MARK 6/12 CPU SELF-TEST/ MANIP MANUAL

PROPRIETARY
Revision A

REVISION RECORD

PUBLICATION NUMBER: HM-0812-0064

Revision	Description	Date
A	Preliminary release	08/31/87

LIST OF EFFECTIVE PAGES

Changes, additions, and deletions to information in this manual are indicated by vertical bars in the margins or by a dot near the page number if the entire page is affected. A vertical bar by the page number indicates pagination rather than content has changed. The effective revision for each page is shown below.

Page	Rev	Page	Rev	Page	Rev
Cover	-				
Title thru vii	A				
1-1 thru 1-21	A				
2-1 thru 2-34	A				
A-1, A-2	A				
Comment Sheet	A				
Mailer	-				
Back Cover	-				

PREFACE

The POINT 4 MARK 6/12 Self-Test/MANIP Manual is designed for maintenance and service technicians of MARK 6 and MARK 12 systems. Its purpose is to provide information and instructions about the CPU Self-Test and the MANIP program.

The manual contains two sections: CPU Self-Test and MANIP. The section on CPU Self-Test provides instructions for accessing the CPU Self-Test, a detailed description of the CPU Self-Test and its operation, information on interpreting HALTS, and a listing of the Self-Test program. The section on MANIP provides instructions for accessing MANIP; a description of the MANIP commands, parameters, and functions; and a listing of the MANIP program.

Related Manuals

Related manuals include:

Title	Pub. Number
MARK 6 System Installation & Maintenance Manual	HM-086-0059
MARK 12 System Installation & Maintenance Manual	HM-0812-0061

CONTENTS

Section 1
CPU SELF-TEST

The MARK 6/12 central processing unit (CPU) Self-Test, has a comprehensive built-in diagnostic program that is contained in an EPROM (Erasable Programmable Read-Only Memory). This diagnostic program is a self-test that tests all CPU logic and performs a comprehensive memory test of all main (onboard) memory. It does not test Data Channel logic; and it tests the Interrupt and I/O logic only insofar as possible without having a separate controller exercise them.

Once the central processing unit (CPU) logic and memory tests have been completed, the CPU Self-Test relocates itself, then repeats. This process continues until an error is found, the STOP switch on the front edge of the CPU is pressed, or a key on the master terminal is pressed.

This section contains the following information and instructions on the CPU Self-Test:

- Accessing the CPU Self-Test
- CPU Self-Test Operation
- Interpreting HALTS
- Detailed Description of the CPU Self-Test
- Self-Test Listing

1.1 ACCESSING THE CPU SELF-TEST

The CPU Self-Test is accessed in three ways: it is automatically initiated when power is turned ON; or it can be accessed through the MANIP program or by the setting of the mini-switches on the front edge of the central processing unit (CPU).

1.1.1 At Power On

When the power is turned ON, the CPU Self-Test is automatically initiated. The power-on sequence includes the following: a firmware self-test routine runs once, MANIP is loaded, and the CPU software self-test repeats four times. If this sequence completes successfully, control is returned to MANIP, and a message is displayed on the master terminal; otherwise a HALT occurs (see Sections 1.2 and 1.3).

1.1.2 Through MANIP

This procedure assumes that all the mini-switches on the front edge of the central processing unit (CPU) are in the standard UP position.

If the operating system is running, shut it down and turn off the disk drive.

To load MANIP, press STOP and then APL on the card chassis mini-panel. To run the CPU Self-Test, enter T on the master terminal. When the CPU halts, press <CONT>. The Self-Test begins to run and runs continuously until the operator presses <ESC>.

1.1.3 By Setting the CPU Mini-Switches

This procedure begins with the power OFF. After removing the front bezel on the card chassis, set the mini-switches on the front edge of the central processing unit (CPU) to octal 100200.

To set the switches to octal 100200 do the following: set the leftmost switch on each of the two banks of mini-switches to the DOWN position and all remaining switches on each bank to the UP position.

Turn the power ON; the system will HALT. PRESS CONT on the card chassis mini-panel. The CPU Self-Test begins to run and runs continuously until any key of the master terminal keyboard is pressed. Be sure to return the mini-switches to their previous positions when testing is completed.

1.2 CPU SELF-TEST OPERATION

Once it has been accessed, the CPU Self-Test should operate as described below. (If it does not operate according to the following description, contact Hardware Technical Support at POINT 4 Data Corporation.)

On the front edge of the central processing unit (CPU), the RUN indicator lights, and the CARRY indicator flashes on and off in an irregular pattern repeating about once every second.

If the master terminal is connected to the system, self-test messages are displayed on the screen in accordance with the stage of operation being completed. (If the master terminal is not connected, the program still operates but no messages are displayed.)

1. After preliminary tests are run, the following is displayed:

   ```
   POINT 4 MARK 12 [or 6] SELF-TEST
   ```

2. After the first completion of the CPU test, the following is displayed:

   ```
   CPU LOGIC OK,
   ```

3. After the first completion of the memory test, the following is displayed:

   ```
   ON-BOARD MEMORY OK.
   ```

4. Thereafter, each time a complete test sequence is completed (approximately once per second), the following is displayed:

   ```
   V
   ```

5. After 64 Vs are displayed, a new line is started. The first item on each new line is the line number (in octal). The line number can be used to estimate how long Self-Test has run.

After running Self-Test 65,536 times, the messages CPU LOGIC OK, and ON-BOARD MEMORY OK. are repeated.

1.3 INTERPRETING HALTS

A HALT is a condition that brings the entire system to a standstill; it indicates an error. If a HALT occurs, two lines of information are displayed on the master terminal screen. The first contains the address of the HALT instruction +1 (the program counter), the four accumulators, and the CPU status word; the second contains the value coded in the mini-switches, and the contents of eight words in main (onboard) memory beginning at that address. For a description of the CPU status word, see Appendix A.

Enter MANIP by pressing APL on the chassis mini-panel, or by pressing <ESC> on the keyboard.

Dump the first few locations of memory. Normally, locations 0, 1, and 2 indicate the following:

- Location 0 indicates the starting location to which Self-Test has relocated itself, i.e., (0) real = 20000 virtual.
- Location 1 is the interrupt vector for the Illegal Interrupt test.
- Location 2 indicates the (real) address of the last test started.

There is one exception to this pattern. If an interrupt has occurred, word 0 contains the value of the program counter at the time of the interrupt.

To have MANIP interpret all addresses as they are in the listing, regardless of their actual location, use the "F" offset (virtual) capability of MANIP. To use "F", enter F, then the content of location 0, then a comma, then 20000. To display the HALT location in virtual form, enter A after the offset has been entered. For more information on MANIP, see Section 2.

By careful analysis of the program listing preceding the HALT, and the contents in the accumulators and temporary storage locations, it is usually possible to discover the reason for the error.

1.4 DETAILED DESCRIPTION OF THE CPU SELF-TEST

The various tests that comprise Self-Test are described below. The tests are numbered; these numbers correspond to the circled numbers on the CPU Self-Test listing provided in Section 1.5.

1. Because a HALT is used to indicate any subsequent error, Self-Test tests the HALT instruction first. This means that when Self-Test is initiated, the central processing unit (CPU) should HALT and the RUN indicator should go out. Press CONT on the front edge of the CPU to resume operation.

2. Self-Test then performs a few preliminary tests to detect certain specific failures. These tests include the compare instructions that will be used in subsequent tests, and the instructions used in the message subroutine.

3. After masking out TTO (master terminal output, device code 11) interrupts, Self-Test sets up the interrupt service vector at location 1 and enables interrupts. If an interrupt occurs subsequently, the CPU HALTs with the device code of the interrupting device in A0.

4. Self-Test then displays the message POINT 4 MARK 12 (or 6) SELF-TEST. This message is displayed only on the first pass. After Self-Test relocates and restarts, this message is suppressed. If no master terminal controller (device code 10/11) is included and operational, the program simply continues with no ill effects.

5. The ALU and Data Bus test increments a counter, using an ISZ instruction, 64K times starting from 0. To test that it takes 64K counts before the counter overflows (resulting in a skip), it uses the four accumulators in four nested loops doing sixteen 1-bit shifts each. This test uses all possible 16-bit numbers as the "destination" input to the arithmetic-logic unit (ALU), and checks that carry propagation can occur from the least significant position up to any other position. It also tests the left and right shift capability for each bit.

6. The Arithmetic-Logic Unit (ALU) Source Operand test sums all numbers from 0 through 64K, and checks that the total is correct (to 16 bits). It uses all possible 16-bit numbers as the "source" input to the ALU.

7. An exhaustive test of all arithmetic-logic unit (ALU) instructions follows. It executes all arithmetic and logical instructions from 100000 = COM 0,0 through 177777 = ANDCS# 3,3,SBN and checks that the final result is correct. This test exercises all operations that the ALU is capable of, using a pseudo-random sequence of operands. It also uses all possible bit combinations in the instruction register (except MSB = 0).

8. The Page 0 and Base 3 Addressing Test writes into each word of page zero that word's own address, using the page zero addressing mode (except when Self-Test is currently in page zero). It then reads the value back using the Base 3 addressing mode and confirms that it is correct.

9. The Relative, Base 2 and Indirect Addressing Modes Test reads each word in the 256-word region addressable by relative addressing three different ways, and it checks that the same value is being read each way. It exercises all possible address displacements in memory-reference type instructions. Each of these displacements is used with the same value in A2 and the program-counter. Different values in A2 and the program counter are tested when Self-Test relocates itself and repeats this test.

10. Auto-Index tests the auto-increment capability using all possible values in location 20. It also tests the auto-decrement capability using all possible values in location 37. Finally, it tests that all other locations differing in only one bit from the range 20-37 do not auto-index.

11. Since no external I/O device is required to run Self-Test, the I/O tests apply only to CPU I/O instructions. The I/O Skip instructions are tested by the use of INTEN and INTDS. When interrupts are enabled, SKPBN CPU should skip and SKPBZ CPU should not, and vice versa when interrupts are disabled. DIA ac,CPU is a READS instruction; it is tested by reading into two different accumulators and checking that they have picked up the same value. DIB ac,CPU is an INTA instruction; it should pick up a zero value. DIC ac,CPU is an IORST instruction; it should not change the content of the accumulator specified.

12. In the Multi-Level Indirect Addressing Test, a three-level addressing chain is tested, checking that the correct value is obtained.

13. When all these tests have been completed successfully, the following message is displayed:

```
64K CPU OK
```

This message is suppressed after the first pass of Self-Test. It will occur again after 64K passes.

14. If Self-Test is currently located below the midpoint of available memory, the Memory Test tests all memory above itself; otherwise, it tests all memory below itself. (Exception: Locations 0 and 1 are always reserved for the current location of Self-Test and for the interrupt vector.) The Memory Test algorithm consists of the following four tests:

    a. Write a 1 into each bit of the first word, then change it to a 0, then change it back to a 1. Do the same for each successive word until all words contain 177777. Now test the first word, check that it contains 177777. Change it to 0, retest, then change it back to 177777. Repeat for each successive word. This algorithm ensures that between the time any word was set to 177777 and the time it is tested, all other words have been toggled back and forth between 0 and 177777.

    b. Repeat the algorithm of test 1 with the 1's and 0's interchanged.

    c. Repeat the algorithm using for each word its own address and its complement as the test value.

    d. Repeat the algorithm using a 73077 (HALT) as the test value. This ensures that if Self-Test ever jumps out of itself, it will HALT, saving the accumulators for failure analysis.

15. When all the above memory tests have completed successfully, the following message is displayed:

    ```
    ON-BOARD MEMORY OK
    ```

    This message is suppressed after the first pass.

16. Self-Test now copies itself to a location slightly more than 20000 words below its current location, wrapping around to the top of memory if necessary. Since its initial location is 20000, the first move will bring it into upper RAM. (Straddling location zero is not allowed.)

