAL REGISTERS
1457          ;:(RO-R5) AND MEM MGMT REGISTERS (KIPDR0-KIPDR7,KIPAR0-KIPAR7,SR2,SR0)
1458          ;ON THE STACK AND SAVES THE STACK POINTER IN PFSTK BELOW.
1459 000502 013746 177560 PDWN: MOV @TKS,-(SP) ;SAVE KEYBOARD STATUS
1460 000506 004767 177572 JSR PC,$SAVR ;GO SAVE REGISTERS ON THE STACK
1461 000512 005737 000752 TST @MMAVA ;CHECK IF MEM MGMT IS AVAILABLE
1462 000516 001417 BEQ 3$ ;BRANCH IF NOT AVAILABLE
1463 000520 013746 177572 MOV @SR0,-(SP) ;SAVE SR0
1464 000524 013746 177576 MOV @SR2,-(SP) ;SAVE SR2
1465 000530 012700 172300 MOV @KIPDR0,RO ;GET ADDRESS OF KIPDR0
1466 000534 012702 000010 MOV @R,R2
1467 000540 010203 MOV R2,R3
1468 000542 012046 1$: MOV (RO)+,-(SP) ;SAVE KIPDR0-KIPDR7
1469 000544 077202 SOB R2,1$
1470 000546 012700 172340 MOV @KIPAR0,RO ;GET ADDRESS OF KIPAR0
1471 000552 012046 2$: MOV (RO)+,-(SP) ;SAVE KIPAR0-KIPAR7
1472 000554 077302 SOB R3,2$
1473 000556 010627 3$: MOV SP,(PC)+ ;SAVE STACK PTR IN FOLLOWING LOCATION
1474 000560 000000 PFSTK: .WORD 0 ;CONTAINS STACK PTR AFTER POWER FAIL
1475 000562 012737 000572 000024 MOV @PUP,@PFVEC ;SET POWER FAIL VECTOR TO PUP ROUTINE
1476 000570 000000 HALT
1477
1478          ;POWER UP ROUTINE.
1479 000572 000240 PUP: NOP
1480 000574 013706 000560 MOV @PFSTK,SP ;SET STACK PTR
1481 000600 005767 000146 TST MMAVA ;CHECK IF MEM MGMT IS AVAILABLE
1482 000604 001417 BEQ 4$
1483 000606 012700 172360 MOV @KIPAR7+2,RO ;GET ADDRESS OF KIPAR7+2
1484 000612 012702 000010 MOV @R,R2
1485 000616 010203 MOV R2,R3
1486 000620 012640 1$: MOV (SP)+,-(RO) ;RESTORE KIPAR7-KIPAR0
1487 000622 077302 SOB R3,1$
1488 000624 012700 172320 MOV @KIPDR7+2,RO ;GET ADDRESS OF KIPDR7+2
1489 000630 012640 2$: MOV (SP)+,-(RO) ;RESTORE KIPDR7-KIPDR0
1490 000632 077202 SOB R2,2$
1491 000634 012637 177576 MOV (SP)+,@SR2 ;RESTORE SR2
1492 000640 012637 177572 MOV (SP)+,@SR0 ;RESTORE SR0
1493 000644 005767 006246 4$: TST PARAVA ;CHECK IF PARITY REGISTERS ARE ENABLED
1494 000650 001402 BEQ 5$ ;BRANCH IF NOT
1495 000652 004767 006170 JSR PC,.$MAMF ;GO ENABLE PARITY REGISTERS
1496 000656 5$:
1497 000656 004767 177446 JSR PC,$RESTR ;RESTORE REGISTERS FROM STACK
1498 000662 012637 177560 MOV (SP)+,@TKS
1499 000666 012737 000502 000024 MOV @PDWN,@PFVEC ;SET POWER FAIL TRAP TO PDWN ROUTINE
1500 000674 005027 CLR (PC)+
1501 000676 000000 10$: .WORD 0
1502 000700 005267 177772 11$: INC 10$ ;DELAY WAITING FOR TTY MOTOR
1503 000704 100375 BPL 11$
1504 000706 004567 000046 JSR RS,$PRINT ;GO TO PRINT ROUTINE
1505 000712 000720 PWRFAIL
1506 000714 000240 5$: NOP
1507 000716 000002 RTI ;RETJRN
1508

```

```

1509 000720 005015 047520 042527 PWRFAIL:.ASCIZ <15><12>'POWER FAILED'<15><12>
1510 000726 020122 040506 046111
1511 000734 042105 005015 000
1512
1513
1514 .SBTTL TAGS & PRINT ROUTINE
1515 000742 000000 ICNT: .EVEN
1516 000744 000000 ICOUNT: .WORD 0 ;CONTAINS PASS COUNT
1517 000746 000000 ERCNT: 0 ;CONTAINS ITERATION PATTERN
1518 000750 000000 LDDISP: 0 ;CONTAINS ERROR COUNT
1519 000752 000000 MMAVA: 0 ;CONTAINS DISPLAY REGISTER IMAGE
1520 ;MEM MGMT AVAILABLE INDICATOR
1521 000754 000000 RELOCF: .WORD 0 ;0=NOT AVAIL,-1=AVAIL
1522 000756 000000 COUNT: .WORD 0 ;CONTAINS RELOCATION FACTOR
1523 ;TEMPORARY WORKING LOCATION
1524 ;ROUTINE TO PASS MESSAGE ADDRESS TO TYPE ROUTINE BELOW
1525 ;CALL: JSR RS, $PRINT
1526 ; MESSAGE ADDRESS
1527 000760 000240 $PRINT: NOP
1528 000762 012567 000016 MOV (RS)+,1$ ;GET MESSAGE ADDRESS
1529 000766 066767 177762 000010 ADD RELOCF,1$ ;ADD RELOCATION FACTOR
1530 000774 013746 177776 MOV #PSW,-(SP) ;PUSH PSW ON THE STACK
1531 001000 004767 000014 JSR PC,..TYPE ;CALL TYPE ROUTINE
1532 001004 000000 1$: .WORD 0 ;CONTAINS MESSAGE ADDRESS
1533 001006 000205 RTS RS ;RETURN
1534
1535 ;ROUTINE TO TYPE ASCII MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
1536 ;THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
1537 ;CALL: TYPE
1538 ; MESADR ;MESADR IS FIRST ADDRESS OF ASCIZ STRING
1539
1540 ;TAGS USED BY THE TYPE ROUTINE BELOW
1541 001010 000 $NULL: .BYTE 0 ;CONTAINS NULL CHARACTER
1542 001011 002 $FILL: .BYTE 2 ;CONTAINS # OF FILLER CHARACTERS
1543 001012 000 $TPFLG: .BYTE 0 ;CONTAINS TELEPRINTER AVAILABLE FLAG
1544 ;0/377 = AVAIL/NOT AVAIL
1545 001013 000 $TKFLG: .BYTE 0 ;CONTAINS KEYBOARD AVAILABLE FLAG
1546 001014 177564 $TPS: .WORD 177564 ;ADDRESS OF TELEPRINTER STATUS REGISTER
1547 001016 177566 $TPB: .WORD 177566 ;ADDRESS OF TELEPRINTER DATA BUFFER
1548 001020 010046 .TYPE: MOV R0,-(SP) ;SAVE R0
1549 001022 017600 000002 MOV #2(SP),R0 ;GET MESSAGE ADDRESS
1550 001026 062766 000002 000002 000002 ADD #2,2(SP) ;ADJUST RETURN PC
1551
1552 001034 112046 1$: MOVB (R0)+,-(SP) ;PUSH CHARACTER TO BE TYPED ONTO STACK
1553 001036 001003 BNE 2$ ;BRANCH IF NOT THE TERMINATOR
1554 001040 005726 TST (SP)+ ;POP TERMINATOR CHAR OFF THE STACK
1555 001042 012600 MOV (SP)+,R0 ;RESTORE R0
1556 001044 000002 RTI ;RETURN TO CALLER
1557
1558 001046 004767 000026 2$: JSR PC,5$ ;TYPE CHARACTER
1559 001052 122726 000012 3$: CMPB #12,(SP)+ ;CHECK IF CHARACTER WAS A LINE FEED
1560 001056 001366 BNE 1$ ;BRANCH IF NOT LINE FEED
1561 001060 016746 177724 MOV $NULL,-(SP) ;GET # OF FILLERS REQUIRED AND FILLER
1562 ;CHARACTER.
1563
1564 001064 105366 000001 4$: DECB 1(SP) ;DECREMENT FILLERS REQ. COUNT

```

```

1565 001070 002770          BLT      3$          ;BRANCH IF NO MORE FILLERS ARE REQUIRED
1566 001072 004767 000002   JSR      PC,5$      ;TYPE FILLER CHARACTER
1567 001076 000772          BR       4$
1568
1569 001100 105777 177710   5$:     TSTB     2$TPS      ;WAIT FOR OUTPUT DEVICE
1570 001104 100375          BPL     .-4
1571 001106 116677 000002 177702   MOVB   2(SP),2$TPB ;OUTPUT CHARACTER
1572 001114 000207          RTS      PC
1573
1574          ;XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
1575          ;ERROR TRAP SERVICE ROUTINE
1576 001116 005737 177570   ERRTRP: TST     2$SWR      ;CHECK IF HALT ON ERROR
1577 001122 100001          BPL     .+4          ;BRANCH IF NO HALT ON ERROR
1578 001124 000000          HALT
1579 001126 005727          TST     (PC)+        ;CHECK IF PREV TRAP TO 4 REPORTED
1580 001130 000000   1$:     .WORD   0          ;CONTAINS ERROR REPORTED FLAG
1581 001132 001013          BNE     2$          ;BRANCH IF NOT REPORTED
1582 001134 010667 177770   MOV     SP,1$        ;SET 'NOT REPORTED'
1583 001140 011602          MOV     (SP),R2      ;GET PC OFF STACK
1584 001142 004767 000352   JSR     PC,$FORMO    ;GO TO FORMAT ROUTINE
1585 001146 004567 177606   JSR     R5,$PRINT    ;GO TO PRINT ROUTINE
1586 001152 001446          TRAP4
1587 001154 004567 177600   JSR     R5,$PRINT    ;GO TO PRINT ROUTINE
1588 001160 002271          DIGITS
1589 001162 000000   2$:     HALT          ;ERROR! SECOND TRAP TO 4 OCCURRED
1590          ;BEFORE FIRST WAS PRINTED
1591 001164 005067 177740   CLR     1$
1592 001170 000137 000200   JMP     2$200        ;RESTART AT 200
1593
1594          .SBTTL  ERROR SERVICE ROUTINE
1595          ;ERROR SERVICE CALLED BY JSR PC,ERROR INSTRUCTION
1596          ;OR HLT (A TRAP INST)
1597 001174 000240          ERROR: NOP
1598 001176 022767 017777 177542   CMP     #17777,ERCNT ;CHECK FOR MAX ERROR CNT
1599 001204 001403          BEQ     4$
1600 001206 062767 000001 177532   ADD     #1,ERCNT     ;INCREMENT ERROR COUNT
1601 001214 032737 001000 177570   4$:     BIT     #BIT9,2$SWR ;SWITCH 9 UP?
1602 001222 001411          BEQ     5$
1603 001224 042767 017777 177516   BIC     #17777,LDDISP ;SAVE RELOCATION BITS
1604 001232 056767 177510 177510   BIS     ERCNT,LDDISP ;LOAD ERROR COUNT
1605 001240 016737 177504 177570   MOV     LDDISP,2$DISPLAY ;LOAD DISPLAY REGISTER
1606 001246 004767 177032   5$:     JSR     PC,$SAVR    ;SAVE REGISTERS ON THE STACK
1607 001252 016602 000014          MOV     14(SP),R2    ;GET PC OF ERROR CALL
1608 001256 010201          MOV     R2,R1
1609 001260 162701 000002          SUB     #2,R1
1610 001264 032737 020000 177570   BIT     #20000,2$SWR ;PRINT OUT DESIRED?
1611 001272 001044          BNE     1$          ;BRANCH IF NO PRINTOUT
1612 001274 004767 000220   JSR     PC,$FORMO    ;GO TO FORMAT ROUTINE
1613 001300 004567 177454   JSR     R5,$PRINT    ;GO TO PRINT ROUTINE
1614 001304 001463          ERRPC
1615 001306 004567 177446   JSR     R5,$PRINT    ;GO TO PRINT ROUTINE
1616 001312 002271          DIGITS
1617 001314 016602 000004          MOV     4(SP),R2    ;GET FAILING ADDRESS (IN R2)
1618 001320 004767 000174   JSR     PC,$FORMO    ;GO TO FORMAT ROUTINE
1619 001324 004567 177430   JSR     R5,$PRINT    ;GO TO PRINT ROUTINE
1620 001330 002247          ADRESS

```

| | | | | | | | | |
|------|--------|--------|--------|--------|----------|--------------|----------------|--------------------------------------------------------------------|
| 1621 | 001332 | 105767 | 005265 | | TSTB | PENFLG | | : BRANCH IF PARITY ERROR DETECTED |
| 1622 | 001336 | 001017 | | | BNE | 11\$ | | : BUT NOT FOUND |
| 1623 | 001340 | 105767 | 005256 | | TSTB | PEFLG | | : BRANCH IF PARITY ERROR DETECTED |
| 1624 | 001344 | 001006 | | | BNE | 10\$ | | : BUT FOUND |
| 1625 | 001346 | 004567 | 177406 | | JSR | R5, \$SPRINT | | : GO TO PRINT ROUTINE |
| 1626 | 001352 | 001467 | | | XMTDAT | | | |
| 1627 | 001354 | 010046 | | | MOV | R0, -(SP) | | : PUSH VALUE TO TYPED ONTO STACK |
| 1628 | 001356 | 004767 | 000360 | | JSR | PC, 02A | | : GO PRINT VALUE |
| 1629 | 001362 | | | 10\$: | | | | |
| 1630 | 001362 | 004567 | 177372 | | JSR | R5, \$SPRINT | | : GO TO PRINT ROUTINE |
| 1631 | 001366 | 001502 | | | RECDAT | | | |
| 1632 | 001370 | 010346 | | | MOV | R3, -(SP) | | : PUSH VALUE TO BE TYPED ONTC STACK |
| 1633 | 001372 | 004767 | 000344 | | JSR | PC, 02A | | |
| 1634 | 001376 | | | 11\$: | | | | |
| 1635 | 001376 | 004567 | 177356 | | JSR | R5, \$SPRINT | | : GO TO PRINT ROUTINE |
| 1636 | 001402 | 015122 | | | \$CRLF | | | |
| 1637 | 001404 | 032737 | 002000 | 177570 | 1\$: | BIT | #2000, @#SWR | : RING BELL ON ERROR |
| 1638 | 001412 | 001403 | | | BEQ | 2\$ | | |
| 1639 | 001414 | 004567 | 177340 | | JSR | R5, \$SPRINT | | : GO TO PRINT ROUTINE |
| 1640 | 001420 | 001515 | | | BELL | | | |
| 1641 | 001422 | 005737 | 177570 | 2\$: | TST | @#SWR | | : HALT AFTER PRINT OUT |
| 1642 | 001426 | 100001 | | | BPL | .+4 | | |
| 1643 | 001430 | 000000 | | | HALT | | | |
| 1644 | 001432 | 004767 | 176672 | | JSR | PC, \$RESTR | | : RESTORE REGISTERS FROM STACK |
| 1645 | 001436 | 010042 | | 3\$: | MOV | R0, -(R2) | | : RESTORE CORRECT DATA TO ADDRESS |
| 1646 | 001440 | 062702 | 000002 | | ADD | #2, R2 | | |
| 1647 | 001444 | 000002 | | | RTI | | | |
| 1648 | | | | | | | | |
| 1649 | 001446 | 051124 | 050101 | 042520 | TRAP4: | .ASCII | 'TRAPPED TO 4' | |
| 1650 | 001454 | 020104 | 047524 | 032040 | | | | |
| 1651 | 001462 | 04C | | | | | | |
| 1652 | 001463 | 120 | 036503 | 000 | ERRPC: | .ASCIZ | 'PC=' | |
| 1653 | 001467 | 107 | 047517 | 020104 | XMTDAT: | .ASCIZ | 'GOOD DATA=' | |
| 1654 | 001474 | 040504 | 040524 | 000075 | | | | |
| 1655 | 001502 | 041040 | 042101 | 042040 | RECDAT: | .ASCIZ | 'BAD DATA=' | |
| 1656 | 001510 | 052101 | 036501 | 000 | | | | |
| 1657 | 001515 | 007 | 000 | | BELL: | .ASCIZ | <7> | |
| 1658 | | 001520 | | | | .EVEN | | |
| 1659 | | | | | | | | : ROUTINE TO PLACE ASCII VALUE OF AN ADDRESS IN TO ADDRESS MESSAGE |
| 1660 | 001520 | 066767 | 177230 | 000014 | \$FORMO: | ADD | RELOC, 11\$+2 | |
| 1661 | 001526 | 066767 | 177222 | 000134 | | ADD | RELOC, 41\$+2 | |
| 1662 | 001534 | 004767 | 176544 | | | JSR | PC, \$SAVR | : GO SAVE REGISTERS ON THE STACK |
| 1663 | 001540 | 012704 | 002271 | | 11\$: | MOV | #DIGITS, R4 | : ADDRESS WHERE ASCII VALUES ARE STORED |
| 1664 | 001544 | 005003 | | | | CLR | R3 | : WORKING & INDEX REGISTER |
| 1665 | 001546 | 162702 | 000002 | | | SUB | #2, R2 | : ADJUST ADDRESS |
| 1666 | 001552 | 010205 | | | | MOV | R2, R5 | : SAVE |
| 1667 | 001554 | 010501 | | | | MOV | R5, R1 | |
| 1668 | 001556 | 005767 | 177170 | | | TST | MMAVA | : CHECK IF MEM MGMT IS AVAILABLE |
| 1669 | 001562 | 001426 | | | | BEQ | 1\$ | : BRANCH IF NOT AVAILABLE |
| 1670 | 001564 | 032737 | 000001 | 177572 | | BIT | #1, @#SRO | : IS MEM MGMT ENABLED |
| 1671 | 001572 | 001422 | | | | BEQ | 1\$ | |
| 1672 | 001574 | 042701 | 017777 | | | BIC | #17777, R1 | : SAVE PAR SELECTOR BITS |
| 1673 | 001600 | 000301 | | | | SWAB | R1 | : SWAP BYTES |
| 1674 | 001602 | 006001 | | | | ROR | R1 | |
| 1675 | 001604 | 006001 | | | | ROR | R1 | : FORM INDEX VALUE |
| 1676 | 001606 | 006001 | | | | ROR | R1 | |

| | | | | | | |
|------|--------|--------|--------|---------|----------------|---------------------------------|
| 1677 | 001610 | 006001 | | ROR | R1 | |
| 1678 | 001612 | 017102 | 001722 | MOV | 2PARTAB(1),R2 | :GET CONTENTS OF PAR |
| 1679 | 001616 | 012700 | 000006 | MOV | #6,R0 | :SHIFT COUNT |
| 1680 | 001622 | 006302 | | ASL | R2 | :SHIFT KIPAR1 6 PLACES LEFT |
| 1681 | 001624 | 006103 | | ROL | R3 | :2 MSB'S GO INTO R3 |
| 1682 | 001626 | 077003 | | SOB | R0,-4 | |
| 1683 | 001630 | 042705 | 160000 | BIC | #160000,R5 | :CLEAR PAR SELECTOR BITS |
| 1684 | 001634 | 060502 | | ADD | R5,R2 | :FORM 18 BIT ADDRESS |
| 1685 | 001636 | 005503 | | ADC | R3 | :IN R2 & R3 |
| 1686 | 001640 | 006302 | | ASL | R2 | :FIRST DIGIT TO R3 |
| 1687 | 001642 | 006103 | | ROL | R3 | |
| 1688 | 001644 | 012700 | 000006 | MOV | #6,R0 | :DIGIT COUNT |
| 1689 | 001650 | 000404 | | BR | 3\$ | :PRINT FIRST DIGIT |
| 1690 | 001652 | 006302 | | ASL | R2 | |
| 1691 | 001654 | 006103 | | ROL | R3 | |
| 1692 | 001656 | 005305 | | DEC | R5 | |
| 1693 | 001660 | 001374 | | BNE | 2\$ | |
| 1694 | 001662 | 012705 | 000003 | MOV | #3,R5 | :DIGIT SHIFT COUNT |
| 1695 | 001666 | 116324 | 002232 | MOV | DIGTAB(3),(4)+ | :LOAD DIGIT INTO MESSAGE |
| 1696 | 001672 | 005003 | | CLR | R3 | :CLEAR INDEX |
| 1697 | 001674 | 005300 | | DEC | R0 | :DEC DIGIT COUNT |
| 1698 | 001676 | 001365 | | BNE | 2\$ | |
| 1699 | 001700 | 004767 | 176424 | JSR | PC,\$RESTR | :RESTORE REGISTERS FROM STACK |
| 1700 | 001704 | 046767 | 177044 | BIC | RELOCF,11\$+2 | |
| 1701 | 001712 | 046767 | 177036 | BIC | RELOCF,41\$+2 | |
| 1702 | 001720 | 000207 | | RTS | PC | :RETURN |
| 1703 | | | | | | |
| 1704 | 001722 | 172340 | | PARTAB: | KIPAR0 | |
| 1705 | 001724 | 172342 | | | KIPAR1 | |
| 1706 | 001726 | 172344 | | | KIPAR2 | |
| 1707 | 001730 | 172346 | | | KIPAR3 | |
| 1708 | 001732 | 172350 | | | KIPAR4 | |
| 1709 | 001734 | 172352 | | | KIPAR5 | |
| 1710 | 001736 | 172354 | | | KIPAR6 | |
| 1711 | 001740 | 172356 | | | KIPAR7 | |
| 1712 | | | | | | |
| 1713 | | | | | | |
| 1714 | | | | | | |
| 1715 | | | | | | |
| 1716 | 001742 | | | | | |
| 1717 | 001742 | 004767 | 176336 | JSR | PC,\$SAVR | :GO SAVE REGISTERS ON THE STACK |
| 1718 | 001746 | 016600 | 000016 | MOV | 16(SP),R0 | :GET VALUE |
| 1719 | 001752 | 012703 | 000006 | MOV | #6,R3 | :COUNTER |
| 1720 | 001756 | 005002 | | CLR | R2 | :WORKING REGISTER |
| 1721 | 001760 | 006100 | | ROL | R0 | |
| 1722 | 001762 | 006102 | | ROL | R2 | |
| 1723 | 001764 | 062702 | 000260 | ADD | #260,R2 | :FORM ASCII VALUE |
| 1724 | 001770 | 010267 | 000040 | MOV | R2,2\$ | :MOVE CHAR TO TYPE LOCATION |
| 1725 | 001774 | 004567 | 176760 | JSR | R5,\$PRINT | :GO TO PRINT ROUTINE |
| 1726 | 002000 | 002034 | | 2\$ | | |
| 1727 | 002002 | 005002 | | CLR | R2 | |
| 1728 | 002004 | 006100 | | ROL | R0 | |
| 1729 | 002006 | 006102 | | ROL | R2 | |
| 1730 | 002010 | 006100 | | ROL | R0 | |
| 1731 | 002012 | 006102 | | ROL | R2 | |
| 1732 | 002014 | 006100 | | ROL | R0 | |


```

1733 002016 006102          ROL      R2
1734 002020 005303          DEC      R3
1735 002022 001360          BNE     1$
1736 002024 004767 176300   JSR     PC,$RESTR      ;RESTORE REGISTERS FROM STACK
1737 002030 012616          MOV     (SP)+,(SP)
1738 002032 000207          RTS     PC
1739 002034 000000   2$:      .WORD 0      ;CONTAINS CHARACTER TO BE TYPED
1740
1741 002036 000000   LODFLO: .WORD 0
1742          ;ROUTINE TO SAVE ABS LOADER
1743 002040 005767 177772   $LDR:   TST     LODFLO
1744 002044 001401          BEQ     3$
1745 002046 000207          RTS     PC
1746 002050 012700 017776   3$:      MOV     #17776,R0
1747 002054 012737 002066 000004   MOV     #2$,$ERRVEC   ;SET TIME OUT TRAP VECTOR
1748 002062 005720          TST     (R0)+
1749 002064 000776          BR      -2
1750 002066 022626   2$:      CMP     (SP)+,(SP)+
1751 002070 162700 002202   SUB     #2202,R0      ;POINT R0 BACK TO LOADER
1752 002074 010067 000102   MOV     R0,$LDR1     ;SAVE FOR RESTORE ROUTINE
1753 002100 012702 001100   MOV     #1100,R2     ;WORD COUNT
1754 002104 012703 015556   MOV     #LODAR,R3    ;WHERE LOADER IS TO BE STORED
1755 002110 012023   1$:      MOV     (R0)+,(R3)+  ;STORE LOADER
1756 002112 005302          DEC     R2
1757 002114 001375          BNE     1$
1758 002116 014367 000042   MOV     -(R3),LSTLOC ;SAVE LAST WORD OF LOADERS
1759 002122 005367 177710   DEC     LODFLO
1760 002126 000207          RTS     PC      ;RETURN
1761
1762          ;ROUTINE TO RESTORE LOADER
1763 002130 005767 177702   $RLDR:  TST     LODFLO
1764 002134 001001          BNE     2$
1765 002136 000207          RTS     PC
1766 002140 016705 000036   2$:      MOV     $LDR1,R5   ;GET FIRST ADDRESS OF WHERE LOADER IS
1767          ;TO BE RESTORED
1768 002144 012704 015556   MOV     #LODAR,R4   ;ADDRESS WHERE LOADER IS STORED
1769 002150 012702 001100   MOV     #1100,R2   ;WORD COUNT
1770 002154 012425   1$:      MOV     (R4)+,(R5)+ ;RESTORE
1771 002156 005302          DEC     R2
1772 002160 001375          BNE     1$
1773 002162 012745          MOV     (PC)+,-(R5) ;RESTORE LAST LOCATION (SAVED BY SAVE
1774 002164 000000   LSTLOC: .WORD 0     ;LOADERS ROUTINE ABOVE)
1775 002166 004567 176566   JSR     R5,$PRINT   ;GO TO PRINT ROUTINE
1776 002172 002204          $LDRM  CLR     LODFLO
1777 002174 005067 177636   CLR     LODFLO
1778 002200 000207          RTS     PC      ;RETURN TO CALLER
1779
1780 002202 000000   $LDR1:  .WORD 0     ;FIRST ADDRESS WHERE LOADERS ARE TO BE
1781          ;RESTORED TO
1782 002204 047514 042101 051105   $LDRM:  .ASCIZ 'LOADER IS RESTORED'<15><12>
1783 002212 044440 020123 042522
1784 002220 052123 051117 042105
1785 002226 005015      000
1786          .EVEN
1787          ;DIGIT TABLE
1788 002232 030460   DIGTAB: "01

```

K04

TEST DDQAB-A 0-124K MEMORY EXERCISER
DDQABA.P11 ERROR SERVICE ROUTINE

MACY11 27(732) 10-SEP-76 10:35 PAGE 49

1789 002234 031462
1790 002236 032464
1791 002240 033466

"23
"45
"67

1792
1793
1794 002242 040514 052123 040
1795 002247 115 046505 051117
1796 002254 020131 042101 051104
1797 002262 051505 020123 051511
1798 002270 040
1799 002271 060 030060 030060
1800 002276 020060 000
1801 002302

;MESSAGES
LST: .ASCII 'LAST '
ADRESS: .ASCII 'MEMORY ADDRESS IS '

DIGITS: .ASCIZ '000000 '

.EVEN

```

1802
1803 002302 000000
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818 002304 012737 002346 000212
1819 002312 012706 000500
1820 002316 004767 177516
1821 002322 004567 176432
1822 002326 012054
1823 002330 005037 000746
1824 002334 005037 000750
1825 002340 013737 000750 177570
1826 002346 012706 000500
1827 002352 005037 006622
1828 002356 012727 002346
1829 002362 000000
1830 002364 005037 000742
1831 002370 005037 000754
1832 002374 012737 000502 000024
1833 002402 005037 000026
1834
1835
1836 002406 005067 176340
1837 002412 032737 010000 177570
1838 002420 001007
1839 002422 012737 002440 000004
1840 002430 005037 177572
1841 002434 005167 176312
1842 002440 004767 004402
1843
1844
1845
1846
1847 002444 012737 002504 000004
1848 002452 010701
1849 002454 004767 004466
1850 002460 012737 007244 000250
1851 002466 012702 020000
1852 002472 010203
1853 002474 010322
1854 002476 062703 000002
1855 002502 000774
1856
1857 002504 012706 000500

```

```

PLACE: .WORD 0
       .SBTTL MEMORY ADDRESS TESTS

```

```

; THIS TEST ADDRESS MEMORY UP TO 128K AND PROVES 'UNIQUENESS' 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 INSTRUCTIONS

```

```

LOAD ADDRESS=200

```

```

PRESS START

```

```

STACK POINTER IS AT 500

```

```

*****RESTART AT 162 TO RESTORE LOADER*****

```

```

MEMORY ADDRESS TEST

```

```

START: MOV #START1, #212 ; CHANGE START ADDRESS
        MOV #STKPTR, SP ; SET UP STACK PTR
        JSR PC, $LDR ; GO SAVE MONITOR & LOADERS
        JSR R5, $PRINT ; GO TO PRINT ROUTINE
        RESLDR
        CLR #ERCNT ; CLEAR ERROR COUNT
        CLR #LDDISP ; CLEAR DISPLAY REGISTER STORAGE LOCN
        MOV #LDDISP, #DISPLAY ; CLEAR DISPLAY REGISTER
START1: MOV #STKPTR, SP ; SET STACK PTR
        CLR #PEFLG ; CLEAR PARITY ERROR INDICATORS
        MOV #START1, (PC)+ ; LOAD PARITY ERROR RESTART ADDRESS
PERSTRT: .WORD 0 ; CONTAINS RESTART ADDRESS AFTER PAR ERR
        CLR #ICNT ; CLEAR PASS COUNT
        CLR #RELOCF ; CLEAR RELOCATION FACTOR
        MOV #PDWN, #PFVEC ; SET POWER FAIL TRAP VECTOR
        CLR #PFVEC+2

```

