PDP11/70

PDP11/70 MEMORY TEST

MD-11-DEMJA-C

EP-DEMJA-C-DL-A

FICHE 1 OF 1

AUG 1977

digital

MADE IN USA

IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DEMJA-C-D
PRODUCT NAME: PDP-11/70 MEMORY TEST
DATE CREATED: MAY 1977
MAINTAINER: DIAGNOSTIC RELEASE ENGINEERING
AUTHOR: JIM LACEY

TABLE OF CONTENTS

1.0 ABSTRACT

PROGRAM DEMJA TESTS CONTIGUOUS MEMORY ADDRESS FROM 000000 TO 17757776. IT VERIFIES THAT EACH ADDRESS IS UNIQUE (AN ADDRESS TEST) AND THAT EACH MEMORY LOCATION CAN BE READ/WRITTEN RELIABLY (WORST CASE NOISE TESTS).
THIS PROGRAM MAY BE USED TO ADJUST/MARGIN MEMORY.

2.0 REQUIREMENTS

2.1 EQUIPMENT

PDP-11/70 FAMILY PROCESSOR WITH 32K MEMORY

2.2 STORAGE
PROGRAM STORAGE - THE PROGRAM USES MEMORY 0-17777

2.3 PRELIMINARY PROGRAMS

DEKBA THROUGH DEKBF

3.0 LOADING AND STARTING PROCEDURE
LOAD PROGRAM INTO MEMORY USING ABS LOADER
LOAD ADDRESS 200
SET SW12 IN DESIRED POSITION (SEE SEC 4.0)
PRESS START.
ASTERISK "*" WILL BE PRINTED AFTER EACH PASS.
"DEMJA DONE!" WILL BE PRINTED AFTER 6 PASSES.
PASS COUNT MAY BE MONITORED IN THE DISPLAY REGISTER.
NOTE: THIS PROGRAM SAVES THE LOADERS (BOOT AND ABS), TO RESTORE THE LOADERS, RESTART AT 162.

3.1 ACT11 OPERATION

IF THE PROGRAM IS RUN IN QUICK VERIFY MODE UNDER ACT11 THE PROGRAM IS DONE AFTER THE FIRST PASS.

4.0 SWITCH SETTINGS

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

SW14 = 1 OR UP.... LOOP SUBTEST

SW13 = 1 OR UP..... INHIBIT ERROR TYPEOUT

SW12 = 1 OR UP....INHIBIT USE OF MEMORY MANAGEMENT
NOTE: INHIBITING THE USE OF MEMORY MANAGEMENT CAN BE DONE ONLY WHEN THE PROGRAM IS STARTED.
IF THE USE OF MEMORY MANAGEMENT IS INHIBITED THE LAST

ADDRESS AS TYPED BY THE PROGRAM WILL ONLY REFLECT THE AMOUNT OF MEMORY UP TO 28K (LAST ADDRESS = 160000).

SW11 = 1 OR UP..... INHIBIT SUBTEST ITERATION

SW10 = 1 OR UP..... RING BELL ON ERROR

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 & RESTORE LOADERS.

5.0 Subroutine Abstracts

5.1 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.

6.0 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 22 bit physical address.

Note: When testing memory locations 0-77776 the PC typed will be a multiple of 100000 greater than relfected in the program listing
the address of the bad data is in (R2) -2
the good data in R0
the bad data in R3
The address of good data is in R4 (Random Data Test only)
when an error is detected when exercising the memory using the worst case noise patterns, the user should restart the program selecting program #2(see sec 9.1 for details) selecting the appropriate parameters. The user can use the PC and address of the failure to select the proper core bank(s) affected and also the specific pattern. This allows maximum scope capabilities.

6.1 Parity Error

If a parity error is detected the program will type:

PARITY ERROR
PC=PPPPPP MEMORY ADDRESS IS AAAAAAAA
PARITY ERROR REG=EEEEEE ?????????? MARGIN

Where PPPPPP is the contents of the PC when the parity error occurred, AAAAAAAA is the address of the word, EEEEEE is the contents of the memory error register, and ?????????? is the margin setting at the time of the parity error.

After reporting the parity error the program will start over.

7.0 Restrictions

7.1 Starting Restriction
Program must not be relocated when restarting

7.2 Operational Restriction
Program checks contiguous memory if a parity error trap occurs when the program is relocated program action is undefined. If parity memory is available or selected the 3xor9 test pattern is for parity memory only. Do not power fail the program when the program is running relocated.

8.0 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 PDP-11/70 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.

8.1 Stack Pointer
The stack pointer is initially set to 520
and is reset to this value at the start of each subtest.

8.2 Pass Count
Six 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..blank 1

8.3 Error Count
Each time an error occurs, the error count is incremented. The error count can be observed by turning the switch to the display position and setting switch 9. (the error count is also stored in location 1002.) the program will count 17777(8) errors; the error count is not incremented past this value..blank 1

8.4 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..blank 1

8.5 Power Fail

The program may be power failed when running. When the power returns the program will continue in sequence.**caution** do not turn power off/on until the message 'power failed' has been typed. this is because the stack may overflow.