1.5 SELF-TEST LISTING

The numbered sections of the CPU Self-Test correlate with the numbers adjacent to the individual tests that appear in Section 1.4.

```
; SELF-TEST PROGRAM FOR POINT 4 MARK 12 CPU [OR MARK 6]
; WRITTEN BY RENNY BOSCH
; 19 SEP 85

;                  All Rights Reserved
;       Copyright (C) 1979, Educational Data Systems
;       Copyright (C) 1985, Point 4 Data Corporation

              1 .TXTM 1
          20000 .LOC  L.SELF
  20000   20000       A.SELF              ;(FOR COMPATIBILITY WITH MARK 5)        (1)

  20001   63077       HALT                ;TEST HALT (MUST PRESS CONT, OR ESC, J)
-------------------------------------------------------------------------------------
; A FEW BASIC ALU TESTS

  20002 102001        ADC    0,0,SKP      ;A0 = 177777
  20003  63077        HALT                ;UNCONDITIONAL "SKP" FAILED TO SKIP
  20004 126424        SUBZ   1,1,SZR      ;A1=0
  20005  63077        HALT
  20006 152000        ADC    2,2
  20007 151404        INC    2,2,SZR      ;A2=0
  20010  63077        HALT
  20011 176000        ADC    3,3
  20012 162415        SNE    3,0          ;A3,A0 SHOULD = 177777
  20013 132414        SEQ    1,2          ;A1,A2 SHOULD = 0
  20014  63077        HALT

; A FEW BASIC JMP, LDA, STA, ISZ TESTS USING RELATIVE ADDRESSING

  20015  20405        LDA    0,.+5
  20016 116414        SEQ    0,3
  20017  63077        HALT                ;A0 & A3 SHOULD = 177777
  20020  30444        LDA    2,C100K
  20021 102621        SUBZR  0,0,SKP
  20022 177777        177777
  20023 112414        SEQ    0,2
  20024  63077        HALT                ;A0 & A2 SHOULD = 100000
  20025  40403        STA    0,COM00
  20026    402        JMP    COM00        ;TEST JMP REL.                          (2)
  20027  63077        HALT                ;SHOULD JUMP OVER THIS
  20030  63077 COM00: HALT                ;PGM CHANGES TO 100000=COM 0,0
  20031 112415        SNE    0,2
  20032  63077        HALT                ;A0 SHOULD = 77777, A2 = 100000
  20033 100000        COM    0,0
  20034  24774        LDA    1,COM00
  20035 112415        SNE    0,2
  20036 132414        SEQ    1,2
  20037  63077        HALT                ;A0, A1, A2 SHOULD = 100000
  20040   4404        JSR    JMP4         ;TEST INSTRS. USED IN "TYPE" S\R
  20041  63277 REF1:  HALT!INTDS
  20042    416 REF2:  JMP    INTSU
  20043  63077        HALT
  20044  54002 JMP4:  STA    3,2          ;LOC. 2 --> LAST TEST BEGUN
  20045  25400        LDA    1,0,3
  20046  20773        LDA    0,REF1
  20047 106414        SEQ    0,1
  20050  63077        HALT                ;A0 & A1 SHOULD = 63277 = (REF1)
  20051 175420        INCZ   3,3
  20052  25400        LDA    1,0,3
  20053  20767        LDA    0,REF2
  20054 106414        SEQ    0,1
  20055  63077        HALT                ;A0 & A1 SHOULD = 416 = (REF2)
  20056   1400        JMP    0,3          ;SHOULD GET TO "INTSU" VIA "REF2"
  20057  63077        HALT
```

```
   20060   4410 INTSU: JSR    SETUP     ;PICK UP POINTER TO INTERRUPT SERVICE
                      ;
 ; INTERRUPT SERVICE ROUTINE
                      ;
   20061  61477 INTSV: INTA   0         ;GET INTERRUPTING DEVICE CODE
   20062  63077        HALT             ;UNEXPECTED INTRPT (A0 = DEV. CODE)
   20063    775        JMP    INTSU     ;IN CASE "CONT" IS PRESSED

   20064 100000 C100K: 100000                                                 (3)
   20065 177777 INIFL:        -1        ;INITIAL FLAG FOR TYPE-OUT
   20066      1 CNTR:         1
   20067      0 FL.4EVER:     0         ;IF NONZERO, RETURN TO MANIP AFTER N PASSES

   20070  54001 SETUP: STA    3,1       ;SET UP INTERRUPT SERVICE VECTOR
   20071  20570        LDA    0,IMASK
   20072  62077        MSKO   0         ;MASK OUT TTO INTERRUPTS
   20073  60177        INTEN
   20074  20771        LDA    0,INIFL
   20075 101015        SNZ    0,0       ;INITIAL ENTRY TO SELF-TEST ?
   20076    426        JMP    TISZ      ;  NO, SKIP TYPE-OUT
   20077 176420        SUBZ   3,3       ;A3 = 0
   20100  54765        STA    3,INIFL   ;INIFL = 0
   20101  60211        NIOC   TTO
   20102  63511        SKPBZ  TTO       ;MAKE SURE TTY WON'T HANG UP
   20103  63077        HALT
   20104   4541        JSR    STYPE     ;"POINT 4 MARK 12 SELF-TEST"
   20105   6412 .TXT "<15><12>
   20106  50117 PO
   20107  44516 IN
   20110  52040 T
   20111  32040 4                                                             (4)
   20112  46501 MA
   20113  51113 RK
   20114  20061  1       0 ——— MARK 6 SUBSTITUTES 0 AND 6 FOR 1 AND 2
   20115  31040 2       6 ———
   20116  51505 SE
   20117  46106 LF
   20120  26524 -T
   20121  42523 ES
   20122  52015 T<15>
   20123   5000 <12>"
   20124 102400 TISZ:  SUB     0,0
   20125  40741        STA     0,CNTR
   20126  10740        ISZ     CNTR      ;TEST ISZ AND DSZ INSTR'S
   20127  14737        DSZ     CNTR
   20130  63077        HALT
   20131  14735        DSZ     CNTR
   20132  10734        ISZ     CNTR
   20133  63077        HALT

; TEST ALU: INCREMENT A COUNTER (VIA ISZ INSTRUCTION) 64K TIMES,
; CHECK THAT NO COUNTS ARE SKIPPED BY USING 4 NESTED LOOPS OF 16 BIT SHIFTS

   20134   4535 TALU:  JSR     PIKUP     ;(SKIPS NEXT WORD)
                       ;
   20135  42263 X:     42263   ;CHECKSUM FOR "EXHAUSTIVE ALU TEST"
                       ;
   20136 176120        ADCZL   3,3       ;A3 = 177776
   20137 152620 LP3:   SUBZR   2,2       ;A2 = 100000
   20140 126220 LP2:   ADCZR   1,1       ;A1 = 77777
   20141 101015 LP1:   SNZ     0,0       ;VERY FIRST PASS ?
   20142 102521        SUBZL   0,0,SKP   ;   YES, SET A0 = 1 & SKIP
   20143  10723 LP0:   ISZ     CNTR      ;   NO, COUNT UP CNTR
   20144 101015        SNZ     0,0                                              5
   20145  63077        HALT
   20146 101123        MOVZL   0,0,SNC   ;A0 SHIFTED 16 BITS ?
   20147    774        JMP     LP0       ;   NO, LOOP
   20150 101015        SNZ     0,0
   20151 124015        COM#    1,1,SNR
   20152  63077        HALT
   20153 102520        SUBZL   0,0
   20154 125242        MOVOR   1,1,SZC   ;HAS A1 SHIFTED 16 TIMES ?
   20155    764        JMP     LP1       ;   NO, REPEAT
   20156 124015        COM#    1,1,SNR
   20157 151015        SNZ     2,2
   20160  63077        HALT
   20161 151223        MOVZR   2,2,SNC   ;A2 SHIFTED 16 TIMES ?
   20162    756        JMP     LP2       ;   NO
   20163 151015        SNZ     2,2
   20164 174015        COM#    3,3,SNR
   20165  63077        HALT
   20166 175142        MOVOL   3,3,SZC   ;A3 SHIFTED 16 TIMES ?
   20167    750        JMP     LP3       ;   NO
   20170  20676        LDA     0,CNTR    ;CNTR HAS NOW BEEN INC'D 16*16*16*16-1
   20171 116415        SNE     0,3       ;   = 177777(8) TIMES
   20172  10674        ISZ     CNTR      ;ONE MORE ISZ SHOULD PRODUCE SKIP
   20173  63077        HALT
```

```
   20174 102400        SUB     0,0       ;TEST EVERY POSS. NO. AS SOURCE OPND.
   20175 126400        SUB     1,1
   20176 107000 ADDLP: ADD     0,1       ;ADD ALL NUMBERS 0 THROUGH 64K-1         6
   20177 101404        INC     0,0,SZR
   20200    776        JMP     ADDLP
   20201 102620        SUBZR   0,0       ;SUM SHOULD = 100000 MOD 64K
   20202 106414        SEQ     0,1
   20203  63077        HALT
```

```
; EXHAUSTIVE TEST OF ALL ALU INSTRUCTIONS

  20204 176220         ADCZR  3,3       ; A3 = 77777 (ARBITRARY INITIAL COND. )
  20205 171300         MOVS   3,2       ; A2 = 177577
  20206 145520         INCZL  2,1       ; A1 = 177400
  20207 102620         SUBZR  0,0       ; A0 = 100000
  20210  40402         STA    0,ALUI
  20211    401         JMP    ALUI      ; TO RELOAD INSTRUCTION PREFETCH QUEUE
                                        ; TO PATCH OUT EXH. ALU TEST, CHANGE 401 TO 412 ***

  20212  63077 ALUI:   HALT             ; CYCLES THROUGH ALL ALU INSTR.
  20213 147100         ADDL   2,1       ; \
  20214 123100         ADDL   1,0       ;  } FOLD RESULT INTO A3                  7
  20215 117100         ADDL   0,3       ; /
  20216  10774         ISZ    ALUI      ; MODIFY INSTRUCTION; ALL DONE ?
  20217    773         JMP    ALUI      ;   NO, CONTINUE
  20220  20715         LDA    0,X       ;   YES
  20221 162414         SEQ    3,0       ; IS FINAL RESULT CORRECT ?
  20222  63077         HALT             ;   NO, ALU ERROR

; BASE 3 ADDRESSING VS. PAGE ZERO

  20223   4446         JSR    PIKUP
  20224    674         R-L. SELF+400
  20225 172032         SGE    3,2       ; IS SELF ABOVE PAGE ZERO ?
  20226    446         JMP    TJSR      ;   NO, SKIP PZ TEST
  20227  20403         LDA    0,LDA02
  20230  40405         STA    0,LDA0
  20231 176521         SUBZL  3,3,SKP   ; SET UP FOR PAGE ZERO TEST
  20232  20002 LDA02:  LDA    0,2                                                8
  20233 175400 B3LP:   INC    3,3       ;   INCREMENT BY 1 WORD
  20234  55400         STA    3,0,3     ; INTO EACH WORD WRITE ITS OWN ADDRESS
  20235  20002 LDA0:   LDA    0,2       ; *******GETS MODIFIED BY PROGRAM******
  20236 116414         SEQ    0,3       ; DID WE GET BACK WHAT WE WROTE?
  20237  63077         HALT             ;   NO
  20240  10775         ISZ    LDA0      ; MODIFY THE LOAD INSTRUCTION
  20241  20423         LDA    0,S377
  20242 162032         SGE    3,0       ; IS A3 < 377
  20243    770         JMP    B3LP      ;   NO, REPEAT LOOP
  20244    430         JMP    TJSR      ; YES, GO ON TO NEXT TEST

  20245  25400 STYPE:  LDA    1,0,3     ; TYPE-OUT SUBROUTINE
  20246 175420         INCZ   3,3
  20247  20411 STYP2:  LDA    0,S377L   ; OUTPUT LEFT BYTE FIRST
  20250 123705         ANDS   1,0,SNR   ; ZERO BYTE (TERMINATOR) ?
  20251   1400         JMP    0,3       ;   YES, RETURN TO CALLER
  20252  61111         DOAS   0,TTO     ; OUTPUT BYTE TO TTY
  20253  63511         SKPBZ  TTO       ; WAIT FOR TTY NOT BUSY                  MISC.
  20254    777         JMP    .-1
  20255 125362         MOVCS  1,1,SZC   ; HAVE WE OUTPUT BOTH BYTES YET ?
  20256    771         JMP    STYP2     ;   NO, DO THE RIGHT BYTE NOW
  20257    766         JMP    STYPE     ;   YES, GET NEXT 2 BYTES

  20260 177400 S377L: 177400
  20261      1 IMASK: 1          ; MASK OUT TTO INTERRUPTS (SET BY MANIP
                                 ;   TO -1 ON POWER-UP TO MASK OUT ALL INTRPTS)
  20262  40101 CURMO: 40101      ; 40000 = 32K, 40001 = MK5, 40101 = MK12 MODE
  20263  40000 M32K:  40000      ; 32K MODE
  20264    377 S377:  377
  20265 177600 CM200: -200
  20266     20 C20:   20
  20267 177737 C40C:  -1-40
  20270  20116 ADR:   LDREL-200         ; USED IN RELATIVE ADDRESSING TEST

; SUB-ROUTINE TO PICK UP POINTER TO CENTRAL REFERENCE POINT

  20271  54002 PIKUP: STA    3,2       ; LOC. 2 --> LAST TEST STARTED
  20272  31400         LDA    2,0,3     ; LOAD PARAMETER WORD
  20273   5401         JSR    1,3       ; SKIP-RETURN WITH POINTER TO "R"
        20274 R=       .         ; REFERENCE POINT USED FOR ADDRESSING EXTENSION
```

```
; BASE 2, RELATIVE, AND INDIRECT ADDRESSING - ALL WITHIN +-200 OF HERE

    20274   4775 TJSR:  JSR    PIKUP
    20275     22        LDREL-R
    20276 173000        ADD    3,2       ;CALC. LOC. OF "LDREL"
    20277  20766        LDA    0,CM200
    20300 143040        ADDO   2,0
    20301  40767        STA    0,ADR     ;SET UP "ADR" = LDREL - 200
    20302  35200        LDA    3,-200,2
    20303  20777        LDA    0,.-1     ;PICK UP BASE 2 INSTR.
    20304  34600        LDA    3,.-200
    20305  34777        LDA    3,.-1     ;PICK UP REL. ADDR. INSTR.
    20306  40404 SETAD: STA    0,LDAB2   ;SET UP BASE 2 INSTRUCTION
    20307  54407        STA    3,LDREL   ;SET UP REL. ADDR. INSTR.
    20310  24755        LDA    1,CM200

    20311   4401 B2LP:  JSR    .+1       ;FOR WORST CASE ADDRESS CALC. TIME
    20312  35200 LDAB2: LDA    3,-200,2 ;*** GETS MODIFIED BY PROGRAM ***
    20313  22755        LDA    0,@ADR
    20314 116414        SEQ    0,3       ;A0 = INDIR., A3 = BASE 2 ADDRESSING
    20315  63077        HALT             ;THEY DON'T MATCH !?
    20316  34600 LDREL: LDA    3,.-200   ;*** GETS MODIFIED BY PROGRAM ***
    20317 116414        SEQ    0,3
    20320  63077        HALT
    20321  10747        ISZ    ADR       ;INCREMENT INDIRECT ADDRESS
    20322  10770        ISZ    LDAB2     ;AND BASE 2 LOAD INSTRUCTION,
    20323  10773        ISZ    LDREL     ;AND RELATIVE LOAD INSTRUCTION
    20324 125404        INC    1,1,SZR   ;HAVE WE TESTED 200 LOCATIONS ?
    20325    764        JMP    B2LP      ;  NOT YET, REPEAT LOOP
    20326  35000        LDA    3,0,2     ;PREPARE FOR 2ND 200 LOCATIONS
    20327  20777        LDA    0,.-1     ;PICK UP BASE 2 INSTR.
    20330  34400        LDA    3,.       ;PICK UP REL. ADDR. INSTR.
    20331 101002        MOV    0,0,SZC   ;HAVE WE DONE 2ND PASS ALREADY ?
    20332    754        JMP    SETAD     ;  NO, DO IT NOW

    20333   4736        JSR    PIKUP     ;PICK UP REFERENCE POINTER
    20334  77262        R-E.SELF+100000
    20335  20726        LDA    0,M32K
    20336 172033        SLS    3,2       ;ARE WE EXECUTING IN UPPER 32K ?
    20337 101400        INC    0,0       ;  YES, THEN CAN'T USE 32K MODE
    20340  40722        STA    0,CURMO
    20341  60377        NIOP   CPU       ;SET CURRENT CPU MODE
```