```

; CHECK IF MEMORY MANAGEMENT IS AVAILABLE

```

```

CLR MAVA ; CLEAR MEM MGMT AVAILABLE INDICATOR
BIT #BIT12, #SWR ; CHECK IF TO RUN WITH MEM MGMT
BNE IS ; DO NOT USE MEM MGMT IF SW12 WAS SET
MOV #IS, #ERRVEC ; SET TIME OUT TRAP
CLR #SRO ; REFERENCE MEM MGMT
COM MAVA ; SET INDICATOR TO -1 IF AVAILABLE
IS: 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

```

```

WRTUP: MOV #DONE0, #ERRVEC ; SET TIME OUT TRAP VECTOR
        MOV PC, R1 ; LOAD TRACE REGISTER
        JSR PC, LDMMO
        MOV #MMABTO, #MMVEC ; SET MEM MGMT ABORT VECTOR
        MOV #20000, R2 ; FIRST ADDRESS
        MOV R2, R3 ; LOAD CONSTANT
        MOV R3, (R2)+ ; WRITE VALUE OF ADDRESS INTO ADDRESS
        ADD #2, R3 ; NEXT VALUE
        BR -.6 ; WRITE UNTIL DONE

```

```

DONE0: MOV #STKPTR, SP ; SET STACK PTR

```

```

1858 002510 004767 177004      JSR    PC,$FORMD      ;GO TO FORMAT ROUTINE
1859 002514 004567 176240      JSR    R5,$SPRINT     ;GO TO PRINT ROUTINE
1860 002520 002242                LST
1861 002522 004567 176232      JSR    R5,$SPRINT     ;GO TO PRINT ROUTINE
1862 002526 015122                $CRLF
1863
1864                                ;ROUTINE TO CHECK THAT VALUE OF MEMORY ADDRESS WAS WRITTEN CORRECTLY
1865 002530 010701                MOV    PC,R1          ;LOAD TRACE REGISTER
1866 002532 012702 020000        MOV    #20000,R2      ;SET R2
1867 002536 012737 002574 000004  MOV    #DONE1,2#ERRVEC ;SET TIME OUT TRAP
1868 002544 010200                MOV    R2,R0
1869 002546 162700 000002        SUB    #2,R0          ;SUBTRACT 2
1870 002552 004767 004370        JSR    PC,LDMMO
1871 002556 062700 000002        1$:   ADD    #2,R0
1872 002562 012203                MOV    (R2)+,R3      ;GET WRITTEN VALUE
1873 002564 020003                CMP    R0,R3         ;CHECK
1874 002566 001773                BEQ    1$
1875 002570 104400                HLT
1876                                ;ERROR! TO DETERMINE WHICH ADDRESS WAS
1877                                ;WRITTEN IMPROPERLY EXAMINE R2. NEXT EXAMINE MEM MGMT REGISTER KIPARI
1878                                ;(IF MEM MGMT IS AVAILABLE). ADD R2 AND KIPARI TOGETHER AS SHOWN BELOW
1879                                ;
1880                                ;      R2-2                0 00X XXX XXX XXX XXX
1881                                ;      KIPARI(772342) 0 000  YYY YYY YYY YYY
1882                                ;      ADDRESS                ZZZ ZZZ ZZZ ZZZ ZZZ ZZZ
1883 002572 000771                BR     1$
1884 002574 012706 000500        DONE1: MOV    #STKPTR,SP ;SET STACK PTR
1885 002600 010701                MOV    PC,R1         ;LOAD TRACE REGISTER
1886
1887                                ;ROUTINE TO WRITE 1'S COMPLEMENT VALUE OF ADDRESS INTO ADDRESS
1888                                ;FOR EXAMPLE ROUTINE WRITES 157777 INTO ADDRESS 20000
1889
1890 002602 005767 176144        TST    MMAVA          ;MEMORY MAGNAGEMENT AVAILABLE?
1891 002606 001420                BEQ    3$
1892 002610 013703 172342        MOV    2#KIPARI,R3   ;FIND LAST ADDRESS IF MEM MANAGE USED
1893 002614 006303                ASL    R3
1894 002616 006303                ASL    R3
1895 002620 006303                ASL    R3
1896 002622 006303                ASL    R3
1897 002624 006303                ASL    R3
1898 002626 006303                ASL    R3
1899 002630 010246                MOV    R2,-(SP)      ;DEVELOP COMPLEMENT OF LAST ADDRESS
1900 002632 042716 020000        BIC    #20000,(SP)   ;SAVE BITS IF MEMORY IS NOT A MULTIPLE OF 4K
1901 002636 062603                ADD    (SP)+,R3
1902 002640 012737 007276 000250  MOV    #MMABT1,2#MMVEC ;SET ABORT VECTOR
1903 002646 000403                BR     2$
1904 002650 162702 000002        3$:   SUB    #2,R2        ;R2=LAST ADDRESS
1905 002654 010203                MOV    R2,R3
1906 002656 005103                2$:   COM    R3          ;COMPLEMENT VALUE IN R3
1907 002660 062703 000002        1$:   ADD    #2,R3
1908 002664 010342                MOV    %3,-(R2)     ;WRITE COMPLIMENT VALUE INTO ADDRESS
1909 002666 102403                BVS    DONE3
1910 002670 020227 017776        CMP    R2,#17776
1911 002674 001371                BNE    1$
1912
1913                                ;SET UP TO CHECK COMPLEMENT DATA WRITTEN DOWN

```

NO4

TEST DDQAB-A 0-124K MEMORY EXERCISER
DDQABA.P11 MEMORY ADDRESS TESTS

MACY11 27(732) 10-SEP-76 10:35 PAGE 52

| | | | | | | | |
|------|--------|--------|--------|--------|---------------------------|--|----------------------------------------------------------|
| 1914 | 002676 | 000240 | | | DONE3: NOP | | |
| 1915 | 002700 | 010701 | | | MOV PC, R1 | | ;LOAD TRACE REGISTER |
| 1916 | 002702 | 005767 | 176044 | | TST MMAVA | | ;CHECK IF MM IS AVAIL |
| 1917 | 002706 | 001406 | | | BEQ 1\$ | | |
| 1918 | 002710 | 012737 | 000200 | 172342 | MOV #200, @#KIPARI | | ;INIT KIPARI |
| 1919 | 002716 | 012737 | 007244 | 000250 | MOV #MMABT0, @#MMVEC | | ;SET ABORT VECTOR |
| 1920 | 002724 | 012737 | 002764 | 000004 | 1\$: MOV #DONE4, @#ERRVEC | | |
| 1921 | 002732 | 012702 | 020000 | | MOV #20000, R2 | | ;FIRST ADDRESS |
| 1922 | 002736 | 010200 | | | MOV R2, RO | | |
| 1923 | 002740 | 005100 | | | COM RO | | ;FIRST DATA (COM OF ADDRESS) |
| 1924 | 002742 | 062700 | 000002 | | ADD #2, RO | | |
| 1925 | 002746 | 162700 | 000002 | | 2\$: SUB #2, RO | | |
| 1926 | 002752 | 012203 | | | MOV (R2)+, R3 | | ;GET VALUE |
| 1927 | 002754 | 020003 | | | CMP RO, R3 | | ;CHECK |
| 1928 | 002756 | 001773 | | | BEQ 2\$ | | |
| 1929 | 002760 | 104400 | | | HLT | | |
| 1930 | 002762 | 000771 | | | BR 2\$ | | |
| 1931 | 002764 | 000240 | | | DONE4: NOP | | |
| 1932 | | | | | | | |
| 1933 | | | | | | | ;ROUTINE TO WRITE BANK # INTO ALL ADDRESSES IN A 4K BANK |
| 1934 | 002766 | 012737 | 003034 | 000004 | MOV #DONE4A, @#ERRVEC | | ;SET TIME OUT TRAP VECTOR |
| 1935 | 002774 | 010701 | | | MOV PC, R1 | | |
| 1936 | 002776 | 004767 | 004144 | | JSR PC, LDMMO | | |
| 1937 | 003002 | 012737 | 007244 | 000250 | MOV #MMABT0, @#MMVEC | | |
| 1938 | 003010 | 012702 | 020000 | | MOV #20000, R2 | | |
| 1939 | 003014 | 005000 | | | CLR RO | | |
| 1940 | 003016 | 005200 | | | 1\$: INC RO | | ;RO WILL BE DATA WRITTEN |
| 1941 | 003020 | 012704 | 010000 | | MOV #4096, R4 | | ;SET 4K COUNTER |
| 1942 | 003024 | 010022 | | | 2\$: MOV RO, (R2)+ | | ;WRITE BANK # INTO ALL ADDRESSES |
| 1943 | 003026 | 005304 | | | DEC R4 | | |
| 1944 | 003030 | 001375 | | | BNE 2\$ | | |
| 1945 | 003032 | 000771 | | | BR 1\$ | | |
| 1946 | | | | | | | |
| 1947 | 003034 | 022626 | | | DONE4A: CMP (SP)+, (SP)+ | | ;ADJUST STACK PTR |
| 1948 | | | | | | | |
| 1949 | | | | | | | ;CHECK THAT DATA WRITTEN ABOVE CAN BE READ |
| 1950 | 003036 | 012737 | 003104 | 000004 | MOV #DONE4B, @#ERRVEC | | |
| 1951 | 003044 | 010701 | | | MOV PC, R1 | | |
| 1952 | 003046 | 004767 | 004074 | | JSR PC, LDMMO | | |
| 1953 | 003052 | 012702 | 020000 | | MOV #20000, R2 | | |
| 1954 | 003056 | 005000 | | | CLR RO | | |
| 1955 | 003060 | 005200 | | | 1\$: INC RO | | |
| 1956 | 003062 | 012704 | 010000 | | MOV #4096, R4 | | |
| 1957 | 003066 | 012203 | | | 2\$: MOV (R2)+, R3 | | |
| 1958 | 003070 | 020003 | | | CMP RO, R3 | | |
| 1959 | 003072 | 001401 | | | BEQ .+4 | | |
| 1960 | 003074 | 104400 | | | HLT | | |
| 1961 | 003076 | 005304 | | | DEC R4 | | |
| 1962 | 003100 | 001372 | | | BNE 2\$ | | |
| 1963 | 003102 | 000766 | | | BR 1\$ | | |
| 1964 | 003104 | 022626 | | | DONE4B: CMP (SP)+, (SP)+ | | |
| 1965 | | | | | | | |
| 1966 | | | | | | | ;ROUTINE TO WRITE CONSTANT DATA INTO 4K |
| 1967 | | | | | | | ;BANK STARTING WITH LAST MEMORY LOCATION |
| 1968 | 003106 | 010701 | | | MOV PC, R1 | | |
| 1969 | 003110 | 012737 | 007276 | 000250 | MOV #MMABT1, @#MMVEC | | |

2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068

003272 012706 000500
003276 004767 003544
003302 004767 006444
003306 012737 003330 000004

003314 010701
003316 012746 000001
003322 005046
003324 004767 004162

003330 012737 001116 000004
003336 012706 000500
003342 010701
003344 012746 000001
003350 005403
003352 010346
003354 004767 004244

003360 005027
003362 000000

003364 012706 000500
003370 010701
003372 012737 003412 000004
003400 012746 000001
003404 005046
003406 004767 004446

003412 012737 001116 000004
003420 016600 000006
003424 005400
003426 010027
003430 000000
003432 012706 000500
003436 010701
003440 012746 000001
003444 010046
003446 004767 004626

003452 005737 007116
003456 001406
003460 005737 003362

SBTTL WORST CASE NOISE TESTS
; THIS TEST WRITES MEMORY WORST CASE NOISE TEST PATTERNS THROUGHOUT
; MEMORY AND CHECKS THAT THEY CAN BE WRITTEN AND READ.
; SET UP TRAP VECTORS
BEGIN1: MOV #STKPTR, SP ; SET STACK PTR
JSR PC, .MAMP ; GO ENABLE PARITY ACTION
JSR PC, CKSWR ; GO CHECK SWITCHES
MOV #DONES, @ERRVEC ; SET UP TIME OUT TRAP

; WRITE 1 XOR 8 TEST PATTERN STARTING AT ADDRESS 20000
MOV PC, R1 ; UPDATE TRACE REGISTER
MOV #1, -(SP) ; PUSH STARTING BANK # ON THE STACK
CLR -(SP) ; PUSH # OF 128. WORD BLOCKS TO WRITE
JSR PC, .1XB ; GO TO ROUTINE TO WRITE 1 XOR 8 PATTERN

; CHECK 1 XOR 8 TEST PATTERN WRITTEN ABOVE
DONES: MOV #ERRTRP, @ERRVEC
IS: MOV #STKPTR, SP ; SET STACK PTR
MOV PC, R1 ; UPDATE TRACE REGISTER
MOV #1, -(SP) ; PUSH STARTING BANK # ON THE STACK
NEG R3 ; R3 CONTAINS # OF 128. WORD BLOCKS
MOV R3, -(SP) ; WRITTEN BY .1XB ROUTINE ABOVE
JSR PC, .1XB ; GO CHECK 1XB PATTERN

CLR (PC)+ ; SET INDICATOR TO WRITE NORMAL 3XB PAT
PARPAT: .WORD 0

; WRITE 3 XOR 9 TEST PATTERN STARTING AT ADDRESS 20000
; NOTE PATTERN IS NORMAL 3 XOR 9 IF NO PARITY MEMORY IS AVAILABLE.
; AND IS A MODIFIED PATTERN IF PARITY MEMORY IS AVAILABLE.
; THE CONTENTS OF PARPAT IF 0/NOT 0 INDICATE IF NORMAL/MODIFIED PATTERN
; IS BEING USED IN TESTS BELOW.
DONES: MOV #STKPTR, SP ; SET STACK PTR
MOV PC, R1 ; UPDATE TRACE REGISTER
MOV #DONES, @ERRVEC ; SET TIME OUT TRAP VECTOR
MOV #1, -(SP) ; PUSH STARTING BANK # ON STACK
CLR -(SP) ; PUSH # OF 256. WORD BLOCKS TO WRITE
JSR PC, .3X9 ; CALL ROUTINE TO WRITE 3XOR9 PATTERN

; CHECK 3 XOR 9 TEST PATTERN WRITTEN ABOVE
DONES: MOV #ERRTRP, @ERRVEC
MOV 6(SP), R0 ; GET # OF 256. WORD BLOCKS WRITTEN
NEG R0 ; FORM TWO'S COMPLEMENT
MOV R0, (PC)+ ; SAVE # OF 256 WORD BLOCKS
WDS.256: .WORD 0 ; CONTAINS # OF 256 WORD BLOCKS IN MEM.
MOV #STKPTR, SP ; SET STACK PTR
MOV PC, R1 ; SET SCOPE PTR
MOV #1, -(SP) ; PUSH BANK # ON THE STACK
MOV R0, -(SP) ; PUSH # OF 256. WORD BLOCKS TO WRITE
JSR PC, .3X9 ; GO CHECK DATA WRITTEN

; SETUP TO RUN MODIFIED 3 XOR 9 PATTERN IF PARITY MEMORY IS AVAILABLE
TST @PARAVA ; BRANCH IF PARITY MEMORY IS NOT AVAIL
BEQ DONES
TST @PARPAT ; BRANCH IF PARITY PAT JUST WRITTEN

```

2069 003464 001003          BNE      DONE8
2070 003466 010637 003362  MOV      SP,2#PARPAT ;SET INDICATOR TO WRITE 3X9 PAR PAT
2071 003472 000734          BR       DONE6        ;REPEAT TEST USING MODIFIED 3X9 PATTERN
2072
2073          ;WRITE 8 XOR 13 TEST PATTERN STARTING AT ADDRESS 40000
2074 003474 012706 000500 000004  DONE8:  MOV      #STKPTR,SP ;SET STACK PTR
2075 003500 012737 003522          MOV      #DONE9,2#ERRVEC ;SET TIME OUT TRAP VECTOR
2076 003506 010701          MOV      PC,R1         ;UPDATE TRACE REGISTER
2077 003510 012746 000002          MOV      #2,-(SP)      ;PUSH STARTING BANK # ON THE STACK
2078 003514 005046          CLR      -(SP)         ;PUSH # OF BANKS TO WRITE ON THE STACK
2079 003516 004767 005226          JSR      PC,.BX13      ;GO TO ROUTINE TO WRITE DATA
2080
2081          ;CHECK 8 XOR 13 TEST PATTERN WRITTEN ABOVE
2082 003522 012706 000500 000004  DONE9:  MOV      #STKPTR,SP ;SET STACK PTR
2083 003526 010701          MOV      PC,R1         ;UPDATE TRACE REGISTER
2084 003530 012737 001116          MOV      #ERRTRP,2#ERRVEC
2085 003536 012746 000002          MOV      #2,-(SP)
2086 003542 005404          NEG      R4
2087 003544 042704 000001          BIC      #1,R4        ;SET 4K BANK COUNT TO 8K INCREMENT
2088 003550 001403          BEQ     DONE10        ;DO NOT CHECK IF ONLY 12K
2089 003552 010446          MOV      R4,-(SP)
2090 003554 004767 005246          JSR      PC...BX13    ;GO CHECK 8 XOR 13 PATTERN WRITTEN ABOVE
2091
2092
2093          ;RELOCATE PROGRAM TO CHECK ADDRESSES FROM 000000-017776 USING 1 XOR 8 PATTERN
2094 003560 000005          DONE10: RESET ;DISABLE MEM MGMT AND PARITY ACTION
2095 003562 005737 000042 1$:      TST      2#42      ;CHECK IF PROGRAM LOADED VIA ACT11
2096 003566 001402          BEQ     2$
2097 003570 000137 004252          JMP     2#RANTST      ;DO NOT RELOCATE IF ACT11
2098 003574 012706 000500 2$:      MOV      #STKPTR,SP ;SET STACK PTR
2099 003600 004567 002544          JSR      5,RELOC      ;RELOCATE PROGRAM CODE
2100 003604 000000          ;FROM 000000 TO
2101 003606 020000          ;20000
2102 003610 005704          TST     R4            ;WAS RELOCATION SUCCESSFUL?
2103 003612 001402          BEQ     3$            ;BRANCH IF SUCCESSFUL I.E. WAS THERE
2104                                ;SUFFICIENT MEMORY TO RELOCATE TO.
2105 003614 000137 004252          JMP     2#RANTST      ;END OF TEST
2106 003620 062707 020000 3$:      ADD     #20000,PC     ;RELOCATE PC
2107 003624 062706 020000          ADD     #20000,SP     ;SET NEW STACK PTR
2108 003630 052767 100000 175112  BIS     #10000,LDDISP ;SET RELOCATION INDICATOR
2109 003636 016737 175106 177570  MOV     LDDISP,2#DISPLAY;LOAD DISPLAY REGISTER
2110
2111          ;*****IMPORTANT NOTE*****
2112          ;PROGRAM IS NOW EXECUTING CODE FROM PC AS SHOWN BELOW +20000.
2113          ;CAUTION: DO NOT ATTEMPT TO RESTART PROGRAM AT 200
2114          ;*****
2115
2116 003644 010701          DONE11: MOV     PC,R1 ;RESTART ADDRESS TO LOOP TEST
2117
2118          ;WRITE 1 XOR 8 TEST PATTERN IN LOCATIONS 000000-017776
2119 003646 005046          CLR     -(SP)         ;PUSH STARTING BANK # ON THE STACK
2120 003650 012746 000040          MOV     #32,-(SP)     ;PUSH # OF 128. WORD BLOCKS TO WRITE
2121 003654 004767 003632          JSR     PC,.1XB       ;GO TO ROUTINE TO WRITE 1 XOR 8 DATA
2122
2123 003660 010701          MOV     PC,R1         ;RESTART & LOOP TEST ADDRESS
2124

```



```

2125
2126 003662 005046
2127 003664 012746 000040
2128 003670 004767 003730
2129
2130
2131
2132 003674 010701
2133 003676 004567 002446
2134 003702 020000
2135 003704 000000
2136 003706 000137 003712
2137 003712 012767 040000 000106
2138 003720 012767 000040 000150
2139 003726 042737 100000 000750
2140 003734 016737 175010 177570
2141 003742 012737 004034 000004
2142 003750 005737 057776
2143 003754 012737 003776 000004
2144 003762 005737 117776
2145 003766 006367 000034
2146 003772 006367 000100
2147 003776 012706 000500
2148 004002 010701
2149 004004 042737 100000 000750
2150 004012 013737 000750 177570
2151 004020 004567 002324
2152 004024 000000
2153 004026 040000
2154 004030 005704
2155 004032 001402
2156 004034 000137 004252
2157 004040 066707 177762
2158 004044 066706 177756
2159 004050 052767 100000 174672
2160 004056 016737 174666 177570
2161
2162
2163
2164
2165
2166
2167 004064 005037 003362
2168 004070 010701
2169
2170
2171 004072 005046
2172 004074 012746
2173 004076 000000
2174
2175 004100 004767 003754
2176
2177
2178 004104 010701
2179 004106 005046
2180 004110 016746 177762

```

```

;CHECK 1 XOR 8 TEST PATTERN AS WRITTEN ABOVE
CLR -(SP) ;PUSH STARTING BANK # ON THE STACK
MOV #32,-(SP) ;PUSH # OF 128. WORD BLOCKS TO WRITE
JSR PC,..1X8 ;GO TO ROUTINE TO CHECK DATA

;RELOCATE PROGRAM TO CHECK ADDRESSES 000000 - 037776 USING
;3 XOR 9 AND 8 XOR 13 PATTERNS
DONE12: MOV PC,R1 ;UPDATE TRACE REGISTER
JSR RS,RELOC ;MOVE PROGRAM BACK TO LOWEST 4K
;FROM 20000 TO
;000000
JMP #.+4 ;RETURN UNRELOCATED
MOV #40000,2S
MOV #32, BLKCNT
BIC #100000,2#LDDISP ;CLEAR RELOCATION INDICATOR
MOV LDDISP,2#DISPLAY ;DISPLAY NOT RELOCATED
MOV #3S,2#ERRVEC ;SET TIME OUT TRAP
TST 2#057776 ;CHECK IF 12K OF MEMORY IS AVAILAELE
MOV #1S,2#ERRVEC
TST 2#117776 ;CHECK IF 20K OF MEMORY
ASL 2S
ASL BLKCNT
1S: MOV #STKPTR,SP ;SET STACK POINTER
MOV PC,R1 ;UPDATE TRACE REGISTER
BIC #100000,2#LDDISP ;CLEAR RELOCATION INDICATOR
MOV 2#LDDISP,2#DISPLAY ;LOAD DISPLAY REGISTER
JSR RS,RELOC ;RELOCATE PROGRAM
;FROM 000000
;TO 40000
2S: TST R4 ;RELOCATION SUCCESSFUL?
BEQ 4S ;YES
3S: JMP 2#RANTST ;GO TO RANDOM DATA TEST
4S: ADD 2S,PC ;RELOCATE PC
ADD 2S,SP ;SET NEW STACK PTR
BIS #100000,LDDISP ;SET RELOCATION INDICATOR
MOV LDDISP,2#DISPLAY;RELOAD DISPLAY REGISTER

;*****IMPORTANT NOTE*****
;PROGRAM IS NOW EXECUTING CODE FROM PC AS SHOWN BELOW +40000 OR +100000
;CAUTION: DO NOT ATTEMPT TO RESTART PROGRAM AT 200
;*****

DONE13: CLR 2#PARPAT ;SET INDICATOR TO WRITE NORMAL 3X9 PAT
MOV PC,R1 ;UPDATE TRACE REGISTER

;WRITE 3X09 TEST PATTERN IN LOCATIONS 000000-037776 OR 000000-077776
CLR -(SP) ;PUSH BANK # 0 ON THE STACK
MOV (PC)+,-(SP) ;PUSH 256. BLOCK WORD COUNT ON STACK
BLKCNT: .WORD 0 ;CONTAINS 256. BLOCK WORD COUNT LOADED
;BY ABOVE ROUTINE. 40/100 IF 8/16K
JSR PC,.3X9

;CHECK PATTERN WRITTEN IN LOCATIONS 000000-037776 OR 000000-077776
MOV PC,R1
CLR -(SP) ;PUSH STARTING BANK # ON THE STACK
MOV BLKCNT,-(SP) ;PUSH 256. WORD BLOCK COUNT ON THE STACK

```

```

2181 004114 004767 004160          JSR      PC,..3X9          ;CHECK
2182
2183          ;ROUTINE TO CHECK IF MODIFIED 3 XOR9 PATTERN SHOULD BE RUN
2184 004120 005737 007116          TST      @#PARAVA          ;BRANCH IF PARITY MEMORY NOT AVAIL
2185 004124 001406                   BEQ      DONE14
2186 004126 005737 003362          TST      @#PARPAT          ;BRANCH IF MODIFIED PATTERN JUST PJM
2187 004132 001003                   BNE     DONE14
2188 004134 010667 177222          MOV      SP,PARPAT          ;SET INDICATOR TO WRITE MODIFIED PAT
2189 004140 000753                   BR       DONE13             ;LOOP TEST USING MODIFIED PATTERN
2190          ;WRITE 8 XOR 13 TEST PATTERN IN LOCATIONS 000000-037777
2191 004142 010701 000000 000000  DONE14: MOV      PC,R1          ;UPDATE RESTART & LOOP ADDRESS
2192 004144 005046                   CLR      -(SP)              ;PUSH STARTING BANK # ON THE STACK
2193 004146 012746 000002          MOV      #2,-(SP)           ;PUSH # OF BANKS TO WRITE ON THE STACK
2194 004152 004767 004572          JSR      PC,..8X13          ;GO TO ROUTINE TO WRITE DATA
2195
2196          ;CHECK 8 XOR 13 PATTERN WRITTEN ABOVE
2197 004156 010701 000000 000000  MOV      PC,R1          ;UPDATE RESTART & LOOP ADDRESS
2198 004160 005046                   CLR      -(SP)              ;PUSH BANK # ON THE STACK
2199 004162 012746 000002          MOV      #2,-(SP)           ;AND # OF BANKS TO CHECK (2)
2200 004166 004767 004634          JSR      PC,..8X13          ;GO TO CHECK ROUTINE
2201
2202          ;RELOCATE PROGRAM BACK TO LOWER MEMORY
2203 004172 012767 040000 000022  DONE15: MOV      #40000,1$
2204 004200 022767 000040 177670  CMP      #40,BLKCNT
2205 004206 001402                   BEQ      .+6
2206 004210 006367 000006          ASL      1$
2207 004214 010701 000000 000000  MOV      PC,R1          ;UPDATE TRACE REGISTER
2208 004216 004567 002126          JSR      RS,RELOC          ;MOVE PROGRAM BACK INTO LOWER
2209 004222 040000 000000 000000  1$:      40000            ;FROM 40000 OR 100000
2210 004224 000000 000000 000000  ;TO 000000
2211 004226 000137 004232 000000  JMP      @#. +4            ;RETURN UNRELOCATED
2212 004232 012706 000500 000500  MOV      #STKPTR,SP        ;RESET STACK POINTER
2213 004236 042737 100000 000750  BIC      #10000,@#LDDISP    ;CLEAR RELOCATION INDICATOR
2214 004244 013737 000750 177570  MOV      @#LDDISP,@#DISPLAY ;LOAD DISPLAY REGISTER
2215
2216          .SBTTL RANDOM DATA ROTATING I/O TESTS
2217          ;RANDOM DATA TEST. THIS TEST MOVES THE PROGRAM CODE THROUGHOUT MEMCP.
2218 004252 010701 004412 000004  RANTST: MOV      PC,R1          ;SET TRACE POINTER
2219 004254 012737 004412 000004  MOV      #7$,@#ERRVEC      ;SET TIME OUT TRAP
2220 004262 005767 174464          TST      MMAVA             ;CHECK IF MEM MGMT IS AVAILABLE
2221 004266 001412                   BEQ      1$                 ;BRANCH IF NOT AVAILABLE
2222 004270 004767 002652          JSR      PC,LDMMO          ;GO SET UP MEM MGMT
2223 004274 105237 172301          INCB     @#KIPDR0+1         ;ALLOW 4K ADDRESSING IN FIRST 4K
2224 004300 012737 077406 172304  MOV      #200*256.-400+UP+RW,@#KIPDR2 ;SET KIPDR2=RW JP 200 3L00+5
2225 004306 012737 000400 172344  MOV      #400,@#KIPAR2
2226 004314 012702 020000 1$:      MOV      #20000,R2        ;SET 'TO' ADDRESS POINTER
2227 004320 005004                   CLR      R4                 ;SET 'FROM' ADDRESS POINTER
2228 004322 012705 004000 2$:      MOV      #2048,R5          ;SET 4K WORD COUNT
2229 004326 012422 3$:      MOV      (R4)+,(R2)+        ;MOVE CODE
2230 004330 012422
2231 004332 005305                   DEC      R5                 ;DECREMENT 4K WORD COUNTER
2232 004334 001374                   BNE     3$
2233
2234 004336 012705 005477 4$:      MOV      #4096.-PLACE+1,R5 ;SET 4K WORD COUNTER
2235 004342 01440C                   MOV      -(R4),R0           ;GET 'GOOD' DATA
2236 004344 014203                   MOV      -(R2),R3           ;GET 'BAD' DATA

```