8.5 Execution Time
Execution time is dependent on the amount of memory.

9.0 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 worst case noise test pattern. A subtest is dedicated to checking the pattern. The test proceeds by exercising each bank of memory using the worst case 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(10) word offset on each subsequent relocation. i.e., First relocation is to 20000, next is to 20100, then 20200, etc. After relocation the code moved is checked against the original code (0-17776). When the random data test is complete the program then successively rotates a 0 bit (ROT0) and a '1' bit (ROT1) through all of memory. When all testing is complete the program 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 6 passes have been completed the program will type 'DEMJA DONE' and restart the program beginning with the memory address tests.

```
                                .NLIST  MD,MC
                                .LIST   ME
                                .ABS
                                .TITLE  MAINDEC-11-DEMJA-C PDP11/70 MEMORY TEST
                                .SBTTL  STARTING INST & DEFINITIONS
                        ;COPYRIGHT 1973 1977 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.

                        ;LOADING AND STARING INSTRUCTIONS
                        ;LOAD ADDRESS 200 AND START
                        ;THIS PROGRAM ALSO RELOCATES THE ABS AND BOOT LOADERS TO ALLOW TESTING
                        ;OF MEMORY. TO RESTORE THE LOADERS RESTART AT 162.
                        ;       STACK POINTER IS SET AT 500
                        ;       AN ASTERISK '*' WILL BE PRINTED ON COMPLETION OF EACH PASS, AND
                        ;       THE PROGRAM NAME WILL BE PRINTED WHEN TEST IS COMPLETE.

                        ;GENERAL REGISTER ASSIGNMENTS
                                R0=%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
```

```
000014          TRTVEC=14               ;ADDRESS OF 'TRACE' TRAP VECTOR
000014  BPTVEC=14               ;ADDRESS OF 'BREAKPOINT' TRAP VECTOR
000020          IOTVEC=20               ;ADDRESS OF IOT TRAP VECTOR
000024          PFVEC=24                ;ADDRESS OF POWER FAIL TRAP VECTOR
000030          EMTVEC=30               ;ADDRESS OF EMT VECTOR
000034          TRAPVEC=34              ;ADDRESS OF TRAP VECTOR
000060          TKVEC=60                ;ADDRESS OF TTY KEYBOARD INTERRUPT VECTOR
000064          TPVEC=64                ;ADDRESS OF TTY PRINTER INTERRUPT VECTOR
000240          PIRVEC=240              ;ADDRESS OF PIRQ VECTOR
000244          FPEVEC=244              ;ADDRESS OF FLOATING POINT INT. VECTOR
000250          MMVEC=250               ;ADDRESS OF MEM MGMT ERROR TRAP VECTOR

        ;REGISTER ADDRESSES
177776          PSW=177776              ;ADDRESS OF STATUS REGISTER
177774          SLR=177774              ;ADDRESS OF STACK LIMIT REGISTER
177772          PIRQ=177772             ;ADDRESS OF PROGRAM INTERRUPT REQUEST
177770          UBREAK=177770           ;ADDRESS OF MICRO BREAK REGISTER
177746          CNTRL=177746            ;ADDRESS OF 11/70 MEMORY CONTROL REGISTER
177560          TKS=177560              ;ADDRESS OF KEYBOARD CSR
177562          TKB=177562              ;ADDRESS OF KEYBOARD BUFFER
177564          TPS=177564              ;ADDRESS OF TELEPRINTER CSR
177566          TPB=177566              ;ADDRESS OF TELEPRINTER BUFFER
177570          SWR=177570              ;ADDRESS OF CONSOL SWITCH REGISTER
177570          DISPLAY=177570          ;ADDRESS OF CONSOL DISPLAY REGISTER

        ;INITIAL STACK POINTER SETTING
000500          STKPTR=500

        ;MISCELLANEOUS BIT ASSIGNMENTS
000100          BIT15=  100
040000          BIT14=  040000
020000          BIT13=  020000
010000          BIT12=  010000
001000          BIT9=   001000
000400          BIT8=   000400
000100          BIT6=   000100

        ;MEMORY MANAGEMENT REGISTER ADDRESS ASSIGNMENTS
177572          SR0=177572              ;ADDRESS OF MEM MGMT REGISTER SR0
177574          SR1=177574              ;   "     "     "          "   SR1
177576          SR2=177576              ;   "     "     "          "    SR2
172516          SR3=172516              ;ADDRESS OF MEM MGMT REGISTER SR3

172300          KIPDR0=172300           ;ADDRESS OF KERNEL 'I' PAGE
172302          KIPDR1=172302           ;DESCRIPTOR REGISTERS
172304          KIPDR2=172304
172306          KIPDR3=172306
172310          KIPDR4=172310
172312          KIPDR5=172312
172314          KIPDR6=172314
172316          KIPDR7=172316

172340          KIPAR0=172340           ;ADDRESSES OD KERNEL 'I' SPACE
172342          KIPAR1=172342           ;PAGE ADRESS REGISTERS
172344          KIPAR2=172344
172346          KIPAR3=172346
```

```
                        172350                          KIPAR4=172350
                        172352                          KIPAR5=172352
                        172354                          KIPAR6=172354
                        172356                          KIPAR7=172356


                                                ;INSTRUCTION EQUATES
                        104400                          HLT=TRAP
                        104000                          SCOPE=EMT               ;SCOPE IS AN EMT TRAP

                                                ;MISC. EQUATES
                        000006                          RW=6                    ;R/W BIT IN PDR REGISTERS
                        000000                          UP=0                    ;UP BIT IN PDR REGISTERS


                        000000                          .=0
000000  000000                                  .WORD   0                       ;SPECIAL TRAP/INTERRUPT CATCHER IF PRO-
000002  000000                                  .WORD   0                       ;GRAM HALTS AT 0 THEN ADDRESS WAS NOT
                                                                                ;LOADED PROPERLY FROM VECTOR.
000004  001126                                  .WORD   ERRTRP
000006  000002                                  .WORD   RTI
                        000034                  .=TRAPVEC
000034  001204                                  .WORD   ERROR
000036  000340                                  .WORD   PRTY7
                        000046                  .=46
000046  004260                                  $ENDAD


                        000052                  .=52
000052  040000                                  40000
                        000100                  .=100
000100  004567  000664          CRLF:   JSR     R5,$PRINT
000104  000746                                  $CRLF
000106  000207                                  RTS     PC
000110  000000                  RELFL:  .WORD   0
000112  000000                  SAVPC2: .WORD   0
                        000162                  .=162
000162  012706  000500          PONE:   MOV     #500,SP ;STARTING ADDRESS TO RELOCATE LOADERS.
000166  004767  002016                  JSR     PC,$RLDR
000172  000000                          HALT
000174  000401                          BR      PTWO
                        000200                  .=200
000200  012706  000500          PTWO:   MOV     #500,SP ;STARTING ADDRESS OF MEMORY TEST.
```

```
                                        JMP     @#START         ;GO TO START OF TEST
                                        .=250
000250  000000                          .WORD   0       ;MEMORY MANAGEMENT TRAP VECTOR.
000252  000000                          .WORD   0

                                ;
                                ;ROUTINE TO SAVE REGISTERS ON THE STACK
                                ;CALLED BY SAVE MACRO OR JSR    PC,$SAVR
000254  012667  000016          $SAVR:  MOV     (SP)+,1$        ;SAVE RETURN PC
000260  010546                          MOV     R5,-(SP)
000262  010446                          MOV     R4,-(SP)
000264  010346                          MOV     R3,-(SP)
000266  010246                          MOV     R2,-(SP)
000270  010146                          MOV     R1,-(SP)
000272  010046                          MOV     R0,-(SP)
000274  012707                          MOV     (PC)+,PC        ;RETURN
000276  000000                  1$:     0                       ;CONTAINS RETURN ADDRESS

                                ;ROUTINE TO RESTORE REGISTERS SAVED ON THE STACK
                                ;CALLED BY RESTORE MACRO OR JSR  PC,$RESTR
000300  012667  000016          $RESTR: MOV     (SP)+,1$        ;SAVE RETURN PC
000304  012600                          MOV     (SP)+,R0
000306  012601                          MOV     (SP)+,R1
000310  012602                          MOV     (SP)+,R2
000312  012603                          MOV     (SP)+,R3
000314  012604                          MOV     (SP)+,R4
000316  012605                          MOV     (SP)+,R5
000320  012707                          MOV     (PC)+,PC        ;RETURN
000322  000000                  1$:     0                       ;CONTAINS RETURN ADDRESS

                                        .SBTTL  POWER FAIL ROUTINE
                000502                  .=502
                                ;POWER FAIL ROUTINE
                                ;THE POWER DOWN ROUTINE SAVES THE KEYBOARD STATUS,THE GENERAL REGISTERS
                                ;(R0-R5) AND MEM MGMT REGISTERS (KIPDR0-KIPDR7,KIPAR0-KIPAR7,SR3,SR2,SR0)
                                ;ON THE STACK AND SAVES THE STACK POINTER IN PFSTK BELOW.
000502  013746  177560          PDWN:   MOV     @#TKS,-(SP)     ;SAVE KEYBOARD STATUS
000506  004767  177542                  JSR     PC,$SAVR        ;GO SAVE REGISTERS ON THE STACK
000512  005737  000762                  TST     @#MMAVA         ;CHECK IF MEM MGMT IS AVAILABLE
000516  001421                          BEQ     3$              ;BRANCH IF NOT AVAILABLE
000520  013746  177572                  MOV     @#SR0,-(SP)     ;SAVE SR0
000524  013746  177576                  MOV     @#SR2,-(SP)     ;SAVE SR2
000530  013746  172516                  MOV     @#SR3,-(SP)     ;SAVE SR3
000534  012700  172300                  MOV     #KIPDR0,R0      ;GET ADDRESS OF KIPDR0
000540  012702  000010                  MOV     #8.,R2
000544  010203                          MOV     R2,R3
000546  012046                  1$:     MOV     (R0)+,-(SP)     ;SAVE KIPDR0-KIPDR7
000550  077202                          SOB     R2,1$
000552  012700  172340                  MOV     #KIPAR0,R0      ;GET ADDRESS OF KIPAR0
000556  012046                  2$:     MOV     (R0)+,-(SP)     ;SAVE KIPAR0-KIPAR7
000560  077302                          SOB     R3,2$
000562  010627                  3$:     MOV     SP,(PC)+        ;SAVE STACK PTR IN FOLLOWING LOCATION
000564  000000                  PFSTK:  .WORD   0               ;CONTAINS STACK PTR AFTER POWER FAIL
000566  012737  000576  000024        MOV     #PUP,@#PFVEC    ;SET POWER FAIL VECTOR TO PUP ROUTINE
000574  000000                          HALT
```

```
                                        ;POWER UP ROUTINE.
000576  000240                  PUP:    NOP
000600  013706  000564                  MOV     @#PFSTK,SP      ;SET STACK PTR
000604  005767  000152                  TST     MMAVA           ;CHECK IF MEM MGMT IS AVAILABLE
000610  001421                          BEQ     4$
000612  012700  172360                  MOV     #KIPAR7+2,R0    ;GET ADDRESS OF KIPAR7+2
000616  012702  000010                  MOV     #8.,R2
000622  010203                          MOV     R2,R3
000624  012640              1$:         MOV     (SP)+,-(R0)     ;RESTORE KIPAR7-KIPAR0
000626  077302                          SOB     R3,1$
000630  012700  172320                  MOV     #KIPDR7+2,R0    ;GET ADDRESS OF KIPDR7+2
000634  012640              2$:         MOV     (SP)+,-(R0)     ;RESTORE KIPDR7-KIPDR0
000636  077202                          SOB     R2,2$
000640  012637  172516                  MOV     (SP)+,@#SR3     ;RESTORE SR3
000644  012637  177576                  MOV     (SP)+,@#SR2     ;RESTORE SR2
000650  012637  177572                  MOV     (SP)+,@#SR0     ;RESTORE SR0
000654  005767  004604              4$: TST     PARAVA          ;CHECK IF PARITY REGISTERS ARE ENABLED
000660  001402                          BEQ     5$              ;BRANCH IF NOT
000662  004767  004474                  JSR     PC,.MAMF        ;GO ENABLE PARITY REGISTERS
000666                              5$:
000666  004767  177406                  JSR     PC,$RESTR       ;RESTORE REGISTERS FROM STACK
000672  012637  177560                  MOV     (SP)+,@#TKS
000676  012737  000502  000024          MOV     #PDWN,@#PFVEC   ;SET POWER FAIL TRAP TO PDWN ROUTINE
000704  005027                          CLR     (PC)+
000706  000000              10$:        .WORD   0
000710  005267  177772      11$:        INC     10$             ;DELAY WAITING FOR TTY MOTOR
000714  100375                          BPL     11$
000716  004567  000046                  JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
000722  000730                          PWRFAIL
000724  000240              6$:         NOP
000726  000002                          RTI                     ;RETURN

000730  005015  047520  042527  PWRFAIL:.ASCII  <15><12>'POWER FAILED'
000736  020122  040506  046111
000744  042105
000746  005015      000         $CRLF:  .ASCIZ  <15><12>

                                        .SBTTL  TAGS & PRINT ROUTINE
                    000752              .EVEN
000752  000000              ICNT:   .WORD   0                       ;CONTAINS PASS COUNT
000754  000000              ICOUNT: .WORD   0               ;CONTAINS ITERATION PATTERN
000756  000000              ERCNT:  0                       ;CONTAINS ERROR COUNT
000760  000000              LDDISP: 0                       ;CONTAINS DISPLAY REGISTER IMAGE
000762  000000              MMAVA:  0                       ;MEM MGMT AVAILABLE INDICATOR
                                                            ;0=NOT AVAIL,-1=AVAIL(18 BIT MODE)
                                                            ;-2=AVAIL(22 BIT MODE)
000764  000000              RELOCF: .WORD   0               ;CONTAINS RELOCATION FACTOR
000766  000000              COUNT:  .WORD   0               ;TEMPORARY WORKING LOACATION

                                        ;ROUTINE TO PASS MESSAGE ADDRESS TO TYPE ROUTINE BELOW
                                        ;CALL:  JSR     R5,$PRINT
                                        ;       MESSAGE ADDRESS
000770  000240              $PRINT: NOP
000772  012567  000016                  MOV     (R5)+,1$        ;GET MESSAGE ADDRESS
```

```
000776  066767  177762  000010          ADD     RELOCF,1$       ;ADD RELOCATION FACTOR
001004  013746  177776                  MOV     @#PSW,-(SP)     ;PUSH PSW ON THE STACK
001010  004767  000014                  JSR     PC,.TYPE        ;CALL TYPE ROUTINE
001014  000000                  1$:     .WORD   0               ;CONTAINS MESSAGE ADDRESS
001016  000205                          RTS     R5              ;RETURN

                                ;ROUTINE TO TYPE ASCII MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
                                ;THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
                                ;CALL:  TYPE
                                ;               MESADR          ;MESADR IS FIRST ADDRESS OF ASCIZ STRING

                                ;TAGS USED BY THE TYPE ROUTINE BELOW
001020      000                 $NULL:  .BYTE   0               ;CONTAINS NULL CHARACTER
001021      002                 $FILL:  .BYTE   2               ;CONTAINS # OF FILLER CHARACTERS
001022      000                 $TPFLG: .BYTE   0               ;CONTAINS TELEPRINTER AVAILABLE FLAG
                                                                ;0/377 = AVAIL/NOT AVAIL
001023      000                 $TKFLG: .BYTE   0               ;CONTAINS KEYBOARD AVAILABLE FLAG
001024  177564                  $TPS:   .WORD   177564          ;ADDRESS OF TLEPRINTER STATUS REGISTER
001026  177566                  $TPB:   .WORD   177566          ;ADDRESS OF TELEPRINTER DATA BUFFER
001030  010046                  .TYPE:  MOV     R0,-(SP)        ;SAVE R0
001032  017600  000002                  MOV     @2(SP),R0       ;GET MESSAGE ADDRESS
001036  062766  000002  000002          ADD     #2,2(SP)        ;ADJUST RETURN PC

001044  112046                  1$:     MOVB    (R0)+,-(SP)     ;PUSH CHARACTER TO BE TYPED ONTO STACK
001046  001003                          BNE     2$              ;BRANCH IF NOT THE TERMINATOR
001050  005726                          TST     (SP)+           ;POP TERMINATOR CHAR OFF THE STACK
001052  012600                          MOV     (SP)+,R0        ;RESTORE R0
001054  000002                          RTI                     ;RETURN TO CALLER

001056  004767  000026          2$:     JSR     PC,TYPIT        ;TYPE CHARACTER
001062  122726  000012          3$:     CMPB    #12,(SP)+       ;CHECK IF CHARACTER WAS A LINE FEED
001066  001366                          BNE     1$              ;BRANCH IF NOT LINE FEED
001070  016746  177724                  MOV     $NULL,-(SP)     ;GET # OF FILLERS REQUIRED AND FILLER
                                                                ;CHARACTER.

001074  105366  000001          4$:     DECB    1(SP)           ;DECREMENT FILLERS REQ. COUNT
001100  002770                          BLT     3$              ;BRANCH IF NO MORE FILLERS ARE REQUIRED
001102  004767  000002                  JSR     PC,TYPIT        ;TYPE FILLER CHARACTER