(9)

```
; TEST AUTO INDEX CAPABILITY (IN 32K AND MK5 MODE ONLY)

   20342   4727 AUTOX: JSR    PIKUP
   20343    334        R-L. SELF+40
   20344 172432        SGR    3,2       ;DOES SELF OVERLAP LOC. 0-40 ?
   20345    454        JMP    IOTST     ;  YES, SKIP AUTO INDEX TESTS
   20346 152000        ADC    2,2       ;NO
   20347  50020        STA    2,20      ;-1 IN AUTO INCREMENT CELL
   20350  34712        LDA    3,CURMO
   20351 175200        MOVR   3,3       ;32K: C = 0; 64K: C = 1
   20352 176600        SUBR   3,3       ;32K: 100000; 64K: 0
   20353  54037        STA    3,37      ;0 (OR 100000) IN AUTO DECR. CELL
   20354 151400 AXLP:  INC    2,2
   20355  22020        LDA    0,@20     ;CAUSES (20) TO INCREMENT
   20356  24020        LDA    1,20
   20357 132414        SEQ    1,2       ;DID THE VALUE IN 20 INCREMENT ?
   20360  63077        HALT             ;  NO - ERROR
   20361  25000        LDA    1,0,2
   20362 106414        SEQ    0,1       ;DID WE LOAD @20 CORRECTLY ?
   20363  63077        HALT             ;  NO ?!
   20364 174400        NEG    3,3       ;DECREMENT A3
   20365 174000        COM    3,3
   20366  22037        LDA    0,@37     ;CAUSES (37) TO DECREMENT
   20367  20037        LDA    0,37
   20370 116414        SEQ    0,3       ;DID CONTENT OF 37 DECREMENT ?
   20371  63077        HALT             ;  NO, ERROR
   20372 175014        SKZ    3,3
   20373    761        JMP    AXLP                                          (10)

; NOW TEST THAT <20 AND >37 DO NOT AUTO INDEX

   20374  20672        LDA    0,C20
   20375  40005        STA    0,5
   20376  26005        LDA    1,@5      ;LOC. 5 SHOULD NOT AUTO-INDEX
   20377  24005        LDA    1,5
   20400 106414        SEQ    0,1       ;DID WE GET BACK WHAT WE WROTE ?
   20401  63077        HALT             ;  NO !?
   20402 111000        MOV    0,2       ;NOW A0 = A2 = 20
   20403 113000 NAXLP: ADD    0,2       ;TEST 40, THEN 60, 120, 220, 420, ETC.
   20404  25000        LDA    1,0,2
   20405 125113        SSN    1,1       ;PREVENT POSS. INFINITE INDIRECT CHAIN
   20406  37000        LDA    3,@0,2    ;SHOULD NOT AUTO-INDEX
   20407  35000        LDA    3,0,2
   20410 136414        SEQ    1,3       ;DID 2 LDA INSTR'S GIVE SAME RESULT ?
   20411  63077        HALT             ;  NO, ERROR
   20412 112400        SUB    0,2
   20413 101122        MOVZL  0,0,SZC   ;PREPARE FOR NEXT LOCATION: ALL DONE ?
   20414    405        JMP    IOTST     ;  YES
   20415  24645        LDA    1,CURMO   ;NO
   20416 125213        SKO    1,1       ;32K MODE
   20417 101113        SSN    0,0       ;  AND ABOUT TO TEST 100020 ?
   20420    763        JMP    NAXLP     ;    NO, STAY IN LOOP (ELSE DONE)
```

```
; I\O TESTS

   20421    4650 IOTST: JSR     PIKUP      ;(SKIPS NEXT WORD)
                        ;
   20422     623 JTYPE: JMP     STYPE      ;ELEVATOR TO "STYPE"
                        ;
   20423   60277        INTDS
   20424   63477        SKPBN   CPU
   20425   63577        SKPBZ   CPU
   20426   63077        HALT               ;ION SHOULD BE OFF
   20427   60177        INTEN
   20430   63577        SKPBZ   CPU
   20431   63477        SKPBN   CPU
   20432   63077        HALT               ;ION SHOULD BE ON
   20433   20627        LDA     0,CURMO
   20434  101212        SKE     0,0        ;ARE WE CURRENTLY IN 32K MODE ?
   20435     416        JMP     IO64K      ;  NO, CAN'T TEST IORST
   20436   62677        IORST              ;YES
   20437   63477        SKPBN   CPU
   20440   63577        SKPBZ   CPU
   20441   63077        HALT               ;ION SHOULD BE OFF AGAIN

   20442  102521        SUBZL   0,0,SKP
                        ;
   20443     626 JPIK:  JMP     PIKUP      ;ELEVATOR TO "PIKUP"
                        ;
   20444  126220        ADCZR   1,1

   20445   62477 DIC07: DIC     0,77       ; = IORST
   20446   66477 DIC17: DIC     1,77       ;(SHOULD NOT CHANGE ACCUMULATOR)
   20447  106415        SNE     0,1
   20450   63077        HALT               ;A0 S/B 1, A1 S/B 77777
   20451   20610        LDA     0,IMASK
   20452   62077        MSKO    0          ;MASK OUT TTO INTERRUPTS

   20453   61477 IO64K: DIB     0,CPU      ; = INTA
   20454  101014        SKZ     0,0
   20455   63077        HALT               ;NO DEVICE SHD. ACK. INTERRUPT
   20456   60677        DIAC    0,CPU      ; = READS
   20457   64577        DIAS    1,CPU      ;ALSO ENABLES ION
   20460  106414        SEQ     0,1
   20461   63077        HALT               ;A0 AND A1 SHD = MINI-SWITCHES
; NOW CHECK THAT DEVICE CODES OTHER THAN 77 DO A DIC NORMALLY

   20462   30605        LDA     2,C40C

   20463   20762 IOLP:  LDA     0,DIC07
   20464   24762        LDA     1,DIC17
   20465  143400        AND     2,0
   20466  147400        AND     2,1
   20467   40403        STA     0,IOINS
   20470   44403        STA     1,IOINS+1
   20471     401        JMP     IOINS      ;TO AVOID SELF-MOD. PROBLEM
   20472   63077 IOINS: HALT               ;PGM CHANGES TO DIC 0,37/57/67/73/75/76
   20473   63077        HALT               ;PGM CHANGES TO DIC 1,37/57/67/73/75/76
   20474  106414        SEQ     0,1
   20475   63077        HALT               ;DIC INSTR'S SHD PICK UP SAME VALUE
   20476  151242        MOVOR   2,2,SZC    ;HAVE WE DONE DEVICE 76 ?
   20477     764        JMP     IOLP       ;  NO, GO BACK
```

11

```
; MULTI-LEVEL @ IF IN 32K MODE -- OR 16-BIT @ IF IN 64K MODE

   20500   4743 MULTI: JSR    JPIK        ;PICK UP POINTER TO R
                       ;
   20501 120507 SA:    @SB
                       ;
   20502  54404        STA    3,SC        ;SET UP C TO POINT TO R
   20503  21400        LDA    0,0,3       ;SAVE THE VALUE AT R
   20504  31766        LDA    2,CURMO-R,3
   20505   4403        JSR    .+3
                       ;
   20506  20274 SC:    R
   20507 120506 SB:    @SC
                       ;
   20510 177245        ADDOR  3,3,SNR     ;COMLEMENT MSB OF A3 (GIVES C')
   20511    455        JMP    CPUDN       ;PROTECT WORD 1 (INTSV)
   20512  45401        STA    1,1,3       ;PUT A KNOWN VALUE IN B'
   20513 151212        SKE    2,2         ;ARE WE IN 64K MODE ?
   20514 121000        MOV    1,0         ;   YES, REMEMBER THE VALUE IN B'
   20515  54772        STA    3,SB        ;SET UP B TO POINT @C (IF 32K)
   20516 175400        INC    3,3
   20517  54762        STA    3,SA        ;A POINTS @B (32K) OR AT B' (64K)
   20520  26761        LDA    1,@SA       ;THIS IS THE LOAD @ INSTRUCTION
                                          ;32K. @A = A --> B --> C --> R
                                          ;64K: @A = A --> B' (= B W.OPP.MSB)
   20521 106414        SEQ    0,1
   20522  63077        HALT               ;MULTI @ FAILED (IF CURMO = 32K)

   20523   4720        JSR    JPIK        ;PICK UP POINTER TO R
                       ;
   20524    676 JJTYP: JMP    JTYPE       ;ELEVATOR TO "TYPE" (PGM SKIPS)
                       ;
   20525  21766        LDA    0,CURMO-R,3
   20526 101212        SKE    0,0         ;CURR. IN 32K MODE ?
   20527    427        JMP    CKM12       ;   NO, CHECK IF MK12 MODE          12
   20530 101400        INC    0,0         ;YES, THEN DO MK5 MODE NEXT
   20531  41766        STA    0,CURMO-R,3
   20532  60377        NIOP   CPU

; UPPER CORE ACCESS TEST (DONE IN MK5 MODE ONLY)

   20533   4403        JSR    .+3         ;MOVE SOME CODE TO UPPER 32K
                       ;
   20534  63077 HALT.: HALT               ;THESE TWO WORDS WILL BE MOVED
   20535   5401        JSR    1,3         ;TO UPPER 32K
                       ;
   20536  20776        LDA    0,.-2
   20537  24776        LDA    1,.-2
   20540 177240        ADDOR  3,3         ;SET MSB OF A3
   20541  45400        STA    1,0,3       ;MOVE THE 2 WORDS,
   20542  41401        STA    0,1,3       ;   REVERSING THEIR ORDER
   20543  24771        LDA    1,HALT.
   20544 122414        SEQ    1,0         ;DID WE CLOBBER LOWER CORE ?
   20545  63077        HALT               ;   YES - 64K MODE NOT WORKING
   20546 171400        INC    3,2         ;SAVE A3 VALUE
   20547   5400        JSR    0,3         ;JMP TO THE JSR 1,3 IN UPPER 32K
   20550  63077        HALT               ;IT SHD. RETURN TO THE NEXT LOCATION:
   20551 172414        SEQ    3,2         ;DID IT PICK UP THE RIGHT A3 ?
   20552  63077        HALT               ;   NO
   20553   4670        JSR    JPIK        ;PICK UP POINTER TO "R"
                       ;
   20554    667 JJPIK: JMP    JPIK        ;ELEVATOR TO "PIKUP" (SKIPS NEXT LOC.)
                       ;
   20555   1446        JMP    AUTOX-R,3;REDO TESTS IN MK5 MODE

   20556  24407 CKM12: LDA    1,C100
   20557 107414        AND#   0,1,SZR     ;HAVE WE DONE MK12 MODE YET ?
   20560    406        JMP    CPUDN       ;   YES, THEN CPU TEST IS DONE
   20561 123000        ADD    1,0         ;NO, SET MK12 MODE
   20562  41766        STA    0,CURMO-R,3
   20563  60377        NIOP   CPU
   20564    635        JMP    IOTST

   20565    100 C100:  100
--------------------------------------------------------------------------
; CPU LOGIC TEST IS NOW COMPLETED

   20566  10573 CPUDN: ISZ    INIFC       ;IS THIS FIRST PASS OF SELF-TEST ?
   20567    411        JMP    TMEM        ;   YES, SKIP TYPE-OUT
   20570   4734        JSR    JJTYP       ;NO, TYPE "CPU LOGIC OK,"         13
   20571  41520 .TXT "CP
   20572  52440 U
   20573  46117 LO
   20574  43511 GI
   20575  41440 C
   20576  47513 OK
   20577  26000 ,"
```

```
; ******************* MEMORY TEST *********************

; FIRST PASS: SET A BIT TO 1, SET IT TO 0, THEN SET
; IT BACK TO 1, THEN DO THE SAME TO NEXT BIT, ETC.
; SECOND PASS: TEST THAT THE BIT = 1, TOGGLE IT TO 0, RETEST,
; AND BACK TO 1, THEN DO SAME FOR NEXT BIT --
; THUS EACH BIT IS TESTED AFTER ALL OTHER BITS HAVE BEEN TOGGLED.
; THEN REPEAT THE WHOLE TEST WITH 0'S AND 1'S INTERCHANGED
; THIRD TEST: USE EACH WORD'S ADDRESS (COMPLEMENTED) IN PLACE OF 0'S OR 1'S
; FOURTH TEST : USE 73077 HALT (HAS ODD PARITY) IN PLACE OF ADDRESS

    20600   4754 TMEM:  JSR    JJPIK
    20601    516        E.SELF-R
    20602 173000        ADD    3,2       ;A2 = FIRST LOC. ABOVE SELF-TEST
    20603  34565        LDA    3,NWDS    ;A3 = -(LENGTH OF SELF-TEST)
    20604 151112        SSP    2,2       ;ARE WE CURRENTLY IN UPPER RAM ?
    20605 157001        ADD    2,3,SKP   ;  YES, TEST LOWER MEMORY
    20606 176001        ADC    3,3,SKP   ;  NO, TEST UPPER PORTION
    20607  30551        LDA    2,MIN.F   ;PROTECT LOC. 0-2, PLUS 4 WORDS
    20610  50561        STA    2,FIRST
    20611 102040        ADCO   0,0       ;SET A0 = 177777, C = 1
    20612  30557 MTEST: LDA    2,FIRST   ;FIRST PASS - SET UP MEMORY
    20613 101003 LOOP1: MOV    0,0,SNC   ;IS THIS THE THIRD TEST ?
    20614 140000        COM    2,0       ;  YES: USE COMPLEMENT OF ADDRESS
    20615  41000        STA    0,0,2
    20616 104000        COM    0,1
    20617  45000        STA    1,0,2     ;TOGGLE MEMORY WORD
    20620  41000        STA    0,0,2     ;TOGGLE BACK AGAIN
    20621 151400        INC    2,2
    20622 156032        SGE    2,3       ;ALL SET UP ?
    20623    770        JMP    LOOP1     ;  NOT YET
    20624  30545        LDA    2,FIRST   ;SECOND PASS - TEST MEMORY
    20625 101002 LOOP2: MOV    0,0,SZC   ;ARE WE ON THE THIRD TEST ?
    20626    406        JMP    LP2.1     ;  NO
    20627 140000        COM    2,0       ;YES, USE COMPLEMENTED ADDRESS
    20630  51374        STA    2,-4,2    ;EXACERBATE MEMORY TIMING
    20631  25000        LDA    1,0,2     ;COMPLEMENTED ADDRESS
    20632  31374        LDA    2,-4,2    ;TRUE ADDRESS
    20633 101000        MOV    0,0       ;DELAY TO INHIBIT ACHGD LOGIC
    20634  25000 LP2.1: LDA    1,0,2
    20635 106414        SEQ    0,1
    20636  63077        HALT             ;A0 = S/B, A1 = IS, A2 = ADR.
    20637 104000        COM    0,1
    20640  45000        STA    1,0,2     ;TOGGLE MEMORY
    20641  25004        LDA    1,+4,2    ;TO EXACERBATE MEMORY TIMING
    20642  25000        LDA    1,0,2     ;RETEST
    20643 124000        COM    1,1
    20644 106414        SEQ    0,1
    20645  73077 HALTI: 73077
    20646  41000        STA    0,0,2     ;TOGGLE MEMORY WORD BACK AGAIN
    20647 151400        INC    2,2
    20650 156032        SGE    2,3       ;TESTED ALL LOCATIONS ?
    20651    754        JMP    LOOP2     ;  NO

; PREPARE FOR NEXT MEMORY TEST

                        ;TEST 1:  A0 = 177777,  C = 1
                        ;TEST 2:  A0 =      0,  C = 1
                        ;TEST 3:  A0 =      1,  C = 0 (VALUE = ADDR.)
                        ;TEST 4:  A0 =  73077,  C = 1

    20652 101466        INCC   0,0,SEZ   ;NOW PREPARE FOR NEXT TEST
    20653  20772        LDA    0,HALTI   ;GET THE HALT INSTRUCTION
    20654  24771        LDA    1,HALTI   ;GET 73077 INSTRUCTION
    20655 122014        ADC#   1,0,SZR   ;HAVE WE DONE FOUR TESTS?
    20656    734        JMP    MTEST     ;NO, DO NEXT TEST
```