```

2237 004346 020003          CMP      R0,R3          ;COMPARE 'GOOD' & 'BAD' DATA
2238 004350 001403          BEQ      5$
2239 004352 005722          TST      (R2)+          ;STEP ADDRESS FOR ERROR ROUTINE
2240 004354 104400          HLT
2241 004356 005742          TST      -(R2)         ;REPORT ERROR
2242 004360 005305          5$: DEC     R5           ;RESTORE ADDRESS POINTER
2243 004362 001367          BNE      4$           ;DECREMENT 4K WORD COUNTER
2244
2245 004364 005767 174362          TST      MAVA          ;CHECK IF MEM MGMT IS AVAILABLE
2246 004370 001405          BEQ      6$           ;BRANCH IF NOT AVAILABLE
2247 004372 005237 172342          INC     @#KIPAR1
2248 004376 005237 172344          INC     @#KIPAR2
2249 004402 000744          BR      1$
2250 004404 062702 000100          6$: ADD     @64.,R2      ;STEP ADDRESS
2251 004410 000744          BR      2$
2252 004412 012706 000500          7$: MOV     @STKPTR,SP   ;RESET STACK PTR
2253 004416 012737 001116 000004          MOV     @ERRTRP,@#ERRVEC;RESTORE ERROR TRAP VECTOR
2254
2255          ;ROTATING 0 TEST. THIS TEST ROTATES A SINGLE '0' THROUGH MEMORY
2256 004424 012767 177777 001052          ROT0: MOV     @-1,CONST ;SET CONSTANT =177777
2257 004432 012746 000001          MOV     @1, -(SP)      ;SET BANK #1
2258 004436 016746 176766          MOV     WDS.256, -(SP) ;GET # OF 256. WORD BLOCKS IN MEMORY
2259 004442 004767 005004          JSR     PC, USER      ;GO WRITE 1'S THROUGHOUT MEMORY
2260 004446 010701          MOV     PC,R1         ;SET SCOPE PTR
2261 004450 012746 000001          MOV     @1, -(SP)      ;SET STARTING BANK #
2262 004454 016746 176750          MOV     WDS.256, -(SP) ;SET # OF 256. WORD BLOCKS TO CHECK
2263 004460 004767 004536          JSR     PC, .ROT0     ;GO TO ROTATE 0 ROUTINE
2264
2265          ;ROTATING 1 TEST THIS TEST ROTATES A SINGLE '1' BIT THROUGH ALL OF
2266          ;MEMORY
2267 004464 005067 001014          ROT1: CLR     .CONST    ;CLEAR CONSTANT
2268 004470 012746 000001          MOV     @1, -(SP)      ;PUSH STARTING BANK ONTO STACK
2269 004474 016746 176730          MOV     WDS.256, -(SP) ;AND # OF 256. WORD BLOCKS IN MEMORY
2270 004500 004767 004746          JSR     PC, USER      ;GO WRITE 0'S THROUGHOUT MEMORY
2271 004504 010701          MOV     PC,R1         ;SET SCOPE PTR
2272 004506 012746 000001          MOV     @1, -(SP)      ;SET STARTING BANK #
2273 004512 016746 176712          MOV     WDS.256, -(SP) ;SET # OF 256. WORD BLOCKS TO CHECK
2274 004516 004767 004574          JSR     PC, .ROT1     ;GO ROTATE A '1' BIT THROUGHOUT MEMORY
2275
2276          ;END OF CYCLE
2277 004522 012767 000050 000072          END: MOV     @50,TEMPO
2278 004530 000005          1$: RESET
2279 004532 005367 000064          DEC     TEMPO
2280 004536 001374          BNE     1$
2281 004540 032767 000100 173022          BIT     @SW06,SWR
2282 004546 001407          BEQ     2$
2283 004550 005067 173474          CLR     EOPHLT
2284 004554 012767 000207 173470          MOV     @207,252
2285 004562 004767 173462          JSR     PC, EOPHLT
2286 004566 010701          2$: MOV     PC,R1
2287 004570 012706 000500          MOV     @STKPTR,SP    ;UPDATE TRACE REGISTER
2288 004574 005237 000742          INC     @#ICNT        ;SET STACK PTR
2289 004600 022737 000007 000742          CMP     @7,@#ICNT     ;INCREMENT PASS COUNT
2290 004606 001406          BEQ     DONE          ;8 PASSES?
2291 004610 004567 174144          JSR     R5,$PRINT     ;BRANCH IF 8 PASSES COMPLETED
2292 004614 012442          A$      ;GO TO PRINT ROUTINE
                ASTERISK
    
```

```

2293 004616 000137 003272          JMP      @#BEGIN1
2294
2295 004622 000000          TEMPO:  .WORD  0
2296
2297 004624          DONE:
2298 004624 004567 174130          JSR      RS, $PRINT      ;GO TO PRINT ROUTINE
2299 004630 012444          ENDMSG
2300 004632 105737 177564          TSTB    @#TPS           ;WAIT FOR BELL TO RING
2301 004636 100375          BPL     -4
2302 004640 013700 000042          MOV     @#42, RO        ;GET DECTAPE MONITOR RETURN ADDRESS
2303 004644 001407          BEQ     FINISH
2304 004646 004767 175256          JSR     PC, $RLDR       ;RESTORE MONITOR & LOADERS
2305 004652 000005          RESET
2306 004654 004710          LOGICAL: JSR    PC, (RO)  ;GO TO DECTAPE MONITOR
2307 004656 000240          NOP
2308 004660 000240          NOP
2309 004662 000240          NOP
2310 004664 000167 175456          FINISH: JMP     START1
2311
2312          .SBTTL  USER TESTS
2313          .SBTTL  PROGRAM # 2
2314          ;PROGRAM # 2
2315          ;THIS PROGRAM ALLOWS THE USER TO SELECT A STARTING 4K BANK #, # OF 4K
2316          ;BANKS TO TEST, AND A WORST CASE PATTERN TO WRITE AND CHECK.
2317 004670 005067 001124          PRG2:   CLR     PRG3FLG      ;CLEAR PROGRAM # 3 FLAG
2318          ;NOTE: PROGRAM 3 ENTERS PROGRAM # 2 HERE, WITH PRG3FLG = 1.
2319 004674 012706 000500          PRG2A:  MOV     #STKPTR, SP   ;SET STACK PTR
2320 004700 005037 006622          CLR     @#PEFLG          ;CLEAR PARITY ERROR INDICATORS
2321 004704 005027          CLR     (PC)+            ;CLEAR RUNNING ALL PATTERNS FLAG
2322 004706 000000          ALLFLG: .WORD  0         ;CONTAINS RUNNING ALL PATERNS FLAG
2323          ;0/-1 = RUNNING SELECTED/ALL PATTERNS
2324 004710 012737 000502 000024          MOV     #PDWN, @#PFVEC    ;SET POWER FAIL TRAP VECTOR
2325 004716 012737 000006 000004          MOV     #ERRVEC+2, @#ERRVEC
2326 004724 005067 174022          CLR     MMAVA
2327 004730 000412          BR     1$                ;TELCO SYSTEM DOES NOT HAVE MEM. MANAGEMENT
2328          ;HENCE SW12 IS BEING DISCONNECTED.
2329          ;CAN BE RECONNECTED BY DELETING
2330          ;THIS BR INSTRUCTION.
2331 004732 032737 010000 177570          BIT     @#BIT12, @#SWR    ;CHECK IF TEST IS TO BE RUN
2332 004740 001006          BNE    1$                ;WITH MEM MGMT ENABLED
2333 004742 000261          SEC
2334 004744 005737 177572          TST     @#SRO            ;CHECK IF MEM MGMT IS AVAILABLE
2335 004750 103402          BCS    1$                ;BRANCH IF NOT AVAILABLE
2336 004752 005167 173774          COM     MMAVA
2337 004756 012737 001116 000004 1$:          MOV     #ERRTRP, @#ERRVEC
2338 004764 005037 000742          CLR     @#ICNT
2339 004770 005037 000750          CLR     @#LDDISP
2340 004774 005037 003362          CLR     @#PARPAT        ;SET INDICATOR TO WRITE NORM 3X9 PAT
2341 005000 032767 000002 172562          BIT     @#SW01, SWR      ;IS THERE A TTY?
2342 005006 001414          BEQ     5$
2343 005010 000000          HALT
2344 005012 005767 000034          TST     .PARIT
2345 005016 001402          BEQ     3$
2346 005020 004767 002022          JSR     PC, .MAMF
2347 005024 005767 000050          3$:    TST     .STBANK
2348 005030 001002          BNE     4$
    
```

| | | | | | | | |
|------|--------|--------|--------|----------|--------|--------------|----------------------------------------|
| 2349 | 005032 | 004767 | 001414 | | JSR | PC,RELOC | |
| 2350 | 005036 | 000442 | | 4\$: | BR | PRG2R | |
| 2351 | 005040 | 004567 | 173714 | 5\$: | JSR | RS,\$SPRINT | ;GO TO PRINT ROUTINE |
| 2352 | 005044 | 012120 | | | PARITY | | |
| 2353 | 005046 | 004767 | 004616 | | JSR | PC,RECD | ;GO GET ANSWER TO THE PARITY QUESTION |
| 2354 | 005052 | 000000 | | .PARIT: | .WORD | 0 | ;TYPE 1 IF PARITY DESIRED 0 IF NOT |
| 2355 | 005054 | 005767 | 177772 | | TST | .PARIT | |
| 2356 | 005060 | 001402 | | | BEQ | 1\$ | ;BRANCH IF PARITY NOT DESIRED |
| 2357 | 005062 | 004767 | 001760 | | JSR | PC,.MAMF | ;GO ENABLE PARITY ACTION |
| 2358 | 005066 | | | 1\$: | | | |
| 2359 | 005066 | 004567 | 173666 | | JSR | RS,\$SPRINT | ;GO TO PRINT ROUTINE |
| 2360 | 005072 | 012155 | | | STBANK | | ;ASK USER FOR STARTING BANK |
| 2361 | 005074 | 004767 | 004570 | | JSR | PC,RECD | |
| 2362 | 005100 | 000000 | | .STBANK: | .WORD | 0 | ;CONTAINS STARTING BANK # |
| 2363 | 005102 | 005767 | 177772 | | TST | .STBANK | ;CHECK IF STARTING AT BANK #0 |
| 2364 | 005106 | 001002 | | | BNE | 1\$ | |
| 2365 | 005110 | 004767 | 001336 | | JSR | PC,RELOC | ;GO RELOCATE THE PROGRAM TO TOP OF MEM |
| 2366 | 005114 | | | 1\$: | | | |
| 2367 | 005114 | 004567 | 173640 | | JSR | RS,\$SPRINT | ;GO TO PRINT ROUTINE |
| 2368 | 005120 | 012204 | | | BANKS | | ;ASK USER FOR # OF 4K BANKS TO TEST |
| 2369 | 005122 | 004767 | 004542 | | JSR | PC,RECD | |
| 2370 | 005126 | 000000 | | .BANKS: | .WORD | 0 | ;CONTAINS # OF BANKS TO CHECK |
| 2371 | 005130 | 004567 | 173624 | | JSR | RS,\$SPRINT | ;GO TO PRINT ROUTINE |
| 2372 | 005134 | 012241 | | | PAT | | ;ASK USER WHICH PATTERN |
| 2373 | 005136 | 004767 | 004526 | | JSR | PC,RECD | |
| 2374 | 005142 | 000000 | | .PAT: | .WORD | 0 | ;CONTAINS PATTERN # |
| 2375 | 005144 | 012767 | 005144 | 175210 | PRG2R: | MOV | #PRG2R,PERSTR |
| 2376 | 005152 | 004767 | 004574 | | JSR | PC,CKSWR | ;SET PAR ERROR RESTART ADDRESS |
| 2377 | 005156 | 016700 | 177760 | | MOV | .PAT,RO | ;GO CHECK SWITCHES |
| 2378 | 005162 | 006300 | | | ASL | RO | ;SHIFT PATTERN # TO FORM INDEX |
| 2379 | 005164 | 066700 | 173564 | | ADD | RELOC,RO | ;ADD RELOCATION FACTOR |
| 2380 | 005170 | 016000 | 005202 | | MOV | WRTTAB(0),RO | ;GET UNRELOCATED PC OF ROUTINE |
| 2381 | 005174 | 066700 | 173554 | | ADD | RELOC,RO | ;ADD RELOCATION FACTOR |
| 2382 | 005200 | 010007 | | | MOV | RO,PC | ;GO TO APPROPRIATE ROUTINE |
| 2383 | | | | | | | |
| 2384 | | | | | | | |
| 2385 | 005202 | 005222 | | | | | |
| 2386 | 005204 | 005312 | | | | | |
| 2387 | 005206 | 005400 | | | | | |
| 2388 | 005210 | 005472 | | | | | |
| 2389 | 005212 | 005600 | | | | | |
| 2390 | 005214 | 005674 | | | | | |
| 2391 | 005216 | 005770 | | | | | |
| 2392 | 005220 | 006000 | | | | | |
| 2393 | | | | | | | |
| 2394 | | | | | | | |
| 2395 | 005222 | 016746 | 177652 | | | | |
| 2396 | 005226 | 016746 | 177674 | | | | |
| 2397 | 005232 | 006316 | | | | | |
| 2398 | 005234 | 005767 | 000560 | | | | |
| 2399 | 005240 | 001004 | | | | | |
| 2400 | 005242 | 006316 | | | | | |
| 2401 | 005244 | 006316 | | | | | |
| 2402 | 005246 | 006316 | | | | | |
| 2403 | 005250 | 006316 | | | | | |
| 2404 | 005252 | 011627 | | | | | |

| | | | |
|-----------------------------------------------|-------|--------|-----------------------------|
| :TABLE OF ROUTINES TO WRITE SELECTED PATTERNS | | | |
| WRTTAB: | .WORD | \$1X8 | ;1 XOR 8 ROUTINE (0) |
| | .WORD | \$3X9 | ;3 XOR 9 ROUTINE (1) |
| | .WORD | \$8X13 | ;8 XOR 13 ROUTINE (2) |
| | .WORD | \$USER | ;USER ROUTINE (3) |
| | .WORD | \$ROTO | ;ROTATING '0' ROUTINE (4) |
| | .WORD | \$ROT1 | ;ROTATING '1' ROUTINE (5) |
| | .WORD | \$3X9P | ;PARITY 3 XOR 9 PATTERN (6) |
| | .WORD | \$ALL | ;ALL EXCEPT USER (7) |

| | | | |
|-----------|-----|---------------|--------------------------------|
| :ROUTINES | | | |
| \$1X8: | MOV | .STBANK,-(SP) | ;GET STARTING BANK # |
| | MOV | .BANKS,-(SP) | ;GET # OF 4K BANKS |
| | ASL | (SP) | ;MULTIPLY BY 32. |
| | TST | PRG3FLG | ;IF PROGRAM # 3 STOP WITH 255. |
| | BNE | 1\$ | ;WORD BLOCK COUNT |
| | ASL | (SP) | ;TO FORM 128. |
| | ASL | (SP) | ;WORD BLOCK |
| | ASL | (SP) | ;COUNT |
| 1\$: | MOV | (SP),(PC)+ | ;SAVE |

| | | | | | | | |
|------|--------|--------|--------|---------|-------|---------------|---------------------------------------|
| 2405 | 005254 | 000000 | | 2\$: | WORD | 0 | ;CONTAINS 128. WORD BLOCK COUNT |
| 2406 | 005256 | 004767 | 002230 | | JSR | PC,.1X8 | ;GO WRITE 1 XOR 8 TEST PATTERN |
| 2407 | | | | | | | |
| 2408 | 005262 | 016746 | 177612 | | MOV | .STBANK,-(SP) | ;GET STARTING BANK # |
| 2409 | 005266 | 016746 | 177762 | | MOV | 2\$,-(SP) | ;GET 128. WORD BLOCK COUNT |
| 2410 | 005272 | 004767 | 002326 | | JSR | PC,.1X8 | ;GO CHECK PATTERN |
| 2411 | 005276 | 005267 | 173440 | | INC | ICNT | |
| 2412 | 005302 | 005767 | 177400 | | TST | ALLFLG | ;CHECK IF RUNNING ALL PATERNS |
| 2413 | 005306 | 001001 | | | BNE | \$3X9 | ;GO START 3X9 PATTERN IF RUNNING ALL |
| 2414 | 005310 | 000744 | | | BR | \$1X8 | ;OTHERWISE LOOP |
| 2415 | | | | | | | |
| 2416 | 005312 | 016746 | 177562 | \$3X9: | MOV | .STBANK,-(SP) | ;GET STARTING BANK # |
| 2417 | 005316 | 016746 | 177604 | | MOV | .BANKS,-(SP) | ;GET # OF BANKS |
| 2418 | 005322 | 005767 | 000472 | | TST | PRG3FLG | ;IF PROGRAM # 2 STOP WITH 256. |
| 2419 | 005326 | 001004 | | | BNE | 1\$ | ;WORD BLOCK COUNT |
| 2420 | 005330 | 006316 | | | ASL | (SP) | ;MULTIPLY BY 16. |
| 2421 | 005332 | 006316 | | | ASL | (SP) | ;TO FORM |
| 2422 | 005334 | 006316 | | | ASL | (SP) | ;256. WORD |
| 2423 | 005336 | 006316 | | | ASL | (SP) | ;BLOCK COUNT |
| 2424 | 005340 | 011627 | | 1\$: | MOV | (SP),(PC)+ | ;SAVE |
| 2425 | 005342 | 000000 | | 2\$: | WORD | 0 | ;CONTAINS 256. WORD BLOCK COUNT |
| 2426 | 005344 | 004767 | 002510 | | JSR | PC,.3X9 | ;GO WRITE PATTERN |
| 2427 | | | | | | | |
| 2428 | | | | | | | |
| 2429 | 005350 | 016746 | 177524 | | | | |
| 2430 | 005354 | 016746 | 177762 | 3\$: | MOV | .STBANK,-(SP) | ;GET STARTING BANK # |
| 2431 | 005360 | 004767 | 002714 | | MOV | 2\$,-(SP) | ;GET # OF 256. WORD BLOCKS |
| 2432 | 005364 | 005267 | 173352 | | JSR | PC,.3X9 | ;GO CHECK PATTERN |
| 2433 | 005370 | 005767 | 177312 | | INC | ICNT | |
| 2434 | 005374 | 001001 | | | TST | ALLFLG | ;CHECK IF RUNNING ALL PATTERNS |
| 2435 | 005376 | 000764 | | | BNE | \$8X13 | ;GO START 8X13 PATTERN IF RUNNING ALL |
| 2436 | | | | | BR | 3\$ | |
| 2437 | 005400 | 005737 | 006020 | \$8X13: | TST | PRG3FLG | ;CANNOT DO 8X13 PATTERN USING |
| 2438 | 005404 | 001007 | | | BNE | 1\$ | ;PROGRAM # 3 |
| 2439 | 005406 | 016746 | 177466 | | MOV | .STBANK,-(SP) | ;GET STARTING BANK # |
| 2440 | 005412 | 016746 | 177510 | | MOV | .BANKS,-(SP) | ;AND # OF 4K BANKS |
| 2441 | 005416 | 032716 | 000001 | | BIT | #1,(SP) | ;MUST BE AN EVEN # OF 4K BANKS |
| 2442 | 005422 | 001405 | | | BEG | 2\$ | |
| 2443 | 005424 | | | 1\$: | | | |
| 2444 | 005424 | 004567 | 173330 | | JSR | R5,\$PRINT | ;GO TO PRINT ROUTINE |
| 2445 | 005430 | 012257 | | | QUEST | | ;PRINT ? |
| 2446 | 005432 | 000167 | 177232 | | JMP | PRG2 | ;RESTART |
| 2447 | | | | | | | |
| 2448 | 005436 | 004767 | 003306 | 2\$: | JSR | PC,.8X13 | ;GO WRITE PATTERN |
| 2449 | 005442 | 016746 | 177432 | | MOV | .STBANK,-(SP) | |
| 2450 | 005446 | 016746 | 177454 | | MOV | .BANKS,-(SP) | |
| 2451 | 005452 | 004767 | 003350 | | JSR | PC,.8X13 | ;GO CHECK PATTERN |
| 2452 | 005456 | 005267 | 173260 | | INC | ICNT | |
| 2453 | 005462 | 005767 | 177220 | | TST | ALLFLG | ;CHECK IF RUNNING ALL PATERNS |
| 2454 | 005466 | 001044 | | | BNE | \$ROTO | ;GO DO ROTO PATTERN IF ALL SELECTED |
| 2455 | 005470 | 000743 | | | BR | \$8X13 | |
| 2456 | | | | | | | |
| 2457 | | | | | | | |
| 2458 | | | | | | | |
| 2459 | 005472 | | | | | | |
| 2460 | 005472 | 004567 | 173262 | | JSR | R5,\$PRINT | ;GO TO PRINT ROUTINE |

;ROUTINE TO WRITE & CHECK USER CONSTANT

\$USER: JSR R5,\$PRINT ;GO TO PRINT ROUTINE

| | | | | | | | |
|------|--------|--------|--------|--------|--------------|---------------|---------------------------------------|
| 2461 | 005476 | 012263 | | | CONST | | ;ASK FOR USER CONSTANT |
| 2462 | 005500 | 004767 | 004164 | | JSR | PC,RECD | |
| 2463 | 005504 | 000000 | | | CONST: .WORD | 0 | ;CONTAINS USER CONSTANT |
| 2464 | 005506 | 016746 | 177366 | | i\$: MOV | .STBANK,-(SP) | ;GET STARTING BANK # |
| 2465 | 005512 | 016746 | 177410 | | MOV | .BANKS,-(SP) | ;GET 4K COUNT |
| 2466 | 005516 | 005767 | 000276 | | TST | PRG3FLG | |
| 2467 | 005522 | 001004 | | | BNE | 2\$ | |
| 2468 | 005524 | 006316 | | | ASL | (SP) | ;MULTIPLY 4K BANK COUNT BY 16. |
| 2469 | 005526 | 006316 | | | ASL | (SP) | ;TO FORM 256. WORD BLOCK COUNT |
| 2470 | 005530 | 006316 | | | ASL | (SP) | |
| 2471 | 005532 | 006316 | | | ASL | (SP) | |
| 2472 | 005534 | 011627 | | | 2\$: MOV | (SP),(PC)+ | ;SAVE |
| 2473 | 005536 | 000000 | | | 3\$: .WORD | 0 | |
| 2474 | 005540 | 004767 | 003706 | | JSR | PC,USER | ;GO WRITE USER CONSTANT |
| 2475 | 005544 | 016746 | 177330 | | MOV | .STBANK,-(SP) | ;GET STARTING BANK # |
| 2476 | 005550 | 016746 | 177762 | | MOV | 3\$,-(SP) | ;AND # OF 256. WORD BLOCKS |
| 2477 | 005554 | 004767 | 003744 | | JSR | PC,USER | ;GO TO USER CHECK ROUTINE |
| 2478 | 005560 | 005267 | 173156 | | INC | ICNT | |
| 2479 | 005564 | 105737 | 177560 | | TSTB | @TKS | ;CHECK IF USER HAS TYPED A CHARACTER |
| 2480 | 005570 | 100346 | | | BPL | 1\$ | |
| 2481 | 005572 | 005737 | 177562 | | TST | @TKB | ;CLEAR FLAG |
| 2482 | 005576 | 000735 | | | BR | \$USER | |
| 2483 | | | | | | | |
| 2484 | | | | | | | |
| 2485 | 005600 | 016746 | 177274 | | \$ROTO: MOV | .STBANK,-(SP) | ;GET STARTING BANK # |
| 2486 | 005604 | 016746 | 177316 | | MOV | .BANKS,-(SP) | ;GET # OF BANKS |
| 2487 | 005610 | 005767 | 000204 | | TST | PRG3FLG | |
| 2488 | 005614 | 001004 | | | BNE | 2\$ | |
| 2489 | 005616 | 006316 | | | ASL | (SP) | ;MULTIPLY 4K BANK COUNT BY 16. |
| 2490 | 005620 | 006316 | | | ASL | (SP) | ;TO FORM 256. WORD BLOCK COUNT |
| 2491 | 005622 | 006316 | | | ASL | (SP) | |
| 2492 | 005624 | 006316 | | | ASL | (SP) | |
| 2493 | 005626 | 011627 | | | 2\$: MOV | (SP),(PC)+ | ;SAVE |
| 2494 | 005630 | 000000 | | | 3\$: .WORD | 0 | ;CONTAINS 256. WORD BLOCK COUNT |
| 2495 | 005632 | 012767 | 177777 | 177644 | MOV | -1,CONST | ;SET CONSTANT |
| 2496 | 005640 | 004767 | 003606 | | JSR | PC,USER | ;GO WRITE ALL 1'S THROUGH MEMORY |
| 2497 | 005644 | 016746 | 177230 | | MOV | .STBANK,-(SP) | ;GET STARTING BANK # |
| 2498 | 005650 | 016746 | 177754 | | MOV | 3\$,-(SP) | ;GET # OF 256. WORD BLOCKS TO CHECK |
| 2499 | 005654 | 004767 | 003342 | | JSR | PC,ROTO | ;GO CHECK ROTATING 0 PATTERN |
| 2500 | 005660 | 005267 | 173056 | | INC | ICNT | ;INCREMENT DISPLAY COUNT |
| 2501 | 005664 | 005767 | 177016 | | TST | ALLFLG | ;CHECK IF RUNNING ALL PATTERNS |
| 2502 | 005670 | 001001 | | | BNE | \$ROT1 | ;GO TO \$ROT1 IF RUNNING ALL PATTERNS |
| 2503 | 005672 | 000742 | | | BR | \$ROTO | ;LOOP |
| 2504 | | | | | | | |
| 2505 | 005674 | 016746 | 177200 | | \$ROT1: MOV | .STBANK,-(SP) | ;GET STARTING BANK # |
| 2506 | 005700 | 016746 | 177222 | | MOV | .BANKS,-(SP) | ;GET # OF 4 K BANKS |
| 2507 | 005704 | 005767 | 000110 | | TST | PRG3FLG | ;CHECK IF RUNNING PROGRAM 3 |
| 2508 | 005710 | 001004 | | | BNE | 2\$ | ;BRANCH IF RUNNING PROGRAM 3 |
| 2509 | 005712 | 006316 | | | ASL | (SP) | ;SHIFT 4K BANK COUNT BY 16. |
| 2510 | 005714 | 006316 | | | ASL | (SP) | ;TO FORM 256. WORD BLOCK COUNT |
| 2511 | 005716 | 006316 | | | ASL | (SP) | |
| 2512 | 005720 | 006316 | | | ASL | (SP) | |
| 2513 | 005722 | 011627 | | | 2\$: MOV | (SP),(PC)+ | ;SAVE |
| 2514 | 005724 | 000000 | | | 3\$: .WORD | 0 | ;CONTAINS 256. WORD BLOCK COUNT |
| 2515 | 005726 | 005067 | 177552 | | CLR | CONST | ;SET CONSTANT |
| 2516 | 005732 | 004767 | 003514 | | JSR | PC,USER | ;GO WRITE 0'S THROUGHOUT |

| | | | | | | |
|------|--------|--------|--------|----------------|--------------------|----------------------------------------------------------------------|
| 2517 | 005736 | 016746 | 177136 | MOV | .STBANK, -(SP) | ; GET STARTING BANK # |
| 2518 | 005742 | 016746 | 177756 | MOV | 35, -(SP) | ; AND 256 WORD BLOCK COUNT |
| 2519 | 005746 | 004767 | 003344 | JSR | PC, ROT1 | ; GO CHECK ROTATING 1 PATTERN |
| 2520 | 005752 | 005267 | 172764 | INC | ICNT | ; INCREMENT PASS COUNT |
| 2521 | 005756 | 005767 | 176724 | TST | ALLFLG | ; CHECK IF RUNNING ALL PATTERNS |
| 2522 | 005762 | 001744 | | BEQ | \$ROT1 | ; LOOP IF NOT RUNNING ALL PATTERNS |
| 2523 | 005764 | 000167 | 177232 | JMP | \$1X8 | ; GO DO \$1X8 PATTERN |
| 2524 | | | | | | |
| 2525 | | | | | | ; ROUTINE TO CHECK MEMORY USING 3 XOR 9 PARITY PATTERN |
| 2526 | 005770 | 010667 | 175366 | \$3X9P: MOV | SP, PARPAT | ; SET INDICATOR TO WRITE PARITY PATTERN |
| 2527 | 005774 | 000167 | 177312 | JMP | \$3X9 | |
| 2528 | | | | | | |
| 2529 | | | | | | ; ALL PATTERNS |
| 2530 | 006000 | 010667 | 176702 | \$ALL: MOV | SP, ALLFLG | ; SET INDICATOR |
| 2531 | 006004 | 000167 | 177212 | JMP | \$1X8 | ; BEGIN WITH 1X8 TEST PATTERN |
| 2532 | | | | | | .SBTTL PROGRAM # 3 |
| 2533 | | | | | | ; THIS PROGRAM IS THE SAME AS PROGRAM # 2 ABOVE EXCEPT THAT 256 WORD |
| 2534 | | | | | | ; DATA BLOCKS MAY BE WRITTEN |
| 2535 | 006010 | 012706 | 000500 | PRG3: MOV | #STKPTR, SP | ; SET STACK PTR |
| 2536 | 006014 | 012727 | 000001 | MOV | #1, (PC)+ | ; SET PROGRAM 3 FLAG |
| 2537 | 006020 | 000000 | | PRG3FLG: WORD | 0 | ; CONTAINS PRG3 INDICATOR |
| 2538 | 006022 | 004567 | 172732 | JSR | R5, \$PRINT | ; GO TO PRINT ROUTINE |
| 2539 | 006026 | 012303 | | PRG3M | | |
| 2540 | 006030 | 004567 | 172724 | JSR | R5, \$PRINT | ; GO TO PRINT ROUTINE |
| 2541 | 006034 | 012204 | | BANKS | | |
| 2542 | 006036 | 000167 | 176632 | JMP | PRG2A | ; GO TO PROGRAM 2 |
| 2543 | | | | | | |
| 2544 | | | | | | .SBTTL PROGRAM # 4 |
| 2545 | | | | | | ; THIS PROGRAM MAY BE USED TO READ/WRITE A USER CONSTANT INTO ANY |
| 2546 | | | | | | ; MEMORY LOCATION |
| 2547 | 006042 | 012706 | 000500 | PRG4: MOV | #STKPTR, SP | ; SET STACK PTR |
| 2548 | 006046 | 005037 | 006622 | CLR | @#PEFLG | ; CLEAR PARITY ERROR INDICATORS |
| 2549 | 006052 | 012737 | 001116 | MOV | @#ERRTRP, @#ERRVEC | |
| 2550 | 006060 | 000005 | | RESET | | |
| 2551 | 006062 | 012707 | 006066 | MOV | #.+4, PC | ; RELOCATE BACK TO FIRST 4K |
| 2552 | 006066 | 004567 | 172666 | JSR | R5, \$PRINT | ; GO TO PRINT ROUTINE |
| 2553 | 006072 | 012364 | | PRG4M | | |
| 2554 | 006074 | 004767 | 003570 | JSR | PC, RECD | |
| 2555 | 006100 | 000000 | | \$ADRSO: .WORD | 0 | ; SONTAINS ADDRESS BITS <15-0> |
| 2556 | 006102 | 016767 | 003634 | MOV | .1617, .EXTAD | ; GET EXTENDED ADDRESS BITS <17-16> |
| 2557 | 006110 | 006037 | 177570 | ROR | @#SWR | ; CHECK SWITCH 0 |
| 2558 | 006114 | 103406 | | BCS | PRG4A | ; GO TO PRG4A IF SET |
| 2559 | 006116 | 004567 | 172636 | JSR | R5, \$PRINT | ; GO TO PRINT ROUTINE |
| 2560 | 006122 | 012263 | | CONST | | |
| 2561 | 006124 | 004767 | 003540 | JSR | PC, RECD | |
| 2562 | 006130 | 000000 | | \$CONST: .WORD | 0 | ; CONTAINS USER CONSTANT |
| 2563 | 006132 | 012767 | 006132 | PRG4A: MOV | #PRG4A, PERSTRT | ; SET RESTART ADDRESS ON PAR ERROR |
| 2564 | 006140 | 016703 | 177734 | MOV | \$ADRSO, R3 | ; GET ADDRESS BITS <15-0> |
| 2565 | 006144 | 016704 | 000176 | MOV | .EXTAD, R4 | ; GET ADDRESS BITS <17-16> |
| 2566 | 006150 | 001020 | | BNE | 10\$ | ; BRANCH IF MEM MGMT REQUIRED |
| 2567 | 006152 | 022703 | 020000 | CMP | #20000, R3 | ; CHECK IF ADDRESS IS LESS THAN 20000 |
| 2568 | 006156 | 101412 | | BLOS | 1\$ | ; BRANCH IF IT IS NOT |
| 2569 | 006160 | 006037 | 177570 | ROR | @#SWR | ; CHECK IF READING |
| 2570 | 006164 | 103431 | | BCS | 3\$ | ; BRANCH IF READING |
| 2571 | 006166 | 004767 | 172112 | JSR | PC, \$SAVR | ; GO SAVE REGISTERS ON THE STACK |
| 2572 | 006172 | 004767 | 000254 | JSR | PC, RELOCP | ; GO RELOCATE PROGRAM |

| | | | | | | | |
|------|--------|--------|--------|---------|----------|--------------------------|-----------------------------------------|
| 2573 | 006176 | 004767 | 172126 | | JSR | PC,\$RESTR | ;RESTORE REGISTERS FROM STACK |
| 2574 | 006202 | 000422 | | | BR | 3\$ | ;GO TO 3\$ |
| 2575 | 006204 | 022703 | 160000 | 1\$: | CMP | #160000,R3 | ;CHECK IF MEM MGMT WILL BE REQUIRED |
| 2576 | 006210 | 101017 | | | BHI | 3\$ | ;GO TO 3\$ IF NOT REQUIRED |
| 2577 | 006212 | 010302 | | 10\$: | MOV | R3,R2 | ;GET ADDRESS BITS <15-0> |
| 2578 | 006214 | 012700 | 000006 | | MOV | #6,R0 | ;SET SHIFT COUNT |
| 2579 | 006220 | 006204 | | 2\$: | ASR | R4 | ;SHIFT 18 BIT ADDRESS |
| 2580 | 006222 | 006003 | | | ROR | R3 | ;6 PLACES RIGHT |
| 2581 | 006224 | 077003 | | | SOB | R0,2\$ | |
| 2582 | 006226 | 004767 | 000714 | | JSR | PC,LDMMO | ;GO SETUP MEM MGMT |
| 2583 | 006232 | 010337 | 172342 | | MOV | R3,#KIPARI | ;SET KIPARI |
| 2584 | 006236 | 042702 | 177700 | | BIC | #177700,R2 | ;CLEAR ADDRESS BITS <15-6> |
| 2585 | 006242 | 052702 | 020000 | | BIS | #20000,R2 | ;SET ADDRESS REGISTER |
| 2586 | 006246 | 000401 | | | BR | 4\$ | ;GO TO 4\$ |
| 2587 | 006250 | 010302 | | 3\$: | MOV | R3,R2 | ;SET ADDRESS REGISTER |
| 2588 | 006252 | 016700 | 177652 | 4\$: | MOV | \$CONST,R0 | ;GET USER CONSTANT |
| 2589 | 006256 | 012737 | 006042 | 000060 | MOV | #PRG4,#TKVEC | ;SET KEYBOARD INTERRUPT VECTOR |
| 2590 | 006264 | 052737 | 000100 | 177560 | BIS | #100,#TKS | ;SET IE BIT IN KEYBOARD CSR |
| 2591 | 006272 | 006037 | 177570 | | ROR | #SWR | ;CHECK SWITCH 0 |
| 2592 | 006276 | 103014 | | | BCC | 5\$ | ;BRANCH IF NOT SET |
| 2593 | 006300 | 011246 | | | MOV | (R2),-(SP) | ;PUSH DATA TO BE TYPED ONTO STACK |
| 2594 | 006302 | 004767 | 173434 | | JSR | PC,02A | ;GO TYPE DATA |
| 2595 | 006306 | 004567 | 172446 | | JSR | R5,\$PRINT | ;GO TO PRINT ROUTINE |
| 2596 | 006312 | 015122 | | | \$CRLF | | |
| 2597 | 006314 | 062767 | 000002 | 177556 | ADD | #2,\$ADR50 | ;STEP ADDRESS |
| 2598 | 006322 | 005567 | 000020 | | ADC | .EXTAD | |
| 2599 | 006326 | 000701 | | | BR | PRG4A | |
| 2600 | 006330 | 010012 | | 5\$: | MOV | R0,(R2) | ;WRITE USER CONSTANT INTO ADDRESS |
| 2601 | 006332 | 012203 | | | MOV | (R2)+,R3 | ;GET DATA WRITTEN |
| 2602 | 006334 | 020003 | | | CMP | R0,R3 | ;CHECK DATA |
| 2603 | 006336 | 001401 | | | BEQ | 6\$ | |
| 2604 | 006340 | 104400 | | | HLT | | ;REPORT ERROR |
| 2605 | 006342 | 005742 | | 6\$: | TST | -(R2) | ;RESTORE ADDRESS |
| 2606 | 006344 | 000771 | | | BR | 5\$ | ;LOOP BACK |
| 2607 | 006346 | 000000 | | .EXTAD: | .WORD | 0 | ;CONTAINS EXTENDED ADDRESS BITS |
| 2608 | | | | | .SBTTL | PROGRAM SUBROUTINES | |
| 2609 | | | | | .SBTTL | RELOCATION ROUTINES | |
| 2610 | | | | | .ROUTINE | TO RELOCATE PROGRAM CODE | |
| 2611 | 006350 | 012500 | | RELOC: | MOV | (R5)+,R0 | ;GET FROM ADDRESS |
| 2612 | 006352 | 011502 | | | MOV | (R5),R2 | ;GET TO ADDRESS |
| 2613 | 006354 | 010203 | | | MOV | R2,R3 | |
| 2614 | 006356 | 062703 | 017776 | | ADD | #17776,R3 | ;MOVES 4K |
| 2615 | 006362 | 012737 | 006432 | 000004 | MOV | #4\$,#ERRVEC | ;SET TIME OUT TRAP |
| 2616 | 006370 | 005004 | | | CLR | R4 | ;CLEAR RELOCATION SUCCESSFUL INDICATOR |
| 2617 | 006372 | 005723 | | | TST | (R3)+ | ;CHECK IF MEMORY IS AVAILABLE |
| 2618 | 006374 | 012022 | | 1\$: | MOV | (R0)+,(R2)+ | ;RELOCATE |
| 2619 | 006376 | 020203 | | | CMP | R2,R3 | ;RELOCATION COMPLETE? |
| 2620 | 006400 | 001375 | | | BNE | 1\$ | |
| 2621 | 006402 | 011503 | | | MOV | (R5),R3 | |
| 2622 | 006404 | 020203 | | 2\$: | CMP | R2,R3 | |
| 2623 | 006406 | 001413 | | | BEQ | 5\$ | ;BRANCH IF DONE |
| 2624 | 006410 | 024042 | | | CMP | -(R0),-(R2) | ;CHECK THAT DATA WAS RELOCATED PROPERLY |
| 2625 | 006412 | 001774 | | | BEQ | 2\$ | |
| 2626 | 006414 | 005703 | | | TST | R3 | ;CHECK IF RELOCATING BACK TO 000000 |
| 2627 | 006416 | 001403 | | | BEQ | 3\$ | |
| 2628 | 006420 | 104400 | | | HLT | | ;ERROR! CANNOT RELOCATE PROGRAM CODE |