001106  000772                          BR      4$

001110  105777  177710          TYPIT:  TSTB    @$TPS           ;WAIT FOR OUTPUT DEVICE
001114  100375                          BPL     .-4
001116  116677  000002  177702          MOVB    2(SP),@$TPB     ;OUTPUT CHARACTER
001124  000207                          RTS     PC

                                ;XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
                                ;ERROR TRAP SERVICE ROUTINE
001126  005737  177570          ERRTRP: TST     @#SWR           ;CHECK IF HALT ON ERROR
001132  100001                          BPL     .+4             ;BRANCH IF NO HALT ON ERROR
001134  000000                          HALT                    ;HALT
001136  005727                          TST     (PC)+           ;CHECK IF PREV TRAP TO 4 REPORTED
001140  000000                  1$:     .WORD   0               ;CONTAINS ERROR REPORTED FLAG
001142  001013                          BNE     2$              ;BRANCH IF NOT REPORTED
001144  010667  177770                  MOV     SP,1$           ;SET 'NOT REPORTED'
001150  011602                          MOV     (SP),R2         ;GET PC OFF STACK
001152  004767  000376                  JSR     PC,$FORMO       ;GO TO FORMAT ROUTINE
```

```
001156  004567  177606                  JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
001162  001460                          IRAP4
001164  004567  177600                  JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
001170  002351                          DIGITS
001172  000000                  2$:     HALT                    ;ERROR! SECOND TRAP TO 4 OCCURRED
                                                                ;BEFORE FIRST WAS PRINTED
001174  005067  177740                  CLR     1$
001200  000137  000200                  JMP     @#200           ;RESTART AT 200

                                        .SBTTL  ERROR SERVICE ROUTINE
                                ;ERROR SERVICE CALLED BY JSR  PC,ERROR INSTRUCTION
                                ;OR HLT (A TRAP INST)
001204  000240                  ERROR:  NOP
001206  022767  017777  177542          CMP     #17777,ERCNT    ;CHECK FOR MAX ERROR CNT
001214  001403                          BEQ     4$
001216  062767  000001  177532          ADD     #1,ERCNT        ;INCREMENT ERROR COUNT
001224  032737  001000  177570  4$:     BIT     #BIT9,@#SWR     ;SWITCH 9 UP?
001232  001411                          BEQ     5$
001234  042767  017777  177516          BIC     #17777,LDDISP   ;SAVE RELOCATION BITS
001242  056767  177510  177510          BIS     ERCNT,LDDISP    ;LOAD ERROR COUNT
001250  016737  177504  177570          MOV     LDDISP,@#DISPLAY        ;LOAD DISPLAY REGISTER
001256  005737  177570          5$:     TST     @#SWR           ;HALT ON ERROR
001262  100002                          BPL     .+6
001264  000000                          HALT
001266  000470                          BR      3$
001270  032737  020000  177570          BIT     #20000,@#SWR    ;PRINT OUT DESIRED?
001276  001051                          BNE     1$              ;BRANCH IF NO PRINTOUT
001300  004767  176750                  JSR     PC,$SAVR        ;GO SAVE REGISTERS ON THE STACK
001304  016602  000014                  MOV     14(SP),R2       ;GET PC OF ERROR CALL
001310  004767  000240                  JSR     PC,$FORMO       ;GO TO FORMAT ROUTINE
001314  004567  177450                  JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
001320  001475                          ERRPC
001322  004567  177442                  JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
001326  002351                          DIGITS
001330  016602  000004                  MOV     4(SP),R2        ;GET FAILING ADDRESS (IN R2)
001334  004767  000214                  JSR     PC,$FORMO       ;GO TO FORMAT ROUTINE
001340  004567  177424                  JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
001344  002327                          ADRESS
001346  105767  003175                  TSTB    PENFLG          ;BRANCH IF PARITY ERROR DETECTED
001352  001017                          BNE     11$             ;BUT NOT FOUND
001354  105767  003166                  TSTB    PEFLG           ;BRANCH IF PARITY ERROR DETECTED
001360  001006                          BNE     10$             ;BUT FOUND
001362  004567  177402                  JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
001366  001501                          XMTDAT
001370  010046                          MOV     R0,-(SP)        ;PUSH VALUE TO TYPED ONTO STACK
001372  004767  000416                  JSR     PC,02A          ;GO PRINT VALUE
001376                          10$:
001376  004567  177366                  JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
001402  001514                          RECDAT
001404  010346                          MOV     R3,-(SP)        ;PUSH VALUE TO BE TYPED ONTO STACK
001406  004767  000402                  JSR     PC,02A
001412  004767  176462          11$:    JSR     PC,CRLF
001416  004767  176656                  JSR     PC,$RESTR       ;RESTORE REGISTERS FROM STACK
001422  032737  002000  177570  1$:     BIT     #2000,@#SWR     ;RING BELL ON ERROR
001430  001403                          BEQ     2$
001432  004567  177332                  JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
```

```
                                                BELL
                                2$:     TST     @#SWR           ;HALT AFTER PRINT OUT
                                        BPL     .+4
                                        HALT
                                3$:     MOV     R0,-(R2)        ;RESTORE CORRECT DATA TO ADDRESS
                                        ADD     #2,R2
                                        RTI

                        TRAP4:  .ASCII  'TRAPPED TO 4 '
                        ERRPC:  .ASCIZ  'PC='
                        XMTDAT: .ASCIZ  'GOOD DATA='
                        RECDAT: .ASCIZ  ' BAD DATA='
                        BELL:   .ASCIZ  <7>
                        PARREG: .ASCIZ  /PARITY ERROR REG=/

                                .EVEN
                        ;ROUTINE TO PLACE ASCII VALUE OF AN ADDRESS IN TO ADDRESS MESSAGE
                        $FORMO: ADD     RELOCF,11$+2
                                ADD     RELOCF,41$+2
                                JSR     PC,$SAVR        ;GO SAVE REGISTERS ON THE STACK
                        11$:    MOV     #DIGITS,R4      ;ADDRESS WHERE ASCII VALUES ARE STORED
                                CLR     R3              ;WORKING & INDEX REGISTER
                                SUB     #2,R2           ;ADJUST ADDRESS
                                MOV     R2,R5           ;SAVE
                                MOV     R5,R1
                                TST     MMAVA           ;CHECK IF MEM MGMT IS AVAILABLE
                                BEQ     1$              ;BRANCH IF NOT AVAILABLE
                                BIT     #1,@#SR0        ;IS MEM MGMT ENABLED
                                BEQ     1$
                                BIC     #17777,R1       ;SAVE PAR SELECTOR BITS
                                SWAB    R1              ;SWAP BYTES
                                ROR     R1
                                ROR     R1              ;FORM INDEX VALUE
                                ROR     R1
                                ROR     R1
                                MOV     @PARTAB(1),R2   ;GET CONTENTS OF PAR
                                MOV     #6,R0           ;SHIFT COUNT
                                ASL     R2              ;SHIFT KIPAR1 6 PLACES LEFT
                                ROL     R3              ;MSB'S GO INTO R3
                                SOB     R0,.-4
                                BIC     #160000,R5      ;CLEAR PAR SELECTOR BITS
                                ADD     R5,R2           ;FORM 22 BIT ADDRESS
                                ADC     R3              ;IN R2 & R3
                        1$:     CLR     R1
                                MOV     #5,R0
                        12$:    ROR     R3
                                ROR     R2
                                ROR     R1
                                DEC     R0
                                BNE     12$
```

```
                                        MOV     #8.,R0          ;DIGIT COUNT
                                        BR      3$              ;PRINT FIRST DIGIT
                                2$:     ASL     R1
                                        ROL     R2
                                        ROL     R3
                                        DEC     R5
                                        BNE     2$
                                3$:     MOV     #3,R5           ;DIGIT SHIFT COUNT
                                41$:    MOVB    DIGTAB(3),(4)+  ;LOAD DIGIT INTO MESSAGE
                                        CLR     R3              ;CLEAR INDEX
                                        DEC     R0              ;DEC DIGIT COUNT
                                        BNE     2$
                                        JSR     PC,$RESTR       ;RESTORE REGISTERS FROM STACK
                                        BIC     RELOCF,11$+2
                                        BIC     RELOCF,41$+2
                                        RTS     PC              ;RETURN

                                PARTAB: KIPAR0
                                        KIPAR1
                                        KIPAR2
                                        KIPAR3
                                        KIPAR4
                                        KIPAR5
                                        KIPAR6
                                        KIPAR7

                                ;ROUTINE TO TYPE OCTAL VALUE PUSHED ONTO STACK
                                ;CALL:  MOV     VALUE,-(SP)     ;PUSH VALUE ONTO STACK
                                ;       JSR     PC,O2A          ;CALL ROUTINE
002014                          O2A:
002014  004767  176234          O2A:    JSR     PC,$SAVR        ;GO SAVE REGISTERS ON THE STACK
002020  016600  000016                  MOV     16(SP),R0       ;GET VALUE
002024  012703  000006                  MOV     #6,R3           ;COUNTER
002030  005002                          CLR     R2              ;WORKING REGISTER
002032  006100                          ROL     R0
002034  006102                          ROL     R2
002036  062702  000260          1$:     ADD     #260,R2         ;FORM ASCII VALUE
002042  010267  000040                  MOV     R2,2$           ;MOVE CHAR TO TYPE LOCATION
002046  004567  176716                  JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
002052  002106                          2$
002054  005002                          CLR     R2
002056  006100                          ROL     R0
002060  006102                          ROL     R2
002062  006100                          ROL     R0
002064  006102                          ROL     R2
002066  006100                          ROL     R0
002070  006102                          ROL     R2
002072  005303                          DEC     R3
002074  001360                          BNE     1$
002076  004767  176176                  JSR     PC,$RESTR       ;RESTORE REGISTERS FROM STACK
002102  012616                          MOV     (SP)+,(SP)
002104  000207                          RTS     PC
002106  000000          2$:     .WORD   0               ;CONTAINS CHARACTER TO BE TYPED

002110  000000          LODFLO: .WORD   0
                        ;ROUTINE TO SAVE ABS LOADER
```

```
                                ;ERROR SERVICE ROUTINE
002112  005767  177772          $LDR:   TST     LODFLO
002116  001401                          BEQ     3$
002120  000207                          RTS     PC
002122  012700  017776          3$:     MOV     #17776,R0
002126  012737  002140  000004          MOV     #2$,@#ERRVEC    ;SET TIME OUT TRAP VECTOR
002134  005720                          TST     (R0)+
002136  000776                          BR      .-2
002140  022626                  2$:     CMP     (SP)+,(SP)+
002142  022700  020000                  CMP     #20000,R0       ;4K MACHINE?
002146  001417                          BEQ     4$              ;YES--GET OUT
002150  162700  005672                  SUB     #1500.+1*2,R0   ;POINT R0 BACK TO LOADER
002154  010067  000102                  MOV     R0,$LDR1        ;SAVE FOR RESTORE ROUTINE
002160  012702  002734                  MOV     #1500.,R2       ;WORD COUNT
002164  012703  010200                  MOV     #LODAR,R3       ;WHERE LOADER IS TO BE STORED
002170  012023                  1$:     MOV     (R0)+,(R3)+     ;STORE LOADER
002172  005302                          DEC     R2
002174  001375                          BNE     1$
002176  014367  000042                  MOV     -(R3),LSTLOC    ;SAVE LAST WORD OF LOADERS
002202  005367  177702                  DEC     LODFLO
002206  000207                  4$:     RTS     PC              ;RETURN
                                ;ROUTINE TO RESTORE LOADER
002210  005767  177674          $RLDR:  TST     LODFLO
002214  001001                          BNE     2$
002216  000207                          RTS     PC
002220  016705  000036          2$:     MOV     $LDR1,R5        ;GET FIRST ADDRESS OF WHERE LOADER IS
                                                                ;TO BE RESTORED
002224  012704  010200                  MOV     #LODAR,R4       ;ADDRESS WHERE LOADER IS STORED
002230  012702  002734                  MOV     #1500.,R2       ;WORD COUNT
002234  012425                  1$:     MOV     (R4)+,(R5)+     ;RESTORE
002236  005302                          DEC     R2
002240  001375                          BNE     1$
002242  012745                          MOV     (PC)+,-(R5)     ;RESTORE LAST LOCATION (SAVED BY SAVE
002244  000000                  LSTLOC: .WORD   0               ;LOADERS ROUTINE ABOVE)
002246  004567  176516                  JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
002252  002264                          $LDRM
002254  005067  177630                  CLR     LODFLO
002260  000207                          RTS     PC              ;RETURN TO CALLER

002262  000000          $LDR1:  .WORD   0               ;FIRST ADDRESS WHERE LOADERS ARETO BE
                                                        ;RESTORED TO
002264  047514  042101  051105  $LDRM:  .ASCIZ  'LOADER IS RESTORED'<15><12>
002272  044440  020123  042522
002300  052123  051117  042105
002306  005015     000
        002312                          .EVEN
                                ;DIGIT TABLE
002312  030460                  DIGTAB: "01
002314  031462                          "23
002316  032464                          "45
002320  033466                          "67

                                ;MESSAGES
002322  040514  052123     040  LST:    .ASCII  'LAST '
002327     115  046505  051117  ADRESS: .ASCII  'MEMORY ADDRESS IS '
002334  020131  042101  051104
```

```
                                ;THIS TEST ADDRESS MEMORY UP TO 128K AND PROVES 'UNIQNESS' OF ALL
                                ;MEMORY ADDRESS IN A 32K SEGMENT. THE TEST WRITES INTO EACH MEMORY
                                ;ADDRESS THE VALUE OF THAT ADDRESS AND THEN CHECKS FOR THE CORRECT
                                ;DATA IN EACH ADDRESS.
                                ;THE TWELVE MOST SIGNIFICANT BITS OF THE LAST AVAILABLE MEMORY ADDRESS
                                ;IS STORED IN R5.
                                ;STARTING INTRUCTIONS
                                ;       LOAD ADDRESS=200
                                ;       PRESS START
                                ;       STACK POINTER IS AT 500
                                ;*************RESTART AT 162 TO RESTORE LOADER*******************
                                ;MEMORY ADDRESS TEST
002376  012737  002440  000212  START:  MOV     #START1,@#212   ;CHANGE START ADDRESS
002404  012706  000500          MOV     #STKPTR,SP      ;SET UP STACK PTR
002410  004767  177476          JSR     PC,$LDR         ;GO SAVE MONITOR & LOADERS
002414  004567  176350          JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
002420  007520                  RESLDR
002422  005037  000756          CLR     @#ERCNT         ;CLEAR ERROR COUNT
002426  005037  000760          CLR     @#LDDISP        ;CLEAR DISPLAY REGISTER STORAGE LOCN
002432  013737  000760  177570  MOV     @#LDDISP,@#DISPLAY      ;CLEAR DISPLAY REGISTER
002440  012706  000500  START1: MOV     #STKPTR,SP      ;SET STACK PTR
002444  005037  004546          CLR     @#PEFLG         ;CLEAR PARITY ERROR INDICATORS
002450  052737  000014  177746  BIS     #14,@#CNTRL     ;DISABLE CACHE
002456  012727  002440          MOV     #START1,(PC)+   ;LOAD PARITY ERROR RESTART ADDRESS
002462  000000  PERSTRT:.WORD   0                       ;CONTAINS RESTART ADDRESS AFTER PAR ERR
002464  005037  000752          CLR     @#ICNT          ;CLEAR PASS COUNT
002470  005037  000764          CLR     @#RELOCF        ;CLEAR RELOACATION FACTOR
002474  012737  000502  000024  MOV     #PDWN,@#PFVEC   ;SET POWER FAIL TRAP VECTOR
002502  005037  000026          CLR     @#PFVEC+2

                                ;CHECK IF MEMORY MANAGEMENT IS AVAILABLE
002506  005067  176250          CLR     MMAVA           ;CLEAR MEM MGMT AVAILABLE INDICATOR
002512  032737  010000  177570  BIT     #BIT12,@#SWR    ;CHECK IF TO RUN WITH MEM MGMT