(14)

```
20657   10503 TFP:   ISZ     INIFM     ; IS THIS FIRST PASS OF SELF-TEST ?
20660     415        JMP     TP.VS     ;   NO, SKIP TYPE-OUT
20661    4643        JSR     JJTYP     ; TYPE " ON-BOARD MEMORY OK. "
20662   20117 .TXT " O
20663   47055 N-
20664   41117 BO
20665   40522 AR
20666   42040 D
20667   46505 ME
20670   46517 MO
20671   51131 RY
20672   20117  O
20673   45456 K.
20674       0 "

20675   14467 TP.VS: DSZ     CCNT      ; END OF A LINE ?
20676     426        JMP     TYPV      ;   NO, JUST TYPE A V
20677   20464        LDA     0, LCNT   ; YES
20700   40464        STA     0, CCNT
20701    4623        JSR     JJTYP
20702    6412 .TXT "<15><12>
20703       0 "

20704   24466        LDA     1, LNO
20705  125400        INC     1, 1      ; INCREMENT LINE NUMBER
20706   44464        STA     1, LNO
20707   20464        LDA     0, M1BIT  ; MASK FOR 1 BIT
20710  101120 TPNXO: MOVZL   0, 0      ; SET CARRY APPROPRIATELY
20711  125105        MOVL    1, 1, SNR ; INITIALLY INSERTS "PUSHER" BIT
20712     410        JMP     TPODN     ; EXIT WHEN "PUSHER" BIT IS GONE
20713  101103        MOVL    0, 0, SNC
20714     775        JMP     .-3
20715   61111        DOAS    0, TTO
20716   63511        SKPBZ   TTO
20717     777        JMP     .-1
20720   20454        LDA     0, M3BIT
20721     767        JMP     TPNXO

20722    4602 TPODN: JSR     JJTYP
20723   35000        ":*400
20724    4600 TYPV:  JSR     JJTYP
20725   53000        "V*400
```

(15)

```
; MOVE TEST PROGRAM THROUGH CORE AND REPEAT

   20726   4626 MOVE:  JSR     JJPIK
   20727 130406        -DIST*2         ;TENT. ASSUME DOUBLE MOVE REQUIRED
   20730  20435        LDA     0,R.MIN
   20731  24435        LDA     1,R.MAX
   20732 162433        SLE     3,0     ;IS SELF WHERE SINGLE MOVE WOULD
   20733 166033        SLS     3,1     ;    CAUSE STRADDLING WORDS 0-3 ?
   20734 151240        MOVOR   2,2     ;  NO, THEN DO SINGLE MOVE
   20735  24432        LDA     1,R.OFS
   20736 136400        SUB     1,3     ;A3 = CURRENT LOC. OF SELF
   20737 173000        ADD     3,2     ;A2 = NEW LOCATION OF SELF
   20740  21467        LDA     0,FL.4EVER-L.SELF,3
   20741 101014        SKZ     0,0     ;DOING SELF-TEST "N" TIMES
   20742  15467        DSZ     FL.4EVER-L.SELF,3;AND "N" EXPIRED ?
   20743    402        JMP     MOVE2   ;  NO; CHANGE 402 TO 1402 TO PREVENT RELOCATION ***
   20744    431        JMP     RT.MANIP ;  YES, RETURN TO MANIP

   20745  24423 MOVE2: LDA     1,NWDS
   20746  21400 MOVLP: LDA     0,0,3   ;NOW DO THE MOVE LOOP
   20747  41000        STA     0,0,2
   20750 175400        INC     3,3
   20751 151400        INC     2,2
   20752 125404        INC     1,1,SZR ;MOVE DONE ?
   20753    773        JMP     MOVLP   ;  NO                                                  (16)
   20754  24414        LDA     1,NWDS
   20755 133000        ADD     1,2
   20756  50000        STA     2,0     ;FOR EASILY FINDING SELF WHEN MOVED
   20757   1002        JMP     2,2

   20760      7 MIN.F: 7      ;MINIMUM VALUE OF "FIRST"
   20761 177777 INIFC: 177777 ;INITIAL FLAG FOR "CPU OK." MESSAGE
   20762 177777 INIFM: 177777 ;INITIAL FLAG FOR "MEMORY OK." MESSAGE
   20763    100 LCNT:  100    ;LINE COUNT = NO. OF V'S BEFORE A CR/LF
   20764      1 CCNT:  1      ;COLUMN COUNT OF V'S TYPED
   20765  23056 R.MIN: DIST-E.SELF+R-1
   20766  24075 R.MAX: DIST+R-L.SELF+4
   20767    274 R.OFS: R-L.SELF
   20770 176766 NWDS:  L.SELF-E.SELF ; - NO. OF WORDS IN SELF PROGRAM
   20771      0 FIRST: 0
   20772      0 LNO:   0      ;LINE NUMBER
   20773 140014 M1BIT: 140014 ;MASK FOR 1-BIT ACSII CHARACTER
   20774  10003 M3BIT: 10003  ;MASK FOR 3-BIT ASCII CHARACTER

          23575 DIST=  23575  ;DISTANCE SELF IS MOVED EACH TIME
```

```
; RETURN TO MANIP OR POWER-FAIL AUTO-RESTART

         20775 RT.MANIP:
  20775  60277          INTDS
  20776 102400          SUB     0,0
  20777  24404          LDA     1,M.SIZE
  21000  30404          LDA     2,M.LOAD
  21001  34404          LDA     3,.MAN1
  21002  60077          NIO     CPU

  21003   1000 M.SIZE:          E.MANIP-L.ASM
  21004 177000 M.LOAD:          A.MANIP
  21005 177024 .MAN1:           MAN.1+A.MANIP-L.MANIP

  21006      4 .BLK  4          ;MEMORY TEST USES

         21012 E.SELF:


               .EOT  ;MARK 12 SELF-TEST [OR MARK 6]
```

Section 2
MANIP

MANIP is a software program that enables an operator to manipulate the operations of the central processing unit (CPU) from the master terminal.

This section contains the following information and instructions on MANIP:

- Accessing MANIP
- MANIP Command Descriptions
- MANIP Listing

2.1 ACCESSING MANIP

When the power is turned ON, MANIP is loaded.

To access MANIP so that the operator can control designated central processing unit (CPU) functions, remove the card chassis bezel and make certain that all the mini-switches on the front edge of the CPU are in the standard UP position.

Press the STOP and APL buttons on the card chassis mini-panel. The CARRY indicator flashes while MANIP waits for a command to be entered on the master terminal. The MANIP prompt (->) appears on the screen. Enter ? and the MANIP menu is displayed as follows:

```
POINT 4 DATA CORP.      MARK 12 [OR MARK 6] VIRTUAL CONSOLE       CPU/FW Rev. nnnn
COPYRIGHT (C) 1986                d mmm, yy                       MANIP rev.   n.n
==================================================================================
_FORMAT__  ____________DESCRIPTION (all parameters octal)_______________
A           Display ADRS: AC0 AC1 AC2 AC3 CPU-STAT (msb = Carry)
Cx,y        CHANGE ACx or CRY/CPU-STAT to y; Cx (x>4): change real adrs to virtual
Dx*         DISPLAY memory contents in octal, starting at adrs x
Ex* or x:   EXAMINE/ENTER into location x. ^ opens prev. adrs. ESC to end
Fx,y        Establish OFFSET for virtual adrss, x=real, y=virtual adrs
Jx*         JUMP to adrs x and execute; default = continue from last HALT
Kx,y,z      Store CONSTANT z in locations x through y
Mx,y,z      MOVE memory block x through y to location z
Nx,y,z,m    SEARCH memory x through y for NOT-EQUAL z, with optional mask m
Ox*         OUTPUT memory in ASCII, starting at
Px          PROGRAM LOAD (IPL) from device code x; default=(mini-switches), or 27
Sx,,y,z,m   SEARCH memory x through y for the value z, with optional mask m
T           Run Mark 12[/6] SELF-TEST.  After HALT, press CONT, or type ESC>J<RETURN>
Ux,y,z,a    UNLOAD PROM:x=PROM adrs.,y=#wds,z=mem.adrs,a=opt.exec.adrs; dflt=DBUG
Xx,y        Calc, and print CHECKSUM over mem. block x through y
Yx          Set up CRT new-line DELAY; 0=max. delay, 177777=none
  * Opt.   adrss mode: 0=word,real; 1=byte,virt; 2=byte,lower 64KB; 3=byte,upper
<CTRL-X>   etc.  CTU (Cassette Tape Unit) access commands
?           Shows this Help menu
->
```

The MANIP commands and parameters are described in Section 2.2.

2.2 MANIP COMMAND DESCRIPTIONS

A MANIP command consists of a single letter, which is the command identifier, and parameters that specify addressing modes, memory addresses, and data input. All parameters must be entered in octal. The letters x, y, z, a, m, and n are used to represent octal parameters.

For some commands, MANIP allows either word or byte addressing using either real memory addresses or offset memory addresses (see the F command). These optional addressing modes are invoked by the parameter "a". They can be used with the commands D, E, J and O. The "a" parameter definitions are as follows:

a Parameter	Definition
Omitted	Word address, including "F" offset, if any
0	Word address, absolute
1	Byte address, using offset, if any
2	Byte address, lower 64KB
3	Byte address, upper 64KB

NOTE

The J command does not permit byte addresses.

If no "a" parameter is given, the addressing mode is "word address, including offset, if any". If there is no "a" parameter, the preceding comma is optional.

Table 2-1 shows the MANIP commands, parameters, and definitions. Table 2-2 lists the MANIP commands used to control a cassette tape unit (CTU).