```

2629                                     ;TO UPPER MEMORY BANK PROPERLY
2630 006422 000000                       HALT
2631 006424 000767                       BR      2$
2632 006426 000000                       3$: HALT
2633                                     ;CONTINUE RELOCATING AT YOUR PERIL
2634 006430 000777                       BR      ;ERROR! CANNOT RELOCATE CODE BACK TO
2635 006432 022626                       4$: CMP  (SP)+,(SP)+ ;TO 000000 PROPERLY
2636 006434 005104                       COM     R4
2637 006436 000240                       5$: NOP
2638 006440 012702 000754                MOV     #RELOC,F,R2 ;GET ADDRESS OF RELOCATION FACTOR
2639 006444 061502                       ADD     (R5),R2      ;ADD FACTOR
2640 006446 012512                       MOV     (R5)+,(R2)  ;RELOCATED RELOC F NOW CONTAINS RELOCATION
2641                                     ;FACTOR
2642 006450 000205                       RTS     5            ;RETURN, R4=-1 IF NO RELOCATION
2643
2644
2645 ;ROUTINE TO RELOCATE PROGRAM CODE FROM ORIGINAL POSITION (0-4K) TO
2646 ;TOP OF MEMORY.
2647 006452 012700 020000 000004          RELOCP: MOV  #20000,R0 ;SET UP TO SCAN FOR TOP OF MEMORY
2648 006456 012737 000006 000004          MOV  #ERRVEC+2,2#ERRVEC
2649 006464 062700 020000                  1$: ADD  #20000,R0 ;INCREMENT SCAN ADDRESS
2650 006470 000261                          SEC
2651 006472 005710                          TST  (R0) ;SET TIME OUT INDICATOR
2652 006474 103373                          BCC  1$ ;CHECK FOR EXISTANT MEMORY
2653 006476 012737 001116 000004          MOV  #ERRTRP,2#ERRVEC ;'C' WILL BE CLEAR IF MEMORY EXISTS
2654 006504 162700 020000                  SUB  #20000,R0 ;ADJUST TO LAST EXISTANT 4K
2655 006510 010067 000006                  MOV  R0,2$ ;PASS RELOCATION ADDRESS TO RELOC ROUTINE
2656 006514 004567 177630                  JSR  R5,RELOC ;RELOCATE PROGRAM
2657 006520 000000                          000000 ;FROM ADDRESS 000000
2658 006522 000000                          2$: .WORD 0 ;TO LAST 4K BANK
2659 006524 004567 172230                  JSR  R5,$PRINT ;GO TO PRINT ROUTINE
2660 006530 012403                          RELOCM
2661 006532 016746 177764                  MOV  2$,-(SP) ;PASS TO 02A ROUTINE
2662 006536 062716 012460                  ADD  #REL24K,(SP) ;SET UP RESTART ADDRESS
2663 006542 004767 173174                  JSR  PC,02A ;TYPE RESTART ADDRESS
2664 006546 011667 000006                  MOV  (SP),3$ ;SAVE RETURN ADDRESS IN 3$ BELOW
2665 006552 066706 177744                  ADD  2$,SP ;RESET STACK PTR
2666 006556 012716                          MOV  (PC)+,(SP) ;GET RETURN ADDRESS
2667 006560 000000                          3$: .WORD 0 ;CONTAINS RETURN PC
2668 006562 066716 177734                  ADD  2$,(SP) ;ADJUST RETURN PC
2669 006566 000207                          RTS     PC
2670
2671 ;SBTTL PARITY ERROR SERVICE ROUTINE
2672 ;WHEN A PARITY ERROR IS DETECTED THIS ROUTINE SCANS MEMORY FOR THE AD-
2673 ;DRESS CAUSING THE PARITY ERROR. WHEN THE ADDRESS IS LOCATED THE ROUTINE
2674 ;HALTS WITH THE ADDRESS+2 IN R0. TO CONTINUE AFTER THE ERROR PRESS CONTINUE.
2675 006570 010067 000170                  .PARSRV: MOV  R0,SAVRO ;SAVE R0 IN SAVRO
2676 006574 012700 006766                  MOV  #SAVRO+2,R0
2677 006600 010120                          MOV  R1,(R0)+
2678 006602 010220                          MOV  R2,(R0)+
2679 006604 010320                          MOV  R3,(R0)+
2680 006606 010420                          MOV  R4,(R0)+
2681 006610 010520                          MOV  R5,(R0)+
2682 006612 004567 172142                  JSR  R5,$PRINT ;GO TO PRINT ROUTINE
2683 006616 007000                          PARERR
2684 006620 005027                          CLR  (PC)+ ;CLEAR PARITY ERROR INDICATORS

```

```

2685 006622 000 FEFLG: .BYTE 0 ;NOT 0/0 =PAR ERR/NO PAR ERR
2686 006623 000 PENFLG: .BYTE 0 ;NOT 0/0=PAR ERR DETECTED/NOT DETECTED ON SCAN
2687 006624 012737 006672 000114 MOV #25,2#PARVEC ;SET PARITY ERROR TRAP
2688 006632 012737 006730 000004 MOV #45,2#ERRVEC ;SET TIME OUT TRAP VECTOR
2689 006640 005002 CLR R2
2690 006642 005767 172104 TST MMVA ;CHECK IF MEM MGMT IS AVAILABLE
2691 006646 001407 BEQ 1$ ;BRANCH IF NOT AVAILABLE
2692 006650 004767 000272 JSR PC,LDMMO ;SET UP MEM MGMT
2693 006654 105237 172301 INCB @#KIPDR0+! ;ALLOW FULL 4K PAGE ADDRESSING
2694 006660 012737 007244 000250 MOV #MMABTO,2#MMVEC ;SET MEM MGMT ABORT TRAP VECTOR
2695 006666 012200 1$: MOV (R2)+,R0 ;SCAN ALL ADDRESSES
2696 006670 00077 BR 1$
2697 006672 110667 177724 2$: MOVA SP,PEFLG ;SET PARITY ERROR FOUND INDICATOR
2698 006676 010003 MOV R0,R3
2699 006700 104400 HLT ;PARITY ERROR! ADDRESS+2 IS IN R2 DATA
2700 ;IS IN R0
2701 006702 000002 RTI ;CONTINUE SCAN
2702 006704 000240 3$: NOP ;INSERT HALT INST TO EXAMINE PARITY REGS
2703 006706 005067 177710 CLR PEFLG ;CLEAR PARITY ERROR INDICATORS
2704 006712 012706 000500 MOV #STKPTR,SP ;RESET STACK PTR
2705 006716 000005 RESET
2706 006720 004767 000122 JSR PC,MAMF ;GO ENABLE PARITY ERROR DETECTION
2707 006724 000177 173432 JMP @PERSTR ;RESTART SELECTED PROGRAM
2708
2709 ;SERVICE ROUTINE IF PARITY ERROR NOT DETECTED ON SCAN
2710 006730 105767 177666 4$: TSTB PEFLG ;BRANCH IF PARITY ERROR WAS
2711 006734 001363 BNE 3$ ;DETECTED ON SCAN
2712 006736 016602 000004 MOV 4(SP),R2 ;GET PC AT TIME OF ERROR
2713 006742 162702 000002 SUB #2,R2 ;BACK IT UP
2714 006746 110667 177651 MOVB SP,PENFLG ;SET IND = NO PAR ERROR DETECTED ON SCAN
2715 006752 004567 172002 JSR R5,SPRINT ;GO TO PRINT ROUTINE
2716 006756 007021 NOFIND
2717 006760 104400 HLT ;ERROR! PARITY ERROR NOT DETECTED ON SCAN
2718 006762 000750 BR 3$
2719 ;THE BELOW 6 WORDS CONTAINS THE SAVED CONTENTS OF R0-R5 WHEN THE
2720 ;PARITY ERROR OCCURRED
2721 006764 000000 SAVR0: .WORD 0
2722 006766 000000 SAVR1: .WORD 0
2723 006770 000000 SAVR2: .WORD 0
2724 006772 000000 SAVR3: .WORD 0
2725 006774 000000 SAVR4: .WORD 0
2726 006776 000000 SAVR5: .WORD 0
2727
2728 007000 005015 040520 044522 PARERR: .ASCIZ <15><12>'PARITY ERROR'<15><12>
2729 007006 054524 042440 051122
2730 007014 051117 005015 000
2731 007021 116 052117 043040 NOFIND: .ASCIZ 'NOT FOUND ON SCAN'<15><12>
2732 007026 052517 042116 047440
2733 007034 020116 041523 047101
2734 007042 005015 000
2735 007046 .EVEN
2736
2737 ;ROUTINE TO ENABLE PARITY ERROR ACTION ON MA/MF PARITY MEMORIES
2738 172100 PARCSR=172100 ;ADDRESS OF FIRST PARITY REGISTER
2739 000114 PARVEC=114 ;PARITY ERROR INTERRUPT VECTOR ADDRESS
2740

```

```

2741 007046 032737 000040 177570 .MAMF: BIT #40,2#SWR ;CHECK IF PARITY ERROR DETECTION IS TO
2742 007054 001033 DISPAR BNE DISPAR ;BE ENABLED. BRANCH IF NOT TO BE ENABLED
2743 007056 013746 000004 MOV 2#ERRVEC, -(SP) ;SAVE ERROR TRAP VECTOR
2744 007062 012737 000006 000004 MOV 2#ERRVEC+2, 2#ERRVEC ;SET TIME OUT TRAP TO RETURN (VIA RTI)
2745 007070 012737 006570 000114 MOV 2#PARSRV, 2#PARVEC ;SET PARITY ERROR TRAP VECTOR
2746 007076 012737 000340 000116 MOV #340, 2#PARVEC+2 ;PRIORITY LEVEL 7 ON TRAP
2747 007104 012700 172100 MOV 2#PARCSR, R0 ;GET FIRST ADDRESS OF PARITY REGISTER
2748 007110 012702 000001 MOV #1, R2
2749 007114 005027 CLR (PC)+ ;CLEAR AVAILABILITY INDICATOR
2750 007116 000000 PARAVA: .WORD 0 ;CONTAINS AVAILABILITY INDICATOR
2751
2752 ;ENABLE ALL AVAILABLE PARITY REGISTERS
2753 007120 000262 1S: SEV ;SET TIME OUT INDICATOR
2754 007122 012720 000001 MOV #1, (R0)+ ;SET ACTION ENABLE IF AVAILABLE
2755 007126 102402 BVS 2S ;BRANCH IF NO PARITY AVAILABLE
2756 007130 050267 177762 BIS R2, PARAVA ;SET AVAILABILITY INDICATOR
2757 007134 006302 2S: ASL R2 ;SHIFT INDICATOR
2758 007136 103370 BCC 1S
2759 007140 012637 000004 MOV (SP)+, 2#ERRVEC ;RESTORE ERROR TRAP VECTOR
2760 007144 000207 DISPAR: RTS PC ;RETURN
2761
2762 .SBTTL MEM MGMT ROUTINES
2763 ;ROUTINE TO INITIALIZE MEMORY MANAGEMENT REGISTERS
2764 007146 000240 LDMNO: NOP
2765 007150 005767 171576 TST MMAVA
2766 007154 001432 BEQ 1S
2767 007156 012737 077006 172300 MOV #177*256, -400+UP+RW, 2#KIPDR0 ;SET KIPDR0=RW UP 177 BLOCKS
2768 007164 012737 077406 172302 MOV #200*256, -400+UP+RW, 2#KIPDR1 ;SET KIPDR1=RW UP 200 BLOCKS
2769 007172 005037 172304 CLR 2#KIPDR2
2770 007176 012737 000000 172344 MOV #0, 2#KIPAR2
2771 007204 012737 077406 172316 MOV #200*256, -400+UP+RW, 2#KIPDR7 ;SET KIPDR7=RW UP 200 BLOCKS
2772 007212 005037 172340 CLR 2#KIPAR0
2773 007216 012737 000200 172342 MOV #200, 2#KIPAR1
2774 007224 012737 007600 172356 MOV #7600, 2#KIPAR7
2775 007232 012737 000001 177572 MOV #1, 2#SRO ;ENABLE MEM MGMT
2776 007240 000240 NOP
2777 007242 000207 1S: RTS PC
2778
2779 ;MEMORY MANAGEMENT ABORT ROUTINE FOR WRITE UP
2780 007244 012702 020000 MMABTO: MOV #20000, R2 ;RESET R2
2781 007250 062737 000200 172342 ADD #200, 2#KIPAR1 ;ADVANCE TO NEXT 4K
2782 007256 013716 177576 MOV 2#SR2, (SP) ;RETURN TO INSTRUCTION THAT
2783 007262 005037 177572 CLR 2#SRO ;DISABLE MEM MGMT
2784 007266 012737 000001 177572 MOV #1, 2#SRO ;ENABLE MEM MGMT
2785 007274 000002 RTI ;CAUSED THE ABORT
2786
2787 ;MEM MGMT ABORT SERVICE FOR WRITE DOWN
2788 007276 012702 040000 MMABTI: MOV #40000, R2 ;RESET R2
2789 007302 162737 000200 172342 SUB #200, 2#KIPAR1
2790 007310 001406 BEQ 2S
2791 007312 013716 177576 MOV 2#SR2, (SP)
2792 007316 012737 000001 177572 MOV #1, 2#SRO ;ENABLE MEM MGMT
2793 007324 000002 RTI
2794
2795 2S: CLR 2#SRO ;DISABLE MEM MGMT
2796 007326 005037 177572 BIS #V, 2(SPI)
2797 007332 052766 000002

```

```

2797 007340 000002 RTI
2798
2799
2800 007342 005702 :ROUTINE TO SET UP MEMORY MANAGEMENT FOR PATTERN TESTS
2801 007344 001442 STMM2: TST R2 ;CHECK IF TESTING BANK # 0
2802 007346 005767 171400 BEQ 25 ;EXIT IF BANK # 0
2803 007352 001005 TST MMAVA
2804 007354 006002 BNE 15 ;BRANCH IF MEM MGMT AVAILABLE
2805 007356 006002 ROR R2 ;ADJUST ADDRESS
2806 007360 006002 ROR R2
2807 007362 006002 ROR R2
2808 007364 000207 RTS PC ;RETURN
2809
2810 007366 004767 177554 15: JSR PC,LDMMO ;GO MAKE INITIAL SET UP
2811 007372 000302 SWAB R2
2812 007374 006002 ROR R2
2813 007376 010237 172344 MOV R2,@#KIPAR2
2814 007402 062702 000200 ADD #200,R2
2815 007406 010237 172346 MOV R2,@#KIPAR3
2816 007412 012737 077406 172304 MOV #200*256.-400+UP+RW,@#KIPDR2 ;SET KIPDR2=RW UP 200 BLOCKS
2817 007420 012737 077406 172306 MOV #200*256.-400+UP+RW,@#KIPDR3 ;SET KIPDR3=RW UP 200 BLOCKS
2818 007426 005037 172310 CLR @#KIPDR4
2819 007432 012702 040000 MOV #40000,R2
2820 007436 012737 007454 000250 MOV #MMAVT2,@#MMVEC
2821 007444 012737 000001 177572 MOV #1,@#SRO ;ENABLE MEM MGMT
2822 007452 000207 25: RTS PC
2823
2824 :ROUTINE TO SERVICE 8 XOR 13 ABORTS
2825 007454 000240 MMABT2: NOP
2826 007456 012702 040000 MOV #40000,R2
2827 007462 062737 000400 172344 ADD #400,@#KIPAR2
2828 007470 062737 000400 172346 ADD #400,@#KIPAR3
2829 007476 013716 177576 MOV @#SR2,(SP) ;SET RETURN TO INSTRUCTION THAT ABORTED
2830 007502 012737 000001 177572 MOV #1,@#SRO ;ENABLE MEM MGMT
2831 007510 000002 RTI
2832
2833 .SBTTL 1 XOR 8 ROUTINES
2834 :ROUTINE TO WRITE 1 XOR 8 WORST CASE NOISE PATTERN
2835 :CALL: MOV BANK #,-(SP) ;PUSH STARTING BANK # ON THE STACK
2836 : MOV BLKCNT,-(SP) ;PUSH 128. WORD BLOCK COUNT ON THE STACK
2837 : JSR PC,.1X8
2838
2839 007512 016603 000002 .1X8: MOV 2(SP),R3 ;GET # OF 128. WORD BLOCKS TO WRITE
2840 007516 016602 000004 MOV 4(SP),R2 ;GET STARTING BANK #
2841 007522 004767 177614 JSR PC,STMM2 ;GO SET UP MEM MGMT
2842 007526 012700 177777 15: MOV #-1,R0 ;SET UP DATA REGISTERS
2843 007532 010005 MOV R0,R5
2844 007534 005105 COM R5
2845
2846 007536 005100 25: COM R0
2847 007540 005105 COM R5
2848 007542 012704 000010 MOV #8,R4 ;SET 128. WORD COUNTER
2849 007546 010022 35: MOV R0,(R2)+ ;WRITE 128. WORDS
2850 007550 010522 MOV R5,(R2)+
2851 007552 010022 MOV R0,(R2)+
2852 007554 010522 MOV R5,(R2)+

```

```

2853
2854 007556 010022      MOV      R0,(R2)+
2855 007560 010522      MOV      R5,(R2)+
2856 007562 010022      MOV      R0,(R2)+
2857 007564 010522      MOV      R5,(R2)+
2858
2859 007566 010022      MOV      R0,(R2)+
2860 007570 010522      MOV      R5,(R2)+
2861 007572 010022      MOV      R0,(R2)+
2862 007574 010522      MOV      R5,(R2)+
2863
2864 007576 010022      MOV      R0,(R2)+
2865 007600 010522      MOV      R5,(R2)+
2866 007602 010022      MOV      R0,(R2)+
2867 007604 010522      MOV      R5,(R2)+
2868
2869 007606 005304      DEC      R4          ;DECREMENT 128. WORD COUNTER
2870 007610 001356      BNE     3$
2871 007612 005303      DEC      R3          ;DECREMENT BLOCK COUNT
2872 007614 001350      BNE     2$
2873 007616 012616      MOV     (SP)+,(SP)  ;ADJUST STACK
2874 007620 012616      MOV     (SP)+,(SP)
2875 007622 000207      RTS      PC          ;RETURN TO CALLER
2876
2877          ;ROUTINE TO CHECK 1 XOR 8 PATTERN WRITTEN ABOVE
2878          ;CALL: MOV      BANK #,-(SP)  ;PUSH STARTING BANK # ON THE STACK
2879          ;      MOV      BLKCNT,-(SP)  ;PUSH 128. WORD BLOCK COUNT ON STACK
2880          ;      JSR      PC,..1X8
2881
2882 007624 000240      ..1X8: NOP
2883 007626 004767 002120      JSR     PC,CKSMR    ;GO CHECK SWITCH REGISTER
2884 007632 016667 000002 171116 10$: MOV     2(SP),COUNT ;GET BLOCK COUNT
2885 007640 016602 000004      MOV     4(SP),R2    ;GET STARTING BANK #
2886 007644 004767 177472      JSR     PC,STAM2    ;GO SET UP MEM MGMT
2887 007650 005000      1$: CLR     R0        ;CLEAR TEST WORD
2888 007652 005767 171066      TST     ICOUNT      ;IF BIT 15 OF ICOUNT =1 THEN PATTERN
2889 007656 100401      BMI     .+4         ;IS COMPLEMENTED
2890 007660 005100      COM     R0          ;COMPLEMENT TEST WORD
2891 007662 012705 000040      2$: MOV     #32.,R5   ;SET 128. WORD COUNTER
2892
2893          3$: COM     R0
2894 007670 012203      MOV     (R2)+,R3    ;GET TEST DATA
2895 007672 020003      CMP     R0,R3       ;COMPARE WITH CHECK WORD
2896 007674 001403      BEQ     .+10
2897 007676 005046      CLR     -(SP)       ;PUSH FAKE STATUS ON THE STACK
2898 007700 004767 17127C      JSR     PC,ERROR    ;ERROR! MEM DATA (R3) NOT = TEST DATA
2899          ;(R0), ADDRESS=(R2)-2
2900
2901 007704 005100      COM     R0
2902 007706 012203      MOV     (R2)+,R3    ;GET TEST DATA
2903 007710 020003      CMP     R0,R3       ;COMPARE WITH CHECK WORD
2904 007712 001403      BEQ     .+10
2905 007714 005046      CLR     -(SP)       ;PUSH FAKE STATUS ON THE STACK
2906 007716 004767 171252      JSR     PC,ERROR    ;ERROR! MEM DATA (R3) NOT = TEST DATA
2907          ;(R0), ADDRESS=(R2)-2
2908

```