002520  001020                  BNE     1$              ;DO NOT USE MEM MGMT IF SW12 WAS SET
002522  012737  002562  000004  MOV     #1$,@#ERRVEC    ;SET TIME OUT TRAP
002530  005037  177572          CLR     @#SR0           ;REFERENCE MEM MGMT
002534  005167  176222          COM     MMAVA           ;SET INDICATOR TO -1 IF AVAILABLE
002540  012737  000020  172516  MOV     #20,@#SR3       ;SET 22 BIT MODE
002546  022737  000020  172516  CMP     #20,@#SR3       ;DID IT SET?
002554  001002                  BNE     1$              ;NO--BRANCH
002556  006367  176200          ASL     MMAVA           ;YES--SET INDICATOR TO -2
002562  004767  002574  1$:     JSR     PC,.MAMF        ;GO ENABLE PARITY ACTION


                                ;ROUTINE TO WRITE VALUE OF MEMORY ADDRESS INTO MEMORY ADDRESS
```

```
                                        ;FOR EXAMPLE ROUTINE WRITES 20000 INTO LOCATION 20000
002566  012737  002626  000004  WRTUP:  MOV     #DONE0,@#ERRVEC ;SET TIME OUT TRAP VECTOR
002574  010701                          MOV     PC,R1           ;LOAD TRACE REGISTER
002576  004767  002712                  JSR     PC,LDMM0
002602  012737  005616  000250          MOV     #MMABT0,@#MMVEC ;SET MEM MGMT ABORT VECTOR
002610  012702  020000                  MOV     #20000,R2       ;FIRST ADDRESS
002614  010203                          MOV     R2,R3           ;LOAD CONSTANT
002616  010322                  1$:     MOV     R3,(R2)+        ;WRITE VALUE OF ADDRESS INTO ADDRESS
002620  062703  000002                  ADD     #2,R3           ;NEXT VALUE
002624  000774                          BR      1$              ;WRITE UNTIL DONE

002626  012706  000500          DONE0:  MOV     #STKPTR,SP      ;SET STACK PTR
002632  004767  176716                  JSR     PC,$FORM0       ;GO TO FORMAT ROUTINE
002636  004567  176126                  JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
002642  002322                          LST
002644  004767  175230                  JSR     PC,CRLF

                                        ;ROUTINE TO CHECK THAT VALUE OF MEMORY ADDRESS WAS WRITTEN CORRECTLY
002650  010701                          MOV     PC,R1           ;LOAD TRACE REGISTER
002652  012702  020000                  MOV     #20000,R2       ;SET R2
002656  012737  002732  000004          MOV     #DONE1,@#ERRVEC ;SET TIME OUT TRAP
002664  010200                          MOV     R2,R0
002666  162700  000002                  SUB     #2,R0           ;SUBTRACT 2
002672  004767  002616                  JSR     PC,LDMM0
002676  062700  000002          1$:     ADD     #2,R0
002702  012203                          MOV     (R2)+,R3        ;GET WRITTEN VALUE
002704  020003                          CMP     R0,R3           ;CHECK
002706  001402                          BEQ     2$
002710  104400                          HLT                     ;ERROR! TO DETERMINE WHICH ADDRESS WAS
002712  000771                          BR      1$
002714  005142          2$:     COM     -(R2)
002716  005112                  COM     (R2)
002720  012203                  MOV     (R2)+,R3
002722  020003                  CMP     R0,R3
002724  001764                  BEQ     1$
002726  104400                  HLT
                                ;WRITTEN IMPROPERLY EXAMINE R2. NEXT EXAMINE MEM MGMT REGISTER KIPAR1
                                ;(IF MEM MGMT IS AVAILABLE). ADD R2 AND KIPAR1 TOGETHER AS SHOWN BELOW

                                ;       R2-2                    0 00X XXX XXX XXX XXX
                                ;       KIPAR1(772342)  Y YYY YYY YYY YYY YYY
                                ;       ADDRESS         Z ZZZ ZZZ ZZZ ZZZ ZZZ ZZZ ZZZ

002730  000762                          BR      1$
002732  012706  000500          DONE1:  MOV     #STKPTR,SP      ;SET STACK PTR
002736  010701                          MOV     PC,R1           ;LOAD TRACE REGISTER

                                ;ROUTINE TO WRITE 1'S COMPLEMENT VALUE OF ADDRESS INTO ADDRESS
                                ;FOR EXAMPLE ROUTINE WRITES 157777 INTO ADDRESS 20000

002740  005767  176016                  TST     MMAVA           ;MEMORY MAGNAGEMENT AVAILABLE?
002744  001420                          BEQ     3$
002746  013703  172342                  MOV     @#KIPAR1,R3     ;FIND LAST ADDRESS IF MEM MANAGE USED
002752  006303                          ASL     R3
002754  006303                          ASL     R3
002756  006303                          ASL     R3
```

```
673  002760  006303                          ASL     R3
674  002762  006303                          ASL     R3
675  002764  006303                          ASL     R3
676  002766  010246                          MOV     R2,-(SP)        ;DEVELOP COMPLEMENT OF LAST ADDRESS
677  002770  042716  020000                  BIC     #20000,(SP)     ;SAVE BITS IF MEMORY IS NOT A MULTIPLE OF 4K
678  002774  062603                          ADD     (SP)+,R3
679  002776  012737  005650  000250          MOV     #MMABT1,@#MMVEC ;SET ABORT VECTOR
680  003004  000403                          BR      2$
681  003006  162702  000002          3$:     SUB     #2,R2           ;R2=LAST ADDRESS
682  003012  010203                          MOV     R2,R3
683  003014  005103          2$:     COM     R3                      ;COMPLEMENT VALUE IN R3
684  003016  062703  000002          1$:     ADD     #2,R3
685  003022  010342                          MOV     R3,-(R2)        ;WRITE COMPLIMENT VALUE INTO ADDRESS
686  003024  102403                          BVS     DONE3
687  003026  020227  017776                  CMP     R2,#17776
688  003032  001371                          BNE     1$
689
690                                  ;SET UP TO CHECK COMPLEMENT DATA WRITTEN DOWN
691  003034  000240          DONE3:  NOP
692  003036  010701                          MOV     PC,R1           ;LOAD TRACE REGISTER
693  003040  005767  175716                  TST     MMAVA           ;CHECK IF MM IS AVAIL
694  003044  001406                          BEQ     1$
695  003046  012737  000200  172342          MOV     #200,@#KIPAR1   ;INIT KIPAR1
696  003054  012737  005616  000250          MOV     #MMABT0,@#MMVEC ;SET ABORT VECTOR
697  003062  012737  003122  000004  1$:     MOV     #DONE4,@#ERRVEC
698  003070  012702  020000                  MOV     #20000,R2       ;FIRST ADDRESS
699  003074  010200                          MOV     R2,R0
700  003076  005100                          COM     R0              ;FIRST DATA (COM OF ADDRESS)
701  003100  062700  000002                  ADD     #2,R0
702  003104  162700  000002          2$:     SUB     #2,R0
703  003110  012203                          MOV     (R2)+,R3        ;GET VALUE
704  003112  020003                          CMP     R0,R3           ;CHECK
705  003114  001773                          BEQ     2$
706  003116  104400                          HLT
707  003120  000771                          BR      2$
708  003122  000240          DONE4:  NOP
709
710                                  ;ROUTINE TO WRITE BANK # INTO ALL ADDRESSES IN A 4K BANK
711  003124  012737  003172  000004          MOV     #DONE4A,@#ERRVEC;SET TIME OUT TRAP VECTOR
712  003132  010701                          MOV     PC,R1
713  003134  004767  002354                  JSR     PC,LDMMO
714  003140  012737  005616  000250          MOV     #MMABT0,@#MMVEC
715  003146  012702  020000                  MOV     #20000,R2
716  003152  005000                          CLR     R0
717  003154  005200          1$:     INC     R0                      ;R0 WILL BE DATA WRITTEN
718  003156  012704  010000                  MOV     #4096.,R4       ;SET 4K COUNTER
719  003162  010022          2$:     MOV     R0,(R2)+                ;WRITE BANK # INTO ALL ADDRESSES
720  003164  005304                          DEC     R4
721  003166  001375                          BNE     2$
722  003170  000771                          BR      1$
723
724  003172  022626          DONE4A: CMP     (SP)+,(SP)+             ;ADJUST STACK PTR
725
726                                  ;CHECK THAT DATA WRITTEN ABOVE CAN BE READ
727  003174  012737  003242  000004          MOV     #DONE4B,@#ERRVEC
728  003202  010701                          MOV     PC,R1
```

```
                                                JSR     PC,LDMM0
                                                MOV     #20000,R2
                                                CLR     R0
                                        1$:     INC     R0
                                                MOV     #4096.,R4
                                        2$:     MOV     (R2)+,R3
                                                CMP     R0,R3
                                                BEQ     .+4
                                                HLT
                                                DEC     R4
                                                BNE     2$
                                                BR      1$
                                        DONE4B: CMP     (SP)+,(SP)+

                                        ;ROUTINE TO WRITE CONSTANT DATA INTO 4K
                                        ;BANK STARTING WITH LAST MEMORY LOCATION
                                                MOV     PC,R1
                                                MOV     #MMABT1,@#MMVEC
                                                SUB     #2,R2
                                                CLR     R0
                                        1$:     DEC     R0
                                                MOV     #4096.,R4
                                        2$:     MOV     R0,-(R2)
                                                BVS     DONE4C
                                                CMP     R2,#17776       ;CHECK IF DONE
                                                BEQ     DONE4C
                                                DEC     R4
                                                BNE     2$
                                                BR      1$

                                        DONE4C: MOV     #DONE4D,@#ERRVEC
                                                MOV     PC,R1
                                                JSR     PC,LDMM0
                                                MOV     #MMABT0,@#MMVEC ;SET ABORT VECTOR
                                                MOV     #20000,R2
                                        1$:     CMP     #4096.,R4       ;CHECK IF WRITE ABOVE STARTED ON
                                                                        ;4K BOUNDARY
                                                BEQ     2$
                                                MOV     (R2)+,R3
                                                CMP     R0,R3
                                                BEQ     4$
                                                HLT
                                                BR      5$
                                        4$:     COM     -(R2)
                                                COM     (R2)
                                                MOV     (R2)+,R3
                                                CMP     R0,R3
                                                BEQ     5$
                                                HLT
                                        5$:     INC     R4
                                                BNE     1$
                                        2$:     INC     R0
                                                MOV     #4096.,R4
                                        3$:     MOV     (R2)+,R3
                                                CMP     R0,R3
                                                BEQ     .+4
```

```
                                                HLT
                                                DEC     R4
                                                BNE     3$
                                                BR      2$

                                DONE4D: CMP     (SP)+,(SP)+
                                                TST     @#42            ;BRANCH IF PROGRAM WAS NOT
                                                BEQ     BEGIN1          ;LOADED VIA ACT11 IN QV OR AA MODES
                                                TST     $ENDAD+2        ;BRANCH IF NOT IN QV MODE
                                                BPL     BEGIN1
                                                MOV     #1,@#ENDCT      ;SET ENDCT TO DO 1 PASS ONLY IN QV
                                                .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
                                                BIS     #14,@#CNTRL     ;DISABLE CACHE
                                                JSR     PC,.MAMF        ;GO ENABLE PARITY ACTION
                                                JSR     PC,CKSWR        ;GO CHECK SWITCHES
                                                CLR     (PC)+           ;SET INDICATOR TO WRITE NORMAL 3X9 PAT
                                PARPAT: .WORD   0
                                                CMP     #-2,MMAVA       ;22 BIT MODE
                                                BNE     DONE6           ;NO--BRANCH
                                                JSR     PC,MARGIN       ;YES--GO SETUP MARGINS

                                ;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.
                                DONE6:  MOV     #STKPTR,SP      ;SET STACK PTR
                                                MOV     PC,R1           ;UPDATE TRACE REGISTER
                                                MOV     #DONE7,@#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
                                DONE7:  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
                                                CMP     #-2,MMAVA
                                                BEQ     1$
                                                TST     @#PARAVA        ;BRANCH IF PARITY MEMORY IS NOT AVAIL
                                                BEQ     DONE8
                                1$:             TST     @#PARPAT        ;BRANCH IF PARITY PAT JUST WRITTEN
```

```
003620  001003                          BNE     DONE8
003622  010637  003472                  MOV     SP,@#PARPAT     ;SET INDICATOR TO WRITE 3X9 PAR PAT
003626  000730                          BR      DONE6           ;REPEAT TEST USING MODIFIED 3X9 PATTERN

                                ;WRITE 8 XOR 13 TEST PATTERN STARTING AT ADDRESS 40000
003630  012706  000500          DONE8:  MOV     #STKPTR,SP      ;SET STACK PTR
003634  012737  003656  000004          MOV     #DONE9,@#ERRVEC ;SET TIME OUT TRAP VECTOR
003642  010701                          MOV     PC,R1           ;UPDATE TRACE REGISTER
003644  012746  000002                  MOV     #2,-(SP)        ;PUSH STARTING BANK # ON THE STACK
003650  005046                          CLR     -(SP)           ;PUSH # OF BANKS TO WRITE ON THE STACK
003652  004767  003222                  JSR     PC,.8X13        ;GO TO ROUTINE TO WRITE DATA

                                ;CHECK 8 XOR 13 TEST PATTERN WRITTEN ABOVE
003656  012706  000500          DONE9:  MOV     #STKPTR,SP      ;SET STACK PTR
003662  010701                          MOV     PC,R1           ;UPDATE TRACE REGISTER
003664  012737  001126  000004          MOV     #ERRTRP,@#ERRVEC
003672  012746  000002                  MOV     #2,-(SP)
003676  005404                          NEG     R4
003700  042704  000001                  BIC     #1,R4           ;SET 4K BANK COUNT TO 8K INCREMENT
003704  001403                          BEQ     DONE10          ;DO NOT CHECK IF ONLY 12K
003706  010446                          MOV     R4,-(SP)
003710  004767  003172                  JSR     PC,..8X13       ;GO CHECK 8 XOR 13 PATTERN WRITTEN ABOVE

003714  000005          DONE10: RESET                           ;DISABLE MEM MGMT AND PARITY ACTION

                                .SBTTL  RANDOM DATA,ROTATING I/O TESTS
                        ;RANDOM DATA TEST. THIS TEST MOVES THE PROGRAM CODE THROUGHOUT MEMORY
003716  010701          RANTST: MOV     PC,R1           ;SET TRACE POINTER
003720  012737  004056  000004          MOV     #7$,@#ERRVEC    ;SET TIME OUT TRAP
003726  005767  175030                  TST     MMAVA           ;CHECK IF MEM MGMT IS AVAILABLE
003732  001412                          BEQ     1$              ;BRANCH IF NOT AVAILABLE
003734  004767  001554                  JSR     PC,LDMMO        ;GO SET UP MEM MGMT
003740  105237  172301                  INCB    @#KIPDR0+1      ;ALLOW 4K ADDRESSING IN FIRST 4K
003744  012737  077406  172304          MOV     #200*256.-400+UP+RW,@#KIPDR2    ;SET KIPDR2=RW UP 200 BLOCKS
003752  012737  000400  172344          MOV     #400,@#KIPDR2
003760  012702  020000          1$:     MOV     #20000,R2       ;SET 'TO' ADDRESS POINTER
003764  005004                          CLR     R4              ;SET 'FROM' ADDRESS POINTER
003766  012705  004000          2$:     MOV     #2048.,R5       ;SET 4K WORD COUNT
003772  012422          3$:     MOV     (R4)+,(R2)+     ;MOVE CODE
003774  012422                          MOV     (R4)+,(R2)+
003776  005305                          DEC     R5              ;DECREMENT 4K WORD COUNTER
004000  001374                          BNE     3$

004002  012705  005405                  MOV     #4096.-PLACE+1,R5       ;SET 4K WORD COUNTER
004006  014400          4$:     MOV     -(R4),R0        ;GET 'GOOD' DATA