TABLE 2-1. MANIP COMMAND DESCRIPTIONS (1 of 7)

Command & Parameters	Definition
A	Displays initial value of program counter (PC) saved in first location of MANIP, contents of accumulators A0, A1, A2, A3, and CPU status word as they were at the time MANIP was entered. The MSB of the CPU status word represents the carry flip-flop. For a description of the CPU status word, see Appendix A.
Cx,y	Changes accumulator or CPU status word, or address representation. ● If x is 0, 1, 2, or 3, then y is stored as saved value for accumulator x (A0, A1, A2, A3, respectively). ● If x is 4, then the CPU status word (MSB = carry) is set equal to y. ● If x is greater than 4 and an address offset has been established (see F command), x is interpreted as a real address and converted to a virtual address using the offset previously established, and displayed on the master terminal. In this case, the y parameter is not used. ● Parameter Description x - 1 octal digit 0-7 or one word octal y - 1 word octal
Dx,a	Dumps memory in octal, beginning at location x, using address mode a. Eight words (or bytes if using a byte address mode) are displayed per line, with the address of the first word (byte) at the beginning of each line. To temporarily pause output, press <CTRL-S> (XOFF); to start, press <CTRL-Q> (XON); to terminate output, press <ESC> or any other key. ● Parameter Description x - octal number representing a 16-bit memory address a - optional one digit (0-3) representing an address mode

TABLE 2-1. MANIP COMMAND DESCRIPTIONS (2 OF 7)

Command & Parameters	Definition
Ex,a	Enables entry at address x, using address mode a. The address (changed to a word address if it was a byte address) is printed, followed by a colon. An octal value can then be entered into the memory location, followed by <RETURN>. The next address (x+1) is then printed and opened for entry. Entry can be continued into sequential address locations until terminated by pressing <RETURN> and then <ESC>. ● If there is no entry before <RETURN>, the present content of the opened location is displayed in octal to allow examination of a value before entering a new one. If <RETURN> is pressed again without an entry, the current value is saved; and the next address is printed and opened for entry. ● If a caret (^) is entered instead of <RETURN>, the previous address is printed and opened. This feature is convenient for confirming an entry just made. ● Parameter Description x - octal number representing a 16-bit memory address a - optional one digit (0-3) representing an address mode
Fx,y	Establishes an address offset (a fixed difference between real memory addresses and virtual addresses as entered and listed in MANIP). The difference x-y is added to an address entered and subtracted from a memory address before listing. If y is not entered, it is assumed to be zero. Whenever a nonzero offset is established, an F is printed at the beginning of each line. To revert to real memory addressing, enter F0. ● Parameter Description x - octal number representing a real memory address y - octal number representing listing address equivalent to address specified in x

TABLE 2-1. MANIP COMMAND DESCRIPTIONS (3 of 7)

Command & Parameters	Definition
F	Saves current offset value, and reinstates previous offset. Displays offset being reinstated. Provides a convenient way to toggle back and forth between two offsets (or one offset and real memory addressing).
Jx,a	Jumps to location x (using address mode a) with accumulators and carry restored. ● Parameter Description x - octal number representing a 16-bit memory address a - optional one digit (0) representing absolute word address mode (Note: byte address modes are not available for J command.)
Kx,y,z	Stores the octal constant z in locations x through y, inclusive. ● Parameter Description x - octal number representing 16-bit beginning memory address y - octal number representing 16-bit ending memory address z - octal number representing constant
Mx,y,z	Move block in main memory. Locations x through y, inclusive, are moved to the area starting at location z. ● Source and destination areas can overlap in either direction without bad effects. ● Can move MANIP as long as destination area does not overlap source area. ● Parameter Description x - octal number representing a 16-bit beginning memory address y - octal number representing a 16-bit ending memory address z - octal number representing a 16-bit beginning memory address of a new location

TABLE 2-1. MANIP COMMAND DESCRIPTIONS (4 of 7)

Command & Parameters	Definition
Nx,y,z,m	Searches locations x through y, inclusive, for values not equal to constant z. Each word is ANDed with mask m before comparison with z. ● If m is not entered, 177777 is assumed. ● Parameter Description x - octal number representing a 16-bit beginning memory address y - octal number representing a 16-bit ending memory address z - octal number representing constant m - octal number representing mask; if omitted defaults to 177777. **Example of the use of mask:** The command Nx,y,0,170000 searches locations x through y for any value whose four MSBs are nonzero, i.e., for any value greater than 7777. If such a value is found, its address and contents are displayed in octal.
Ox,a	Outputs ASCII. Contents of memory starting at location x (using address mode a) are displayed as text. If a zero byte is found, the output is terminated. ● Parameter Description x - octal number representing a 16-bit memory address a - optional one digit (0-3) representing an address mode

TABLE 2-1. MANIP COMMAND DESCRIPTIONS (5 of 7)

Command & Parameters	Definition
Px	Program loads from disk or other DMA device. Performs standard bootstrap APL (gives a NIOS instruction with device code x, then idles at location 377 waiting for the disk to overwrite that location). If x is omitted, P reads the CPU mini-switches and uses their contents as the device code; if mini-switches are not set to a valid device code, P uses device code 27. **If the switch representing the 200 bit is set in addition to the device code switches, MANIP cannot be accessed. Pressing APL causes MANIP to try to boot from the disk.** ● Parameter Description x - One or two-digit octal number (1 through 76) representing the device code from which the program is to be loaded
Sx,y,z,m	Searches locations x through y, inclusive, for constant z. Each word is ANDed with mask m before comparison with z. ● If m is omitted, 177777 is assumed. ● Parameter Description x - octal number representing a 16-bit beginning memory address y - octal number representing a 16-bit ending memory address z - octal number representing a constant m - octal number representing a mask; if omitted defaults to the value 177777 **Example of the use of mask.** The command Sx,y,60025,160077 searches location x through y for any I/O instruction for device 25. If a comparison is found, its address and contents are displayed in octal.
T	Runs software self-test. CPU halts. Press <CONT> or press <ESC> J <RETURN>.

TABLE 2-1. MANIP COMMAND DESCRIPTION (6 of 7)

Command & Parameters	Definition
Ux,y,z,a	Loads from APL PROM beginning at PROM location x, and reading y words into main memory starting at location z, then jumps to starting address a. ● Parameter Description x - an octal number representing a beginning PROM location y - an octal number representing the number of words to be loaded z - an octal number representing a beginning memory address a - optional octal number representing a starting address
U	Loads DBUG from APL PROM into main memory starting at location 73000 and jumps into it.
Xx,y	Calculates and prints checksum over memory locations x through y. Uses a revolving checksum (using a SUBL 0,1 instruction with A0 = each word from x through y, and A1 = accumulating checksum; initially 0). This ensures that if two words in memory are swapped, the swap is detected by the checksum. Useful for determining if any word in memory has changed. ● Parameter Description x - an octal number representing a 16-bit beginning memory address y - an octal number representing a 16-bit ending memory address
Yx	Sets up a <RETURN> delay. Required on some terminals for proper scrolling. After each carriage return/line feed, MANIP counts up an accumulator from x to 0 before proceeding. For maximum delay set x=0, for no delay set x=177777. ● Parameter Description x - octal number representing <RETURN> delay value

TABLE 2-1. MANIP COMMAND DESCRIPTIONS (7 of 7)

Command & Parameters	Definition
x:y	Octal value y is stored at location x, and the next cell is opened (see E command). ● Parameter Description x - octal number representing a 16-bit memory address y - 1 to 6 digits representing an octal value
<CTRL-X> or other control character	Any control character is interpreted as a cassette tape unit (CTU) command. MANIP passes the command to the CTU and displays responses from the CTU, if any. CTU commands in MANIP are described in Table 2-2. For more information, refer to the IRIS R9 System Manager Manual.

TABLE 2-2. MANIP CTU COMMAND DESCRIPTIONS

Command & Parameters	Definition
<CTRL-D>	List directory (index) from tape, if tape is so formatted
<CTRL-E>	Enquire (error status).
<CTRL-K>file	Kill the named file.
<CTRL-O>file	Open named file if it is in the directory.
<CTRL-O>file,x,y	Create a directory entry for named file starting at block x and containing y+1 blocks of 128 words each.
<CTRL-R>	Read the open file from tape into memory.
<CTRL-R>x,y	Read from tape into memory; read y+1 blocks starting at block x.
<CTRL-S>x	Seek to block x on tape. <CTRL-S>999 winds tape all the way forward.
<CTRL-T>n	Select track n (0 or 1).
<CTRL-X>	Cancel partially entered command.
<CTRL-Z>	Rewind tape to starting position.
<ESC>	Exits CTU mode and reverts to MANIP.