```

2909 007722 005100 COM RO
2910 007724 012203 MOV (R2)+,R3 ;GET TEST DATA
2911 007726 020003 CMP RO,R3 ;COMPARE WITH CHECK WORD
2912 007730 001403 BEQ .+10
2913 007732 005046 CLR -(SP) ;PUSH FAKE STATUS ON THE STACK
2914 007734 004767 171234 JSR PC,ERROR ;ERROR! MEM DATA (R3) NOT = TEST DATA
2915 ;(R0), ADDRESS=(R2)-2
2916
2917 007740 005100 COM RO
2918 007742 012203 MOV (R2)+,R3 ;GET TEST DATA
2919 007744 020003 CMP RO,R3 ;COMPARE WITH CHECK WORD
2920 007746 001403 BEQ .+10
2921 007750 005046 CLR -(SP) ;PUSH FAKE STATUS ON THE STACK
2922 007752 004767 171216 JSR PC,ERROR ;ERROR! MEM DATA (R3) NOT = TEST DATA
2923 ;(R0), ADDRESS=(R2)-2
2924
2925 007756 005305 DEC R5 ;DECREMENT 128. WORD COUNTER
2926 007760 001342 BNE 3$
2927 007762 005100 COM RO ;COMPLEMENT CHECK WORD
2928 007764 005367 170766 DEC COUNT ;DECREMENT BLOCK COUNT
2929 007770 001334 BNE 2$
2930 007772 016602 000004 MOV 4(SP),R2 ;GET BANK #
2931 007776 004767 177340 JSR PC,STMM2
2932 010002 016667 000002 170746 4$: MOV 2(SP),COUNT ;GET # OF 128. WORD BLOCKS TO COMPLEMENT
2933 010010 006367 170730 ASL ICOUNT
2934 010014 102306 BVC 10$
2935 010016 012705 000040 50$: MOV #32.,R5
2936 010022 005122 5$: COM (R2)+ ;COMPLEMENT PATTERN
2937 010024 005122 COM (R2)+
2938 010026 005122 COM (R2)+
2939 010030 005122 COM (R2)+
2940 010032 005305 DEC R5
2941 010034 001372 BNE 5$
2942 010036 005367 170714 DEC COUNT
2943 010042 001365 BNE 50$
2944 010044 005767 170674 TST ICOUNT
2945 010050 001270 BNE 10$
2946 010052 012616 MOV (SP)+,(SP)
2947 010054 012616 MOV (SP)+,(SP)
2948 010056 000207 RTS PC
2949
2950 .SBTTL 3 XOR 9 ROUTINES
2951 ;ROUTINE TO WRITE 3XOR9 WORST CASE NOISE TEST PATTERN
2952 ;CALL: MOV BANK #,-(SP) ;PUSH STARTING BANK # ON STACK
2953 ; MOV BLKCNT,-(SP) ;PUSH 256. WORD BLOCK COUNT ON STACK
2954 ; JSR PC,.3X9 ;CALL ROUTINE
2955
2956 010060 016602 000004 .3X9: MOV 4(SP),R2 ;GET STARTING BANK #
2957 010064 004767 177252 JSR PC,STMM2
2958 010070 005000 CLR RO
2959 010072 010003 MOV RO,R3
2960 010074 005103 COM R3 ;RO (0) AND R3 (-1) IS THE DATA WRITTEN
2961 010076 005767 173260 TST PARPAT ;BRANCH IF PARITY MEMORY PATTERN IS
2962 010102 001402 BEQ 1$ ;NOT TO BE WRITTEN
2963
2964 010104 012700 000401 MOV #401,R0 ;WRITE PARITY 3X9 PATTERN

```

```

2965 010110 012704 000020 1$: MOV #16.,R4 ;EACH LOOP WRITES 256. WORDS
2966
2967 010114 010022 2$: MOV R0,(R2)+
2968 010116 010022 MOV R0,(R2)+
2969 010120 010022 MOV R0,(R2)+
2970 010122 010022 MOV R0,(R2)+
2971
2972 010124 010322 MOV R3,(R2)+
2973 010126 010322 MOV R3,(R2)+
2974 010130 010322 MOV R3,(R2)+
2975 010132 010322 MOV R3,(R2)+
2976
2977 010134 010022 MOV R0,(R2)+
2978 010136 010022 MOV R0,(R2)+
2979 010140 010022 MOV R0,(R2)+
2980 010142 010022 MOV R0,(R2)+
2981
2982 010144 010322 MOV R3,(R2)+
2983 010146 010322 MOV R3,(R2)+
2984 010150 010322 MOV R3,(R2)+
2985 010152 010322 MOV R3,(R2)+
2986
2987 010154 005304 DEC R4
2988 010156 001356 BNE 2$
2989 010160 005100 COM R0
2990 010162 005103 COM R3
2991 010164 005767 173172 TST PARPAT ;BRANCH IF PARITY MEMORY PATTERN IS
2992 010170 001402 BEQ 3$ ;NOT TO BE WRITTEN
2993
2994 010172 004767 000014 JSR PC,XOR39 ;GO GET CONSTANTS
2995 010176 005366 000002 3$: DEC 2(SP) ;DECREMENT 256. WORD BLOCK COUNT
2996 010202 001342 BNE 1$
2997 010204 012616 MOV (SP)+,(SP) ;ADJUST STACK
2998 010206 012616 MOV (SP)+,(SP)
2999 010210 000207 RTS PC
3000
3001 ;ROUTINE TO SET CONSTANTS FOR WRITING/CHECKING 3 XOR PATTERN WITH
3002 ;PARITY.
3003 010212 032702 000010 .XOR39: BIT #10,R2 ;CHECK BIT 3
3004 010216 001404 BEQ .3IS0 ;BRANCH IF BIT 3 = 0
3005 010220 032702 00100C .3IS1: BIT #1000,R2 ;CHECK BIT 9
3006 010224 001404 BEQ .3NOT9 ;BRANCH IF BIT 9 = 0

```


| | | | |
|------|--------|--------|--------|
| 3007 | 010226 | 000407 | |
| 3008 | 010230 | 032702 | 001000 |
| 3009 | 010234 | 001404 | |
| 3010 | 010236 | 005767 | 170502 |
| 3011 | 010242 | 100004 | |
| 3012 | 010244 | 100410 | |
| 3013 | 010246 | 005767 | 170472 |

| | | |
|---------|-----|----------|
| | BR | .3IS9 |
| .3ISO: | BIT | #1000,R2 |
| | BEQ | .3IS9 |
| .3NOT9: | TST | ICOUNT |
| | BPL | LDCOMP |
| | BMI | LDNORM |
| .3IS9: | TST | ICOUNT |

```

;CHECK BIT 9
;BRANCH IF 0
;CHECK IF NORMAL OR COMPLEMENT DATA
;GO LOAD COMPLEMENT CONSTANTS
;GO LOAD NORMAL CONSTANTS
;CHECK IF NORMAL OR COMPLEMENT DATA

```

```

3014 010252 100005
3015 010254 012700 177777
3016 010260 012703 000401
3017 010264 000207
3018 010266 012700 000401
3019 010272 012703 177777
3020 010276 000207
3021
3022
3023
3024
3025

```

```

LDNORM: BPL LDNORM ;GO LOAD NORMAL CONSTANTS
MOV #-1,R0 ;SET COMPLEMENT CONSTANTS
MOV #401,R3
RTS PC ;RETURN
LDNORM: MOV #401,R0 ;LOAD NORMAL CONSTANTS
MOV #-1,R3
RTS PC

:ROUTINE TO CHECK 3 XOR 9 WORST CASE NOISE PATTERN
:CALL: MOV BANK# -(SP) ;PUSH STARTING BANK # ONTC STACK
: MOV BLKCNT, -(SP) ;AND 256. WORD BLOCK COUNT
: JSR PC...3X9 ;CALL ROUTINE

```

J06

TEST DDQAB-A 0-124K MEMORY EXERCISER
DDQABA.P11 3 XOR 9 ROUTINES

MACY11 27(732) 10-SEP-76 10:35 PAGE 74

3026
3027 010300 000240
3028 010302 004767 001444
3029
3030

..3X9: NOP
JSR PC,CKSWR ;GO CHECK SWITCH REGISTER
;CHECK WORST CASE PATTERN

K06

TEST DDQAB-A 0-124K MEMORY EXERCISER
DDQABA.P11 3 XOR 9 ROUTINES

MACY11 27(732) 10-SEP-76 10:35 PAGE 75

3C31 010306 016604 000002

1\$: MOV 2(SP),R4 ;GET 256. BLOCK WORD COUNT

L06

TEST DDQAB-A 0-124K MEMORY EXERCISER
DDQABA.P1: 3 XOR 9 ROUTINES

MACY11 27(732) 10-SEP-76 10:35 PAGE 76

3032 010312 016602 000004

MOV 4(SP),R2 ;GET FIRST BANK #

M06

TEST DDQAB-A 0-124K MEMORY EXERCISER
DDQABA.P11 3 XOR 9 ROUTINES

MACY11 27(732) 10-SEP-76 10:35 PAGE 77

3033 010316 004767 177020
3034 010322 005000
3035 010324 005767 170414

JSR PC,STMM2
CLR RO
TST ICOUNT

;GO SET UP MEM MGMT
;SET CHECK WORD
;IF ICOUNT IS NEG AM CHECKING COMP-

TEST DDQAB-A 0-124K MEMORY EXERCISER
 DDQABA.P11 3 XOR 9 ROUTINES

MACY11 27(732) 10-SEP-76 10:35 PAGE 78

| | | | | | | |
|------|--------|--------|--------|----------|---------|-------------------------------------|
| 3036 | 010330 | 100001 | | BPL | .+4 | ;LEMENED PATTERN |
| 3037 | 010332 | 005100 | | COM | RO | ;50 COMPLEMENT CHECK WORD |
| 3038 | 010334 | 012705 | 000100 | 2\$: MOV | #64.,R5 | ;SET 256. WORD COUNTER |
| 3039 | | | | | | |
| 3040 | 010340 | 005767 | 173016 | 3\$: TST | PARPAT | ;BRANCH IF PARITY MEMORY PATTERN IS |
| 3041 | 010344 | 001402 | | BEQ | 30\$ | ;NOT TO BE CHECKED |

```

3042
3043 010346 004767 177640
3044 010352
3045 010352 012203
3046 010354 020003
3047 010356 001403
3048 010360 005046
3049 010362 004767 170606
3050
3051
3052 010366 012203
3053 010370 020003
3054 010372 001403
3055 010374 005046
3056 010376 004767 170572
3057
3058
3059 010402 012203
3060 010404 020003
3061 010406 001403
3062 010410 005046
3063 010412 004767 170556
3064
3065
3066 010416 012203
3067 010420 020003
3068 010422 001403
3069 010424 005046
3070 010426 004767 170542
3071
3072
3073
3074 010432 005100
3075 010434 005305
3076 010436 001340
3077 010440 005100
3078 010442 005304
3079 010444 001333
3080
3081 010446 032737 040000 177570
3082 010454 001314
3083 010456 016667 000002 170272 40$:
3084 010464 016602 000004
3085 010470 004767 176646
3086
3087 ;CHECK WORST CASE BIT COMPLEMENT PATTERN
3088 010474 005000
3089 010476 005767 170242
3090 010502 100001
3091 010504 005100
3092 010506 012704 000100 4$:
3093 010512 012705 000004 5$:
3094 010516 005767 172640 6$:
3095 010522 001402
3096 010524 004767 177462
3097 010530 012203 60$:

```

```

JSR PC, .XOR39 ;GO GET CONSTANT
30$: MOV (R2)+,R3 ;GET TEST DATA
CMP RO,R3 ;COMPARE WITH CHECK WORD
BEQ .+10
CLR -(SP) ;PUSH FAKE STATUS ON THE STACK
JSR PC,ERROR ;ERROR! MEM DATA (R3) NOT = TEST DATA
;(RO), ADDRESS=(R2)-2

MOV (R2)+,R3 ;GET TEST DATA
CMP RO,R3 ;COMPARE WITH CHECK WORD
BEQ .+10
CLR -(SP) ;PUSH FAKE STATUS ON THE STACK
JSR PC,ERROR ;ERROR! MEM DATA (R3) NOT = TEST DATA
;(RO), ADDRESS=(R2)-2

MOV (R2)+,R3 ;GET TEST DATA
CMP RO,R3 ;COMPARE WITH CHECK WORD
BEQ .+10
CLR -(SP) ;PUSH FAKE STATUS ON THE STACK
JSR PC,ERROR ;ERROR! MEM DATA (R3) NOT = TEST DATA
;(RO), ADDRESS=(R2)-2

MOV (R2)+,R3 ;GET TEST DATA
CMP RO,R3 ;COMPARE WITH CHECK WORD
BEQ .+10
CLR -(SP) ;PUSH FAKE STATUS ON THE STACK
JSR PC,ERROR ;ERROR! MEM DATA (R3) NOT = TEST DATA
;(RO), ADDRESS=(R2)-2

MOV (R2)+,R3 ;GET TEST DATA
CMP RO,R3 ;COMPARE WITH CHECK WORD
BEQ .+10
CLR -(SP) ;PUSH FAKE STATUS ON THE STACK
JSR PC,ERROR ;ERROR! MEM DATA (R3) NOT = TEST DATA
;(RO), ADDRESS=(R2)-2

COM RO ;COMPLEMENT CHECK WORD
DEC R5 ;DECREMENT 256. WORD COUNTER
BNE 3$
COM RO ;COMPLEMENT CHECK WORD
DEC R4 ;DECREMENT BLOCK COUNTER
BNE 2$

BIT #40000,2#SWR ;LOOP ON TEST?
BNE 1$ ;BRANCH IF LOOP ON TEST DESIRED
MOV 2(SP),COUNT ;GET # OF 256. WORD BLOCKS TO CHECK
MOV 4(SP),R2 ;GET STARTING BANK #
JSR PC,STMM2 ;GO SET UP MEM MGMT IF REQUIRD

;CHECK WORST CASE BIT COMPLEMENT PATTERN
CLR RO
TST ICOUNT ;CHECK IF COMPLEMENT PATERN
BPL .+4
COM RO ;COMPLEMENT CHECK WORD
MOV #64,R4 ;SET 256. WORD COUNTER
MOV #4,R5 ;SET 4 WORD COUNTER
TST PARPAT ;BRANCH IF PARITY MEMORY PATTERN IS
BEQ 60$ ;NOT TO BE CHECKED
JSR PC, .XOR39
MOV (R2)+,R3 ;GET DATA

```


| | | | | | | | | |
|------|--------|--------|--------|--------|-------|------|--------------|--------------------------------|
| 3098 | 010532 | 020003 | | | | CMP | R0,R3 | ;CHECK DATA |
| 3099 | 010534 | 001403 | | | | BEG | .+10 | |
| 3100 | 010536 | 005046 | | | | CLR | -(SP) | |
| 3101 | 010540 | 004767 | 170430 | | | JSR | PC,ERROR | |
| 3102 | 010544 | 005100 | | | | COM | R0 | ;COMPLEMENT CHECK WORD |
| 3103 | 010546 | 005142 | | | | COM | -(R2) | ;COMPLEMENT TEST DATA |
| 3104 | 010550 | 012203 | | | | MOV | (R2)+,R3 | ;GET DATA |
| 3105 | 010552 | 020003 | | | | CMP | R0,R3 | ;CHECK |
| 3106 | 010554 | 001403 | | | | BEG | .+10 | |
| 3107 | 010556 | 005046 | | | | CLR | -(SP) | ;PUSH FAKE STATUS ON THE STACK |
| 3108 | 010560 | 004767 | 170410 | | | JSR | PC,ERROR | |
| 3109 | 010564 | 005100 | | | | COM | R0 | ;COMPLEMENT CHECK WORD |
| 3110 | 010566 | 005162 | 177776 | | | COM | -2(R2) | ;RESTORE DATA |
| 3111 | 010572 | 005305 | | | | DEC | R5 | ;DECREMENT 4 WORD COUNTER |
| 3112 | 010574 | 001350 | | | | BNE | 6\$ | |
| 3113 | 010576 | 005100 | | | | COM | R0 | ;COMPLEMENT CHECK WORD |
| 3114 | 010600 | 005304 | | | | DEC | R4 | ;DECREMENT 256. WORD COUNTER |
| 3115 | 010602 | 001343 | | | | BNE | 5\$ | |
| 3116 | 010604 | 005100 | | | | COM | R0 | ;COMPLEMENT CHECK WORD |
| 3117 | 010606 | 005367 | 170144 | | | DEC | COUNT | ;DECREMENT BLOCK COUNTER |
| 3118 | 010612 | 001335 | | | | BNE | 4\$ | |
| 3119 | | | | | | | | |
| 3120 | 010614 | 016602 | 000004 | | | MOV | 4(SP),R2 | ;GET BANK # |
| 3121 | 010620 | 004767 | 176516 | | | JSR | PC,STAMP2 | |
| 3122 | 010624 | 016603 | 000002 | | | MOV | 2(SP),R3 | ;GET BLOCK COUNT |
| 3123 | 010630 | 032737 | 040000 | 177570 | | BIT | #40000,2#SWR | ;LOOP ON TEST |
| 3124 | 010636 | 001307 | | | | BNE | 40\$ | ;BRANCH IF LOOP ON TEST |
| 3125 | 010640 | 006367 | 170100 | | | ASL | ICOUNT | |
| 3126 | 010644 | 102220 | | | | BVC | 1\$ | |
| 3127 | 010646 | 012705 | 000040 | | 7\$: | MOV | #32.,R5 | ;COMPLEMENT PATTERN |
| 3128 | 010652 | 011200 | | | 10\$: | MOV | (R2),R0 | ;GET FIRST DATA WORD |
| 3129 | 010654 | 016204 | 000010 | | | MOV | 10(R2),R4 | ;GET FIFTH DATA WORD |
| 3130 | 010660 | 110422 | | | | MOVB | R4,(R2)+ | ;SWAP WORDS 1-4 |
| 3131 | 010662 | 110422 | | | | MOVB | R4,(R2)+ | ;WITH 3-8 |

```

3132 010664 110422
3133 010666 110422
3134 010670 110422
3135 010672 110422
3136 010674 110422
3137 010676 110422
3138 010700 110022
3139 010702 110022
3140 010704 110022
3141 010706 110022
3142 010710 110022
3143 010712 110022
3144 010714 110022
3145 010716 110022
3146 010720 005305
3147 010722 001353
3148 010724 005303
3149 010726 001347
3150
3151 010730 005757 170010
3152 010734 001402
3153 010736 000167 177344
3154 010742 012616
3155 010744 012616
3156 010746 000207
3157
3158
3159
3160
3161
3162
3163
3164 010750 016604 000002
3165 010754 016602 000004
3166 010760 004767 176356
3167 010764 005000
3168 010766 012705 000040
3169 010772 005100
3170 010774 012703 000200
3171 011000 005100
3172 011002 010022
3173 011004 005303
3174 011006 001375
3175 011010 005305
3176 011012 001370
3177 011014 005304
3178 011016 001363
3179 011020 012616
3180 011022 012616
3181 011024 000207
3182
3183
3184
3185
3186
3187

```

```

MOV      R4,(R2)+
MOV      R4,(R2)+
MOV      R4,(R2)+
MOV      R4,(R2)+
MOV      R4,(R2)+
MOV      R4,(R2)+
MOV      R0,(R2)+      ;AND VICE VERSA
MOV      R0,(R2)+
MOV      R0,(R2)+
MOV      R0,(R2)+
MOV      R0,(R2)+
MOV      R0,(R2)+
MOV      R0,(R2)+
MOV      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

```

```

3188
3189 011026 000240      ..8X13: NOP
3190 011030 004767 000716 JSR    PC,CKSWR      ;CO CHECK SWITCH REGISTER
3191 011034 012700 177777 MOV    #-1,R0       ;SET TEST DATA WORD
3192 011040 016602 000004 10$: MOV    4(SP),R2   ;GET BANK #
3193 011044 004767 176272 JSR    PC,STMM2     ;GO SET MEM MGMT IF REQUIRED
3194 011050 016667 000002 167700 MOV    2(SP),COUNT ;GET # OF 4K BANKS TO CHECK
3195
3196 011056 012704 000040 1$: MOV    #32.,R4   ;SET 4K WORD COUNTER
3197 011062 005100 2$: COM    R0       ;COMPLEMENT TEST WORD
3198 011064 012705 000100 MOV    #64.,R5     ;SET 128 WORD COUNTER
3199
3200 011070 3$:
3201 011070 012203      MOV    (R2)+,R3   ;GET TEST DATA
3202 011072 020003      CMP    R0,R3     ;COMPARE WITH CHECK WORD
3203 011074 001403      BEQ    .+10
3204 011076 005046      CLR    -(SP)     ;PUSH FAKE STATUS ON THE STACK
3205 011100 004767 170070 JSR    PC,ERROR   ;ERROR! MEM DATA (R3) NOT = TEST DATA
3206
3207
3208 011104 012203      MOV    (R2)+,R3   ;GET TEST DATA
3209 011106 020003      CMP    R0,R3     ;COMPARE WITH CHECK WORD
3210 011110 001403      BEQ    .+10
3211 011112 005046      CLR    -(SP)     ;PUSH FAKE STATUS ON THE STACK
3212 011114 004767 170054 JSR    PC,ERROR   ;ERROR! MEM DATA (R3) NOT = TEST DATA
3213
3214
3215 011120 005305      DEC    R5        ;DECREMENT 128 WORD COUNTER
3216 011122 001362      BNE    3$
3217 011124 005304      DEC    R4        ;DECREMENT 4096. WORD COUNTER
3218 011126 001355      BNE    2$
3219 011130 005100      COM    R0
3220 011132 005367 167620 DEC    COUNT     ;ALL 4K BANKS CHECKED?
3221 011136 001347      BNE    1$
3222
3223 011140 016602 000004 MOV    4(SP),R2   ;GET FIRST BANK ADDRESS
3224 011144 004767 176172 JSR    PC,STMM2   ;GO SET UP MEM MGMT IF REQUIRED
3225 011150 016604 000002 MOV    2(SP),R4   ;GET # OF 4K BANKS
3226 011154 006367 167564 ASL    ICOUNT     ;SHIFT ITERATION PATTERN
3227 011160 001401      BEQ    .+4
3228 011162 102326      BVC    10$
3229 011164 012705 004000 40$: MOV    #2048.,R5  ;SET 4096. WORD COUNTER
3230 011170 005122 4$: COM    (R2)+   ;COMPLEMENT TEST PATTERN
3231 011172 005122      COM    (R2)+
3232 011174 005305      DEC    R5
3233 011176 001374      BNE    4$
3234 011200 005304      DEC    R4
3235 011202 001370      BNE    40$
3236 011204 005100      COM    R0        ;COMPLEMENT TEST WORD
3237 011206 005767 167532 TST    ICOUNT
3238 011212 001312      BNE    10$
3239 011214 012616      MOV    (SP)+,(SP)
3240 011216 012616      MOV    (SP)+,(SP)
3241 011220 000207      RTS    PC        ;RETURN
3242
3243

```

.SBTTL ROTATING 1'S & 0'S ROUTINES

```

3244      ;ROUTINE TO CHECK ROTATING '0' BIT THROUGH FIELD OF 1'S
3245      :CALL:  MOV   BANK#,-(SP)      ;SET STARTING BANK #
3246      :      MOV   BLKCNT,-(SP)     ;SET 256. WORD BLOCK COUNT
3247      :      JSR   PC,.ROT0         ;CALL ROUTINE
3248
3249      .ROT0: JSR   PC,CKSWR         ;GO CHECK SWITCHES
3250      :      MOV   2(SP),R4         ;GET 256. WORD BLOCK COUNT
3251      :      MOV   4(SP),R2         ;GET FIRST BANK #
3252      :      JSR   PC,STAM2        ;GO SET UP MEM MGMT (IF AVAIL)
3253      :      MOV   #-1,R0          ;SET CHECK WORD
3254
3255      1$:  MOV   #256.,R5           ;SET 256. WORD COUNT
3256      2$:  CLC                    ;CLEAR CARRY BIT IN PSW
3257      :      JSR   PC,ROTATE       ;
3258      :      MOV   -2(R2),R3       ;GET RESULT
3259      :      BCS   3$              ;BRANCH IF 'C' BIT WAS SET
3260      :      CMP   R0,R3           ;CHECK RESULT
3261      :      BEQ   4$              ;
3262      3$:  CLR   -(SP)             ;ERROR! COULD NOT ROTATE '0' BIT
3263      :      JSR   PC,ERROR        ;THROUGH ADDRESS IN R2
3264      4$:  DEC   R5                ;DECREMENT 256. WORD COUNT
3265      :      BNE   2$              ;LOOP UNTIL DONE
3266      :      DEC   R4              ;DECREMENT 256. WORD BLOCK COUNT
3267      :      BNE   1$              ;LOOP UNTIL DONE
3268      :      MOV   (SP)+,(SP)      ;POP CONSTANTS OFF THE STACK
3269      :      MOV   (SP)+,(SP)      ;
3270      :      RTS   PC              ;RETURN TO CALLER
3271
3272      ;ROUTINE TO CHECK ROTATING '1' BIT THROUGH A FIELD OF 0'S
3273      :CALL:  MOV   BANK#,-(SP)      ;SET STARTING BANK #
3274      :      MOV   BLKCNT,-(SP)     ;SET # OF 256. WORD BLOCKS TO CHECK
3275      :      JSR   PC,.ROT1         ;CALL ROUTINE
3276
3277      .ROT1: JSR   PC,CKSWR         ;GO CHECK SWITCHES
3278      :      MOV   2(SP),R4         ;GET # OF 256. WORD BLOCKS TO CHECK
3279      :      MOV   4(SP),R2         ;GET STARTING BANK #
3280      :      JSR   PC,STAM2        ;GO SET UP MEM MGMT (IF AVAIL)
3281      :      CLR   R0              ;SET CHECK WORD
3282
3283      1$:  MOV   #256.,R5           ;SET 256. WORD COUNTER
3284      2$:  SEC                    ;SET 'C' BIT IN PSW
3285      :      JSR   PC,ROTATE       ;GO ROTATE '1' BIT
3286      :      MOV   -2(R2),R3       ;GET RESULT
3287      :      BCC   3$              ;BRANCH IF 'C' IS CLEAR
3288      :      CMP   R0,R3           ;CHECK RESULT
3289      :      BEQ   .+4              ;
3290      3$:  HLT                    ;ERROR! COULD NOT ROTATE '1' BIT
3291      :      :                      ;THROUGH ADDRESS IN R2
3292      :      DEC   R5                ;DECREMENT 256. WORD COUNT
3293      :      BNE   2$              ;
3294      :      DEC   R4              ;DECREMENT 256. WORD BLOCK COUNT
3295      :      BNE   1$              ;
3296      :      MOV   (SP)+,(SP)      ;ADJUST RETURN ADDRESS
3297      :      MOV   (SP)+,(SP)      ;
3298      :      RTS   PC              ;RETURN TO CALLER
3299

```

```

3300          ;ROUTINE TO ROTATE 'C' BIT THROUGH A MEMORY LOCATION.
3301 011404 106112 ROTATE: ROLB (R2) ;(R2)=177776 OR 000001
3302 011406 106112          ROLB (R2) ;(R2)=177775 OR 000002
3303 011410 106112          ROLB (R2) ;(R2)=177773 OR 000004
3304 011412 106112          ROLB (R2) ;(R2)=177767 OR 000010
3305 011414 106112          ROLB (R2) ;(R2)=177757 OR 000020
3306 011416 106112          ROLB (R2) ;(R2)=177737 OR 000040
3307 011420 106112          ROLB (R2) ;(R2)=177677 OR 000100
3308 011422 106112          ROLB (R2) ;(R2)=177777 OR 000000
3309 011424 106122          ROLB (R2)+ ;(R2)=177577 OR 000200
3310 011426 106112          ROLB (R2) ;(R2)=177377 OR 000400
3311 011430 106112          ROLB (R2) ;(R2)=176777 OR 001000
3312 011432 106112          ROLB (R2) ;(R2)=175777 OR 002000
3313 011434 106112          ROLB (R2) ;(R2)=173777 OR 004000
3314 011436 106112          ROLB (R2) ;(R2)=167777 OR 010000
3315 011440 106112          ROLB (R2) ;(R2)=157777 OR 020000
3316 011442 106112          ROLB (R2) ;(R2)=137777 OR 040000
3317 011444 106112          ROLB (R2) ;(R2)=077777 OR 100000
3318 011446 106122          ROLB (R2)+ ;(R2)=177777 OR 000000
3319 011450 000207          RTS      PC      ;RETURN
3320
3321          ;ROUTINE TO WRITE USER SLECTED PATTERN INTO MEMORY
3322          ;CALL: MOV      BANK#, -(SP) ;PUSH STARTING BANK # ONTO STACK
3323          ;      MOV      BLKCNT, -(SP) ;AND 128. WORD BLOCK COUNT
3324          ;      JSR      PC, ..USER ;CALL ROUTINE
3325
3326 011452 016604 000002 .USER: MOV      2(SP), R4 ;GET BLOCK COUNT
3327 011456 016602 000004      MOV      4(SP), R2 ;GET STARTING BANK #
3328 011462 004767 175654      JSR      PC, STMM2 ;GO SET UP MEM MGMT
3329 011466 016700 174012      MOV      .CONST, R0 ;GET USER CONSTANT
3330 011472 012703 000100 1$: MOV      #64, R3 ;SET 256. WORD COUNTER
3331 011476 010022 2$: MOV      R0, (R2)+ ;WRITE 256. WORDS
3332 011500 010022      MOV      R0, (R2)+
3333 011502 010022      MOV      R0, (R2)+
3334 011504 010022      MOV      R0, (R2)+
3335 011506 005303      DEC      R3 ;DECREMENT 256. WORD COUNTER
3336 011510 001372      BNE      2$ ;LOOP UNTIL 256. WORDS HAVE BEEN WRITTEN
3337 011512 005304      DEC      R4 ;DECREMENT BLOCK COUNT
3338 011514 001366      BNE      1$
3339 011516 012616      MOV      (SP)+, (SP) ;ADJUST STACK
3340 011520 012616      MOV      (SP)+, (SP)
3341 011522 000207          RTS      PC
3342
3343          .SBTTL USER PATTERN ROUTINE
3344          ;ROUTINE TO CHECK USER SELECTED PATTERN
3345          ;CALL: MOV      BANK#, -(SP) ;PUSH STARTING BANK # ONTO STACK
3346          ;      MOV      BLKCNT, -(SP) ;AND 256. WORD BLOCK COUNT
3347          ;      JSR      PC, ..USER ;CALL ROUTINE
3348
3349 011524 004767 000222 ..USER: JSR      PC, CKSWR ;GO CHECK SWITCH REGISTER
3350 011530 016700 173750      MOV      .CONST, R0 ;GET USER CONSTANT
3351 011534 016604 000002 1$: MOV      2(SP), R4 ;GET # OF 256. WORD BLOCKS
3352 011540 016602 000004      MOV      4(SP), R2 ;GET STARTING BANK #
3353 011544 004767 175572      JSR      PC, STMM2 ;GO SET UP MEM MGMT IF REQUIRED
3354
3355 011550 012705 000100 2$: MOV      #64, R5 ;SET WORD COUNT

```