004010  014203                          MOV     -(R2),R3        ;GET 'BAD' DATA
004012  020003                          CMP     R0,R3           ;COMPARE 'GOOD' & 'BAD' DATA
004014  001403                          BEQ     5$
004016  005722                          TST     (R2)+           ;STEP ADDRESS FOR ERROR ROUTINE
004020  104400                          HLT                     ;REPORT ERROR
004022  005742                          TST     -(R2)           ;RESTORE ADDRESS POINTER
004024  005305          5$:     DEC     R5              ;DECREMENT 4K WORD COUNTER
004026  001367                          BNE     4$              ;LOOP UNTIL 4K WORDS CHECKED
```

```
                                                        ;CHECK IF MEM MGMT IS AVAILABLE
                                                        ;BRANCH IF NOT AVAILABLE
                                                        ;STEP ADDRESS
                                                        ;RESET STACK PTR
                                                        ;RESTORE ERROR TRAP VECTOR

                                                        ;ROTATING 0 TEST. THIS TEST ROTATES A SINGLE '0' THROUGH MEMORY
                                                        ;SET CONSTANT =177777
                                                        ;SET BANK #1
                                                        ;GET # OF 256. WORD BLOCKS IN MEMORY
                                                        ;GO WRITE 1'S THROUGHOUT MEMORY
                                                        ;SET SCOPE PTR
                                                        ;SET STARTING BANK #
                                                        ;SET # OF 256. WORD BLOCKS TO CHECK
                                                        ;GO TO ROTATE 0 ROUTINE

                                                        ;ROTATING 1 TEST THIS TEST ROTATES A SINGLE '1' BIT THROUGH ALL OF
                                                        ;MEMORY
                                                        ;CLEAR CONSTANT
                                                        ;PUSH STARTING BANK ONTO STACK
                                                        ;AND # OF 256. WORD BLOCKS IN MEMORY
                                                        ;GO WRITE 0'S THROUGHOUT MEMORY
                                                        ;SET SCOPE PTR
                                                        ;SET STARTING BANK #
                                                        ;SET # OF 256. WORD BLOCKS TO CHECK
                                                        ;GO ROTATE A '1' BIT THROUGHOUT MEMORY

                                                        ;END OF CYCLE
                                                        ;RESET MACHINE TO KEY-START STATE
                                                        ;UPDATE TRACE REGISTER
                                                        ;SET STACK PTR
                                                        ;INCREMENT PASS COUNT
                                                        ;CHECK FOR LAST PASS
                                                        ;MAKE 5 PASSES
                                                        ;PASS COUNT ADDRESS
                                                        ;BRANCH IF LAST PASS COMPLETED
                                                        ;GO TO PRINT ROUTINE

                                                        ;GO TO PRINT ROUTINE

                                                        ;WAIT FOR BELL TO RING

                                                        ;GET DECTAPE MONITOR RETURN ADDRESS

                                                        ;RESTORE MONITOR & LOADERS
                                                        ;GO TO DECTAPE MONITOR
```

```
                                        FINISH: JMP     START1
                                                .SBTTL  PROGRAM SUBROUTINES
                                                .SBTTL  RELOCATION ROUTINES
                                        ;ROUTINE TO RELOCATE PROGRAM CODE
004274  012500                          RELOC:  MOV     (R5)+,R0        ;GET FROM ADDRESS
004276  011502                                  MOV     (R5),R2         ;GET TO ADDRESS
004300  010203                                  MOV     R2,R3
004302  062703  017776                          ADD     #17776,R3       ;MOVES 4K
004306  012737  004356  000004                  MOV     #4$,@#ERRVEC    ;SET TIME OUT TRAP
004314  005004                                  CLR     R4              ;CLEAR RELOCATION SUCCESSFUL INDICATOR
004316  005723                                  TST     (R3)+           ;CHECK IF MEMORY IS AVAILABLE
004320  012022                          1$:     MOV     (R0)+,(R2)+     ;RELOCATE
004322  020203                                  CMP     R2,R3           ;RELOCATION COMPLETE?
004324  001375                                  BNE     1$
004326  011503                                  MOV     (R5),R3
004330  020203                          2$:     CMP     R2,R3
004332  001413                                  BEQ     5$              ;BRANCH IF DONE
004334  024042                                  CMP     -(R0),-(R2)     ;CHECK THAT DATA WAS RELOCATED PROPERLY
004336  001774                                  BEQ     2$
004340  005703                                  TST     R3              ;CHECK IF RELOCATING BACK TO 000000
004342  001403                                  BEQ     3$
004344  104400                                  HLT                     ;ERROR! CANNOT RELOCATE PROGRAM CODE
                                                                        ;TO UPPER MEMORY BANK PROPERLY
004346  000000                                  HALT
004350  000767                                  BR      2$              ;CONTINUE RELOCATING AT YOUR PERIL
004352  000000                          3$:     HALT                    ;ERROR! CANNOT RELOCATE CODE BACK TO
                                                                        ;TO 000000 PROPERLY
004354  000777                                  BR      .
004356  022626                          4$:     CMP     (SP)+,(SP)+     ;RESTORE STACK PTR
004360  005104                                  COM     R4
004362  000240                          5$:     NOP
004364  012702  000764                          MOV     #RELOCF,R2      ;GET ADDRESS OF RELOCATION FACTOR
004370  061502                                  ADD     (R5),R2         ;ADD FACTOR
004372  012512                                  MOV     (R5)+,(R2)      ;RELOCATED RELOCF NOW CONTAINS RELOCATION
                                                                        ;FACTOR
004374  000205                                  RTS     5               ;RETURN, R4=-1 IF NO RELOCATION

                                        ;ROUTINE TO RELOCATE PROGRAM CODE FROM ORIGINAL POSITION (0-4K) TO
                                        ;TOP OF MEMORY.
004376  012700  020000                  RELOCP: MOV     #20000,R0       ;SET UP TO SCAN FOR TOP OF MEMORY
004402  012737  000006  000004                  MOV     #ERRVEC+2,@#ERRVEC
004410  062700  020000                  1$:     ADD     #20000,R0       ;INCREMENT SCAN ADDRESS
004414  000261                                  SEC                     ;SET TIME OUT INDICATOR
004416  005710                                  TST     (R0)            ;CHECK FOR EXISTANT MEMORY
004420  103373                                  BCC     1$              ;'C' WILL BE CLEAR IF MEMORY EXISTS
004422  012737  001126  000004                  MOV     #ERRTRP,@#ERRVEC
004430  162700  020000                          SUB     #20000,R0       ;ADJUST TO LAST EXISTANT 4K
004434  010067  000006                          MOV     R0,2$           ;PASS RELOCATION ADDRESS TO RELOC ROUTINE
004440  004567  177630                          JSR     R5,RELOC        ;RELOCATE PROGRAM
004444  000000                                  000000                  ;FROM ADDRESS 000000
004446  000000                          2$:     .WORD   0               ;TO LAST 4K BANK
004450  004567  174314                          JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
004454  010047                                  RELOCM
004456  016746  177764                          MOV     2$,-(SP)        ;PASS TO 02A ROUTINE
```

```
1009  004462  062716  010124                ADD     #REL24K,(SP)    ;SET UP RESTART ADDRESS
1010  004466  004767  175322                JSR     PC,$2A          ;TYPE RESTART ADDRESS
1011  004472  011667  000006                MOV     (SP),3$         ;SAVE RETURN ADDRESS IN 3$ BELOW
1012  004476  066706  177744                ADD     2$,SP           ;RESET STACK PTR
1013  004502  012716                        MOV     (PC)+,(SP)      ;GET RETURN ADDRESS
1014  004504  000000                3$:     .WORD   0               ;CONTAINS RETURN PC
1015  004506  066716  177734                ADD     2$,(SP)         ;ADJUST RETURN PC
1016  004512  000207                        RTS     PC
1017
1018                                        .SBTTL  MA/MF PARITY ERROR SERVICE ROUTINE
1019                                ;WHEN MA/MF A PARITY ERROR IS DETECTED THIS ROUTINE SCANS MEMORY FOR THE
1020                                ;ADDRESS CAUSING THE PARITY ERROR. WHEN THE ADDRESS IS LOCATED THE ROUTINE
1021                                ;HALTS WITH THE ADDRESS+2 IN R0. TO CONTINUE AFTER THE ERROR PRESS CONTINUE.
1022  004514  010067  000170        .PARSRV:MOV     R0,SAVR0        ;SAVE R0 IN SAVR0
1023  004520  012700  004712                MOV     #SAVR0+2,R0
1024  004524  010120                        MOV     R1,(R0)+
1025  004526  010220                        MOV     R2,(R0)+
1026  004530  010320                        MOV     R3,(R0)+
1027  004532  010420                        MOV     R4,(R0)+
1028  004534  010520                        MOV     R5,(R0)+
1029  004536  004567  174226                JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
1030  004542  004724                        PARERR
1031  004544  005027                        CLR     (PC)+           ;CLEAR PARITY ERROR INDICATORS
1032  004546     000                PEFLG:  .BYTE   0               ;NOT 0/0 =PAR ERR/NO PAR ERR
1033  004547     000                PENFLG: .BYTE   0               ;NOT 0/0=PAR ERR DETECTED/NOT DETECTED ON SCAN
1034  004550  012737  004616  000114        MOV     #2$,@#PARVEC            ;SET PARITY ERROR TRAP
1035  004556  012737  004654  000004        MOV     #4$,@#ERRVEC            ;SET TIME OUT TRAP VECTOR
1036  004564  005002                        CLR     R2
1037  004566  005767  174170                TST     MMAVA           ;CHECK IF MEM MGMT IS AVAILABLE
1038  004572  001407                        BEQ     1$              ;BRANCH IF NOT AVAILABLE
1039  004574  004767  000714                JSR     PC,LDMMO        ;SET UP MEM MGMT
1040  004600  105237  172301                INCB    @#KIPDR0+1      ;ALLOW FULL 4K PAGE ADDRESSING
1041  004604  012737  0056[illegible]  000250        MOV     #MMABT0,@#MMVEC ;SET MEM MGMT ABORT TRAP VECTOR
1042  004612  012200                1$:     MOV     (R2)+,R0        ;SCAN ALL ADDRESSES
1043  004614  000776                        BR      1$
1044  004616  110667  177724        2$:     MOVB    SP,PEFLG        ;SET PARITY ERROR FOUND INDICATOR
1045  004622  010003                        MOV     R0,R3
1046  004624  104400                        HLT                     ;PARITY ERROR! ADDRESS+2 IS IN R2
1047  004626  000002                        RTI                     ;CONTINUE SCAN
1048  004630  000240                3$:     NOP                     ;INSERT HALT INST TO EXAMINE PARITY REGS
1049  004632  005067  177710                CLR     PEFLG           ;CLEAR PARITY ERROR INDICATORS
1050  004636  012706  000500                MOV     #STKPTR,SP      ;RESET STACK PTR
1051  004642  000005                        RESET
1052  004644  004767  000512                JSR     PC,.MAMF        ;GO ENABLE PARITY ERROR DETECTION
1053  004650  000177  175606                JMP     @PERSTRT        ;RESTART SELECTED PROGRAM
1054
1055                                ;SERVICE ROUTINE IF PARITY ERROR NOT DETECTED ON SCAN
1056  004654  105767  177666        4$:     TSTB    PEFLG           ;BRANCH IF PARITY ERROR WAS
1057  004660  001363                        BNE     3$              ;DETECTED ON SCAN
1058  004662  016602  000004                MOV     4(SP),R2        ;GET PC AT TIME OF ERROR
1059  004666  162702  000002                SUB     #2,R2           ;BACK IT UP
1060  004672  110667  177651                MOVB    SP,PENFLG       ;SET IND = NO PAR ERROR DETECTED ON SCAN
1061  004676  004567  174066                JSR     R5,$PRINT       ;GO TO PRINT ROUTINE
1062  004702  004745                        NOFIND
1063  004704  104400                        HLT                     ;ERROR! PARITY ERROR NOT DETECTED ON SCAN
1064  004706  000750                        BR      3$
```

```
                                        ;THE BELOW 6 WORDS CONTAINS THE SAVED CONTENTS OF R0-R5 WHEN THE
                                        ;PARITY ERROR OCCURRED
004710  000000                  SAVR0:  .WORD   0
004712  000000                  SAVR1:  .WORD   0
004714  000000                  SAVR2:  .WORD   0
004716  000000                  SAVR3:  .WORD   0
004720  000000                  SAVR4:  .WORD   0
004722  000000                  SAVR5:  .WORD   0

004724  005015  040520  044522  PARERR: .ASCIZ  <15><12>'PARITY ERROR'<15><12>
004732  054524  042440  051122
004740  051117  005015     000
004745     116  052117  043040  NOFIND: .ASCIZ  'NOT FOUND ON SCAN'<15><12>
004752  052517  042116  047440
004760  020116  041523  047101
004766  005015     000
004772                          .EVEN


177740                  MEMLO=177740
177742                  MEMHI=177744
177744                  MEMERR=177744

004772  005767  177550  .22PAR: TST     PEFLG           ;BEEN HERE BEFORE
004776  001403                  BEQ     1$              ;BRANCH IF NO
005000  000000                  HALT                    ;YES -- DOUBLE PARITY ERROR
005002  000177  175454          JMP     @PERSTRT
005006  010667  177534  1$:     MOV     SP,PEFLG        ;SET PARITY ERROR FLAG
005012  005737  177570          TST     @#SWR           ;HALT ON ERROR?
005016  100001                  BPL     100$            ;BRANCH IF NO
005020  000000                  HALT                    ;YES
005022  013746  177744  100$:   MOV     @#MEMERR,-(SP)  ;SAVE MEMORY ERROR REG
005026  013701  177740          MOV     @#MEMLO,R1      ;GET ADDRESS OF WHERE THE PARITY
005032  013702  177742          MOV     @#MEMHI,R2      ;ERROR OCCURRED
005036  011637  177744          MOV     (SP),@#MEMERR   ;CLEAR THE ERROR REG
005042  032737  020000  177570  BIT     #BIT13,@#SWR    ;INHIBIT ERROR TYPEOUT
005050  001071                  BNE     101$            ;BRANCH IF YES
                        ;PRINT  "PARITY ERROR"
005052  004567  173712          JSR     R5,$PRINT
005056  004724                  PARERR
                        ;PRINT  "PC=XXXXXX"
005060  004567  173704          JSR     R5,$PRINT
005064  001475                  ERRPC
005066  016646  000002          MOV     2(SP),-(SP)     ;GET PC AT TIME OF PARITY ERROR
005072  066716  173666          ADD     RELOCF,(SP)
005076  004767  174712          JSR     PC,02A
005102  004567  173662          JSR     R5,$PRINT
005106  002361                  SPACE1
                        ;CHANGE 22-BIT ADDRESS TO OCTAL-ACSII
005110  012700  002351          MOV     #DIGITS,R0
005114  012704  000010          MOV     #8.,R4
005120  012705  000003  2$:     MOV     #3,R5
005124  005003          3$:     CLR     R3
005126  006301          4$:     ASL     R1
005130  106102                  ROLB    R2
```

```
                                        ROL     R3
                                        SOB     R5,4$
                                        MOVB    DIGTAB(R3),(R0)+
                                        SOB     R4,2$
                                ;PRINT  "MEMORY ADDRESS IS AAAAAAAA"
                                        JSR     R5,$PRINT
                                        ADRESS
                                        JSR     PC,CRLF
                                ;PRINT  "PARITY ERROR REG=XXXXXX"
                                        JSR     R5,$PRINT
                                        PARREG
                                        MOV     (SP),R5
                                        JSR     PC,02A
                                        JSR     PC,$PRINT
                                        SPACE1
                                ;PRINT THE MARGIN SETTING
                                        MOV     ICNT,R0
                                        MOVB    MRGNTB(R0),R0
                                        ADD     #MARTBL,R0
                                        MOV     (R0),5$
                                        JSR     R5,$PRINT
                                5$:     MARTBL
                                        JSR     R5,$PRINT
                                        MARMSG
                                101$:   JMP     @PERSTRT
                                ;MARGIN MESSAGE TABLE
                                MARTBL: NORMAL
                                        0
                                        ESTRB
                                        LSTRB