2.3 MANIP LISTING

This section contains the MANIP listing.

```
; MANIP  --  RELOCATABLE RAM MANIPULATOR AND DEBUGGER
; MARK 12 VERSION WITH CASSETTE TAPE INTERFACE VIA MIGHTY-MUX [OR MARK 6]
; WRITTEN BY RENNY BOSCH
; 2 OCT 85
; LAST EDIT 30 APR 86 BY RB.

;                  All Rights Reserved
;       Copyright (C) 1975, Educational Data Systems
;       Copyright (C) 1986, Point 4 Data Corporation

; OVERVIEW OF CONTENT OF APL PROM:

          17000 L.ASM=17000              ;ASSEMBLY LOCATION (ARBITRARY)

          17000 L.MANIP=        0+L.ASM;MANIP IS FIRST THING
          20000 L.SELF=      1000+L.ASM;SELF-TEST, AND MINOR ROUTINES
          22000 L.DBUG=      3000+L.ASM;DBUG
          25400 L.BZUD=      6400+L.ASM;BZUD FOR LOTUS CONTROLLER

         177000 A.MANIP=     177000     ;CORE ADDRESS WHERE MANIP IS LOADED
          20000 A.SELF=      20000      ;CORE ADDRESS WHERE SELF-TEST IS LOADED
          73000 A.DBUG=      73000      ;CORE ADDRESS WHERE DBUG IS LOADED

              2 MEM=  2
             25 MUX=  25

          17000 .LOC   L.MANIP          ;ACTUAL LOC. = 177000
    17000 177000 PC:    A.MANIP          ;INITIAL PROGRAM COUNTER SAVED HERE

; ON ENTRY TO EACH OF THE "COMMAND LETTER" PROCEDURES,
;       A0 = FIRST OPERAND
;       A1 = SECOND OPERAND
;       A2 = FIRST OPERAND AS AN ADDRESS (INCL. OFFSET IF ANY)
;       A3 = B = CENTRAL REFERENCE POINT

    17001  40546 MANIP:STA    0,A        ;START HERE
    17002  44546       STA    1,A+1      ;SAVE ACCUMULATORS AND CPU STATUS
    17003  50546       STA    2,A+2
    17004  54546       STA    3,A+3
    17005 102520       SUBZL  0,0
    17006  60777       DIAP   0,CPU      ;READ SAVED CPU STATUS
    17007  40544       STA    0,A+4
    17010  24411       LDA    1,C.RUN
    17011 107414       AND#   0,1,SZR    ;DOING POWER-UP ?
    17012    550       JMP    MAN 0      ;  NO, CHECK FOR AUTO-BOOT
    17013   4526       JSR    RDSELF     ;READ SELF-TEST FROM APL PROMS
    17014  20406       LDA    0,N.SELF
    17015  41067       STA    0,FL.4EVER-L.SELF,2;CLEAR FOREVER FLAG
    17016 102000       ADC    0,0
    17017  42404       STA    0,@.IMASK;MASK OUT ALL INTRPTS; DISK MAY BE ON
    17020   1002       JMP    2,2        ;AND SKIP OVER HALT TEST

    17021     20 C.RUN:       20         ;RUN BIT IN CPU STATUS WORD
    17022      4 N.SELF:      4           ;NUMBER OF TIMES SELF-TEST IS RUN
    17023  20261 .IMASK:      IMASK-L.SELF+A.SELF
```

```
; ON POWER UP, RETURN HERE AFTER RUNNING SELF-TEST "N" TIMES

         17024 MAN.1:
17024 102520        SUBZL   0,0
17025  60777        DIAP    0,CPU     ;READ SAVED CPU STATUS WORD
17026  24504        LDA     1,C.AUTO
17027 107414        AND#    0,1,SZR   ;IS KEYSWITCH IN "AUTO" POSITION ?
17030    433        JMP     PFAR      ;   YES, DO POWER-FAIL AUTO-RESTART
17031   4467        JSR     CKSWITCHES;NO, CHECK MINI-SWITCHES
17032  20430        LDA     0,HMPTR   ;PRINT "PRESS ? FOR HELP MENU"
17033    402        SKIP
17034  20425 .QM:   LDA     0,QMPTR   ;"?" = DISPLAY HELP MENU
17035  41407        STA     0,COUNT-B,3
17036  21407 QMLOP: LDA     0,COUNT-B,3;POINTER INTO APL PROM MENU TEXT
17037  11407        ISZ     COUNT-B,3
17040 126520        SUBZL   1,1       ;READ 1 WORD FROM APL PROM
17041   4402        JSR     .+2
17042      0 QMWD:  0                 ;2 BYTES TEXT WILL BE READ INTO HERE
17043 171000        MOV     3,2       ;A2 = CORE ADDRESS
17044   4464        JSR     JRTN1     ;PICK UP POINTER TO "B" IN A3
17045   5654        JSR     RDAPL-B,3;DO THE PROM READ
17046   4462        JSR     JRTN1     ;RECOVER POINTER TO "B" IN A3
17047  20773        LDA     0,QMWD
17050 101015        SNZ     0,0       ;END OF TEXT MESSAGE ?
17051    545        JMP     INCM1     ;   YES, GOTO AWAIT COMMAND INPUT
17052 105224        MOVZR   0,1,SZR   ;IS TEXT WORD = 1 ?
17053    404        JMP     QMTYPE    ;   NO, TYPE 2 CHARACTERS
17054  64777        DIAP    1,CPU     ;YES, READ ID CODE FROM CPU FIRMWARE
17055   5444        JSR     TPOCT-B,3;TYPE IT IN OCTAL FORM
17056    760        JMP     QMLOP

17057   5473 QMTYP: JSR     TP2CH-B,3;TYPE 2 CHARACTERS OF TEXT MESSAGE
17060    756        JMP     QMLOP

17061   7015 QMPTR: L.MENU-L.ASM      ;POINTER TO HELP MENU IN APL PROM
17062   7000 HMPTR: L.HELP-L.ASM      ;POINTER TO "PRESS ?" MESSAGE


         17063 PFAR:                  ;POWER-FAIL AUTO-RESTART
17063 126400        SUB     1,1
17064 152400        SUB     2,2
17065  61402        DIB     0,MEM     ;READ BLOCK 0 --> PAGE 0
17066  24446        LDA     1,V.VIRGIN
17067 103112        ADDL#   0,0,SZC   ;DID A MEMORY RESPOND,
17070 107415        AND#    0,1,SNR   ;AND NON-VIRGIN ?
17071    404        JMP     M.ERROR   ;   NO, ERROR
17072 105222        MOVZR   0,1,SZC   ;YES, WAS TRANSFER ERROR-FREE,
17073 125212        SKE     1,1       ;OR WAS ERROR CORRECTED ?
17074    377        JMP     377       ;   YES, GOTO PFAR BOOTSTRAP
         17075 M.ERROR:               ;NO, MEMORY READ ERROR
17075 105000        MOV     0,1       ;SAVE MEMORY STATUS WORD
17076   4432        JSR     JRTN1
17077   5422        JSR     TCRLF-B,3;TYPE CR, LF
17100   5470        JSR     TYPE-B,3
17101   3534        7*L+"\            ;BELL, BACKSLASH
17102    476        JMP     TOCTI     ;TYPE MEMORY STATUS WORD & GOTO INCMD
```

```
17103  30432 .P:     LDA     2,C77      ;"P" = AUTO PROGRAM LOAD FROM DISC
17104 112033         SLS     0,2        ;OPERAND < 77 ?
17105    516         JMP     ABOR1      ;  NO, ERROR
17106 101015         SNZ     0,0        ;WAS AN OPERAND ENTERED ?
17107  60477         READS   0          ;  NO, READ SWITCHES
17110 143405         AND     2,0,SNR    ;MASK DEVICE CODE: IS IT 0 ?
17111  20425         LDA     0,C27      ;  YES, USE 27 AS A DEFAULT
       17112 LD.IPL:
17112  40344         STA     0,PZ.DEVCO;SAVE DEVICE CODE FOR IPL PROGRAM
17113   4557         JSR     LDOVLAY    ;LOAD IPL ROUTINE FROM APL PROMS
17114   2240           L.IPL-L.ASM
17115     37           E.IPL-L.IPL
17116    345           L.IPL+377-WT377
17117   1000         JMP      IPL-L.IPL,2;AND JUMP INTO IT


       17120 CKSWITCHES:
17120  60477         READS   0
17121  24412         LDA     1,C.SELF
17122 106415         SNE     0,1        ;SWITCHES SET FOR SELF-TEST ?
17123    414         JMP     .T         ;  YES, GOTO SELF-TEST
17124  24555         LDA     1,C200
17125  30410         LDA     2,C77
17126 122404         SUB     1,0,SZR    ;(SWITCHES) BETWEEN 200
17127 112033         SLS     0,2        ;AND 277 ?
17130    561 JRTN1:  JMP     RTN1       ;  NO, RETURN WITH (A3) = B
17131    761         JMP     LD.IPL     ;YES, DO AUTO IPL

17132  20000 C.AUTO:         20000      ;INDICATES KEYSWITCH IN AUTO POSITION
17133 100200 C.SELF:         100200     ;CODE IN MINI-SWITCHES FOR SELF-TEST
17134     20 V.VIRGIN:       20         ;0 MEANS MEMORY DOES NOT HOLD VALID DATA
17135     77 C77:            77
17136     27 C27:            27


17137   4402 .T:     JSR     RDSELF     ;"T" = LOAD AND RUN SELF-TEST
17140   1001         JMP     1,2

       17141 RDSELF:
17141  54406         STA     3,A
17142   4530         JSR     LDOVLAY
17143   1000           L.SELF-L.ASM
17144   1012           E.SELF-L.SELF
17145  20000           A.SELF
17146   2401         JMP     @A
```

```
17147       0 A:     0                ;CPU STATUS IS SAVED HERE: A0
17150       0        0                ;A1
17151       0        0                ;A2
17152       0        0                ;A3
17153       0        0                ;CPU STATUS WORD

17154    5561 .J:    JSR      CNVRT-B,3;"J" = JUMP
17155   24623        LDA      1,PC     ;SAVED VALUE OF PC AT LAST HALT
17156   21412        LDA      0,N.OP-B,3
17157  100234        COMZR#   0,0,SZR  ;ANY OPERAND(S) ENTERED ?
17160   45404        STA      1,ADR1-B,3;  NO, THEN DO A "CONTINUE"
17161     462        JMP      .J2

17162    4736 MAN.0: JSR      CKSWITCHES;NORMAL MANIP ENTRY
17163    5422        JSR      TCRLF-B,3;TYPE A CR, LF
17164   24614 .A:    LDA      1,PC     ;"A" = DUMP PC AND ACCUMULATORS
17165    5442        JSR      TPADR-B,3
17166    5472        JSR      TPCLN-B,3
17167   24760        LDA      1,A
17170    5447        JSR      TPOCL-B,3
17171   24757        LDA      1,A+1
17172    5447        JSR      TPOCL-B,3
17173   24756        LDA      1,A+2
17174    5447        JSR      TPOCL-B,3
17175   24755        LDA      1,A+3
17176    5447        JSR      TPOCL-B,3
17177   24754        LDA      1,A+4    ;   AND CARRY
17200    5447 TOCTI: JSR      TPOCL-B,3
17201     534        JMP      INCMD

17202   30476 .C:    LDA      2,C4     ;"C" = CHANGE ACCUMULATOR, C
17203  112432        SGR      0,2      ;IS FIRST OPND <= 4 ?
17204    4406        JSR      .C1      ;  YES
17205    5470 .CREF: JSR      TYPE-B,3 ;TYPE "=F" AND VIRTUAL EQUIVALENT
17206   43075        "F*L+"=
17207   25400        LDA      1,OP1-B,3
17210    5442        JSR      TPADR-B,3
17211     524        JMP      INCMD

17212  117000 .C1:   ADD      0,3
17213   45742        STA      1,A-.CREF,3;SAVE 2D OPND AS NEW CPU STATUS
17214  112415        SNE      0,2
17215   64377        64377             ;= NIOP CPU WITH A1
17216     517 INCM1: JMP      INCMD
```

```
17217  31412 .U:     LDA     2,N.OP-B,3; "U" = UNLOAD APL PROMS INTO MEMORY
17220 150225         COMZR   2,2,SNR   ;# OPNDS = 3 OR 4 ?
17221    410         JMP     .U1       ;  YES, USE GIVEN PARAMETERS
17222 151235         MOVZR#  2,2,SNR   ;NO, # OPNDS = 1 OR 2 ?
17223    507 ABOR1:  JMP     ABORT     ;  YES
17224   4446         JSR     LDOVLAY   ;NO OPERANDS: LOAD DBUG
17225   3000           L.DBUG-L.ASM
17226   4000           L.BZUD+400-L.DBUG
17227  73000           A.DBUG
17230    412         JMP     .J1       ;LEAVES A.DBUG IN A2

17231  31406 .U1:    LDA     2,ADR3-B,3
17232   4445         JSR     RDAPL     ;READ APL PROMS
17233   4456         JSR     RTN1
17234  21412         LDA     0,N.OP-B,3
17235 101414         INC#    0,0,SZR   ;FOURTH OPERAND GIVEN ?
17236    477         JMP     INCMD     ;  NO, STAY IN MANIP
17237  31403         LDA     2,OP4-B,3; YES
17240  21416         LDA     0,OFSET-B,3
17241 113000         ADD     0,2       ;CONVERT TO AN ADDRESS
17242  50565 .J1:    STA     2,ADR1    ;AND PREPARE TO JUMP THERE
17243  20710 .J2:    LDA     0,A+4     ;CONTINUATION OF "J" COMMAND
17244 101100         MOVL    0,0       ;LOAD CARRY
17245  20702         LDA     0,A       ;AND ACCUMULATORS
17246  24702         LDA     1,A+1
17247  30702         LDA     2,A+2
17250  34702         LDA     3,A+3
17251  60211         NIOC    TTO
17252   2555         JMP     @ADR1     ;JUMP TO USER PROGRAM


17253 106400 .M:     SUB     0,1       ;"M" = MOVE
17254  35406         LDA     3,ADR3-B,3
17255 102520         SUBZL   0,0       ;TENTATIVELY DELTA = 1
17256 156033         SLS     2,3       ;IS DIRECTION OF MOVE DOWNWARD ?
17257    404         JMP     .M1       ;  YES
17260 102000         ADC     0,0       ;  NO - MAKE DELTA = -1 AND
17261 133000         ADD     1,2       ;  START AT OTHER END
17262 137000         ADD     1,3
17263  25000 .M1:    LDA     1,0,2     ;FINALLY DO THE MOVE ITSELF
17264  45400         STA     1,0,3
17265 113000         ADD     0,2       ;ADD DELTA TO SOURCE & DEST. PNTRS.
17266 117000         ADD     0,3
17267  10543         ISZ     COUNT     ;DONE ?
17270    773         JMP     .M1       ;  NO
17271    444         JMP     INCMD     ;  YES
```

```
; * * * * * START OF MANIP'S "PAGE 0" (ACCESSIBLE IF A3 = B) * * * * *
          17272 LDOVLAY:                  ;LOAD AN OVERLAY FROM APL PROMS
 17272    21400        LDA    0,0,3
 17273    25401        LDA    1,1,3
 17274    31402        LDA    2,2,3
 17275 175600          INCR   3,3
 17276 175500          INCL   3,3
 17277    60077 RDAPL: NIO    CPU

 17300        4 C4:    4
 17301      200 C200:  200
 17302 177763 N.TS:   B-TSEND;NO. TS CELLS TO BE CLEARED FOR NEW CMD
 17303 177777 DELAY: -1      ;- DELAY AFTER CR

; BRANCH.  BRANCHES TO THE DESTINATION INDICATED IN TABLE ENTRY IF THE