```

3356 011554
3357 011554 012203
3358 011556 020003
3359 011560 001403
3360 011562 005046
3361 011564 004767 167404
3362
3363
3364 011570 012203
3365 011572 020003
3366 011574 001403
3367 011576 005046
3368 011600 004767 167370
3369
3370
3371 011604 012203
3372 011606 020003
3373 011610 001403
3374 011612 005046
3375 011614 004767 167354
3376
3377
3378 011620 012203
3379 011622 020003
3380 011624 001403
3381 011626 005046
3382 011630 004767 167340
3383
3384
3385 011634 005305
3386 011636 001346
3387 011640 005304
3388 011642 001342
3389
3390 011644 032737 040000 177570
3391 011652 001330
3392 011654 006367 167064
3393 011660 001325
3394 011662 012616
3395 011664 012616
3396 011666 000207
3397
3398
3399
3400
3401
3402
3403 011670
3404 011670 004767 166410
3405 011674 004767 002172
3406 011700 010267 000042
3407 011704 010367 000040
3408 011710 004767 166414
3409 011714 011667 000024
3410 011720 016777 000024 00001E
3411 011726 016767 000014 000006

```

```

35:
MOV (R2)+,R3 ;GET TEST DATA
CMP R0,R3 ;COMPARE WITH CHECK WORD
BEQ .+10
CLR -(SP) ;PUSH FAKE STATUS ON THE STACK
JSR PC,ERROR ;ERROR! MEM DATA (R3) NOT = TEST DATA
;(R0), ADDRESS=(R2)-2

MOV (R2)+,R3 ;GET TEST DATA
CMP R0,R3 ;COMPARE WITH CHECK WORD
BEQ .+10
CLR -(SP) ;PUSH FAKE STATUS ON THE STACK
JSR PC,ERROR ;ERROR! MEM DATA (R3) NOT = TEST DATA
;(R0), ADDRESS=(R2)-2

MOV (R2)+,R3 ;GET TEST DATA
CMP R0,R3 ;COMPARE WITH CHECK WORD
BEQ .+10
CLR -(SP) ;PUSH FAKE STATUS ON THE STACK
JSR PC,ERROR ;ERROR! MEM DATA (R3) NOT = TEST DATA
;(R0), ADDRESS=(R2)-2

MOV (R2)+,R3 ;GET TEST DATA
CMP R0,R3 ;COMPARE WITH CHECK WORD
BEQ .+10
CLR -(SP) ;PUSH FAKE STATUS ON THE STACK
JSR PC,ERROR ;ERROR! MEM DATA (R3) NOT = TEST DATA
;(R0), ADDRESS=(R2)-2

DEC R5 ;DECREMENT WORD COUNT
BNE 35
DEC R4 ;DECREMENT BLOCK COUNT
BNE 25

BIT #40000,2*SWR ;CHECK LOOP SWITCH
BNE 15 ;LOOP CHECKING THIS PATTERN
ASL ICOUNT ;SHIFT PATTERN INDICATOR
BNE 15
MOV (SP)+,(SP) ;ADJUST STACK
MOV (SP)+,(SP)
RTS PC ;RETURN TO CALLER

.SBTTL GET TTY INPUT ROUTINE
;ROUTINE TO GET ASCII INPUT FROM TTY, AND CONVERT TO OCTAL.
;ROUTINE LEAVES THE FIRST 16 BITS IN ADDRESS FOLLOWING THE CALL
;AND THE LAST 2 BITS IN .1617 BELOW.
;CALL: JSR PC,RECD
RECD:
JSR PC,$SAVR ;GO SAVE REGISTERS ON THE STACK
JSR PC,INUM
MOV R2,TEMP2
MOV R3,TEMP3
JSR PC,$RESTR ;RESTORE REGISTERS FROM STACK
MOV (SP),TEMP1
MOV TEMP3,2TEMP1
MOV TEMP2,.1617

```

```

3412 011734 062716 000002
3413 011740 000207
3414 011742 000000
3415 011744 000000
3416 011746 000000
3417 011750 000000
3418
3419
3420
3421
3422 011752 042767 017777 166770
3423 011760 032737 000400 177570
3424 011766 001402
3425 011770 004767 000464
3426 011774 032737 001000 177570
3427 012002 001404
3428 012004 056767 166736 166736
3429 012012 000403
3430 012014 056767 166722 166726
3431 012022 016737 166722 177570
3432 012030 012767 040177 166706
3433 012036 032737 004000 177570
3434 012044 001402
3435 012046 105067 166672
3436 012052 000207
3437
3438
3439 012054 005015 047524 051040
3440 012062 051505 047524 042522
3441 012070 046040 040517 042504
3442 012076 051522 051440 040524
3443 012104 052122 040440 020124
3444 012112 033061 006462 000012
3445 012120 005015 047105 041101
3446 012126 042514 050040 051101
3447 012134 052111 037531 030440
3448 012142 030057 054475 051505
3449 012150 047057 020117 000
3450 012155 015 051412 040524
3451 012162 052122 047111 020107
3452 012170 040502 045516 021440
3453 012176 034050 037451 000040
3454 012204 005015 020043 043117
3455 012212 032040 020113 040502
3456 012220 045516 020123 047524
3457 012226 052040 051505 024124
3458 012234 024470 020077 000
3459 012241 015 050012 052101
3460 012246 042524 047122 021440
3461 012254 020077 000
3462 012257 015 037412 000
3463 012263 015 052012 050131
3464 012270 020105 047503 051516
3465 012276 040524 052116 000
3466 012303 015 044412 050116
3467 012310 052125 021440 047440

```

```

ADD #2,(SP)
RTS PC
.1617: .WORD 0
TEMP1: .WORD 0
TEMP2: .WORD 0
TEMP3: .WORD 0

;ROUTINE TO CHECK THE SWITCH REGISTER
;CHECK SWITCH 9: IF SET, LOAD ERROR COUNT INTO THE DISPLAY REGISTER;
;IF NOT SET, LOAD PASS COUNT INTO THE DISPLAY REGISTER
CKSWR: BIC #17777,LDISP ;SAVE RELOCATION BITS
        BIT #BIT8,#SWR ;CHECK SWITCH 8
        BEQ 10$ ;BRANCH IF SET
        JSR PC,REL24K ;GO RELOCATE PROGRAM BACK TO 4K AND STOP
10$: BIT #BIT9,#SWR ;SWITCH 9 SET ?
        BEQ 1$
        BIS ERCNT,LDISP ;LOAD ERROR COUNT
2$: BR 2$
1$: BIS ICNT,LDISP ;LOAD PASS COUNT
2$: MOV LDISP,#DISPLAY ;LOAD THE DISPLAY REGISTER
        MOV #040177,ICOUNT ;LOAD ITERATION COUNT WORD
        BIT #4000,#SWR ;CHECK SW11
        BEQ +6
        CLRB ICOUNT ;ICOUNT =040000 IF SW11 =1
        RTS PC

;MESSAGES
RESLDR: .ASCIZ <15><12>'TO RESTORE LOADERS START AT 162'<15>'12.'
PARITY: .ASCIZ <15><12>'ENABLE PARITY? 1/0=YES/NO '
STBANK: .ASCIZ <15><12>'STARTING BANK #(8)? '
BANKS: .ASCIZ <15><12>'# OF 4K BANKS TO TEST(8)? '
PAT: .ASCIZ <15><12>'PATTERN #'
QUEST: .ASCIZ <15><12>'?'
CONST: .ASCIZ <15><12>'TYPE CONSTANT'
PRG3M: .ASCIZ <15><12>'INPUT # OF 256. WORD BLOCKS TO TEST INSTEAD OF'

```

```

3468 012316 020106 032462 027066
3469 012324 053440 051117 020104
3470 012332 046102 041517 051513
3471 012340 052040 020117 042524
3472 012346 052123 044440 051516
3473 012354 042524 042101 047440
3474 012362 000106
3475 012364 005015 054524 042520
3476 012372 040440 042104 042522
3477 012400 051523 000
3478 012403 015 052012 020117
3479 012410 042522 052123 051117
3480 012416 020105 051120 043517
3481 012424 040522 020115 052123
3482 012432 051101 020124 052101
3483 012440 000040
3484 012442 000052
3485 012444 042104 040521 020102
3486 012452 047504 042516 000041
3487
3488
3489
3490 012460 010700
3491 012462 042700 017777
3492 012466 010067 000004
3493 012472 004567 173652
3494 012476 000000
3495 012500 000000
3496 012502 012706 000500
3497 012506 042737 100000 000750
3498 012514 013737 000750 177570
3499 012522 005037 000754
3500 012526 000005
3501 012530 005037 000176
3502 012534 000137 000162
3503
3504
3505
3506
3507
3508
3509
3510
3511
3512
3513
3514
3515
3516
3517
3518
3519
3520
3521
3522
3523

```

PRG4M: .ASCIZ <15><12>'TYPE ADDRESS'

RELOCM: .ASCIZ <15><12>'TO RESTORE PROGRAM START AT '

ASTERISK: .ASCIZ '*'
ENDMSG: .ASCIZ 'DDQAB DONE!'

.EVEN

:ROUTINE TO RELOCATE PROGRAM BACK TO 0

```

REL24K: MOV PC,RO ;FORM BASE ADDRESS WHERE CODE
        BIC #17777,RO ;IS RELOCATED
        MOV RO,15 ;PUT FROM ADDRESS INTO SUBROUTINE CALL
        JSR RS,RELOC ;RELOCATE CODE TO
15:      ;LOWEST 4K
        MOV #STKPTR,SP ;SET STACK PTR
        BIC #10000,2#LDDISP ;CLEAR RELOCATION INDICATOR
        MOV 2#LDDISP,2#DISPLAY ;LOAD DISPLAY REGISTER
        CLR 2#RELOCF ;CLEAR RELOCATION FACTOR
        RESET ;DISABLE MEM MGMT
        CLR 2#176 ;PUT A HALT AT 176
        JMP 2#162 ;RESTORE LOADERS & HALT AT 176

```

.SBTTL BRANCH GOBBLE MOS MEMORY TEST

*****PROGRAM DESCRIPTION*****

```

:THIS IS A PSEUDO-MODIFIED VERSION OF THE BRANCH GOBBLE
:MOS MEMORY EXERCISER. PSEUDO-MODIFIED BECAUSE THE
:ORIGINAL CODE, TAKEN FROM THE DZAKA-A INSTRUCTION
:EXERCISER, WHICH IS BRANCH GOBBLE IS INCLUDED
:HERE IN ITS ORIGINAL FORM, BUT MEMORY MANAGEMENT
:CAPABILITIES HAVE BEEN ADDED TO GIVE IT OPERATING
:ABILITIES IN A 0-128K MEMORY ENVIRONMENT.

```

*****OPERATING PROCEDURE*****

```

:WHEN LOADED THIS PROGRAM'S STARTING ADDRESS IS XXXXXX.
:WHEN RUNNING THE FOLLOWING STEPS ARE TAKEN:
:1.) A PROGRAM ID IS TYPED ON THE TTY:
        BRANCH GOBBLE MOS TEST
:1.5) THE PROGRAM DETERMINES IF THERE IS MOS PARITY. IF YES IT IS ENABLED

```


3524
3525
3526
3527
3528
3529
3530
3531
3532
3533
3534
3535
3536
3537
3538
3539
3540
3541
3542
3543
3544
3545
3546
3547
3548
3549
3550
3551
3552
3553
3554
3555
3556
3557
3558
3559
3560
3561
3562
3563
3564
3565
3566
3567
3568
3569
3570
3571
3572
3573
3574
3575
3576
3577
3578
3579

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

```

3580
3581
3582
3583
3584
3585
3586
3587
3588
3589
3590
3591
3592
3593
3594
3595
3596
3597
3598
3599
3600
3601 012540
3602
3603      077406
3604
3605
3606
3607
3608
3609
3610
3611 012540 000000
3612
3613
3614 012542 005067 165352
3615 012546 005067 166200
3616 012552 004567 166202
3617 012556 015310
3618 012560 004767 167254
3619 012564 004767 001566
3620 012570
3621 012570 004567 166164
3622 012574 015243
3623
3624 012576 105737 177560
3625 012602 100375
3626 012604 113746 177562
3627 012610 042716 177600
3628 012614 004767 001520
3629 012620 005726
3630 012622 022766 000131 177776
3631 012630 001513
3632
3633 012632 022766 000116 177776
3634 012640 001406
3635 012642 012746 000077
    
```

```

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.

TOP:
;CONSTANTS:
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.

LODFLG: .WORD 0

BRANCH: CLR RELFL ;INITIALIZE THE RELOCATION INDICATOR.
CLR MMAVA ;INITIALIZE THE MEM. MANAGEMENT FLAG.
JSR RS,$PRINT ;GO TO PRINT ROUTINE
.WORD IDMESS
JSR PC,$LDR
15$: JSR PC,MOSPAR
1$: JSR RS,$PRINT ;GO TO PRINT ROUTINE
.WORD MMMESS ;MEMORY MANAGEMENT SHOULD
;BE ENABLED
2$: TST @TKS ;WAIT FOR A CHARACTER.
BPL 2$
MOVB @TKB,-(SP) ;GET THE CHARACTER.
BIC #177600,(SP)
JSR PC,TYPIT ;ECHO THE CHARACTER.
TST (SP)+ ;RESET THE STACK.
CMP #'Y,-2(SP) ;IF IT IS Y, THEN GO
BEQ YMMBGO ;TO YMMBGO TO ENABLE
;MEMORY MANAGEMENT
CMP #'N,-2(SP) ;IS IT N?
BEQ 3$
MOV #'?,-(SP) ;IF IT WAS NIETHER
    
```

```

3636 012646 004767 001466 JSR PC,TYPIT ;Y OR N THEN ASK
3637 012652 005726 TST (SP)+ ;THE USER AGAIN.
3638 012654 000745 BR 1$
3639 012656 004767 001066 3$: JSR PC,LIMITS ;GO GET THE LIMITING
3640 ;ADDRESSES FOR THE TEST.
3641 ;THEY WILL BE LEFT AS
3642 ;BLOCK NUMBERS IN LOLIM
3643 ;AND HILIM.
3644 012662 022767 001600 001172 SPOT: CMP #1600,HILIM
3645 012670 002010 BGE SPOT2
3646 012672 012700 177770 MOV #SPOT--,RO
3647 012676 060700 ADD PC,RO
3648 012700 062700 177772 ADD #-6,RO
3649 012704 010046 MOV RO,-(SP)
3650 012706 000167 001140 JMP LIMERR
3651 012712 005046 SPOT2: CLR -(SP)
3652 012714 012746 002502 MOV #FIRST1--,-(SP)
3653 012720 060716 ADD PC,(SP)
3654 012722 062716 177772 ADD #-6,(SP)
3655 012726 004767 000774 JSR PC,LOSEG4
3656 012732 004767 000436 JSR PC,LOSEG1
3657 012736 012746 000052 REPET1: MOV #'*,-(SP)
3658 012742 004767 001372 JSR PC,TYPIT
3659 012746 005726 TST (SP)+
3660 012750 022767 000200 001104 CMP #200,HILIM
3661 012756 002026 BGE REPET3
3662 012760 016700 001076 MOV HILIM,RO
3663 012764 004767 000054 4$: JSR PC,ROT
3664 012770 010067 002556 MOV RO,HI ;SET THE PARAMETERS
3665 ;IN THE ACTUAL TEST
3666 ;ROUTINE.
3667 012774 016700 001064 MOV LOLIM,RO ;DO THE SAME FOR THE
3668 ;LOW ADDRESS LIMIT.
3669 013000 004767 000040 JSR PC,ROT
3670 013004 010067 002544 MOV RO,LO
3671 013010 005037 000036 CLR #36 ;SET UP THE VECTORS
3672 013014 012704 000020 MOV #REPET3--,R4 ;FOR AN INTERRUPT
3673 013020 060704 ADD PC,R4 ;FROM A TRAP INSTRUCTION
3674 013022 062704 177772 ADD #-6,R4 ;WHICH WILL BE USED TO
3675 013026 010437 000034 MOV R4,#34 ;RETURN FROM THE TEST
3676 ;ROUTINE WHEN IT IS DONE.
3677 013032 000002 RTI ;START THE TEST.
3678 013034 004767 000404 REPET3: JSR PC,LOSEG2
3679 013040 000167 177672 JMP REPET1
3680 ;
3681 ;THIS ROUTINE IS CALLED TO SHIFT RO TO THE LEFT SIX BITS.
3682 013044 012701 177772 ROT: MOV #-6,R1
3683 013050 006300 1$: ASL RO
3684 013052 005201 INC R1
3685 013054 002775 BLT 1$
3686 013056 000207 RTS PC
3687 ;
3688 ;YMMBGO SETS UP FOR USING MEMORY MANAGEMENT TO
3689 ;DO THE MOS TEST.
3690 ;ALL THE MEMORY MANAGEMENT REGISTERS WHICH ARE
3691 ;TO REMAIN STATIC FOR THE TESTS DURATION ARE SET.

```

```

3692 ; THE LIMITS ARE THEN GOTTEN AND CHECKED FOR VALIDITY.
3693 ; THE TRAP INTERRUPT VECTORS ARE SET AND YMMBG1
3694 ; IS CALLED. YMMBG1 IS A ROUTINE THE SETS THOSE
3695 ; MEMORY MANAGEMENT REGISTERS WHICH NEED TO
3696 ; BE CHANGED DYNAMICALLY DURING THE TEST.
3697
3698 013060 012767 177777 165664 YMMBG0: MOV #-1,MMAVA
3699 013066 005037 177572 CLR @#SRO ;SET ALL THE STATIC
3700 013072 012700 172340 MOV #KIPARO,R0 ;REGISTERS.
3701 013076 012701 172300 MOV #KIPDRO,R1
3702 013102 005003 CLR R3
3703 013104 012704 177770 MOV #-10,R4
3704 013110 010320 1$: MOV R3,(R0)+
3705 013112 062703 000200 ADD #200,R3
3706 013116 012721 077406 MOV #KPOR,(R1)+
3707 013122 005204 INC R4
3708 013124 001371 BNE 1$
3709 013126 004767 000616 JSR PC,LIMITS ;GET THE LIMITS FOR THE PENDING
3710 ;TEST. THEY ARE LEFT IN BLOCK NUMBER
3711 ;FORM IN HILIM AND LOLIM.
3712 013132 004767 000570 JSR PC,LOSEG4 ;SEE IF PERMANENT RELOCATION IS
3713 ;APPROPRIATE FOR THIS TEST SPAN.
3714 013136 012737 007600 172356 MOV #7600,@#KIPAR7 ;MAP THE UNIBUS DEVICE PAGE INTO
3715 ;INTO HIGH VIRTUAL MEMORY.
3716 013144 005046 REPET2: CLR -(SP)
3717 013146 012746 002250 MOV #FIRST1-.,-(SP)
3718 013152 060716 ADD PC,(SP)
3719 013154 062716 177772 ADD #-6,(SP) ;SET UP THE STACK
3720 ;TO SIMULATE THE OCCURRENCE OF AN
3721 ;AN INTERRUPT SO THAT THE TEST CAN
3722 ;BE STARTED USING AN RTI.
3723 013160 016767 000704 000674 MOV HISAV,HILIM
3724 013166 016767 000674 000670 MOV LOSAV,LOLIM
3725 013174 004767 000174 JSR PC,LOSEG1
3726 013200 012746 000052 MOV #*,-(SP)
3727 013204 004767 001130 JSR PC,TYPIT
3728 013210 005726 TST (SP)+
3729 013212 012737 013224 000034 MOV #YMMBG1,@#34
3730 013220 005037 000036 CLR @#36 ;SET UP THE TRAP INTERRUPT VECTOR
3731 ;WHICH WILL BE USED TO RETURN FROM THE
3732 ;TESTING ROUTINE.
3733
3734 ; YMMBG1 IS USED TO DYNAMICALLY ALLOCATE MEMORY UNDER MEMRY MANAGEMENT
3735 ; WHILE A TEST IS IN PROGRESS. WORKING UPWARDS FROM THE LOLIM
3736 ; MEMORY MANAGEMENT IS SET TO ENABLE BRGOB TO WORK THROUGH AS
3737 ; MUCH OF THE TEST SPAN AS POSSIBLE IN A SINGLE MANAGEMENT SET UP
3738 ; BEFORE HAVING TO RESET THE MANAGEMENT REGISTERS.
3739 ; BRGOB ALWAYS IS IN LOW VIRTUAL MEMORY AND UPPER VIRTUAL
3740 ; ADDRESSES ARE ALWAYS MAPPED INTO UNIBUS DEVICE ADDRESSES.
3741
3742 013224 005037 177572 YMMBG1: CLR @#SRO
3743 013230 026767 000626 000626 CMP HILIM,LOLIM ;IS THE TEST DONE?
3744 013236 101451 BLOS DONIT ;YES THEN BRANCH.
3745 013240 012700 172342 MOV #KIPARI,R0 ;ELSE GET READY TO SET
3746 013244 016705 000614 MOV LOLIM,R5 ;THE KERNAL PAGE ADDRESS REGISTERS.
3747 013250 012701 177772 MOV #-6,R1

```

```

3748 013254 012767 020000 002272          MCV      #20000,LO
3749 013262 010520          1$:      MOV      RS,(R0)+      ;RESET THE KIPAR'S
3750 013264 062705 000200          2$:      ADD      #200,RS
3751 013270 026705 000566          25$:     CMP      HILIM,RS      ;REACHED HILIM?
3752 013274 101407          3$:      BLOS    3$           ;YES, GOTO 3$.
3753 013276 005201          INC      R1           ;NO, INCREMENT R1 AND SEE IF ALL THE
3754          BLT      1$           ;KIPAR'S HAVE BEEN SET.
3755 013300 002770          BLT      1$           ;ALL THE KIPAR'S HAVE NOT BEEN SET SO
3756          MOV      RS,HI      ;LOOP TO GET THE NEXT ONE.
3757 013302 010567 002244          MOV      RS,HI      ;DO THIS IF ALL THE TEST SPAN HAS NOT
3758          BR      4$           ;BEEN ALLOCATED TO SOME VIRTUAL ADDRSSES
3759          MOV      HI,HI      ;IS THE KERNAL INSTRUCTION SPACE.
3760 013306 162705 000002          SUB      #2,RS
3761 013312 000403          BR      4$
3762 013314 016767 000542 002230 3$:     MOV      HILIM,HI      ;DO THIS IF ALL THE TEST SPAN HAS BEEN
3763          BR      4$           ;ALLOCATED TO THE VIRTUAL KERNAL SPACE
3764          MOV      HI,HI      ;JUST ALLOCATED.
3765 013322 166767 000536 002222 4$:     SUB      LOLIM,HI      ;COMPUTE THE VIRTUAL LIMIT
3766          MOV      HI,RO      ;OF THE TEST SPAN.
3767          JSR      PC,ROT
3768 013334 004767 177504          JSR      PC,ROT
3769 013340 062700 020000          ADD      #20000,RO
3770 013344 010067 002202          MOV      RO,HI
3771 013350 010567 000510          MOV      RS,LOLIM
3772 013354 005237 177572          INC      #5$RO
3773 013360 000002          RTI
3774          ;TURN ON MEMORY MANAGEMENT.
3775          ;RETURN TO BR008 TO
3776          ;PERFORM THE TEST IN
3777          ;THE SPAN INDICATED
3778          ;BY THE RESULT OF THE
3779          ;ABOVE
3778 013362 004767 000056          DONIT:   JSR      PC,LOSEG2
3779 013366 022626          CMP      (SP)+(SP)+
3780 013370 000167 177550          JMP      REPET2      ;TEST COMPLETED, SO
3781          ;RESTART
3782          ;
3783          ;LOSEG1 IS CALLED TO DECIDE WHETHER OR NOT THE LIMITS ARE SUCH THAT
3784          ;THE FIRST 4K OF MEMORY WILL HAVE TO BE CHECKED IN THE TEST. THAT
3785          ;IS DO THE LIMITS INCLUDE ADDRESSES WHICH LIE IN THE FIRST 4K BLOCK
3786          ;OF MEMORY THUS REQUIRING THAT THE CONTENTS OF THIS FIRST BLOCK OF
3787          ;MEMORY BE MOVED INTO THE SECOND 4K BLOCK SO THAT THE TEST CAN BE
3788          ;RUN THROUGH THE FIRST 4K BLOCK. IF RELOCATION IS NECESSARY LOSEG1
3789          ;SETS LOSFL TO -1 AND RESETS THE LIMITS, LOLIM AND HILIM,
3790          ;APPROPRIATELY.
3791          ;
3792 013374 005067 000324          LOSEG1: CLR      LOSFL      ;INITIALIZE.
3793 013400 026727 000462 000200          CMP      LOSAV,#200    ;SEE IF THE LOW LIMIT OR THE HIGH LIMIT
3794 013406 103011          BHIS    2$           ;LIE IN THE FIRST 4K BLOCK OF MEMORY.
3795 013410 005367 000310          DEC      LOSFL
3796 013414 022767 000200 000446          CMP      #200,HISAV
3797 013422 103004          BHIS    3$
3798 013424 012767 000200 000432 1$:     MOV      #200,LOLIM
3799 013432 000207          2$:     RTS      PC
3800 013434 012767 000200 000420 3$:     MOV      #200,HILIM
3801 013442 000770          BR      1$
3802          ;
3803          ;LOSEG2 IS CALLED TO DO THE ACTUAL RELOCATION OF THE FIRST 4K MEMORY

```

```

3804
3805
3806
3807
3808 013444 005767 000254
3809 013450 100401
3910 013452 000207
3811 013454 012667 000242
3812 013460 005737 000120
3813 013464 100423
3814 013466 004567 172656
3815 013472 000000
3816 013474 020000
3817 013476 012737 177777 000120
3818 013504 012701 020000
3819 013510 060106
3820 013512 060116
3821 013514 060107
3822 013516 060137 000114
3823 013522 060167 000174
3824 013526 012767 177777 164364
3825 013534 016700 000326
3826 013540 004767 177300
3827 013544 020027 000320
3828 013550 002002
3829 013552 012700 000320
3830 013556 010067 001772
3831 013562 016700 000302
3832 013566 022700 000200
3833 013572 002002
3834 013574 012700 000200
3835 013600 020027 000004
3836 013604 003002
3837 013606 000167 000034
3838 013612 004767 177226
3839 013616 010067 001730
3840
3841
3842 013622 012701 000024
3843 013626 060701
3844 013630 062701 177772
3845 013634 010137 000034
3846 013640 005037 000036
3847 013644 000002
3848
3849
3850
3851 013646 016746 000050
3852 013652 005767 000042
3853 013656 100417
3854 013660 004567 172464
3855 013664 020000
3856 013666 000000
3857 013670 005037 000120
3858 013674 012701 020000
3859 013700 160106

```

```

;BANK INTO THE SECOND 4K MEMORY BANK, THEN RUN THE TEST
;THROUGH THE DESIGNATED PARTS OF THE FIRST BANK AND THEN RESTORE
;THE CONTENTS OF THE FIRST BANK BY MOVING THE SECOND BANK'S
;CONTENTS BACK INTO THE FIRST BANK.
LOSEG2: TST      LOSFL      ;SEE IF RELOCATION IS NECESSARY.
        BMI      1$        ;IF NOT RETURN.
        RTS      PC
1$:     MOV      (SP)+,SAVPC
        TST      @#120      ;SEE IF THE PROGRAM IS ALREADY RELOCATED
        BMI      14$
        JSR      RS,RELOC
        .WORD    0
        .WORD    20000
        MOV      @-1,@#120
        MOV      @20000,R1
        ADD      R1,SP
        ADD      R1,(SP)
        ADD      R1,PC
        ADD      R1,@#114
        ADD      R1,SAVPC
        MOV      @-1,RELF
14$:    MOV      LOSAV,RO      ;ESTABLISH VALID LIMITS FOR THE TEST
        JSR      PC,ROT      ;THROUGH THE FIRST 4K MEMORY BANK.
        CMP      RO,@#320
        BGE      15$
        MOV      @#320,RO
15$:    MOV      RO,LO
        MOV      HISAV,RO
        CMP      @#200,RO
        BGE      2$
        MOV      @#200,RO
2$:     CMP      RO,@#4
        BGT      3$
        JMP      LOSEG3
3$:     JSR      PC,ROT
        MOV      RO,HI
        ;
        ;
        MOV      @LOSEG3--,R1
        ADD      PC,R1
        ADD      @-6,R1
        MOV      R1,@#34
        CLR      @#36
        ;PERFORM THE TEST
        ;
        ;LOSEG3 RELOCATES BACK INTO THE FIRST 4K MEMORY BANK.
LOSEG3: MOV      SAVPC,-(SP)
        TST      PERRFL
        BMI      1$
        JSR      RS,RELOC
        .WORD    20000
        .WORD    0
        CLR      @#120
        MOV      @20000,R1
        SUB      R1,SP

```