                                        LCRNT
                                        HCRNT
                                        NORMAL

                                ;MARGIN MESSAGES
005256  047516  046522  046101  NORMAL: .ASCIZ  'NORMAL'
005264     000
005265     105  051101  054514  ESTRB:  .ASCIZ  'EARLY STROBE'
005272  051440  051124  041117
005300  000105
005302  040514  042524  051440  LSTRB:  .ASCIZ  'LATE STROBE'
005310  051124  041117  000105
005316  047514  020127  052503  LCRNT:  .ASCIZ  'LOW CURRENT'
005324  051122  047105  000124
005332  044510  044107  041440  HCRNT:  .ASCIZ  'HIGH CURRENT'
005340  051125  042522  052116
005346     000
005347     040  040515  043522  MARMSG: .ASCIZ  ' MARGIN'<12><15>
005354  047111  006412     000
        005362                  .EVEN

                                ;ROUTINE TO ENABLE PARITY ERROR ACTION ON 11/70 PARITY MEMORIES
        000114                          PARVEC=114              ;PARITY ERROR INTERRUPT VECTOR ADDRESS

005362  032737  000040  177570  .MAMF:  BIT     #40,@#SWR       ;CHECK IF PARITY ERROR DETECTION IS TO
005370  001007                          BNE     1$              ;BE ENABLED. BRANCH IF NOT TO BE ENABLED
```

```
1177                                    ;ENABLE PARITY ERROR DETECTION
1178  005372  042737  000002  177746            BIC     #2,@#CNTRL      ;OTHERWISE, INSURE THAT PARITY ERROR
1179                                                                    ;DETECTION IS ENABLED
1180  005400  012767  000001  000056            MOV     #1,PARAVA       ;SET PARITY ERROR DETECTION INDICATOR
1181  005406  000405                            BR      2$
1182                                    ;DISABLE PARITY ERROR DETECTION
1183  005410  052737  000002  177746  1$:       BIS     #2,@#CNTRL      ;DISABLE PARITY ERROR DETECTION
1184  005416  005067  000042                    CLR     PARAVA          ;CLEAR PARITY ERROR DETECTION INDICATOR
1185                                    ;SET-UP PARITY ERROR SERVICE TRAP FOR 18-BIT OR 22-BIT
1186                                    ;ADDRESSING MODES
1187  005422  012737  004514  000114  2$:       MOV     #.PARSRV,@#PARVEC       ;SET-UP 18-BIT ADDRESS PARITY
1188                                                                    ;ERROR TRAP VECTOR
1189  005430  012737  000340  000116            MOV     #340,@#PARVEC+2 ;PRIORITY LEVEL 7 ON TRAP
1190  005436  022767  177776  173316            CMP     #-2,MMAVA       ;22-BIT ADDRESSING ENABLED?
1191  005444  001006                            BNE     3$              ;BRANCH IF NOT, OTHERWISE
1192  005446  012737  004772  000114            MOV     #.22PAR,@#PARVEC        ;SET-UP 22-BIT ADDRESS PARITY
1193                                                                    ;ERROR TRAP VECTOR
1194  005454  012737  000340  000116            MOV     #340,@#PARVEC+2 ;PRIORITY LEVEL 7 ON TRAP
1195  005462  000207                    3$:     RTS     PC              ;RETURN
1196  005464  000000                    PARAVA: .WORD   0               ;PARITY ERROR DETECTION INDICATOR
1197                                                                    ;0 - PARITY ERROR DETECTION IS DISABLED
1198                                                                    ;1 - PARITY ERROR DETECTION IS ENABLED
1199
1200
1201                                            .SBTTL MARGIN ROUTINE
1202                                    ;ROUTINE TO SET THE MARGINS
1203          177750                    MAINTRG=177750
1204
1205  005466  016700  173260            MARGIN: MOV     ICNT,R0         ;PASS COUNT
1206  005472  005002                            CLR     R2              ;FAST COUNTER
1207  005474  116037  005506  177750            MOVB    MRGNTB(R0),@#MAINTRG ;LOAD MAINTENANCE REG.
1208  005502  077201                    1$:     SOB     R2,1$
1209  005504  000207                            RTS     PC
1210
1211  005506      000                   MRGNTB: .BYTE   0               ;NORMAL
1212  005507      004                           .BYTE   4               ;EARLY STROBE
1213  005510      006                           .BYTE   6               ;LATE STROBE
1214  005511      010                           .BYTE   10              ;LOW CURRENT
1215  005512      012                           .BYTE   12              ;HIGH CURRENT
1216  005513      000                           .BYTE   0               ;NORMAL
1217
1218
1219                                            .SBTTL  MEM MGMT ROUTINES
1220                                    ;ROUTINE TO INITIALIZE MEMORY MANAGEMENT REGISTERS
1221  005514  000240                    LDMMO:  NOP
1222  005516  005767  173240                    TST     MMAVA
1223  005522  001434                            BEQ     1$
1224  005524  012737  000020  172516            MOV     #20,@#SR3       ;22 BIT MODE
1225  005532  012737  077006  172300            MOV     #177*256.-400+UP+RW,@#KIPDR0    ;SET KIPDR0=RW UP 177 BLOCKS
1226  005540  012737  077406  172302            MOV     #200*256.-400+UP+RW,@#KIPDR1    ;SET KIPDR1=RW UP 200 BLOCKS
1227  005546  005037  172304                    CLR     @#KIPDR2
1228  005552  005037  172344                    CLR     @#KIPAR2
1229  005556  012737  077406  172316            MOV     #200*256.-400+UP+RW,@#KIPDR7    ;SET KIPDR7=RW UP 200 BLOCKS
1230  005564  005037  172340                    CLR     @#KIPAR0
1231  005570  012737  000200  172342            MOV     #200,@#KIPAR1
1232  005576  012737  177600  172356            MOV     #177600,@#KIPAR7
```

```
1233  005604  012737  000001  177572          MOV     #1,@#SR0        ;ENABLE MEM MGMT
1234  005612  000240                          NOP
1235  005614  000207                  1$:     RTS     PC
1236
1237                                          ;MEMORY MANAGEMENT ABORT ROUTINE FOR WRITE UP
1238  005616  012702  020000          MMABT0: MOV     #20000,R2       ;RESET R2
1239  005622  062737  000200  172342          ADD     #200,@#KIPAR1   ;ADVANCE TO NEXT 4K
1240  005630  013716  177576                  MOV     @#SR2,(SP)      ;RETURN TO INSTRUCTION THAT
1241  005634  005037  177572                  CLR     @#SR0           ;DISABLE MEM MGMT
1242  005640  012737  000001  177572          MOV     #1,@#SR0        ;ENABLE MEM MGMT
1243  005646  000002                          RTI                     ;CAUSED THE ABORT
1244
1245                                          ;MEM MGMT ABORT SERVICE FOR WRITE DOWN
1246  005650  012702  040000          MMABT1: MOV     #40000,R2       ;RESET R2
1247  005654  162737  000200  172342          SUB     #200,@#KIPAR1
1248  005662  001406                          BEQ     2$
1249  005664  013716  177576                  MOV     @#SR2,(SP)
1250  005670  012737  000001  177572          MOV     #1,@#SR0        ;ENABLE MEM MGMT
1251  005676  000002                          RTI
1252  005700                          2$:
1253  005700  005037  177572                  CLR     @#SR0           ;DISABLE MEM MGMT
1254  005704  052766  000002  000002          BIS     #V,2(SP)
1255  005712  000002                          RTI
1256
1257                                          ;ROUTINE TO SET UP MEMORY MANAGEMENT FOR PATTERN TESTS
1258  005714  005702                  $TMM2:  TST     R2              ;CHECK IF TESTING BANK # 0
1259  005716  001442                          BEQ     2$              ;EXIT IF BANK # 0
1260  005720  005767  173036                  TST     MMAVA
1261  005724  001005                          BNE     1$              ;BRANCH IF MEM MGMT AVAILABLE
1262  005726  006002                          ROR     R2              ;ADJUST ADDRESS
1263  005730  006002                          ROR     R2
1264  005732  006002                          ROR     R2
1265  005734  006002                          ROR     R2
1266  005736  000207                          RTS     PC              ;RETURN
1267
1268  005740  004767  177550          1$:     JSR     PC,LDMM0        ;GO MAKE INITIAL SET UP
1269  005744  000302                          SWAB    R2
1270  005746  006002                          ROR     R2
1271  005750  010237  172344                  MOV     R2,@#KIPAR2
1272  005754  062702  000200                  ADD     #200,R2
1273  005760  010237  172346                  MOV     R2,@#KIPAR3
1274  005764  012737  077406  172304          MOV     #200*256.-400+UP+RW,@#KIPDR2    ;SET KIPDR2=RW UP 200 BLOCKS
1275  005772  012737  077406  172306          MOV     #200*256.-400+UP+RW,@#KIPDR3    ;SET KIPDR3=RW UP 200 BLOCKS
1276  006000  005037  172310                  CLR     @#KIPDR4
1277  006004  012702  040000                  MOV     #40000,R2
1278  006010  012737  006026  000250          MOV     #MMABT2,@#MMVEC
1279  006016  012737  000001  177572          MOV     #1,@#SR0        ;ENABLE MEM MGMT
1280  006024  000207                  2$:     RTS     PC
1281
1282                                          ;ROUTINE TO SERVICE 8 XOR 13 ABORTS
1283  006026  000240                  MMABT2: NOP
1284  006030  012702  040000                  MOV     #40000,R2
1285  006034  062737  000400  172344          ADD     #400,@#KIPAR2
1286  006042  062737  000400  172346          ADD     #400,@#KIPAR3
1287  006050  013716  177576                  MOV     @#SR2,(SP)      ;SET RETURN TO INSTRUCTION THAT ABORTED
1288  006054  012737  000001  177572          MOV     #1,@#SR0        ;ENABLE MEM MGMT
```

```
1289  006062  000002                        RTI
1290
1291                                        .SBTTL  3 XOR 9 ROUTINES
1292                                ;ROUTINE TO WRITE 3XOR9 WORST CASE NOISE TEST PATTERN
1293                                ;CALL:  MOV     BANK #,-(SP)    ;PUSH STARTING BANK # ON STACK
1294                                ;       MOV     BLKCNT,-(SP)    ;PUSH 256. WORD BLOCK COUNT ON STACK
1295                                ;       JSR     PC,.3X9         ;CALL ROUTINE
1296
1297  006064  016602  000004        .3X9:   MOV     4(SP),R2        ;GET STARTING BANK #
1298  006070  004767  177620                JSR     PC,STMM2
1299  006074  005000                        CLR     R0
1300  006076  010003                        MOV     R0,R3
1301  006100  005103                        COM     R3              ;R0 (0) AND R3 (-1) IS THE DATA WRITTEN
1302  006102  005767  175364                TST     PARPAT          ;BRANCH IF PARITY MEMORY PATTERN IS
1303  006106  001402                        BEQ     1$              ;NOT TO BE WRITTEN
1304
1305  006110  012700  000401                MOV     #401,R0         ;WRITE PARITY 3X9 PATTERN
1306  006114  012704  000020        1$:     MOV     #16.,R4         ;EACH LOOP WRITES 256. WORDS
1307
1308  006120  010022                2$:     MOV     R0,(R2)+
1309  006122  010022                        MOV     R0,(R2)+
1310  006124  010022                        MOV     R0,(R2)+
1311  006126  010022                        MOV     R0,(R2)+
1312
1313  006130  010022                        MOV     R0,(R2)+
1314  006132  010022                        MOV     R0,(R2)+
1315  006134  010022                        MOV     R0,(R2)+
1316  006136  010022                        MOV     R0,(R2)+
1317
1318  006140  010322                        MOV     R3,(R2)+
1319  006142  010322                        MOV     R3,(R2)+
1320  006144  010322                        MOV     R3,(R2)+
1321  006146  010322                        MOV     R3,(R2)+
1322
1323  006150  010322                        MOV     R3,(R2)+
1324  006152  010322                        MOV     R3,(R2)+
1325  006154  010322                        MOV     R3,(R2)+
1326  006156  010322                        MOV     R3,(R2)+
1327
1328  006160  005304                        DEC     R4
1329  006162  001356                        BNE     2$
1330  006164  005100                        COM     R0
1331  006166  005103                        COM     R3
1332  006170  005767  175276                TST     PARPAT          ;BRANCH IF PARITY MEMORY PATTERN IS
1333  006174  001402                        BEQ     3$              ;NOT TO BE WRITTEN
1334
1335  006176  004767  000014                JSR     PC,.XOR39       ;GO GET CONSTANTS
1336  006202  005366  000002        3$:     DEC     2(SP)           ;DECREMENT 256. WORD BLOCK COUNT
1337  006206  001342                        BNE     1$
1338  006210  012616                        MOV     (SP)+,(SP)      ;ADJUST STACK
1339  006212  012616                        MOV     (SP)+,(SP)
1340  006214  000207                        RTS     PC
1341
1342                                ;ROUTINE TO SET CONSTANTS FOR WRITING/CHECKING 3 XOR PATTERN WITH
1343                                ;PARITY.
1344  006216  032702  000020        .XOR39: BIT     #20,R2          ;CHECK BIT 3
```

```
1345  006222  001404                        BEQ     .3IS0           ;BRANCH IF BIT 3 = 0
1346  006224  032702  002000        .3IS1:  BIT     #2000,R2        ;CHECK BIT 9
1347  006230  001404                        BEQ     .3NOT9          ;BRANCH IF BIT 9 =0
1348  006232  000407                        BR      .3IS9
1349  006234  032702  002000        .3IS0:  BIT     #2000,R2        ;CHECK BIT 9
1350  006240  001404                        BEQ     .3IS9           ;BRANCH IF 0
1351  006242  005767  172506        .3NOT9: TST     ICOUNT          ;CHECK IF NORMAL OR COMPLEMENT DATA
1352  006246  100004                        BPL     LDCOMP          ;GO LOAD COMPLEMENT CONSTANTS
1353  006250  100410                        BMI     LDNORM          ;GO LOAD NORMAL CONSTANTS
1354  006252  005767  172476        .3IS9:  TST     ICOUNT          ;CHECK IF NORMAL OR COMPLEMENT DATA
1355  006256  100005                        BPL     LDNORM          ;GO LOAD NORMAL CONSTANTS
1356  006260  012700  177777        LDCOMP: MOV     #-1,R0          ;SET COMPLEMENT CONSTANTS
1357  006264  012703  000401                MOV     #401,R3
1358  006270  000207                        RTS     PC              ;RETURN
1359  006272  012700  000401        LDNORM: MOV     #401,R0         ;LOAD NORMAL CONSTANTS
1360  006276  012703  177777                MOV     #-1,R3
1361  006302  000207                        RTS     PC
1362
1363                                 ;ROUTINE TO CHECK 3 XOR 9 WORST CASE NOISE PATTERN
1364                                 ;CALL:  MOV     BANK#,-(SP)     ;PUSH STARTING BANK # ONTO STACK
1365                                 ;       MOV     BLKCNT,-(SP)    ;AND 256. WORD BLOACK COUNT
1366                                 ;       JSR     PC,..3X9        ;CALL ROUTINE
1367
1368  006304  000240                ..3X9:  NOP
1369  006306  004767  001104                JSR     PC,CKSWR        ;GO CHECK SWITCH REGISTER
1370
1371                                 ;CHECK WORST CASE PATTERN
1372  006312  016604  000002        1$:     MOV     2(SP),R4        ;GET 256. BLOCK WORD COUNT
1373  006316  016602  000004                MOV     4(SP),R2        ;GET FIRST BANK #
1374  006322  004767  177366                JSR     PC,STMM2        ;GO SET UP MEM MGMT
1375  006326  005000                        CLR     R0              ;SET CHECK WORD
1376  006332  005767  172420                TST     ICOUNT          ;IF ICOUNT IS NEG AM CHECKING COMP-
1377  006336  100001                        BPL     .+4             ;LEMENTED PATTERN
1378  006340  005100                        COM     R0              ;SO COMPLEMENT CHECK WORD
1379  006344  012705  000040        2$:     MOV     #32.,R5         ;SET 256. WORD COUNTER
1380
1381  006350  005767  175122        3$:     TST     PARPAT          ;BRANCH IF PARITY MEMORY PATTERN IS
1382  006354  001402                        BEQ     30$             ;NOT TO BE CHECKED
1383
1384  006356  004767  177640                JSR     PC,.XOR39       ;GO GET CONSTANT
1385  006362
1386  006362  012203                30$:
                                        MOV     (R2)+,R3        ;GET TEST DATA
1387  006364  020003                        CMP     R0,R3           ;COMPARE WITH CHECK WORD
1388  006366  001403                        BEQ     .+10
1389  006370  005046                        CLR     -(SP)           ;PUSH FAKE STATUS ON THE STACK
1390  006372  004767  172612                JSR     PC,ERROR        ;ERROR! MEM DATA (R3) NOT = TEST DATA
1391                                                                 ;(R0), ADDRESS=(R2)-2
1392
1393  006376  012203                        MOV     (R2)+,R3        ;GET TEST DATA
1394  006400  020003                        CMP     R0,R3           ;COMPARE WITH CHECK WORD
1395  006402  001403                        BEQ     .+10
1396  006404  005046                        CLR     -(SP)           ;PUSH FAKE STATUS ON THE STACK
1397  006406  004767  172576                JSR     PC,ERROR        ;ERROR! MEM DATA (R3) NOT = TEST DATA
1398                                                                 ;(R0), ADDRESS=(R2)-2
1399
1400  006412  012203                        MOV     (R2)+,R3        ;GET TEST DATA
```

```
1401  006410  020003                        CMP     R0,R3           ;COMPARE WITH CHECK WORD
1402  006412  001403                        BEQ     .+10
1403  006414  005046                        CLR     -(SP)           ;PUSH FAKE STATUS ON THE STACK
1404  006416  004767  172562                JSR     PC,ERROR        ;ERROR! MEM DATA (R3) NOT = TEST DATA
1405                                                                  ;(R0), ADDRESS=(R2)-2
1406
1407  006422  012203                        MOV     (R2)+,R3        ;GET TEST DATA