; RIGHT-MOST 7 (OR 5) BITS THEREOF AGREE WITH A0.  CALLING SEQUENCE:
;     JSR    BRNC7      (OR BRNC5 FOR 5-BIT, WITH A1 = 37)
;     DEST1-.-1*K+CHAR1(OR F INSTEAD OF K FOR 5-BIT)
;     DEST2-.-1*K+CHAR2
;
;     END OF LIST IS INDICATED BY 7 (OR 5) LSB'S = 0

; A -1 IN THE TABLE IS USED TO DETERMINE MAX ALLOWABLE NO. OF OPERANDS

 17304    24512 BRNC7: LDA    1,C177
 17305 123400 BRNC5: AND    1,0
 17306    31400        LDA    2,0,3
 17307 175400          INC    3,3
 17310 147415          AND#   2,1,SNR   ;END OF LIST ?
 17311      511 RTN1:  JMP    RTNA3     ;  YES
 17312 150015          COM#   2,2,SNR   ;IS LIST ENTRY = -1 ?
 17313    10522        ISZ    N.OP      ;  YES: MAX. NO. OPNDS. EXCEEDED ?
 17314 112421          SUBZ   0,2,SKP   ;  NO OR YES,NO
 17315      415        JMP    ABORT     ;  YES,YES
 17316 133414          AND#   1,2,SZR   ;MATCH ?
 17317      767        JMP    BRNC5+1   ;  NO
 17320 151113          SSN    2,2       ;IS DISPLACEMENT NEGATIVE ?
 17321 125620          INCZR  1,1       ;  NO - CHANGE A1 TO 100 (OR 20)
 17322 151200          MOVR   2,2
 17323 125224          MOVZR  1,1,SZR   ;SHIFTED 7 (OR 5) PLACES ?
 17324      776        JMP    .-2       ;  NO
 17325    20476        LDA    0,OP1
 17326    24476        LDA    1,OP2
 17327 157000          ADD    2,3       ;YES - ADD TO "." IN LIST & GO THERE
 17330    30477        LDA    2,ADR1
 17331      471        JMP    RTNA3
```

```
17332  60210 ABORT:NIOC   TTI         ;ABORT, TYPE "\"
17333   4560       JSR    TYPE
17334    134       "\
17335  20463 INCMD:LDA    0,F.CUR     ;INPUT A COMMAND
17336  40503       STA    0,OFSET
17337   4506       JSR    TCRLF       ;TYPE CR, LF
17340  54500       STA    3,.TS       ;INITIALIZE OPERAND STORAGE POINTER
17341  14477       DSZ    .TS
17342  24740       LDA    1,N.TS
17343 102400       SUB    0,0
17344  41400       STA    0,0,3       ;CLEAR TEMP STORE AREA
17345 175400       INC    3,3
17346 125404       INC    1,1,SZR
17347    775       JMP    .-3
17350 101400 INCHA:INC    0,0         ;WAIT FOR INPUT
17351  63610       SKPDN  TTI
17352    776       JMP    .-2
17353  60610       DIAC   0,TTI
17354   4730       JSR    BRNC7       ;SEE IF IT'S AN ACTIVE CHARACTER
                   ;
17355  37015       .CR-.-1*K+15       ;CARRIAGE RETURN
17356  62336       .UP-.-1*K+"^       ;UP ARROW (EXAMINE PREVIOUS)
17357 172433       ABORT-.-1*K+33     ;ESCAPE
17360 172210       ABORT-.-1*K+10     ;BACKSPACE (UNAVAILABLE)
17361 112477       .QM-.-1*K+"?       ;QUESTION MARK --> HELP MENU
                   ;
17362  20000 C20K: 20000              ;SERVES AS LIST TERMINATOR
                   ;
17363  30457       LDA    2,C40
17364 112032       SGE    0,2         ;CONTROL CHARACTER ?
17365   1556       JMP    .CTU-B,3    ;  YES, GO TO CTU DRIVER
17366  61111       DOAS   0,TTO       ;NOT ACTIVE, ECHO IT
17367  30454       LDA    2,C60
17370 142400       SUB    2,0
17371  34453       LDA    3,C7
17372 116432       SGR    0,3         ;IS IT AN OCTAL DIGIT ?
17373    404       JMP    OCTAL       ;  YES
17374 143023       ADDZ   2,0,SNC     ;RECONSTITUTE CHAR.; IS IT COMMA ?
17375  40436 INCH2:STA    0,T         ;  NO, SAVE IT
17376    416       JMP    SOCTF

17377  10435 OCTAL:ISZ    OCTFL       ;FIRST OCTAL DIGIT OF A NUMBER ?
17400  10440       ISZ    .TS         ;  YES, ADVANCE PARAMETER POINTER
17401  26437       LDA    1,@.TS      ;PROCESS OCTAL CHARACTER
17402 125120       MOVZL  1,1         ;SHIFT PREV. NO. LEFT 3 BITS
17403 125120       MOVZL  1,1
17404 125120       MOVZL  1,1
17405 107000       ADD    0,1         ;ADD NEW DIGIT TO PREV. NO.
17406  46432       STA    1,@.TS
17407  20432       LDA    0,OFSET
17410 107000       ADD    0,1         ;CONVERT TO ADDRESS: ADD OFFSET IF ANY
17411  30427       LDA    2,.TS
17412  45004       STA    1,4,2       ;SAVE AS AN ADDRESS
17413 102000       ADC    0,0
17414  40420 SOCTF:STA    0,OCTFL     ;SET OCTAL FLAG
17415    733       JMP    INCHA
```

```
17416     177 C177:  177

17417       0 F.PRE:0        ;PREVIOUS VALUE OF "F" OFFSET
17420       0 F.CUR:0        ;CURRENT VALUE OF "F" OFFSET

17421   34415 RTNTS:LDA      3,TS       ;RETURN VIA TS
17422    5400 RTNA3:JSR      0,3        ;RETURN VIA A3

        17423 B=       .     ;USED AS THE CENTRAL LOCATION REFERENCE ***

17423       0 OP1:   0       ;FIRST OPERAND TYPED IN (OCTAL)
                             ;(LAST BYTE RECEIVED IN CTU READ)

17424       0 OP2:   0       ;2D OPND. (VALUE IN C, CONTROL IN D, O, L)

17425       0 OP3:   0       ;THIRD OPERAND (CONSTANT IN K, N, S)

17426       0 OP4:   0       ;FOURTH OPERAND (MASK IN N, S, EXEC. ADDR. IN L)

17427       0 ADR1:  0       ;FIRST OPERAND, CONVERTED TO AN ADDRESS

17430       0 ADR2:  0       ;2D OPND, CONVERTED TO AN ADDRESS

17431       0 ADR3:  0       ;THIRD ADDRESS (DESTINATION IN M, L)

17432       0 COUNT:0        ;NO. WORDS TO BE MOVED OR SEARCHED
                             ;WORD/BYTE INDICATOR IN D

17433       0 T:     0       ;COMMAND LETTER

17434       0 OCTFL:0        ;OCTAL FLAG, CONTROLS OPERAND COUNTING

17435       0 N.OP:  0       ;COUNTS NO. OF OPERANDS ENTERED

17436       0 TS:    0       ;GENERAL SUBROUTINE RETURN ADDRESS

17437       0 FLAG:  0       ;FLAG USED IN S/N AND E/: AND CTU

        17440 TSEND=.        ;END OF VARIABLES INITIALIZED TO 0

17440       0 .TS:   0       ;POINTER TO ABOVE TEMP. STORE (INCMD)

17441       0 OFSET:0        ;WORKING VALUE OF ADDRESS OFFSET


17442      40 C40:   40
17443      60 C60:   60
17444       7 C7:    7
```

```
; TYPE-OUT ROUTINES

    17445  54771 TCRLF: STA    3, TS       ; TYPE CARRIAGE RETURN, LINE FEED
    17446   4445        JSR    TYPE
    17447   5015        12*L+15
    17450   4443        JSR    TYPE
    17451  37055        ">*L+"-            ; PROMPT IS ->
    17452   4441        JSR    TYPE
    17453     40        "
    17454  35660        LDA    3, DELAY-B, 3; WAIT SPECIFIED DELAY AFTER CR, LF
    17455  20764        LDA    0, OFSET    ; (INCL. IN LOOP FOR LONGER MAX. DELAY)
    17456 175404        INC    3, 3, SZR
    17457    776        JMP    .-2
    17460 101015        SNZ    0, 0        ; IS THERE AN OFFSET ?
    17461    740        JMP    RTNTS       ;   NO, RETURN
    17462   4431        JSR    TYPE        ; YES, TYPE "F "
    17463  20106        " *L+"F
    17464    735        JMP    RTNTS

    17465  20754 TPADR: LDA    0, OFSET    ; TYPE A1 AS AN ADDRESS
    17466 106400        SUB    0, 1
    17467 152420 TPOCT: SUBZ   2, 2        ; SUPPRESS LEADING ZEROES
    17470  54746        STA    3, TS
    17471    405        JMP    TPA01

    17472  30750 TPOCL: LDA    2, C40      ; TYPE SPACES FOR LEADING ZEROES
    17473  20747        LDA    0, C40      ; TYPE ONE INITIAL SPACE
    17474  54742        STA    3, TS       ; TYPE THE OCTAL NO. IN A1, AFTER
    17475   4422        JSR    TPCHA       ;   TYPING THE CHARACTER IN A0
    17476 102620 TPA01: SUBZR  0, 0        ; PREPARE TO MOVE MSB OF A1 INTO A0
    17477 101041        MOVO   0, 0, SKP   ; SET CARRY TO FORM "PUSHER" BIT
    17500  20662 TPNXT: LDA    0, C20K     ; LEFT-SHIFT ONE DIGIT FROM A1 INTO A0
    17501 125105        MOVL   1, 1, SNR   ; INITIALLY INSERTS "PUSHER" BIT
    17502    717        JMP    RTNTS       ; EXIT WHEN "PUSHER" BIT IS GONE
    17503 101103        MOVL   0, 0, SNC
    17504    775        JMP    .-3
    17505 101015        SNZ    0, 0        ; NON-ZERO DIGIT ...
    17506 125135        MOVZL# 1, 1, SNR   ;   ... OR LAST DIGIT ?
    17507  30734        LDA    2, C60      ;   YES: ADDEND FOR ASCII DIGIT
    17510 143040        ADDO   2, 0
    17511   4406        JSR    TPCHA
    17512    766        JMP    TPNXT
```

```
17513  21400 TYPE:  LDA     0,0,3      ;TYPE THE CHAR.(S) FOLL. THE JSR
17514 175401        INC     3,3,SKP
17515  20433 TPCLN: LDA     0,COLON    ;TYPE COLON
17516 101020 TP2CH: MOVZ    0,0        ;TYPE 2 CHARACTERS IN A0
17517  54427 TPCHA: STA     3,TPSV3    ;TYPE 1 CHAR. IN A0 IF C=1; 2 IF C=0
17520  40427 TPCH1: STA     0,TPSV0
17521  63610 TPCH2: SKPDN   TTI
17522    414        JMP     TPCH4
17523  60610 TPCH3: DIAC    0,TTI
17524  34672        LDA     3,C177
17525 163400        AND     3,0
17526  34423        LDA     3,C22
17527 116015        ADC#    0,3,SNR
17530    406        JMP     TPCH4
17531 162014        ADC#    3,0,SZR
17532    603        JMP     INCMD
17533  63610 TPCH6: SKPDN   TTI
17534    777        JMP     TPCH6
17535    766        JMP     TPCH3

17536  63511 TPCH4: SKPBZ   TTO
17537    762        JMP     TPCH2
17540  20407        LDA     0,TPSV0
17541  61111        DOAS    0,TTO
17542 101362        MOVCS   0,0,SZC    ;SECOND CHAR. TO BE TYPED ?
17543    755        JMP     TPCH1      ;   YES
17544  34402        LDA     3,TPSV3
17545    655        JMP     RTNA3

17546      0 TPSV3: 0
17547      0 TPSV0: 0
17550     72 COLON: ":
17551     22 C22:   22
```

```
    17552 122000 .CR:   ADC     1,0        ;PROCESS CARRIAGE RETURN
    17553  41407        STA     0,COUNT-B,3
    17554  20664        LDA     0,.TS
    17555  25655        LDA     1,C4-B,3
    17556 122400        SUB     1,0
    17557 162422        SUBZ    3,0,SZC   ;> 4 OPERANDS ENTERED ?
    17560   1707        JMP     ABORT-B,3;  YES, ERROR
    17561  40654        STA     0,N.OP     ;NO. OF OPERANDS - 5
    17562   4404        JSR     OVERLAY
    17563    623        L.OVLAY-L.ASM

    17564 177400 C377L:177400
    17565    377 C377:  377

          17566 OVERLAY:                   ;READ IN AN OVERLAY AND EXECUTE IT
    17566  21400        LDA     0,0,3      ;LOCATION IN APL PROM
    17567  24432        LDA     1,SZOVLAY;SIZE OF OVERLAY AREA
    17570   4432        JSR     GTPOVLAY ;GET CORE ADDRESS FOR OVERLAY
    17571 171000        MOV     3,2
    17572   4406        JSR     NIOCPU     ;READ IN THE OVERLAY
    17573   4627        JSR     RTNA3      ;PREPARE ACCUMULATORS
    17574  20627        LDA     0,OP1
    17575  24627        LDA     1,OP2
    17576  30631        LDA     2,ADR1
    17577    424        JMP     L.OVLAY    ;AND EXECUTE OVERLAY CODE

          17600 NIOCPU:
    17600  60077        NIO     CPU        ;READ APL PROM AND RETURN TO (A3)

          17601 .CTU:                      ;<CTRL-CHAR> = CTU ACCESS COMMAND
    17601  41410        STA     0,T-B,3
    17602   4764        JSR     OVERLAY    ;USE CTU OVERLAY
    17603   2012        L.CTU-L.ASM

; CONVERT ADDRESSING MODE, INCL. VIRTUAL (USING "F" OFFSET) AND/OR
; BYTE (LOWER OR UPPER CORE OR VIRTUAL) - USED IN D,E,I,J,O

    17604 125015 CNVRT:SNZ     1,1        ;BYTE ADDRESSING MODE ?
    17605    405        JMP     CNV2       ;  NO
    17606 131225        MOVZR   1,2,SNR    ;PUT LSB IN C; WAS OP2 = 1 ?
    17607 124020        COMZ    1,1        ;   YES, SET TO -1 & SET C = 0
    17610 101200        MOVR    0,0        ;CONVERT BYTE TO WORD ADDR., USING C
    17611 125103        MOVL    1,1,SNC    ;SAVE C IN LSB OF A1
    17612  30623 CNV2:  LDA     2,N.OP
    17613 151404        INC     2,2,SZR    ;MAX. NO. OPERANDS ?
    17614  30625        LDA     2,OFSET    ;   NO, USE OFFSET
    17615  50624        STA     2,OFSET
    17616 113000        ADD     0,2        ;NOW CALC. DESIRED ADDRESS
    17617  50610        STA     2,ADR1
    17620    602 GETA3: JMP     RTNA3      ;FOR USE BY OVERLAY ROUTINES

; * * * * * * * * * * END OF MANIP'S "PAGE ZERO" * * * * * * * * * * *
```

```
   17621      151 SZOVLAY:      L.MMCW-L.OVLAY

            17622 GTPOVLAY:
   17622    5400        JSR     0,3        ;GET POINTER TO OVERLAY AREA

; ********** OVERLAY AREA BEGINS HERE: **************

            17623 L.OVLAY:

; CONTINUATION OF .CR CODE -- MUST BE FIRST ITEM IN OVERLAY AREA

   17623   20610        LDA     0,T        ;BRANCH ON INITIAL LETTER
   17624   24430        LDA     1,C37
   17625    5662        JSR     BRNC5-B,3;COMMAND LETTER BRANCH TABLE
                        ;
   17626 157425         .U-.-1*F+"U-H      ;UNLOAD PROM