```

3860 013702 160116
3861 013704 160137 000114
3862 013710 160107
3863 013712 160166 000002
3864 013716 000207
3865 013720 000000
3866 013722 000000
3867 013724 000000
3868
3869
3870 013726 005067 177766
3871 013732 026727 000132 000200
3872 013740 101002
3873 013742 005367 177752
3874 013746 000207
3875
3876
3877
3878
3879
3880
3881
3882
3883
3884
3885
3886
3887 013750
3888 013750 004567 165004
3889 013754 015206
3890 013756 004767 000110
3891 013762 004767 000334
3892 013766 004767 000330
3893 013772 010367 000064
3894 013776 004567 164756
3895 014002 015226
3896 014004 004767 000062
3897 014010 004767 000306
3898 014014 004767 000302
3899 014020 010367 000040
3900 014024 026767 000032 000032
3901 014032 003407
3902
3903 014034 016767 000022 000026
3904 014042 016767 000016 000016
3905 014050 000207
3906 014052
3907 014052 004567 164702
3908 014056 015273
3909 014060 000733
3910
3911 014062 000000
3912 014064 000000
3913
3914 014066 000000
3915 014070 000000

```

```

SUB R1,(SP)
SUB R1,20114
SUB R1,PC
SUB R1,2(SP)
RTS PC
IS:
PERRFL: .WORD 0
SAVPC: .WORD 0
LOSFL: .WORD 0
:
LOSEG4: CLR PERRFL
CMP HISAV,#200
BHI IS
DEC PERRFL
IS:
RTS PC
:
:LIMITS IS CALLED TO ASK THE USER FOR BOTH
:THE HIGH AND LOW UNIBUS ADDRESS LI,ITS FOR
:THE IMPENDING TEST. THE TWO LIMITS ARE LEFT IN
:BLOCK NUMBER FORM AT LOCATIONS HILIM AND
:LOLIM (THEY ARE ALSO PUT IN LOSAV AND HISAV FOR
:LATER USE BY THE ROUTINE DONE). A VALIDITY CHECK
:IS MADE TO MAKE SURE THE INDICATED SPAN
:IS A VALID TEST SPAN.
:LIMITS:
JSR RS,SPRINT ;GO TO PRINT ROUTINE
.WORD HIMESS
JSR PC,INUM ;ASSEMBLE THIS NUMBER.
JSR PC,THRR
JSR PC,THRR
MOV R3,HILIM
JSR RS,SPRINT ;GO TO PRINT ROUTINE
.WORD LOMESS
JSR PC,INUM ;ASSEMBLE THIS NUMBER
JSR PC,THRR
JSR PC,THRR
MOV R3,LOLIM
CMP HILIM,LOLIM ;IF LOLIM IS GREATER
BLE LIMERR ;THAN HILIM THEN GOTC
LIMERR, ERROR
2$: MOV HILIM,HISAV ;STORE THE LIMITS IN
MOV LOLIM,LOSAV ;SAVE REGISTERS.
RTS PC ;RETURN
LIMERR: JSR RS,SPRINT ;GO TO PRINT ROUTINE
.WORD ERMESS ;WRITE AN ERROR MESSAGE
BR LIMITS ;AND TRY AGAIN.
:
HILIM: .WORD 0
LOLIM: .WORD 0
:THESE ARE INTERMEDIATE STORGE REGISTERS:
LOSAV: .WORD 0
HISAV: .WORD 0

```

```

3916
3917
3918
3919
3920
3921
3922
3923
3924
3925
3926
3927
3928
3929
3930
3931
3932
3933 014072 005046
3934 014074 005002
3935 014076 005003
3936 014100 012705 177771
3937 014104 005737 177562
3938 014110 012746 000076
3939 014114 004767 000220
3940 014120 005726
3941 014122 105737 177560
3942 014126 100375
3943 014130 013746 177562
3944 014134 042715 177600
3945 014140 022716 000177
3946 014144 001011
3947 014146 005726
3948 014150 005716
3949
3950
3951 014152 001763
3952
3953
3954
3955
3956 014154 012716 000134
3957 014160 004767 000154
3958 014164 005726
3959 014166 000755
3960 014170 004767 000144
3961
3962
3963 014174 022716 000015
3964 014200 001350
3965
3966 014202 012716 000012
3967 014206 004767 000126
3968 014212 005726
3969 014214 005716
3970 014216 001415
3971
    
```

```

: THIS ROUTINE IS CALLED TO ASSEMBLE AN 18-BIT
: NUMBER FROM THE TTY AND TRUNCATE IT DOWN TO
: 12-BITS
: A CALL IS MADE THUS:
:     JMP     PC, INUM
: RES:     .WORD 0
: THE NUMBER IS ASSEMBLED AND THE RESULTING 12-BIT
: TRUNCATED NUMBER IS LEFT IN RES. WHEN AND
: RTS RETURN IS MADE.
: NOTE THAT THE NUMBER SHOULD BE SPECIFIED
: IN OCTAL DIGITS. AN CHARACTERS WHICH DO NOT
: MEET THIS SPECIFICATION IN THE INPUT STRING
: WILL CAUSE AN ERROR WHICH WILL BE SIGNALLED BY
: A ? ON THE TTY FOLLOWED BY A RETRY.

: NUM:  CLR     -(SP)           : PUT A ZERO MARKER ON
:       CLR     R2             : SET UP THE TEMPORARY
:       CLR     R3             : STORAGE AND COUNTER
:       MOV     #7, R5         : REGISTERS.
:       TST     @TKB
:       MOV     #'-(SP)
:       JSR     PC, TYPIT
:       TST     (SP)+
1$:    TSTB     @TKS           : WAIT FOR A CHARACTER
:       BPL     1$
:       MOV     @TKB, -(SP)    : GET IT ONTO THE
:       BIC     #177600, (SP)  : STACK
:       CMP     #177, (SP)     : IS IT RUBOUT?
:       BNE     2$            : IF NOT GOTO 2$
:       TST     (SP)+         : IF IT WAS A RUBOUT
:       TST     (SP)          : FIRST SEE IF THEE
:                               : IS A PREVIOUS
:                               : CHARACTER ON THE STACK.
:                               : IF THERE WAS NO PREVIOUS
:                               : CHARACTER TAKE NO
:                               : RUBOUT ACTION AND
:                               : GO WAIT AT 1$ FOR
:                               : THE NEXT CHARACTER
:                               : IF THERE WAS
:                               : A PREVIOUS CHARACTER
:                               : PRINT A SLASH
2$:    JSR     PC, TYPIT      : IF THE LAST INPUT
:                               : CHARACTER WAS NOT
:                               : RUBOUT ECHO IT
:                               : IS IT CR.
:                               : NO, BRANCH TO 1$ FOR
:                               : NEXT CHARACTER.
:                               : YES, PRINT A LF
3$:    TST     (SP)          : START TO ASSEMBLE
:       BEG     4$           : THE NUMBER. IF THE
:                               : STACK IS AT THE
    
```



```

3972
3973
3974 014220 012604          MOV      (SP)+,R4
3975 014222 062704 177710   ADD      #-70,R4
3976 014226 002022          BGE     INERR
3977 014230 062704 000010   ADD      #10,R4
3978 014234 002417          BLT     INERR
3979 014236 005205          INC     RS
3980 014240 001415          BEQ     INERR
3981 014242 004767 000054   JSR     PC,THRR
3982
3983
3984
3985 014246 010402          MOV     R4,R2
3986
3987 014250 000761          BR      3$
3988
3989 014252 005205          4$: INC     RS
3990 014254 001403          BEQ     5$
3991 014256 004767 000040   JSR     PC,THRR
3992 014262 000773          BR      4$
3993 014264 004767 000022   5$: JSR     PC,ONER
3994 014270 005726          TST     (SP)+
3995 014272 000207          RTS     PC
3996
3997
3998
3999 014274
4000 014274 004567 164460   INERR: JSR     RS,SPRINT ;GO TO PRINT ROUTINE
4001 014300 015237          .WORD  INAMES
4002 014302 005726          1$: TST     (SP)+ ;CLEAR JUNK OFF THE
4003 014304 001376          BNE     1$ ;STACK
4004 014306 000167 177560   JMP     INUM ;THE ROUTINE LIMITS.
4005 014312 000241          ONER: CLC
4006 014314 006002          ROR     R2 ;ROUTINE WHICH ROTATES
4007 014316 006003          ROR     R3 ;TEMP2 AND TEMP3 ONE
4008 014320 000207          RTS     PC ;BIT TO THE RIGHT
4009
4010 014322 004767 177764   THRR: JSR     PC,ONER ;TREATING THEM AS A 32-BIT
4011 014326 004767 177760   JSR     PC,ONER ;OPERAND.
4012 014332 004767 177754   JSR     PC,ONER ;ROUTINE WHICH CALL
4013 014336 000207          RTS     PC ;ONER 3 TIMES.
4014 014340 105737 177564   TYPIT: TSTB  @#TPS ;TYPIT TAKES THE
4015 014344 100375          BPL     TYPIT ;WORD 2 BYTES UP IN
4016 014346 116637 000002 177566   MOVB   2(SP),@#TPB ;THE STACK AND "PRINTS"
4017 014354 000207          RTS     PC ;IT ON THE TTY
4018
4019
4020
4021
4022
4023
4024 172100
4025 014356 004767 000454   PAREGS=172100
4026 014362 012700 172100   MOSPAR: JSR     PC,MPVECT ;SET THE PARITY ERROR TRAP VECTOR.
4027
4027          MOV     #PAREGS,R0 ;GET READY TO LOOK AT THE POSSIBLE
;PARITY REGISTERS PRESENT.
;MOSPAR IS CALLED TO CHECK OUT THE POSSIBILITY OF TURNING ON MOS
;PARITY DURING THE BRANCH GOBBLE TEST. FIRST MOSPAR SEES WHAT
;MOS PARITY REGISTERS EXIST AND THEN IFOME ARE FOUND THEY
;ARE ENABLED AND THE INTERRUPT VECTOR IS SET TO TRAP TO PARERR A ROUTINE
;WHICH WILL NOTIFY THE USER OF THE ERROR AND ITS LOCATION.

```

```

4028 014366 012701 177762
4029 014372 012737 014454 000004
4030 014400 005037 000006
4031 014404 005067 000110
4032
4033
4034
4035
4036
4037
4038 014410 005710
4039
4040
4041 014412 005767 000102
4042 014416 100403
4043 014420 004567 164334
4044 014424 015102
4045
4046 014426 010046
4047 014430 004767 165306
4048 014434 004567 164320
4049 014440 015122
4050 014442 005367 000052
4051 014446 012710 000001
4052 014452 000401
4053
4054
4055 014454 022626
4056 014456 062700 000002
4057 014462 005201
4058 014464 002751
4059 014466 005767 000026
4060
4061 014472 100403
4062 014474 004567 164260
4063 014500 015131
4064 014502 012737 001116 000004
4065 014510 012737 000002 000006
4066 014516 000207
4067 014520 000000
4068
4069
4070
4071
4072
4073
4074
4075
4076
4077
4078
4079
4080
4081
4082 014522
4083 014522 004567 164232
    
```

```

MOV      #16,R1
MOV      #MP2,2#4
CLR      2#6
CLR      MPFL
;THE REAL REGISTERS ARE LOCATED USING A TIME OUT PLAN WHERE BY THE
;POSSIBILITIES ARE REFERENCED AND IF A TIME OUT OCCURS THEY ARE NONEXISTENT
;OR IF NO TIME OUT OCCURS THEN THE REGISTER IS REAL. WHEN A REAL
;REGISTER IS FOUND IT IS WRITTEN INTO AND READ TO DETERMINE
;WHAT KIND OF PARITY REGISTER IT IS, CORE OR MOS. IF IT IS MOS THEN THE
;REGISTER IS SET TO ENABLE PARITY.
MP1:     TST      (R0)
;IF THIS INSTRUCTION TIMES OUT THEN
;THERE IS NO PARITY REGISTER AT THE
;ADDRESS IN R0.
;SEE IF THE TABLE HEADING HAS BEEN OUTPUT.
TST      MPFL
BMI      1$
JSR      R5,$PRINT ;GO TO PRINT ROUTINE
        .WORD    MPMS
!$:      MOV      R0,-(SP) ;OUTPUT AN ENTRY INTO THAT TABLE.
        JSR      PC,02A
        JSR      R5,$PRINT ;GO TO PRINT ROUTINE
        .WORD    $CRLF
DEC      MPFL
MOV      #1,(R0)
BR       MP25
;IF A TIME OCCURS COME HERE. OR IF THE REGISTER WAS NOT MOS PARITY.
MP2:     CMP      (SP)+,(SP)+
MP25:    ADD      #2,R0
        INC      R1
        BLT      MP1 ;BRANCH IF NOT DONE.
        TST      MPFL ;SEE IF ANY MOS PARITY REGISTERS HAVE
;BEEN FOUND.
        BMI      MP3 ;IF NOT DON'T OUT PUT THE NO PARITY MESSAGE.
        JSR      R5,$PRINT ;GO TO PRINT ROUTINE
        .WORD    NOMPAR
MP3:     MOV      #ERRTRP,2#4
        MOV      #2,2#6
        RTS      PC
MPFL:    .WORD    0
;IF A PARITY ERROR IS DETECTED THEN A TRAP IS MADE THROUGH LOCATION
;114 TO THIS ROUTINE. HERE THE USER IS TOLD OF THE PARITY ERROR AND THE
;LOCATION PLUS 2 OF THE INSTRUCTION WHICH CAUSED THE ERROR TO COME OUT.
;THEN A SCAN IS MADE THROUGH MEMORY. IF THE ERROR DOESN'T COME UP DURING
;THE SCAN THEN THE USER IS TOLD THAT THE ERROR WAS NOT FOUND ON THE SCAN.
;IF THE ERROR IS DETECTED ON THE SCAN THEN THE USER IS TOLD WHAT LOCATION
;CAUSED THE ERROR AND WHETHER OR NOT MEMORY MANAGEMENT WAS ON DURING THE
;SCAN. IF IT WAS THEN THE PAR INVOLVED IN RELOCATING THE BAD LOCATION'S
;ADDRESS IS ALSO GIVEN TO THE USER SO HE CAN TAKE THE GIVEN
;ADDRESS AND RELOCATE IT USING THE PAR GIVEN TO FIND THE REAL MEMORY
;ADDRESS CAUSING THE PARITY ERROR. WHEN THIS HAS BEEN DONE THE TEST WHICH
;WAS IN PROGRESS WHEN THE ERROR WAS ENCOUNTERED IS RESTARTED. USING
;THE PARAMETERS WHICH THE USER LAST INPUT TO THE PROGRAM.
PARER2: JSR      R5,$PRINT ;GO TO PRINT ROUTINE
    
```


| | | | | | | | |
|------|--------|--------|--------|---------|-------|------------|---------------------------------------------|
| 4140 | 014742 | 000403 | | BR | RETPR | | ;GO RESTART THE TEST WHICH WAS IN PROGRESS. |
| 4141 | | | | | | | |
| 4142 | 014744 | | | 2s: | | | |
| 4143 | 014744 | 004567 | 164010 | | JSR | R5,SPRINT | ;GO TO PRINT ROUTINE |
| 4144 | 014750 | 015173 | | | .WORD | NOKT11 | ;THE SCAN SO TELL |
| 4145 | 014752 | 004567 | 164002 | RETPR: | JSR | R5,SPRINT | |
| 4146 | 014756 | 015122 | | | .WORD | \$CALF | |
| 4147 | 014760 | 016703 | 000002 | | MOV | SAVPER,R3 | |
| 4148 | 014764 | 000714 | | | BR | CONT | |
| 4149 | | | | | | | ;THE ADDRESS GIVEN FOR THE ERROR, THEN |
| 4150 | | | | | | | ;RESTART THE TEST WHICH WAS IN PROGRESS |
| 4151 | | | | | | | ;WHEN THE ERROR OCCURRED. |
| 4152 | 014766 | 000000 | | SAVPER: | .WORD | 0 | |
| 4153 | | | | | | | |
| 4154 | 014770 | | | PAR4: | | | |
| 4155 | 014770 | 004567 | 163764 | | JSR | R5,SPRINT | ;GO TO PRINT ROUTINE |
| 4156 | 014774 | 015062 | | | .WORD | NOTIND | ;TELL THE USER THAT IS THE CASE AND |
| 4157 | | | | | | | ;RESTART THE SCAN WHICH WAS IN PROGRESS |
| 4158 | | | | | | | ;WHEN THE ERROR WAS ORIGINALLY |
| 4159 | | | | | | | ;ENCOUNTERED. |
| 4160 | | | | : | | | |
| 4161 | 014776 | 012706 | 000500 | | MOV | #500,SP | ;RESTART THE TEST WHICH WAS IN PROGRESS |
| 4162 | 015002 | 000005 | | | RESET | | ;WHEN THE ORIGINAL PARITY ERROR WAS FIRST |
| 4163 | 015004 | 004737 | 000124 | | JSR | PC,2*WHERE | ;ENCOUNTERED. SEE IF RELOCATION INTO THE |
| 4164 | 015010 | 012707 | 015014 | | MOV | #1\$,PC | ;FIRST 4K BANK IS NECESSARY. |
| 4165 | | | | is: | | | |
| 4166 | 015014 | 004767 | 177336 | | JSR | PC,MOSPAR | ;GO RESET THE PARITY REGISTERS BEFORE |

```

4167 015020 005767 163726      TST      MMAVA      ; STARTING EITHER A MEMORY MANAGED
4168 015024 001402                BEQ      2$          ; OR NON MEMORY MANAGED TEST.
4169 015026 000167 176112      JMP      REPET2
4170 015032 000167 175654      2$:      JMP      SPOT2
4171
4172
4173
4174
4175
4176
4177 015036 012700 177464      ; ROUTINE USED TO SET THE TRAP VECTOR 114.
4178 015042 060700      MPVECT: MOV      #PARER2-.,RO
4179 015044 062700 177772      ADD      PC,RO
4180 015050 010037 000114      ADD      #-6,RO
4181 015054 005037 000116      MOV      RO,#114
4182 015060 000207      CLR      @#116
4183
4184
4185 015062 041523 047101 041440      ; MESSAGES USED FOR THESE PARITY ROUTINES.
4186 015070 046517 046120 052105      NOTIND: .ASCIZ  'SCAN COMPLETE'<15><12>
4187 015076 006505 000012
4188 015102 005015 040520 044522      MPMES:  .ASCII  <15><12>'PARITY ENABLED'
4189 015110 054524 042440 040516
4190 015116 046102 042105
4191 015122 005015 000
4192 015125 120 036503 000      $CRLF:  .ASCIZ  <15><12>
4193 015131 015 047012 020117      PERMES:  .ASCIZ  'PC='
4194 015136 040520 044522 054524      NOMPAR:  .ASCIZ  <15><12>'NO PARITY'<15><12>
4195 015144 005015 000
4196
4197
4198
4199 015147 040 040510 030104      ; THESE ARE MESSAGES USED FOR COMMUNICATIONS
4200 015154 005015 052113 030461      ; ON THE TTY BY THE PROGRAM.
4201 015162 047440 020116 040520      LOCBAD:  .ASCIZ  'HAD'
4202 015170 036522 000      KPARAM:  .ASCIZ  <15><12>'KT11 ON PAR='
4203 015173 015 045412 030524      NOKT11:  .ASCIZ  <15><12>'KT11 OFF'
4204 015200 020061 043117 000106
4205 015206 005015 044510 044107      HIMESS:  .ASCIZ  <15><12>'HIGH LIMIT?'<15><12>
4206 015214 046040 046511 052111
4207 015222 006477 000012
4208 015226 047514 020127 044514      LOMESC:  .ASCII  'LOW LIMIT'
4209 015234 044515 124
4210 015237 077 005015 000      INRMES:  .ASCIZ  '?'<15><12>
4211 015243 015 052412 042523      MMESS:  .ASCII  <15><12>'USE KT11? (Y OR N)'<15>
4212 015250 045440 030524 037461
4213 015256 024040 020131 051117
4214 015264 047040 006451
4215 015270 037012 000
4216 015273 116 052117 053040      ERRMES:  .ASCIZ  <12>'>'
4217 015300 046101 042111 006441      ERRMES:  .ASCIZ  'NOT VALID!'<15><12>
4218 015306 000012
4219 015310 005015 051102 047101      IDMESS:  .ASCIZ  <15><12>'BRANCH GOBBLE'<15><12>
4220 015316 044103 043440 041117
4221 015324 046102 006505 000012
4222
    .EVEN
    
```

K08

```

4223
4224
4225
4226
4227 015332 125252
4228 015334 000401
4229 015336 000000
4230 015340 010703
4231 015342 162703 000004
4232 015346 010304
4233 015350 005204
4234 015352 005013
4235 015354 000261
4236 015356 105513
4237 015360 100402
4238 015362 105214
4239 015364 000773
4240 015366 102401
4241 015370 000000
4242 015372 000242
4243 015374 105214
4244 015376 103402
4245 015400 102001
4246 015402 100401
4247 015404 000000
4248 015406 000137 015474
4249 015412
4250 015412 000000
4251 015414 000027
4252 015416 016700 000130
4253 015422 012701 177770
4254 015426 060701
4255 015430 062701 177772
4256 015434 012703 000040
4257 015440 060703
4258 015442 062703 177772
4259 015446 010367 177736
4260 015452 016702 177736
4261 015456 014140
4262 015460 005302
4263 015462 001375
4264 015464 010067 177722
4265 015470 000177 177716
4266 015474 016700 177712
4267 015500 162767 000002 177704
4268 015506 026767 177700 000040
4269 015514 101414
4270 015516 016701 177672
4271 015522 011060 177776
4272 015526 005720
4273 015530 005301
4274 015532 001373
4275 015534 016737 177652 177570
4276 015542 000177 177644
4277 015546 104400
4278 015550 000722

```

THE FOLLOWING IS THE CODE TAKEN DIRECTLY FROM
 THE POP-11 FAMILY INSTRUCTION EXERCISER DZQKA-A.