1408  006424  020003                        CMP     R0,R3           ;COMPARE WITH CHECK WORD
1409  006426  001403                        BEQ     .+10
1410  006430  005046                        CLR     -(SP)           ;PUSH FAKE STATUS ON THE STACK
1411  006432  004767  172546                JSR     PC,ERROR        ;ERROR! MEM DATA (R3) NOT = TEST DATA
1412                                                                  ;(R0), ADDRESS=(R2)-2
1413
1414  006436  012203                        MOV     (R2)+,R3        ;GET TEST DATA
1415  006440  020003                        CMP     R0,R3           ;COMPARE WITH CHECK WORD
1416  006442  001403                        BEQ     .+10
1417  006444  005046                        CLR     -(SP)           ;PUSH FAKE STATUS ON THE STACK
1418  006446  004767  172532                JSR     PC,ERROR        ;ERROR! MEM DATA (R3) NOT = TEST DATA
1419                                                                  ;(R0), ADDRESS=(R2)-2
1420
1421  006452  012203                        MOV     (R2)+,R3        ;GET TEST DATA
1422  006454  020003                        CMP     R0,R3           ;COMPARE WITH CHECK WORD
1423  006456  001403                        BEQ     .+10
1424  006460  005046                        CLR     -(SP)           ;PUSH FAKE STATUS ON THE STACK
1425  006462  004767  172516                JSR     PC,ERROR        ;ERROR! MEM DATA (R3) NOT = TEST DATA
1426                                                                  ;(R0), ADDRESS=(R2)-2
1427
1428  006466  012203                        MOV     (R2)+,R3        ;GET TEST DATA
1429  006470  020003                        CMP     R0,R3           ;COMPARE WITH CHECK WORD
1430  006472  001403                        BEQ     .+10
1431  006474  005046                        CLR     -(SP)           ;PUSH FAKE STATUS ON THE STACK
1432  006476  004767  172502                JSR     PC,ERROR        ;ERROR! MEM DATA (R3) NOT = TEST DATA
1433                                                                  ;(R0), ADDRESS=(R2)-2
1434
1435  006502  012203                        MOV     (R2)+,R3        ;GET TEST DATA
1436  006504  020003                        CMP     R0,R3           ;COMPARE WITH CHECK WORD
1437  006506  001403                        BEQ     .+10
1438  006510  005046                        CLR     -(SP)           ;PUSH FAKE STATUS ON THE STACK
1439  006512  004767  172466                JSR     PC,ERROR        ;ERROR! MEM DATA (R3) NOT = TEST DATA
1440                                                                  ;(R0), ADDRESS=(R2)-2
1441
1442
1443  006516  005100                        COM     R0              ;COMPLEMENT CHECK WORD
1444  006520  005305                        DEC     R5              ;DECREMENT 256. WORD COUNTER
1445  006522  001310                        BNE     3$
1446  006524  005100                        COM     R0              ;COMPLEMENT CHECK WORD
1447  006526  005304                        DEC     R4              ;DECREMENT BLOCK COUNTER
1448  006530  001303                        BNE     2$
1449
1450  006532  032737  040000  177570        BIT     #40000,@#SWR    ;LOOP ON TEST?
1451  006540  001264                        BNE     1$              ;BRANCH IF LOOP ON TEST DESIRED
1452  006542  016667  000002  172216  40$:  MOV     2(SP),COUNT     ;GET # OF 256. WORD BLOCKS TO CHECK
1453  006550  016602  000004                MOV     4(SP),R2        ;GET STARTING BANK #
1454  006554  004767  177134                JSR     PC,STMM2        ;GO SET UP MEM MGMT IF REQUIRD
1455
1456                                ;CHECK WORST CASE BIT COMPLEMENT PATTERN
```

```
1457  006560  005000                        CLR     R0
1458  006562  005767  172166                TST     ICOUNT          ;CHECK IF COMPLEMENT PATERN
1459  006566  100001                        BPL     .+4
1460  006570  005100                        COM     R0              ;COMPLEMENT CHECK WORD
1461  006572  012704  000040        4$:     MOV     #32.,R4         ;SET 256. WORD COUNTER
1462  006576  012705  000010        5$:     MOV     #8.,R5          ;SET 8 WORD COUNTER
1463  006602  005767  174664        6$:     TST     PARPAT          ;BRANCH IF PARITY MEMORY PATTERN IS
1464  006606  001402                        BEQ     60$             ;NOT TO BE CHECKED
1465  006610  004767  177402                JSR     PC,.XOR39
1466  006614  012203                60$:    MOV     (R2)+,R3        ;GET DATA
1467  006616  020003                        CMP     R0,R3           ;CHECK DATA
1468  006620  001403                        BEQ     .+10
1469  006622  005046                        CLR     -(SP)
1470  006624  004767  172354                JSR     PC,ERROR
1471  006630  005100                        COM     R0              ;COMPLEMENT CHECK WORD
1472  006632  005142                        COM     -(R2)           ;COMPLEMENT TEST DATA
1473  006634  012203                        MOV     (R2)+,R3        ;GET DATA
1474  006636  020003                        CMP     R0,R3           ;CHECK
1475  006640  001403                        BEQ     .+10
1476  006642  005046                        CLR     -(SP)           ;PUSH FAKE STATUS ON THE STACK
1477  006644  004767  172334                JSR     PC,ERROR
1478  006650  005100                        COM     R0              ;COMPLEMENT CHECK WORD
1479  006652  005162  177776                COM     -2(R2)          ;RESTORE DATA
1480  006656  005305                        DEC     R5              ;DECREMENT 4 WORD COUNTER
1481  006660  001350                        BNE     6$
1482  006662  005100                        COM     R0              ;COMPLEMENT CHECK WORD
1483  006664  005304                        DEC     R4              ;DECREMENT 256. WORD COUNTER
1484  006666  001343                        BNE     5$
1485  006670  005100                        COM     R0              ;COMPLEMENT CHECK WORD
1486  006672  005367  172070                DEC     COUNT           ;DECREMENT BLOCK COUNTER
1487  006676  001335                        BNE     4$
1488
1489  006700  016602  000004                MOV     4(SP),R2        ;GET BANK #
1490  006704  004767  177004                JSR     PC,STMM2
1491  006710  016603  000002                MOV     2(SP),R3        ;GET BLOCK COUNT
1492  006714  032737  040000  177570        BIT     #40000,@#SWR    ;LOOP ON TEST
1493  006722  001307                        BNE     40$             ;BRANCH IF LOOP ON TEST
1494  006724  006367  172024                ASL     ICOUNT
1495  006730  102402                        BVS     7$
1496  006732  000167  177354                JMP     1$
1497  006736  012705  000020        7$:     MOV     #16.,R5         ;COMPLEMENT PATTERN
1498  006742  011200                10$:    MOV     (R2),R0         ;GET 1ST DATA WORD
1499  006744  016204  000020                MOV     20(R2),R4       ;GET 9TH DATA WORD
1500  006750  110422                        MOVB    R4,(R2)+        ;SWAP WORDS 1-8
1501  006752  110422                        MOVB    R4,(R2)+        ;WITH 9-16
1502  006754  110422                        MOVB    R4,(R2)+
1503  006756  110422                        MOVB    R4,(R2)+
1504  006760  110422                        MOVB    R4,(R2)+
1505  006762  110422                        MOVB    R4,(R2)+
1506  006764  110422                        MOVB    R4,(R2)+
1507  006766  110422                        MOVB    R4,(R2)+
1508  006770  110422                        MOVB    R4,(R2)+
1509  006772  110422                        MOVB    R4,(R2)+
1510  006774  110422                        MOVB    R4,(R2)+
1511  006776  110422                        MOVB    R4,(R2)+
1512  007000  110422                        MOVB    R4,(R2)+
```

```
1513  007002  110422                          MOVB    R4,(R2)+
1514  007004  110422                          MOVB    R4,(R2)+
1515  007006  110422                          MOVB    R4,(R2)+
1516  007010  110022                          MOVB    R0,(R2)+        ;AND VICE VERSA
1517  007012  110022                          MOVB    R0,(R2)+
1518  007014  110022                          MOVB    R0,(R2)+
1519  007016  110022                          MOVB    R0,(R2)+
1520  007020  110022                          MOVB    R0,(R2)+
1521  007022  110022                          MOVB    R0,(R2)+
1522  007024  110022                          MOVB    R0,(R2)+
1523  007026  110022                          MOVB    R0,(R2)+
1524  007030  110022                          MOVB    R0,(R2)+
1525  007032  110022                          MOVB    R0,(R2)+
1526  007034  110022                          MOVB    R0,(R2)+
1527  007036  110022                          MOVB    R0,(R2)+
1528  007040  110022                          MOVB    R0,(R2)+
1529  007042  110022                          MOVB    R0,(R2)+
1530  007044  110022                          MOVB    R0,(R2)+
1531  007046  110022                          MOVB    R0,(R2)+
1532  007050  005305                          DEC     R5
1533  007052  001333                          BNE     10$
1534  007054  005303                          DEC     R3
1535  007056  001327                          BNE     7$
1536
1537  007060  005767  171670                 TST     ICOUNT
1538  007064  001402                          BEQ     11$
1539  007066  000167  177220                 JMP     1$
1540  007072  012616                 11$:    MOV     (SP)+,(SP)
1541  007074  012616                         MOV     (SP)+,(SP)
1542  007076  000207                         RTS     PC
1543
1544                                 ;ROUTINE TO WRITE 8 XOR 13 WORST CASE NOISE TEST PATTERN
1545                                         .SBTTL  8 XOR 13 ROUTINES
1546                                 ;CALL:  MOV     BANK #,-(SP)
1547                                 ;       MOV     #4KBANKS,-(SP)
1548                                 ;       JSR     PC,.8X13
1549
1550  007100  012616                 .8X13:  MOV     (SP)+,(SP)      ;ADJUST STACK
1551  007102  012616                         MOV     (SP)+,(SP)
1552  007104  000207                         RTS     PC
1553
1554                                 ;ROUTINE TO CHECK 8 XOR 13 WORST CASE NOISE TEST PATTERN
1555                                 ;CALL:
1556                                 ;       MOV     BANK#,-(SP)     ;PUSH FIRST BANK # ON THE STACK
1557                                 ;       MOV     #BANKS,-(SP)    ;PUSH # OF 4K BANKS TO CHECK ON THE STACK
1558                                 ;       JSR     PC,..8X13       ;CALL ROUTINE
1559
1560  007106  012616                 ..8X13: MOV     (SP)+,(SP)
1561  007110  012616                         MOV     (SP)+,(SP)
1562  007112  000207                         RTS     PC              ;RETURN
1563
1564                                         .SBTTL  ROTATING 1'S & 0'S ROUTINES
1565                                 ;ROUTINE TO CHECK ROTATING '0' BIT THROUGH FILD OF 1'S
1566                                 ;CALL:  MOV     BANK#,-(SP)     ;SET STARTING BANK #
1567                                 ;       MOV     BLKCNT,-(SP)    ;SET 256. WORD BLOCK COUNT
1568                                 ;       JSR     PC,.ROTO        ;CALL ROUITNE
```

```
1569
1570  007114  004767  000276          .ROT0:  JSR     PC,CKSWR        ;GO CHECK SWITCHES
1571  007120  016604  000002                  MOV     2(SP),R4        ;GET 256. WORD BLOCK COUNT
1572  007124  016602  000004                  MOV     4(SP),R2        ;GET FIRST BANK #
1573  007130  004767  176560                  JSR     PC,STMM2        ;GO SET UP MEM MGMT (IF AVAIL)
1574  007134  012700  177777                  MOV     #-1,R0          ;SET CHECK WORD
1575
1576  007140  012705  000400          1$:     MOV     #256.,R5        ;SET 256. WORD COUNT
1577  007144  000241                  2$:     CLC                     ;CLEAR CARRY BIT IN PSW
1578  007146  004767  000124                  JSR     PC,ROTATE
1579  007152  016203  177776                  MOV     -2(R2),R3       ;GET RESULT
1580  007156  103402                          BCS     3$              ;BRANCH IF 'C' BIT WAS SET
1581  007160  020003                          CMP     R0,R3           ;CHECK RESULT
1582  007162  001403                          BEQ     4$
1583  007164  005046                  3$:     CLR     -(SP)           ;ERROR! COULD NOT ROTATE '0' BIT
1584  007166  004767  172012                  JSR     PC,ERROR        ;THROUGH ADDRESS IN R2
1585  007172  005305                  4$:     DEC     R5              ;DECREMENT 256. WORD COUNT
1586  007174  001363                          BNE     2$              ;LOOP UNTIL DONE
1587  007176  005304                          DEC     R4              ;DECREMENT 256. WORD BLOCK COUNT
1588  007200  001357                          BNE     1$              ;LOOP UNTIL DONE
1589  007202  012616                          MOV     (SP)+,(SP)      ;POP CONSTANTS OFF THE STACK
1590  007204  012616                          MOV     (SP)+,(SP)
1591  007206  000207                          RTS     PC              ;RETURN TO CALLER
1592
1593                                  ;ROUTINE TO CHECK ROTATING '1' BIT THROUGH A FIELD OF 0'S
1594                                  ;CALL:  MOV     BANK#,-(SP)     ;SET STARTING BANK #
1595                                  ;       MOV     BLKCNT,-(SP)    ;SET # OF 256. WORD BLOCKS TO CHECK
1596                                  ;       JSR     PC,.ROT1        ;CALL ROUTINE
1597
1598  007210  004767  000202          .ROT1:  JSR     PC,CKSWR        ;GO CHECK SWITCHES
1599  007214  016604  000002                  MOV     2(SP),R4        ;GET # OF 256. WORD BLOCKS TO CHECK
1600  007220  016602  000004                  MOV     4(SP),R2        ;GET STARTING BANK #
1601  007224  004767  176464                  JSR     PC,STMM2        ;GO SET UP MEM MGMT (IF AVAIL)
1602  007230  005000                          CLR     R0              ;SET CHECK WORD
1603
1604  007232  012705  000400          1$:     MOV     #256.,R5        ;SET 256. WORD COUNTER
1605  007236  000261                  2$:     SEC                     ;SET 'C' BIT IN PSW
1606  007240  004767  000032                  JSR     PC,ROTATE       ;GO ROTATE '1' BIT
1607  007244  016203  177776                  MOV     -2(R2),R3       ;GET RESULT
1608  007250  103002                          BCC     3$              ;BRANCH IF 'C' IS CLEAR
1609  007252  020003                          CMP     R0,R3           ;CHECK RESULT
1610  007254  001401                          BEQ     .+4
1611  007256  104400                  3$:     HLT                     ;ERROR! COULD NOT ROTATE '1' BIT
1612                                                                  ;THROUGH ADDRESS IN R2
1613  007260  005305                          DEC     R5              ;DECREMENT 256. WORD COUNT
1614  007262  001365                          BNE     2$
1615  007264  005304                          DEC     R4              ;DECREMENT 256. WORD BLOCK COUNT
1616  007266  001361                          BNE     1$
1617  007270  012616                          MOV     (SP)+,(SP)      ;ADJUST RETURN ADDRESS
1618  007272  012616                          MOV     (SP)+,(SP)
1619  007274  000207                          RTS     PC              ;RETURN TO CALLER
1620
1621                                  ;ROUTINE TO ROTATE 'C' BIT THROUGH A MEMORY LOCATION.
1622  007276  106112                  ROTATE: ROLB    (R2)            ;(R2)=177776 OR 000001
1623  007300  106112                          ROLB    (R2)            ;(R2)=177775 OR 000002
1624  007302  106112                          ROLB    (R2)            ;(R2)=177773 OR 000004
```

```
1625  007304  106112                          ROLB    (R2)            ;(R2)=177767 OR 000010
1626  007306  106112                          ROLB    (R2)            ;(R2)=177757 OR 000020
1627  007310  106112                          ROLB    (R2)            ;(R2)=177737 OR 000040
1628  007312  106112                          ROLB    (R2)            ;(R2)=177677 OR 000100
1629  007314  106112                          ROLB    (R2)            ;(R2)=177777 OR 000000
1630  007316  106122                          ROLB    (R2)+           ;(R2)=177577 OR 000200
1631  007320  106112                          ROLB    (R2)            ;(R2)=177377 OR 000400
1632  007322  106112                          ROLB    (R2)            ;(R2)=176777 OR 001000
1633  007324  106112                          ROLB    (R2)            ;(R2)=175777 OR 002000
1634  007326  106112                          ROLB    (R2)            ;(R2)=173777 OR 004000
1635  007330  106112                          ROLB    (R2)            ;(R2)=167777 OR 010000
1636  007332  106112                          ROLB    (R2)            ;(R2)=157777 OR 020000
1637  007334  106112                          ROLB    (R2)            ;(R2)=137777 OR 040000
1638  007336  106112                          ROLB    (R2)            ;(R2)=077777 OR 100000
1639  007340  106122                          ROLB    (R2)+           ;(R2)=177777 OR 000000
1640  007342  000207                          RTS     PC              ;RETURN
1641
1642                                  ;ROUTINE TO WRITE ONE WORD PATTERN INTO MEMORY
1643                                  ;CALL:  MOV     BANK#,-(SP)     ;PUSH STARTING BANK # ONTO STACK
1644                                  ;       MOV     BLKCNT,-(SP)    ;AND 128. WORD BLOCK COUNT
1645                                  ;       JSR     PC,WRTPAT       ;CALL ROUTINE
1646
1647  007344  016604  000002         WRTPAT: MOV     2(SP),R4        ;GET BLOCK COUNT