```

MARKER: .WORD 125252
FIRST: BR .+4
        .WORD 0
        MOV PC,R3
        SUB #4,R3
        MOV R3,R4
        INC R4
        CLR (R3)
1$: SEC
        ADCB (R3)
        BMI 2$
        INCB (R4)
        BR 1$
2$: BVS .+4
        HALT
        CLV
        INCB (R4)
        BCS INCB1
        BVC INCB1
        BMI .+4
INCB1: HALT
RTAD: JMP @RELO
LAST:
STRTAD: .WORD 0
LENGTH: .WORD 1+LAST-FIRST/2
FIRST1: MOV HI,R0
        MOV #LAST-.,R1
        ADD PC,R1
        ADD #-6,R1
        MOV #RELO-.,R3
        ADD PC,R3
        ADD #-6,R3
        MOV R3,RTAD+2
        MOV LENGTH,R2
ABC: MOV -(R1),-(R0)
        DEC R2
        BNE ABC
        MOV R0,STRTAD
        JMP @STRTAD
RELO: MOV STRTAD,R0
        SUB #2,STRTAD
        CMP STRTAD,L0
        BLOS RET
        MOV LENGTH,R1
RELO1: MOV (R0),-2(R0)
        TST (R0)+
        DEC R1
        BNE RELO1
        MOV STRTAD,@#177570
        JMP @STRTAD
RET: TRAP
        BR FIRST1

```

| | | | | | |
|------|--------|--------|--------|-------|---|
| 4279 | 015552 | 000000 | HI: | .WORD | 0 |
| 4280 | 015554 | 000000 | LO: | .WORD | 0 |
| 4281 | | | : | | |
| 4282 | | | : | | |
| 4283 | | | : | | |
| 4284 | 015556 | | LODAR: | | |
| 4285 | | 000001 | | .END | |

| | | | | | | | | | | | | | | |
|---------|---------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| LJLIM | 014064 | 3667 | 3724* | 3743 | 3746 | 3765 | 3771* | 3798* | 3899* | 3900 | 3904 | 3912* | | |
| LJMESS | 015226 | 3895 | 4208* | | | | | | | | | | | |
| LOSAY | 014066 | 3724 | 3793 | 3825 | 3904* | 3914* | | | | | | | | |
| LOSEG1 | 013374 | 3656 | 3725 | 3792* | | | | | | | | | | |
| LOSEG2 | 013444 | 3678 | 3778 | 3808* | | | | | | | | | | |
| LOSEG3 | 013646 | 3837 | 3842 | 3851* | | | | | | | | | | |
| LOSEG4 | 013726 | 3655 | 3712 | 3870* | | | | | | | | | | |
| LOSFL | 013724 | 3792* | 3795* | 3808 | 3867* | | | | | | | | | |
| LST | 002242 | 1794* | 1860 | | | | | | | | | | | |
| LSTLOC | 002164 | 1758* | 1774* | | | | | | | | | | | |
| MARKER | 015332 | 4227* | | | | | | | | | | | | |
| MAABTO | 007244 | 1850 | 1919 | 1937 | 1985 | 2694 | 2780* | | | | | | | |
| MAABT1 | 007276 | 1902 | 1969 | 2788* | | | | | | | | | | |
| MAABT2 | 007454 | 2820 | 2825* | | | | | | | | | | | |
| MAVA | 000752 | 1461 | 1481 | 1519* | 1668 | 1836* | 1841* | 1890 | 1916 | 2220 | 2245 | 2326* | 2336* | 2590 |
| | | 2765 | 2802 | 3615* | 3698* | 4167 | | | | | | | | |
| MAHES | 015243 | 3622 | 4211* | | | | | | | | | | | |
| MAVEC = | 000250 | 1281* | 1850* | 1902* | 1919* | 1937* | 1969* | 1985* | 2694* | 2820* | | | | |
| MOSPAR | 014356 | 3619 | 4025* | 4166 | | | | | | | | | | |
| MPFL | 014520 | 4031* | 4041 | 4050* | 4059 | 4067* | | | | | | | | |
| MPHES | 015102 | 4044 | 4188* | | | | | | | | | | | |
| MPVECT | 015036 | 4025 | 4177* | | | | | | | | | | | |
| MP1 | 014410 | 4038* | 4058 | | | | | | | | | | | |
| MP2 | 014454 | 4029 | 4055* | | | | | | | | | | | |
| MP25 | 014456 | 4052 | 4056* | | | | | | | | | | | |
| MP3 | 014502 | 4061 | 4064* | | | | | | | | | | | |
| N = | 000010 | 1255* | | | | | | | | | | | | |
| NOFIND | 007021 | 2716 | 2731* | | | | | | | | | | | |
| NOKT11 | 015173 | 4144 | 4203* | | | | | | | | | | | |
| NOMPAR | 015131 | 4063 | 4193* | | | | | | | | | | | |
| NOTIND | 015062 | 4156 | 4185* | | | | | | | | | | | |
| ONER | 014312 | 3993 | 4005* | 4010 | 4011 | 4012 | | | | | | | | |
| O2A | 001742 | 1628 | 1633 | 1716* | 2594 | 2663 | 4047 | 4087 | 4120 | 4126 | 4139 | | | |
| PARAVA | 007116 | 1493 | 2066 | 2184 | 2750* | 2756* | | | | | | | | |
| PARCSA= | 172100 | 2738* | 2747 | | | | | | | | | | | |
| PARCS= | 172100 | 4024* | 4026 | | | | | | | | | | | |
| PARERR | 007000 | 2683 | 2728* | 4084 | | | | | | | | | | |
| PARER2 | 014522 | 4082* | 4177 | | | | | | | | | | | |
| PARITY | 012120 | 2352 | 3445* | | | | | | | | | | | |
| PARPAT | 003362 | 2039* | 2068 | 2070* | 2167* | 2186 | 2188* | 2340* | 2526* | 2961 | 2991 | 3040 | 3094 | |
| PARR3 | 014622 | 4091 | 4116* | | | | | | | | | | | |
| PARR4 | 014770 | 4096 | 4154* | | | | | | | | | | | |
| PARTAB | 001722 | 1678 | 1704* | | | | | | | | | | | |
| PARVEC= | 000114 | 2687* | 2739* | 2745* | 2746* | 4094* | 4095* | | | | | | | |
| PAT | 012241 | 2372 | 3459* | | | | | | | | | | | |
| PC = | %000007 | 1243* | 1376* | 1395* | 1402* | 1406* | 1407* | 1410* | 1413* | 1416* | 1422* | 1425* | 1438* | 1450* |
| | | 1460* | 1473* | 1495* | 1497* | 1500* | 1531* | 1558* | 1566* | 1572* | 1579 | 1584* | 1606* | 1612* |
| | | 1618* | 1628* | 1633* | 1644* | 1662* | 1699* | 1702* | 1717* | 1736* | 1738* | 1745* | 1760* | 1765* |
| | | 1773 | 1778* | 1820* | 1828* | 1842* | 1848 | 1849* | 1858* | 1865 | 1870* | 1885 | 1915 | 1935 |
| | | 1936* | 1951 | 1952* | 1968 | 1983 | 1984* | 2019* | 2020* | 2024 | 2027* | 2032 | 2036* | 2038* |
| | | 2047 | 2051* | 2057* | 2060 | 2063* | 2076 | 2079* | 2083 | 2090* | 2106* | 2116 | 2121* | 2123 |
| | | 2128* | 2132 | 2148 | 2157* | 2168 | 2172 | 2175* | 2178 | 2181* | 2191 | 2194* | 2197 | 2200* |
| | | 2207 | 2218 | 2222* | 2259* | 2260 | 2263* | 2270* | 2271 | 2274* | 2285* | 2286 | 2304* | 2306* |
| | | 2321* | 2346* | 2349* | 2353* | 2357* | 2361* | 2365* | 2369* | 2373* | 2376* | 2382* | 2404* | 2406* |
| | | 2410* | 2424* | 2426* | 2431* | 2448* | 2451* | 2462* | 2472* | 2474* | 2477* | 2493* | 2496* | 2499* |
| | | 2513* | 2516* | 2519* | 2536* | 2551* | 2554* | 2561* | 2571* | 2572* | 2573* | 2582* | 2594* | 2596* |

| | | | | | | | | | | | | | | |
|--------|----------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| | | 2967* | 2968* | 2969* | 2970* | 2972* | 2973* | 2974* | 2975* | 2977* | 2978* | 2979* | 2980* | 2982* |
| | | 2983* | 2984* | 2985* | 3003 | 3005 | 3008 | 3032* | 3045 | 3052 | 3059 | 3066 | 3084* | 3097 |
| | | 3103* | 3104 | 3110* | 3120* | 3128 | 3129 | 3130* | 3131* | 3132* | 3133* | 3134* | 3135* | 3136* |
| | | 3137* | 3138* | 3139* | 3140* | 3141* | 3142* | 3143* | 3144* | 3145* | 3165* | 3172* | 3192* | 3201 |
| | | 3208 | 3223* | 3230* | 3231* | 3251* | 3258 | 3279* | 3286 | 3301* | 3302* | 3303* | 3304* | 3305* |
| | | 3306* | 3307* | 3308* | 3309* | 3310* | 3311* | 3312* | 3313* | 3314* | 3315* | 3316* | 3317* | 3318* |
| | | 3327* | 3331* | 3332* | 3333* | 3334* | 3352* | 3357 | 3364 | 3371 | 3378 | 3406 | 3934* | 3985* |
| | | 4006* | 4260* | 4262* | | | | | | | | | | |
| R3 | =:000003 | 1239* | 1467* | 1472* | 1485* | 1487* | 1632 | 1664* | 1681* | 1685* | 1687* | 1691* | 1696* | 1719* |
| | | 1734* | 1754* | 1755* | 1758 | 1852* | 1853 | 1854* | 1872* | 1873 | 1892* | 1893* | 1894* | 1895* |
| | | 1896* | 1897* | 1898* | 1901* | 1905* | 1906* | 1907* | 1926* | 1927 | 1957* | 1958 | 1990* | 1991 |
| | | 1998* | 1999 | 2034* | 2035 | 2236* | 2237 | 2564* | 2567 | 2575 | 2577 | 2580* | 2583 | 2587 |
| | | 2601* | 2602 | 2613* | 2614* | 2617 | 2619 | 2621* | 2622 | 2626 | 2679 | 2698* | 2839* | 2871* |
| | | 2894* | 2895 | 2902* | 2903 | 2910* | 2911 | 2918* | 2919 | 2959* | 2960* | 2972 | 2973 | 2974 |
| | | 2975 | 2982 | 2983 | 2984 | 2985 | 2990* | 3016* | 3019* | 3045* | 3046 | 3052* | 3053 | 3059* |
| | | 3060 | 3066* | 3067 | 3097* | 3098 | 3104* | 3105 | 3122* | 3148* | 3170* | 3173* | 3201* | 3202 |
| | | 3208* | 3209 | 3258* | 3260 | 3286* | 3288 | 3330* | 3335* | 3357* | 3358 | 3364* | 3365 | 3371* |
| | | 3372 | 3378* | 3379 | 3407 | 3702* | 3704 | 3705* | 3893 | 3899 | 3935* | 4007* | 4101* | 4104 |
| | | 4116* | 4117 | 4118* | 4119 | 4134* | 4135* | 4136 | 4147* | 4230* | 4231* | 4232 | 4234* | 4236* |
| | | 4256* | 4257* | 4258* | 4259 | | | | | | | | | |
| R4 | =:000004 | 1240* | 1663* | 1768* | 1770 | 1941* | 1943* | 1956* | 1961* | 1973* | 1978* | 1987 | 1994* | 1997* |
| | | 2002* | 2086* | 2087* | 2089 | 2102 | 2154 | 2227* | 2229 | 2230 | 2235 | 2565* | 2579* | 2616* |
| | | 2636* | 2680 | 2848* | 2869* | 2965* | 2987* | 3031* | 3078* | 3092* | 3114* | 3129* | 3130 | 3131 |
| | | 3132 | 3133 | 3134 | 3135 | 3136 | 3137 | 3164* | 3177* | 3196* | 3217* | 3225* | 3234* | 3250* |
| | | 3266* | 3278* | 3294* | 3326* | 3337* | 3351* | 3387* | 3672* | 3673* | 3674* | 3675 | 3703* | 3707* |
| | | 3974* | 3975* | 3977* | 3985 | 4232* | 4233* | 4238* | 4243* | | | | | |
| R5 | =:000005 | 1241* | 1397* | 1504* | 1528 | 1533* | 1585* | 1587* | 1613* | 1615* | 1619* | 1625* | 1630* | 1635* |
| | | 1639* | 1666* | 1667 | 1683* | 1684 | 1692* | 1694* | 1725* | 1766* | 1770* | 1773* | 1775* | 1821* |
| | | 1859* | 1861* | 2133* | 2151* | 2208* | 2228* | 2231* | 2234* | 2242* | 2291* | 2298* | 2351* | 2359* |
| | | 2367* | 2371* | 2444* | 2460* | 2538* | 2540* | 2552* | 2559* | 2595* | 2611 | 2612 | 2621 | 2639 |
| | | 2640 | 2656* | 2659* | 2681 | 2682* | 2715* | 2843* | 2844* | 2847* | 2850 | 2852 | 2855 | 2857 |
| | | 2860 | 2862 | 2865 | 2867 | 2891* | 2925* | 2935* | 2940* | 3038* | 3075* | 3093* | 3111* | 3127* |
| | | 3146* | 3168* | 3175* | 3198* | 3215* | 3229* | 3232* | 3255* | 3264* | 3283* | 3292* | 3355* | 3385* |
| | | 3493* | 3616* | 3621* | 3746* | 3749 | 3750* | 3751 | 3757 | 3760* | 3771 | 3814* | 3854* | 3898* |
| | | 3894* | 3907* | 3936* | 3979* | 3989* | 4000* | 4043* | 4048* | 4062* | 4083* | 4085* | 4088* | 4121* |
| | | 4123* | 4137* | 4143* | 4145* | 4155* | | | | | | | | |
| SAVPC | 013722 | 3811* | 3823* | 3851 | 3866* | | | | | | | | | |
| SAVPC2 | 000122 | 1380* | 1396* | 1400 | | | | | | | | | | |
| SAVPER | 014766 | 4119* | 4147 | 4152* | | | | | | | | | | |
| SAVR0 | 006764 | 2675* | 2676 | 2721* | | | | | | | | | | |
| SAVR1 | 006766 | 2722* | | | | | | | | | | | | |
| SAVR2 | 006770 | 2723* | | | | | | | | | | | | |
| SAVR3 | 006772 | 2724* | | | | | | | | | | | | |
| SAVR4 | 006774 | 2725* | | | | | | | | | | | | |
| SAVR5 | 006776 | 2726* | | | | | | | | | | | | |
| SCOPE | = 104000 | 1337* | | | | | | | | | | | | |
| SLP | = 177774 | 1285* | | | | | | | | | | | | |
| SM | = 040000 | 1260* | | | | | | | | | | | | |
| SP | =:000006 | 1242* | 1396 | 1400* | 1405* | 1409* | 1412* | 1415* | 1421* | 1424* | 1431 | 1432* | 1433* | 1434* |
| | | 1435* | 1436* | 1437* | 1443 | 1444 | 1445 | 1446 | 1447 | 1448 | 1449 | 1459* | 1463* | 1464* |
| | | 1468* | 1471* | 1473 | 1480* | 1486 | 1489 | 1491 | 1492 | 1498 | 1530* | 1548* | 1549 | 1550* |
| | | 1552* | 1554 | 1555 | 1559 | 1561* | 1564* | 1571 | 1582 | 1583 | 1607 | 1617 | 1627* | 1632* |
| | | 1718 | 1737* | 1750 | 1819* | 1826* | 1857* | 1884* | 1899* | 1900* | 1901 | 1947 | 1964 | 2006 |
| | | 2018* | 2025* | 2026* | 2031* | 2033* | 2035* | 2046* | 2049* | 2050* | 2055 | 2059* | 2061* | 2062* |
| | | 2070 | 2074* | 2077* | 2078* | 2082* | 2085* | 2089* | 2098* | 2107* | 2119* | 2120* | 2126* | 2127* |
| | | 2147* | 2158* | 2171* | 2172* | 2179* | 2180* | 2188 | 2192* | 2193* | 2199* | 2199* | 2212* | 2252* |

| | | | | | |
|--------|--------|-------|-------|-------|-------|
| .XOR39 | 010212 | 2994 | 3003# | 3043 | 3096 |
| ..USER | 011524 | 2477 | 3349# | | |
| ..1X8 | 007624 | 2036 | 2128 | 2410 | 2882# |
| ..3X9 | 010300 | 2063 | 2181 | 2431 | 3027# |
| ..8X13 | 011026 | 2090 | 2200 | 2451 | 3189# |
| ..1X8 | 007512 | 2027 | 2121 | 2406 | 2839# |
| 1617 | 011742 | 2556 | 3411# | 3414# | |
| .3ISO | 010230 | 3004 | 3008# | | |
| .3IS1 | 010220 | 3005# | | | |
| .3IS9 | 010246 | 3007 | 3009 | 3013# | |
| .3NOT9 | 010236 | 3006 | 3010# | | |
| .3X9 | 010060 | 2051 | 2175 | 2426 | 2956# |
| .813 | 010750 | 2079 | 2194 | 2448 | 3164# |

\$TYPE 1201*

| | | | | | | | | | | | | | | | |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| ADC | 1685 | 2598 | | | | | | | | | | | | | |
| ADCB | 4236 | | | | | | | | | | | | | | |
| ADD | 1529 | 1550 | 1600 | 1646 | 1660 | 1661 | 1684 | 1723 | 1854 | 1871 | 1901 | 1907 | 1924 | 2106 | 2107 |
| | 2157 | 2158 | 2250 | 2379 | 2381 | 2597 | 2614 | 2639 | 2649 | 2662 | 2665 | 2668 | 2781 | 2814 | 2827 |
| | 2828 | 3412 | 3647 | 3648 | 3653 | 3654 | 3673 | 3674 | 3705 | 3718 | 3719 | 3750 | 3769 | 3819 | 3820 |
| | 3821 | 3822 | 3823 | 3843 | 3844 | 3975 | 3977 | 4056 | 4092 | 4093 | 4097 | 4098 | 4116 | 4118 | 4135 |
| | 4178 | 4179 | 4254 | 4255 | 4257 | 4258 | | | | | | | | | |
| ASL | 1680 | 1686 | 1690 | 1893 | 1894 | 1895 | 1896 | 1897 | 1898 | 2145 | 2146 | 2206 | 2378 | 2397 | 2400 |
| | 2401 | 2402 | 2403 | 2420 | 2421 | 2422 | 2423 | 2468 | 2469 | 2470 | 2471 | 2489 | 2490 | 2491 | 2492 |
| | 2509 | 2510 | 2511 | 2512 | 2757 | 2933 | 3125 | 3226 | 3392 | 3683 | | | | | |
| ASR | 2579 | | | | | | | | | | | | | | |
| BCC | 2592 | 2652 | 2758 | 3287 | | | | | | | | | | | |
| BOS | 2335 | 2558 | 2570 | 3259 | 4244 | | | | | | | | | | |
| BEO | 1462 | 1482 | 1494 | 1599 | 1602 | 1638 | 1669 | 1671 | 1744 | 1874 | 1891 | 1917 | 1928 | 1959 | 1977 |
| | 1989 | 1992 | 2000 | 2008 | 2067 | 2088 | 2096 | 2103 | 2155 | 2185 | 2205 | 2221 | 2238 | 2246 | 2282 |
| | 2290 | 2303 | 2342 | 2345 | 2356 | 2442 | 2522 | 2603 | 2623 | 2625 | 2627 | 2691 | 2766 | 2790 | 2801 |
| | 2896 | 2904 | 2912 | 2920 | 2962 | 2992 | 3004 | 3006 | 3009 | 3041 | 3047 | 3054 | 3061 | 3068 | 3095 |
| | 3099 | 3106 | 3152 | 3203 | 3210 | 3227 | 3261 | 3289 | 3359 | 3366 | 3373 | 3380 | 3424 | 3427 | 3434 |
| | 3631 | 3634 | 3951 | 3970 | 3980 | 3990 | 4129 | 4168 | | | | | | | |
| BGE | 3645 | 3661 | 3828 | 3833 | 3976 | | | | | | | | | | |
| BGT | 3836 | | | | | | | | | | | | | | |
| BHI | 2576 | 3872 | | | | | | | | | | | | | |
| BHIS | 3794 | 3797 | | | | | | | | | | | | | |
| BIC | 1603 | 1672 | 1683 | 1700 | 1701 | 1900 | 2087 | 2139 | 2149 | 2213 | 2584 | 3422 | 3491 | 3497 | 3627 |
| | 3944 | 4134 | | | | | | | | | | | | | |
| BIS | 1604 | 2108 | 2159 | 2585 | 2590 | 2756 | 2796 | 3428 | 3430 | | | | | | |
| BIT | 1601 | 1610 | 1637 | 1670 | 1837 | 2281 | 2331 | 2341 | 2441 | 2741 | 3003 | 3005 | 3008 | 3081 | 3123 |
| | 3390 | 3423 | 3426 | 3433 | 4128 | | | | | | | | | | |
| BLE | 3901 | | | | | | | | | | | | | | |
| BLOS | 2568 | 3744 | 3752 | 4269 | | | | | | | | | | | |
| BLT | 1565 | 3685 | 3755 | 3978 | 4058 | | | | | | | | | | |
| BMI | 1394 | 2889 | 3012 | 3809 | 3813 | 3853 | 4042 | 4061 | 4237 | 4246 | | | | | |
| BNE | 1553 | 1560 | 1581 | 1611 | 1622 | 1624 | 1693 | 1698 | 1735 | 1757 | 1764 | 1772 | 1838 | 1911 | 1944 |
| | 1962 | 1979 | 1995 | 2003 | 2069 | 2187 | 2232 | 2243 | 2280 | 2332 | 2348 | 2364 | 2399 | 2413 | 2419 |
| | 2434 | 2438 | 2454 | 2467 | 2488 | 2502 | 2508 | 2566 | 2620 | 2711 | 2742 | 2803 | 2870 | 2872 | 2925 |
| | 2929 | 2941 | 2943 | 2945 | 2988 | 2996 | 3076 | 3079 | 3082 | 3112 | 3115 | 3118 | 3124 | 3147 | 3149 |
| | 3174 | 3176 | 3178 | 3216 | 3218 | 3221 | 3233 | 3235 | 3238 | 3265 | 3267 | 3293 | 3295 | 3336 | 3338 |
| | 3386 | 3388 | 3391 | 3393 | 3708 | 3946 | 3964 | 4003 | 4263 | 4274 | | | | | |
| BPL | 1503 | 1570 | 1577 | 1642 | 2010 | 2301 | 2480 | 3011 | 3014 | 3036 | 3090 | 3625 | 3942 | 4015 | |
| BR | 1567 | 1689 | 1749 | 1855 | 1883 | 1903 | 1930 | 1945 | 1963 | 1980 | 2004 | 2071 | 2189 | 2249 | 2251 |
| | 2327 | 2350 | 2414 | 2435 | 2455 | 2482 | 2503 | 2574 | 2586 | 2599 | 2606 | 2631 | 2634 | 2696 | 2718 |
| | 3007 | 3429 | 3638 | 3761 | 3801 | 3909 | 3959 | 3987 | 3992 | 4052 | 4105 | 4140 | 4148 | 4228 | 4239 |
| | 4278 | | | | | | | | | | | | | | |
| BVC | 2934 | 3126 | 3228 | 4245 | | | | | | | | | | | |
| BVS | 1909 | 1975 | 2755 | 4240 | | | | | | | | | | | |
| CLC | 3256 | 4005 | | | | | | | | | | | | | |
| CLR | 1401 | 1500 | 1591 | 1664 | 1696 | 1720 | 1727 | 1777 | 1823 | 1824 | 1827 | 1830 | 1831 | 1833 | 1836 |
| | 1840 | 1939 | 1954 | 1971 | 2026 | 2038 | 2050 | 2078 | 2119 | 2126 | 2167 | 2171 | 2179 | 2192 | 2198 |
| | 2227 | 2267 | 2283 | 2317 | 2320 | 2321 | 2326 | 2338 | 2339 | 2340 | 2515 | 2548 | 2616 | 2684 | 2689 |
| | 2703 | 2749 | 2769 | 2772 | 2783 | 2795 | 2818 | 2887 | 2897 | 2905 | 2913 | 2921 | 2958 | 3034 | 3048 |
| | 3055 | 3062 | 3069 | 3088 | 3100 | 3107 | 3167 | 3204 | 3211 | 3262 | 3281 | 3360 | 3367 | 3374 | 3381 |
| | 3499 | 3501 | 3614 | 3615 | 3651 | 3671 | 3699 | 3702 | 3716 | 3730 | 3742 | 3792 | 3846 | 3857 | 3870 |
| | 3933 | 3934 | 3935 | 4030 | 4031 | 4095 | 4100 | 4101 | 4181 | 4234 | | | | | |
| CLRB | 3435 | | | | | | | | | | | | | | |
| CLV | 4242 | | | | | | | | | | | | | | |
| CMP | 1598 | 1750 | 1873 | 1910 | 1927 | 1947 | 1958 | 1964 | 1976 | 1987 | 1991 | 1999 | 2006 | 2204 | 2237 |

| | | | | | | | | | | | | | | | |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| | 2289 | 2567 | 2575 | 2602 | 2619 | 2622 | 2624 | 2635 | 2895 | 2903 | 2911 | 2919 | 3046 | 3053 | 3060 |
| | 3067 | 3098 | 3105 | 3202 | 3209 | 3260 | 3288 | 3358 | 3365 | 3372 | 3379 | 3630 | 3633 | 3644 | 3660 |
| | 3743 | 3751 | 3779 | 3793 | 3796 | 3827 | 3832 | 3835 | 3871 | 3900 | 3945 | 3963 | 4055 | 4268 | |
| CMPB | 1559 | | | | | | | | | | | | | | |
| COM | 1841 | 1906 | 1923 | 2336 | 2636 | 2844 | 2846 | 2847 | 2890 | 2893 | 2901 | 2909 | 2917 | 2927 | 2936 |
| | 2937 | 2938 | 2939 | 2960 | 2989 | 2990 | 3037 | 3074 | 3077 | 3091 | 3102 | 3103 | 3109 | 3110 | 3113 |
| | 3116 | 3169 | 3171 | 3197 | 3219 | 3230 | 3231 | 3236 | | | | | | | |
| DEC | 1692 | 1697 | 1734 | 1756 | 1759 | 1771 | 1943 | 1961 | 1972 | 1978 | 2002 | 2231 | 2242 | 2279 | 2869 |
| | 2871 | 2925 | 2928 | 2940 | 2942 | 2987 | 2995 | 3075 | 3078 | 3111 | 3114 | 3117 | 3146 | 3148 | 3173 |
| | 3175 | 3177 | 3215 | 3217 | 3220 | 3232 | 3234 | 3264 | 3266 | 3292 | 3294 | 3335 | 3337 | 3385 | 3387 |
| | 3795 | 3873 | 4050 | 4262 | 4273 | | | | | | | | | | |
| DECB | 1564 | | | | | | | | | | | | | | |
| HALT | 1337 | | | | | | | | | | | | | | |
| INC | 1372 | 1408 | 1476 | 1578 | 1589 | 1643 | 2343 | 2630 | 2632 | 4241 | 4247 | | | | |
| | 1502 | 1940 | 1955 | 1994 | 1996 | 2247 | 2248 | 2288 | 2411 | 2432 | 2452 | 2478 | 2500 | 2520 | 3684 |
| | 3707 | 3753 | 3772 | 3979 | 3989 | 4057 | 4233 | | | | | | | | |
| INCB | 2223 | 2693 | 4238 | 4243 | | | | | | | | | | | |
| JMP | 1411 | 1414 | 1417 | 1423 | 1426 | 1592 | 2097 | 2105 | 2136 | 2156 | 2211 | 2293 | 2310 | 2446 | 2523 |
| | 2527 | 2531 | 2542 | 2707 | 3153 | 3502 | 3650 | 3679 | 3780 | 3837 | 4004 | 4169 | 4170 | 4248 | 4265 |
| | 4276 | | | | | | | | | | | | | | |
| JSR | 1397 | 1406 | 1407 | 1410 | 1413 | 1416 | 1422 | 1425 | 1460 | 1495 | 1497 | 1504 | 1531 | 1558 | 1566 |
| | 1584 | 1585 | 1587 | 1606 | 1612 | 1613 | 1615 | 1618 | 1619 | 1625 | 1628 | 1630 | 1633 | 1635 | 1639 |
| | 1644 | 1662 | 1699 | 1717 | 1725 | 1736 | 1775 | 1820 | 1821 | 1842 | 1849 | 1858 | 1859 | 1861 | 1870 |
| | 1936 | 1952 | 1984 | 2019 | 2020 | 2027 | 2036 | 2051 | 2063 | 2079 | 2090 | 2099 | 2121 | 2128 | 2133 |
| | 2151 | 2175 | 2181 | 2194 | 2200 | 2208 | 2222 | 2259 | 2263 | 2270 | 2274 | 2285 | 2291 | 2298 | 2304 |
| | 2306 | 2346 | 2349 | 2351 | 2353 | 2357 | 2359 | 2361 | 2365 | 2367 | 2369 | 2371 | 2373 | 2376 | 2406 |
| | 2410 | 2426 | 2431 | 2444 | 2448 | 2451 | 2460 | 2462 | 2474 | 2477 | 2496 | 2499 | 2516 | 2519 | 2538 |
| | 2540 | 2552 | 2554 | 2559 | 2561 | 2571 | 2572 | 2573 | 2582 | 2594 | 2595 | 2656 | 2659 | 2663 | 2682 |
| | 2692 | 2706 | 2715 | 2810 | 2841 | 2883 | 2886 | 2898 | 2906 | 2914 | 2922 | 2931 | 2957 | 2994 | 3028 |
| | 3033 | 3043 | 3049 | 3056 | 3063 | 3070 | 3085 | 3096 | 3101 | 3108 | 3121 | 3166 | 3190 | 3193 | 3205 |
| | 3212 | 3224 | 3249 | 3252 | 3257 | 3263 | 3277 | 3280 | 3285 | 3328 | 3349 | 3353 | 3361 | 3368 | 3375 |
| | 3382 | 3404 | 3405 | 3408 | 3425 | 3493 | 3616 | 3618 | 3619 | 3621 | 3628 | 3636 | 3639 | 3655 | 3656 |
| | 3658 | 3663 | 3669 | 3678 | 3709 | 3712 | 3725 | 3727 | 3768 | 3778 | 3814 | 3826 | 3838 | 3854 | 3888 |
| | 3690 | 3891 | 3892 | 3894 | 3896 | 3897 | 3898 | 3907 | 3939 | 3957 | 3960 | 3967 | 3981 | 3991 | 3993 |
| | 4000 | 4010 | 4011 | 4012 | 4025 | 4043 | 4047 | 4048 | 4062 | 4083 | 4085 | 4087 | 4088 | 4120 | 4121 |
| | 4123 | 4126 | 4130 | 4131 | 4132 | 4133 | 4137 | 4139 | 4143 | 4145 | 4155 | 4163 | 4166 | | |
| MOV | 1396 | 1400 | 1405 | 1409 | 1412 | 1415 | 1421 | 1424 | 1431 | 1432 | 1433 | 1434 | 1435 | 1436 | 1437 |
| | 1438 | 1443 | 1444 | 1445 | 1446 | 1447 | 1448 | 1449 | 1450 | 1459 | 1463 | 1464 | 1465 | 1466 | 1467 |
| | 1468 | 1470 | 1471 | 1473 | 1475 | 1480 | 1483 | 1484 | 1485 | 1486 | 1488 | 1489 | 1491 | 1492 | 1498 |
| | 1499 | 1528 | 1530 | 1548 | 1549 | 1555 | 1561 | 1582 | 1583 | 1605 | 1607 | 1608 | 1617 | 1627 | 1632 |
| | 1645 | 1663 | 1666 | 1667 | 1678 | 1679 | 1688 | 1694 | 1718 | 1719 | 1724 | 1737 | 1746 | 1747 | 1752 |
| | 1753 | 1754 | 1755 | 1758 | 1766 | 1768 | 1769 | 1770 | 1773 | 1818 | 1819 | 1825 | 1826 | 1828 | 1832 |
| | 1839 | 1847 | 1848 | 1850 | 1851 | 1852 | 1853 | 1857 | 1865 | 1866 | 1867 | 1868 | 1872 | 1884 | 1885 |
| | 1892 | 1899 | 1902 | 1905 | 1908 | 1915 | 1918 | 1919 | 1920 | 1921 | 1922 | 1926 | 1934 | 1935 | 1937 |
| | 1938 | 1941 | 1942 | 1950 | 1951 | 1953 | 1956 | 1957 | 1968 | 1969 | 1973 | 1974 | 1982 | 1983 | 1985 |
| | 1986 | 1990 | 1997 | 1998 | 2011 | 2018 | 2021 | 2024 | 2025 | 2030 | 2031 | 2032 | 2033 | 2035 | 2046 |
| | 2047 | 2048 | 2049 | 2054 | 2055 | 2057 | 2059 | 2060 | 2061 | 2062 | 2070 | 2074 | 2075 | 2076 | 2077 |
| | 2082 | 2083 | 2084 | 2085 | 2089 | 2098 | 2109 | 2116 | 2120 | 2123 | 2127 | 2132 | 2137 | 2138 | 2140 |
| | 2141 | 2143 | 2147 | 2148 | 2150 | 2160 | 2168 | 2172 | 2178 | 2180 | 2188 | 2191 | 2193 | 2197 | 2199 |
| | 2203 | 2207 | 2212 | 2214 | 2218 | 2219 | 2224 | 2225 | 2226 | 2228 | 2229 | 2230 | 2234 | 2235 | 2236 |
| | 2252 | 2253 | 2256 | 2257 | 2258 | 2260 | 2261 | 2262 | 2268 | 2269 | 2271 | 2272 | 2273 | 2277 | 2284 |
| | 2286 | 2287 | 2302 | 2319 | 2324 | 2325 | 2337 | 2375 | 2377 | 2380 | 2382 | 2395 | 2396 | 2404 | 2408 |
| | 2409 | 2416 | 2417 | 2424 | 2429 | 2430 | 2439 | 2440 | 2449 | 2450 | 2464 | 2465 | 2472 | 2475 | 2476 |
| | 2485 | 2486 | 2493 | 2495 | 2497 | 2498 | 2505 | 2506 | 2513 | 2517 | 2518 | 2526 | 2530 | 2535 | 2536 |
| | 2547 | 2549 | 2551 | 2556 | 2563 | 2564 | 2565 | 2577 | 2578 | 2583 | 2587 | 2588 | 2589 | 2593 | 2600 |
| | 2601 | 2611 | 2612 | 2613 | 2615 | 2618 | 2621 | 2638 | 2640 | 2647 | 2648 | 2653 | 2655 | 2661 | 2664 |

| | | | | | | | | | | | | | | | |
|--------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| .MCALL | 1201 | | | | | | | | | | | | | | |
| .NLIST | 7 | 1197 | 1198 | | | | | | | | | | | | |
| .REM | 8 | | | | | | | | | | | | | | |
| .SBTTL | 1203 | 1453 | 1513 | 1594 | 1804 | 2014 | 2216 | 2312 | 2313 | 2532 | 2544 | 2608 | 2609 | 2671 | 2762 |
| | 2833 | 2950 | 3159 | 3243 | 3343 | 3398 | 3504 | | | | | | | | |
| .TITLE | 1202 | | | | | | | | | | | | | | |
| .WORD | 1358 | 1359 | 1361 | 1362 | 1364 | 1365 | 1367 | 1369 | 1379 | 1380 | 1398 | 1399 | 1419 | 1420 | 1474 |
| | 1501 | 1515 | 1516 | 1521 | 1522 | 1532 | 1546 | 1547 | 1580 | 1739 | 1741 | 1774 | 1780 | 1803 | 1829 |
| | 2039 | 2058 | 2173 | 2295 | 2322 | 2354 | 2362 | 2370 | 2374 | 2385 | 2386 | 2387 | 2388 | 2389 | 2390 |
| | 2391 | 2392 | 2405 | 2425 | 2463 | 2473 | 2494 | 2514 | 2537 | 2555 | 2562 | 2607 | 2658 | 2667 | 2721 |
| | 2722 | 2723 | 2724 | 2725 | 2726 | 2750 | 3414 | 3415 | 3416 | 3417 | 3611 | 3617 | 3622 | 3815 | 3816 |
| | 3855 | 3856 | 3865 | 3866 | 3867 | 3889 | 3895 | 3908 | 3911 | 3912 | 3914 | 3915 | 4001 | 4044 | 4049 |
| | 4063 | 4067 | 4084 | 4086 | 4089 | 4122 | 4124 | 4138 | 4144 | 4146 | 4152 | 4156 | 4227 | 4229 | 4250 |
| | 4251 | 4279 | 4280 | | | | | | | | | | | | |

ERRORS DETECTED: 0
 DEFAULT GLOBALS GENERATED: 0

*DDQABA, DDQABA, SEQ/SOL/CRF/DS:ERFZ/EN:ABS=DSKM:DDQABA.P11
 RUN-TIME: 21 25 5 SECONDS
 RUN-TIME RATIO: 131/52=2.4
 CORE USED: 10K (20 PAGES)