1648  007350  016602  000004                 MOV     4(SP),R2        ;GET STARTING BANK #
1649  007354  004767  176334                 JSR     PC,STMM2        ;GO SET UP MEM MGMT
1650  007360  012700                         MOV     (PC)+,R0        ;GET USER CONSTANT
1651  007362  000000                 .CONST: 0
1652  007364  012703  000100         1$:     MOV     #64.,R3         ;SET 256. WORD COUNTER
1653  007370  010022                 2$:     MOV     R0,(R2)+        ;WRITE 256. WORDS
1654  007372  010022                         MOV     R0,(R2)+
1655  007374  010022                         MOV     R0,(R2)+
1656  007376  010022                         MOV     R0,(R2)+
1657  007400  005303                         DEC     R3              ;DECREMENT 256. WORD COUNTER
1658  007402  001372                         BNE     2$              ;LOOP UNTIL 256. WORDS HAVE BEEN WRITTEN
1659  007404  005304                         DEC     R4              ;DECREMENT BLOCK COUNT
1660  007406  001366                         BNE     1$
1661  007410  012616                         MOV     (SP)+,(SP)      ;ADJUST STACK
1662  007412  012616                         MOV     (SP)+,(SP)
1663  007414  000207                         RTS     PC
1664
1665
1666                                  ;ROUTINE TO CHECK THE SWITCH REGISTER
1667                                  ;CHECK SWITCH 9: IF SET, LOAD ERROR COUNT INTO THE DISPLAY REGISTER;
1668                                  ;IF NOT SET, LOAD PASS COUNT INTO THE DISPLAY REGISTER
1669  007416  042767  017777  171334 CKSWR:  BIC     #17777,LDDISP   ;SAVE RELOCATION BITS
1670  007424  032737  000400  177570         BIT     #BIT8,@#SWR     ;CHECK SWITCH 8
1671  007432  001402                         BEQ     10$             ;BRANCH IF SET
1672  007434  004767  000464                 JSR     PC,REL24K       ;GO RELOCATE PROGRAM BACK TO 4K AND STOP
1673  007440  032737  001000  177570 10$:    BIT     #BIT9,@#SWR     ;SWITCH 9 SET ?
1674  007446  001404                         BEQ     1$
1675  007450  056767  171302  171302         BIS     ERCNT,LDDISP    ;LOAD ERROR COUNT
1676  007456  000403                         BR      2$
1677  007460  056767  171266  171272 1$:     BIS     ICNT,LDDISP     ;LOAD PASS COUNT
1678  007466  016737  171266  177570 2$:     MOV     LDDISP,@#DISPLAY;LOAD THE DISPLAY REGISTER
1679  007474  012767  040177  171252         MOV     #040177,ICOUNT  ;LOAD ITERATION COUNT WORD
1680  007502  032737  004000  177570         BIT     #4000,@#SWR     ;CHECK SW11
```

```
1681  007510  001402                          BEQ     .+6
1682  007512  105067  171236                  CLRB    ICOUNT          ;ICOUNT =040000 IF SW11 =1
1683  007516  000207                          RTS     PC
1684
1685                                          ;MESSAGES
1686  007520  005015  047524  051040  RESLDR: .ASCIZ  <15><12>'TO RESTORE LOADERS START AT 162'<15><12>
1687  007526  051505  047524  042522
1688  007534  046040  040517  042504
1689  007542  051522  051440  040524
1690  007550  052122  040440  020124
1691  007556  033061  006462  000012
1692  007564  005015  047105  041101  PARITY: .ASCIZ  <15><12> 'ENABLE PARITY? 1/0=YES/NO '
1693  007572  042514  050040  051101
1694  007600  052111  037531  030440
1695  007606  030057  054475  051505
1696  007614  047057  020117     000
1697  007621     015  051412  040524  STBANK: .ASCIZ  <15><12>'STARTING BANK #(8)? '
1698  007626  052122  047111  020107
1699  007634  040502  045516  021440
1700  007642  034050  037451  000040
1701  007650  005015  020043  043117  BANKS:  .ASCIZ  <15><12>'# OF 4K BANKS TO TEST(8)? '
1702  007656  032040  020113  040502
1703  007664  045516  020123  047524
1704  007672  052040  051505  024124
1705  007700  024470  020077     000
1706  007705     015  050012  052101  PAT:    .ASCIZ  <15><12>'PATTERN #? '
1707  007712  042524  047122  021440
1708  007720  020077     000
1709  007723     015  037412     000  QUEST:  .ASCIZ  <15><12>'?'
1710  007727     015  052012  050131  CONST:  .ASCIZ  <15><12>'TYPE CONSTANT'
1711  007734  020105  047503  051516
1712  007742  040524  052116     000
1713  007747     015  044412  050116  PRG3M:  .ASCIZ  <15><12>'INPUT # OF 256. WORD BLOCKS TO TEST INSTEAD OF'
1714  007754  052125  021440  047440
1715  007762  020106  032462  027066
1716  007770  053440  051117  020104
1717  007776  046102  041517  051513
1718  010004  052040  020117  042524
1719  010012  052123  044440  051516
1720  010020  042524  042101  047440
1721  010026  000106
1722  010030  005015  054524  042520  PRG4M:  .ASCIZ  <15><12>'TYPE ADDRESS'
1723  010036  040440  042104  042522
1724  010044  051523     000
1725  010047     015  052012  020117  RELOCM: .ASCIZ  <15><12>'TO RESTORE PROGRAM START AT '
1726  010054  042522  052123  051117
1727  010062  020105  051120  043517
1728  010070  040522  020115  052123
1729  010076  051101  020124  052101
1730  010104  000040
1731  010106  000052                  ASTERISK:.ASCIZ '*'
1732  010110  042504  045115  020101  ENDMSG: .ASCIZ  'DEMJA DONE!'
1733  010116  047504  042516  000041
1734                                          .EVEN
1735
1736                                          ;ROUTINE TO RELOCATE PROGRAM BACK TO 0
```

```
1737  010124  010700                  REL24K: MOV     PC,R0           ;FORM BASE ADDRESS WHERE CODE
1738  010126  042700  017777                  BIC     #17777,R0       ;IS RELOCATED
1739  010132  010067  000004                  MOV     R0,1$           ;PUT FROM ADDRESS INTO SUBROUTINE CALL
1740  010136  004567  174132                  JSR     R5,RELOC        ;RELOCATE CODE TO
1741  010142  000000                  1$:     0                       ;LOWEST 4K
1742  010144  000000                          0
1743  010146  012706  000500                  MOV     #STKPTR,SP      ;SET STACK PTR
1744  010152  042737  100000  000760          BIC     #100000,@#LDDISP ;CLEAR RELOCATION INDICATOR
1745  010160  013737  000760  177570          MOV     @#LDDISP,@#DISPLAY ;LOAD DISPLAY REGISTER
1746  010166  005037  000764                  CLR     @#RELOCF        ;CLEAR RELOCATION FACTOR
1747  010172  000005                          RESET                   ;DISABLE MEM MGMT
1748  010174  000137  000162                  JMP     @#PONE          ;RESTORE LOADERS & HALT
1749                                  ;
1750  010200                          LODAR:
1751          000001                          .END
```

```
ADRESS  002327            374     559#   1127
ASTERI  010106            939    1731#
BANKS   007650           1701#
BEGIN1  003446            792     794     800#    940
BELL    001527            393     409#
BIT12 = 010000             89#    604
BIT13 = 020000             88#   1101
BIT14 = 040000             87#
BIT15 = 000100             86#
BIT6  = 000100             92#
BIT8  = 000400             91#   1670
BIT9  = 001000             90#    353    1673
BPTVEC= 000014             58#
C     = 000001             38#
CKSWR   007416            803    1369    1570    1598    1669#
CNTRL = 177746             74#    594*    801*    930*   1178*   1183*
CONST   007727           1710#
COUNT   000766            274#   1452*   1486*
CRLF    000100            157#    388     632    1128
DIGITS  002351            340     370     419     563#   1115
DIGTAB  002312            457     552#   1123
DISPLA= 177570             80#    357*    591*   1678*   1745*
DONE    004232            937     941#
DONE0   002626            618     628#
DONE1   002732            637     661#
DONE10  003714            860     866#
DONE3   003034            686     691#
DONE4   003122            697     708#
DONE4A  003172            711     724#
DONE4B  003242            727     741#
DONE4C  003310            752     754     759#
DONE4D  003422            759     790#
DONE6   003510            807     816#    843
DONE7   003536            818     824#
DONE8   003630            839     841     846#
DONE9   003656            847     854#
EMTVEC= 000030             61#
END     004166            929#
ENDCT   004212            795*    935#
ENDMSG  010110            943    1732#
ERCNT   000756            267#    350     352*    356     589*   1675
ERROR   001204            149     349#   1390    1397    1404    1411    1418    1425    1432    1439    1470    1477    1584
ERRPC   001475            368     404#   1108
ERRTRP  001126            146     328#    824     856     905    1000
ERRVEC= 000004             54#    509*    606*    618*    637*    697*    711*    727*    759*    818*    824*    847*    856*
                          871*    905*    962*    995*   1000*   1035*
ESTRB   005265           1149    1158#
FINISH  004270            947     953#
FPEVEC= 000244             66#
HCRNT   005332           1152    1165#
HLT   = 104400            120#    645     652     706     737     770     777     785     892     975    1046    1063    1611
ICNT    000752            265#    597*    933*    936    1137    1205    1677
ICOUNT  000754            266#   1351    1354    1376    1458    1494*   1537    1679*   1682*
IOTVEC= 000020             59#
KIPAR0= 172340            109#    217     466    1230*
KIPAR1= 172342            110#    467     669     695*    899*   1231*   1239*   1247*
```

```
KIPAR2= 172344          111#    468     877*    900*    1228*   1271*   1285*
KIPAR3= 172346          112#    469    1273*   1286*
KIPAR4= 172350          113#    470
KIPAR5= 172352          114#    471
KIPAR6= 172354          115#    472
KIPAR7= 172356          116#    230     473    1232*
KIPDR0= 172300          100#    212     875*   1040*   1225*
KIPDR1= 172302          101#   1226*
KIPDR2= 172304          102#    876*   1227*   1274*
KIPDR3= 172306          103#   1275*
KIPDR4= 172310          104#   1276*
KIPDR5= 172312          105#
KIPDR6= 172314          106#
KIPDR7= 172316          107#    235    1229*
KM    =  000000          45#
LCRNT   005316         1151    1163#
LDCOMP  006260         1352    1356#
LDDISP  000760          268#    355*    356*    357     590*    591     1669*   1675*   1677*   1678    1744*   1745
LDMMO   005514          620     640     713     729     761     874     1039    1221#   1268
LDNORM  006272         1353    1355    1359#
LODAR   010200          518     532    1750#
LODFLO  002110          503#    505     523*    527     541*
LST     002322          558#    631
LSTLOC  002244          522*    538#
LSTRB   005302         1150    1161#
MAINTR= 177750         1203#   1207*
MARGIN  005466          808    1205#
MARMSG  005347         1144    1168#
MARTBL  005240         1139    1142    1147#
MEMERR= 177744         1086#   1097    1100*
MEMHI = 177742         1085#   1039
MEMLO = 177740         1084#   1038
MMABTO  005616          621     696     714     762     1041    1238#
MMABTI  005650          679     746    1246#
MMABT2  006026         1278    1283#
MMAVA   000762          207     228     269#    424     603*    608*    612*    667     693     806     836     872     897
                       1037    1190    1222    1260
MMVEC =  000250           67#    621*    679*    696*    714*    746*    762*   1041*   1278*
MRGNTB  005506         1138    1207    1211#
N     =  000010          41#
NOFIND  004745         1062    1077#
NORMAL  005256         1147    1153    1156#
O2A     002014          382     387     478#   1010    1111    1133
PARAVA  005464          241     838    1180*   1184*   1196#
PARERR  004724         1030    1074#   1105
PARITY  007564         1692#
PARPAT  003472          805#    840     842*   1302    1332    1381    1463
PARREG  001531          410#   1131
PARTAB  001774          434     466#
PARVEC= 000114         1034*   1173#   1187*   1189*   1192*   1194*
PASSMG  002363          566#
PASSNM  002370          567#
PAT     007705         1706#
PDWN    000502          205#    247     599
PEFLG   004546          377     593*   1032#   1044*   1049*   1056    1089    1093*
PENFLG  004547          375    1033#   1060*
```

```
PERSTR  002462           596#   1053    1092    1145
PFSTK   000564           221#    227
PFVEC = 000024            60#    222*    247*    599*    600*
PIRQ  = 177772            72#
PIRVEC= 000240            65#
PKM   = 000000            48#
PLACE   002374           569#    886
PONE    000162           163#   1748
PRG3M   007747          1713#
PRG4M   010030          1722#
PRTY4 = 000200            44#
PRTY7 = 000340            43#    150
PSM   = 010000            49#
PSW   = 177776            70#    282
PTWO    000200           166     168#
PUM   = 030000            50#
PUP     000576           222     226#
PWRFAI  000730           253     257#
QUEST   007723          1709#
RANTST  003716           870#
RECDAT  001514           385     407#
REG   = 004000            51#
RELFL   000110           160#
RELOC   004274           958#   1003    1740
RELOCF  000764           273#    281     416     417     462     463     598*    985     1110    1746*
RELOCM  010047          1007    1725#
RELOCP  004376           994#
REL24K  010124          1009    1672    1737#
RESLDR  007520           588    1686#
RESVEC= 000010            55#
ROTATE  007276          1578    1606    1622#
ROTO    004070           908#
ROT1    004130           919#
RW    = 000006           124#    876     1225    1226    1229    1274    1275
SAVPC2  000112           161#
SAVR0   004710          1022*   1023    1067#
SAVR1   004712          1068#
SAVR2   004714          1069#
SAVR3   004716          1070#
SAVR4   004720          1071#
SAVR5   004722          1072#
SCOPE = 104000           121#
SLR   = 177774            71#
SM    = 040000            46#
SPACE1  002361           565#   1113    1135
SR0   = 177572            95#    209     240*    426     607*    1233*   1241*   1242*   1250*   1253*   1279*   1288*
SR1   = 177574            96#
SR2   = 177576            97#    210     239*   1240    1249    1287
SR3   = 172516            98#    211     238*    609*    610    1224*
START   002376           169     584#
START1  002440           584     592#    595     953
STBANK  007621          1697#
STKPTR= 000500            83#    585     592     628     661     800     816     829     846     854     904     932     1050
                         1743
STMM2   005714          1258#   1298    1374    1454    1490    1573    1601    1649
SWR   = 177570            79#    328     353     358     362     390     394     604    1094    1101    1175    1450    1492
```

```
                        1670    1673    1680
T       = 000020         42#
TBITVE= 000014           56#
TKB     = 177562         76#
TKS     = 177560         75#     205     246*
TKVEC   = 000060         63#
TPB     = 177566         78#
TPS     = 177564         77#     944
TPVEC   = 000064         64#
TRAPVE= 000034           62#     148
TRAP4   001460          338      401#
TRTVEC= 000014           57#
TYPIT   001110          310      318     321#
UBREAK= 177770           73#
UM      = 140000         47#
UP      = 000000        125#     876    1225    1226    1229    1274    1275
V       = 000002         39#    1254
WDS.25  003554          828#     910     914     921     925
WRTPAT  007344          911      922    1647#
WRTUP   002566          618#
XMTDAT  001501          380      405#
Z       = 000004         40#
$CRLF   000746          158      260#
$ENDAD  004260          152      793     949#
$FILL   001021          294#
$FORMD  001554          336      366     372     416#    629
$LDR    002112          505#     586
$LDRM   002264          540      546#
$LDR1   002262          516*     530     544#
$NULL   001020          293#     313
$PRINT  000770          157      252     279#    337     339     367     369     373     379     384     392     487     539
                        587      630     938     942    1006    1029    1061    1104    1107    1112    1126    1130    1134
                       1141    1143
$RESTR  000300          189#     245     389     461     498
$RLDR   002210          164      527#    948
$SAVR   000254          177#     206     364     418     479
$TKFLG  001023          297#
$TPB    001026          299#     323*
$TPFLG  001022          295#
$TPS    001024          298#     321
$SSTAG= 016070         1751#
.       = 010200         127#    142#    148#    151#    154#    156#    162#    167#    170#    200#    264#    322     329
                         359     395     414#    438     511     550#    568#    736     784     945     981    1081#   1170#
                        1377    1388    1395    1402    1409    1416    1423    1430    1437    1459    1468    1475    1610
                        1681    1751
.CONST  007362          908*    919*    1651#
.MAMF   005362          243     613      802    1052    1175#
.PARSR  004514         1022#   1187
.ROT0   007114          915    1570#
.ROT1   007210          926    1598#
.TYPE   001030          283     300#
.XOR39  006216         1335    1344#    1384    1465
..3X9   006304          833    1368#
..8X13  007106          862    1560#
.22PAR  004772         1089#   1192
.3ISO   006234         1345    1349#
```

E04


```
.3IS1   006224            1346#
.3IS9   006252            1348    1350    1354#
.3NOT9  006242            1347    1351#
.3X9    006064             821    1297#
.8X13   007100             851    1550#
```

F04


```
$TYPE       48

. ABS.  010200     000


 ERRORS DETECTED:  0

 DSKZ:DEMJAC.BIN,DSKZ:DEMJAC.LST/CRF/SOL/NL:TOC=DSKZ:DEMJAC.P11
 RUN-TIME:  4  3  .4 SECONDS
 RUN-TIME RATIO: 154/8=18.2
 CORE USED:  25K  (49 PAGES)
```