   17627    1316        .N-.-1*F+"N-H      ;SEARCH FOR NOT EQUAL
   17630    1223        .S-.-1*F+"S-H      ;SEARCH
   17631 177777                 -1         ;MAX 3 OPERANDS HEREAFTER
   17632    4013        .K-.-1*F+"K-H      ;ENTER A CONSTANT
   17633 160755        .M-.-1*F+"M-H      ;MOVE A BLOCK
   17634 177777                 -1         ;MAX 2 OPERANDS HEREAFTER
   17635 156203        .C-.-1*F+"C-H      ;CHANGE ACCUMULATOR, ETC.
   17636    1044        .D-.-1*F+"D-H      ;DUMP MEMORY
   17637    1705        .E-.-1*F+"E-H      ;ENTER OR EXAMINE
   17640    2706        .F-.-1*F+"F-H      ;ESTABLISH AN ADDRESS OFFSET
   17641 154512        .J-.-1*F+"J-H      ;JUMP TO A LOCATION, OR CONTINUE
   17642    3717        .O-.-1*F+"O-H      ;OUTPUT IN ASCII
   17643    4570        .X-.-1*F+"X-H      ;COMPUTE A CHECKSUM
   17644     772        .CLN-.-1*F+":-F    ;OPEN A CELL FOR ENTRY OR EXAM.
   17645 177777                 -1         ;MAX 1 OPERAND HEREAFTER
   17646 154641        .A-.-1*F+"A-H      ;DISPLAY ACCUMULATORS, ETC.
   17647 151560        .P-.-1*F+"P-H      ;PROGRAM LOAD FROM DMA DEVICE
   17650 153324        .T-.-1*F+"T-H      ;LOAD SELF-TEST AND HALT
   17651     431        .Y-.-1*F+"Y-H      ;DELAY FOR CRT SCROLLING
                        ;
   17652       0        0                  ;LIST TERMINATOR
   17653    1707        JMP     ABORT-B,3

   17654      37 C37:   37

   17655   15414 .S:    DSZ     FLAG-B,3 ;"S" = SEARCH FOR GIVEN VALUE
   17656    4710 .N:    JSR     OVERLAY  ;"N" = SEARCH FOR NOT-EQUAL
   17657    2161        L.SN-L.ASM

   17660    4706 .D:    JSR     OVERLAY  ;"D" = DUMP OCTAL - WORDS OR BYTES
   17661    2207        L.D-L.ASM

   17662   41660 .Y:    STA     0,DELAY-B,3;"Y" = SET RETURN DELAY
   17663    1712        JMP     INCMD-B,3
```

```
17664  11412 .CLN:  ISZ     N.OP-B,3 ;INPUT = COLON: TWO OPERANDS ?
17665    423        JMP     .CLN1    ;   NO, DISPLAY CONTENT
17666  45000        STA     1,0,2    ;   YES, STORE OP2 AT ADR1
17667 102521 .CLN2: SUBZL   0,0,SKP  ;<< FROM .CLN1
17670 102000 .UP:   ADC     0,0      ;"^" = EXAMINE PREVIOUS ADDRESS
17671  25400        LDA     1,OP1-B,3
17672 107000        ADD     0,1
17673  45400        STA     1,OP1-B,3
17674  31404        LDA     2,ADR1-B,3
17675 113001        ADD     0,2,SKP
17676   5561 .E:    JSR     CNVRT-B,3;"E" = PREPARE TO ENTER DATA
17677  51404        STA     2,ADR1-B,3
17700   5422 NXTL:  JSR     TCRLF-B,3
17701  25404        LDA     1,ADR1-B,3
17702   5442        JSR     TPADR-B,3
17703  45414        STA     1,FLAG-B,3;SET "EXAMINE" FLAG = 0
17704  45401 .CLN3: STA     1,OP2-B,3;PREPARE FOR OCTAL INPUT --> OP2
17705  55415        STA     3,.TS-B,3;(PRETEND ONE OPERAND HAS COME IN)
17706   5472        JSR     TPCLN-B,3;TYPE A COLON
17707   1752        JMP     INCH2-B,3;COUNT AS ONE OPERAND, SET T = ":"

17710  21414 .CLN1: LDA     0,FLAG-B,3;2D OPERAND NOT TYPED IN
17711 101014        SKZ     0,0      ;HAVE WE ALREADY EXAMINED IT ?
17712    755        JMP     .CLN2    ;   YES, GO TO NEXT LINE
17713  11414        ISZ     FLAG-B,3 ;   NO
17714  25000        LDA     1,0,2
17715   5447        JSR     TPOCL-B,3;TYPE THE VALUE AT ADR1
17716    766        JMP     .CLN3    ;TYPE A COLON & WAIT FOR INPUT


17717 122400 .F:    SUB     1,0      ;"F" = ESTABLISH ADDRESS OFFSET
17720 105000        MOV     0,1
17721  21775        LDA     0,F.CUR-B,3
17722  31412        LDA     2,N.OP-B,3
17723 150234        COMZR#  2,2,SZR  ;ANY OPERANDS ENTERED ?
17724  25774        LDA     1,F.PRE-B,3;   NO, SWAP WITH PREVIOUS OFFSET
17725 106414        SEQ     0,1      ;IS NEW VALUE DIFF. FROM CURRENT ?
17726  41774        STA     0,F.PRE-B,3;   YES, SAVE CURRENT VALUE
17727  45775        STA     1,F.CUR-B,3
17730   5472        JSR     TPCLN-B,3
17731   5447 TPOCI: JSR     TPOCL-B,3;TYPE THE NEW OFFSET VALUE
17732   1712        JMP     INCMD-B,3


17733  21402 .K:    LDA     0,OP3-B,3;"K" = ENTER A CONSTANT IN CORE
17734  41000        STA     0,0,2
17735 151400        INC     2,2
17736  11407        ISZ     COUNT-B,3
17737    775        JMP     .-3
17740   1712        JMP     INCMD-B,3
```

```
17741   5561 .O:     JSR    CNVRT-B,3;"O" = OUTPUT ASCII
17742   5472         JSR    TPCLN-B,3
17743 125200         MOVR   1,1       ;PUT INITIAL BYTE INDICATOR IN C
17744  25542 .02:    LDA    1,C377-B,3
17745  21000         LDA    0,0,2
17746 101003         MOV    0,0,SNC   ;INITIAL PASS TYPE 1 BYTE ?
17747 101300         MOVS   0,0       ;   NO
17750 107405         AND    0,1,SNR   ;IS FIRST BYTE = 0 ?
17751   1712         JMP    INCMD-B,3;   YES
17752   5474         JSR    TPCHA-B,3;   NO, TYPE 1 OR 2 BYTES
17753 106415         SNE    0,1       ;WAS SECOND BYTE = 0 ?
17754   1712         JMP    INCMD-B,3;   YES
17755 151400         INC    2,2       ;   NO
17756    766         JMP    .02

17757 126440 .X:     SUBO   1,1       ;"X" = CALCULATE CHECKSUM
17760  21000         LDA    0,0,2
17761 106500         SUBL   0,1       ;FORM NEGATIVE ROTATING CHECKSUM
17762 151400         INC    2,2
17763  11407         ISZ    COUNT-B,3;DONE ?
17764    774         JMP    .-4       ;   NO
17765 125200         MOVR   1,1
17766    743         JMP    TPOCI     ;   YES, TYPE IT OUT

        17774 L.MMCW=       L.MANIP+774

17767       5 .BLK  L.MMCW-.

; MIGHTY-MUX CONTROL WORDS -- KEPT HERE FOR EASE OF MODIFICATION

17774   66000 PORTO:66000  ;MIGHTY-MUX PORT 0 CONTROL BLOCK POINTER
17775   50057 PCONO:50057  ;MIGHTY-MUX PORT 0 (CRT) CONTROL WORD
17776   66040 PORT1:66040  ;MIGHTY-MUX PORT 1 CONTROL BLOCK POINTER
17777   40377 PCON1:40377  ;MIGHTY-MUX PORT 1 (CTU) CONTROL WORD


        20000 E.MANIP:

           40 F=      40
          100 H=      100
          200 K=      200
          400 L=      400

              .EOT  ;MANIP
```

```
; MARK 12 MANIP OVERLAYS
; 23 OCT 85
; LAST EDIT 30 APR 86 BY RB.

; CTU -- CASSETTE TAPE UNIT DRIVER, FOR ADPI-1 VIA MIGHTY-MUX PORT 1

         21012 L.CTU:

          1167 D.CTU=L.CTU-L.OVLAY   ;DELTA BETWEEN MANIP AND CTU OVERLAY

 21012   4546         JSR    GETREF
 21013  31402         LDA    2,PORT0+D.CTU-E.CTU,3
 21014  71025         DOA    2,MUX     ;SET UP MUX CONTROL AREA
 21015 102400         SUB    0,0
 21016  41000         STA    0,0,2     ;CLEAR ICW
 21017  25403         LDA    1,PCON0+D.CTU-E.CTU,3
 21020  45001         STA    1,1,2     ;SET UP PORT CONTROL PARAMETERS
 21021  31404         LDA    2,PORT1+D.CTU-E.CTU,3
 21022  41000         STA    0,0,2     ;ALSO INITIALIZE PORT 1
 21023  25405         LDA    1,PCON1+D.CTU-E.CTU,3
 21024  45001         STA    1,1,2
 21025  63025         DOC    0,MUX     ;TURN MIGHTY-MUX ON
 21026   4761         JSR    GETA3+D.CTU
 21027  21410         LDA    0,T-B,3  ;COMMAND LETTER
 21030   4511         JSR    OUTP1     ;SEND IT TO PORT 1
 21031   4473 CTUP0:  JSR    INP0      ;IS THERE INPUT FROM PORT 0 (CRT) ?
 21032    407         JMP    CTUP1     ;   NO, EXAMINE PORT 1
 21033  25773         LDA    1,C177-B,3;YES, MASK TO 7 BITS
 21034 123400         AND    1,0
 21035  24522         LDA    1,C33
 21036 106415         SNE    0,1       ;IS IT ESCAPE ?
 21037    431         JMP    CTUDN     ;   YES
 21040   4501         JSR    OUTP1     ;NO, SEND IT TO PORT 1
 21041   4465 CTUP1:  JSR    INP1      ;IS THERE INPUT FROM PORT 1 (CTU) ?
 21042    767         JMP    CTUP0     ;   NO, RE-EXAMINE PORT 0
 21043   4474         JSR    OUTP0     ;YES, SEND IT TO PORT 0
 21044  24454         LDA    1,C12
 21045 106415         SNE    0,1       ;WAS IT LF ?
 21046    407         JMP    CTULF     ;   YES
 21047  25414         LDA    1,FLAG-B,3;NO, SAVE LAST 2 INPUTS
 21050  31542 CTUSV:  LDA    2,C377-B,3
 21051 147700         ANDS   2,1
 21052 107000         ADD    0,1
 21053  45414         STA    1,FLAG-B,3
 21054    755         JMP    CTUP0
```

```
21055  21414 CTULF: LDA    0, FLAG-B, 3
21056  24443        LDA    1, C1515
21057 106414        SEQ    0, 1       ; WAS INPUT CR, CR, LF ?
21060    405        JMP    CTUL2      ;   NO
21061  21410        LDA    0, T-B, 3  ; YES = COMMAND ACKNOWLEDGED
21062  24441        LDA    1, C. CR
21063 122445        SUBO   1, 0, SNR  ; ARE WE DOING A CTU READ?
21064    411        JMP    CTUR1      ;   YES
21065  24435 CTUL2: LDA    1, C0715
21066 106414        SEQ    0, 1       ; WAS INPUT BELL, CR, LF ?
21067    742        JMP    CTUPO      ;   NO, STAY IN CTU MODE
21070   4717 CTUDN: JSR    GETA3+D. CTU
21071  11403        ISZ    OP4-B, 3   ; ALLOW TIME TO XMIT LAST CHAR.
21072    777        JMP    . -1
21073  60225        NIOC   MUX        ; TURN OFF MIGHTY-MUX
21074   1712        JMP    INCMD-B, 3; GET BACK TO MANIP

21075  41400 CTUR1: STA    0, OP1-B, 3; READ CTU --> CORE
21076   4430        JSR    INP1       ; INPUT A BYTE FROM CTU
21077    777        JMP    . -1
21100  25421        LDA    1, C7-B, 3
21101 106414        SEQ    0, 1       ; IS IT A BELL ?
21102    406        JMP    CTUR2      ;   NO
21103   4423        JSR    INP1       ; YES, GET NEXT BYTE
21104    777        JMP    . -1
21105  25421        LDA    1, C7-B, 3
21106 106414        SEQ    0, 1       ; ANOTHER BELL ?
21107    741        JMP    CTUSV      ;   NO, EXIT READ MODE
21110  25400 CTUR2: LDA    1, OP1-B, 3; RETRIEVE LAST INPUT BYTE
21111 125362        MOVCS  1, 1, SZC  ; IS THIS THE 2ND OF A PAIR ?
21112    763        JMP    CTUR1      ;   NO, GET ANOTHER ONE
21113 107000        ADD    0, 1       ; YES, COMBINE 2 BYTES INTO A WORD
21114  47404        STA    1, @ADR1-B, 3; AND STORE IT AWAY
21115  11404        ISZ    ADR1-B, 3
21116    757        JMP    CTUR1

21117  60000 C60K:  60000  ; MMUX SINGLE-CHAR. OUTPUT CODE
21120     12 C12:   12     ; LF
21121   6415 C1515: 15*L+15; CR, CR, LF = COMMAND ACK.
21122   3415 C0715: 7*L+15 ; BELL, CR, LF = COMMAND DONE
21123     22 C. CR: "R-100 ; CTRL R = CTU READ BLOCKS CMD.
```

```
21124  30437 INP0:  LDA    2,PORT0+D.CTU;TEST PORT 0 FOR INPUT
21125    402        JMP    .+2
21126  30437 INP1:  LDA    2,PORT1+D.CTU;TEST PORT 1 FOR INPUT
21127  21000        LDA    0,0,2
21130 101113        SSN    0,0      ;IS THERE ANY INPUT ?
21131    656        JMP    GETA3+D.CTU;  NO, RETURN
21132  24424        LDA    1,K377
21133  45000        STA    1,0,2    ;CLEAR ICW
21134 123400        AND    1,0      ;MASK INPUT BYTE
21135 175400        INC    3,3
21136    651        JMP    GETA3+D.CTU;SKIP RETURN

21137  30424 OUTP0: LDA    2,PORT0+D.CTU;DO OUTPUT TO PORT 0
21140    402        JMP    .+2
21141  30424 OUTP1: LDA    2,PORT1+D.CTU;DO OUTPUT TO PORT 1
21142  26421 OUTPL: LDA    1,@PORT0+D.CTU
21143 125113        SSN    1,1      ;IS THERE ANY INPUT ON PORT 0,
21144  63710        SKPDZ  TTI      ;OR ON TTY INTERFACE ?
21145    723        JMP    CTUDN    ;   YES, GO BACK TO TTY MODE
21146  25001        LDA    1,1,2    ;NO
21147 125113        SSN    1,1      ;HAS PREVIOUS OUTPUT BEEN SENT ?
21150    772        JMP    OUTPL    ;  NO, STAY IN OUTPUT LOOP
21151  24746        LDA    1,C60K   ;YES
21152 107000        ADD    0,1      ;ADD OUTPUT CONTROL CODE TO BYTE
21153  45001        STA    1,1,2    ;AND PUT IT IN OCW
21154  61025        DOA    0,MUX    ;PROD MMUX OUTPUT
21155    632        JMP    GETA3+D.CTU;RETURN

21156    377 K377:  377
21157     33 C33:   33              ;ESCAPE

       21160 GETREF:
21160   5400        JSR    0,3      ;PICK UP POINTER TO HERE

       21161 E.CTU:

21161      2 .BLK   L.MMCW+D.CTU-.  ;OVERFLOW CHECK
       21161 .LOC   E.CTU
```

```
; SEARCH ROUTINES: S (SEARCH) AND N (SEARCH FOR NOT-EQUAL)

         21161 L.SN:

          1336 D.SN= L.SN-L.OVLAY

21161  31404 .SN2:  LDA    2,ADR1-B,3
21162  21000        LDA    0,0,2
21163  25403        LDA    1,OP4-B,3;4TH OPERAND (IF ANY) IS THE MASK
21164 125014        SKZ    1,1
21165 123400        AND    1,0
21166  25402        LDA    1,OP3-B,3;THIRD OPERAND IS THE COMPAREE
21167 122404        SUB    1,0,SZR   ;A0 = 0 MEANS EQUAL
21170 102520        SUBZL  0,0       ;A0 = 1 MEANS NOT EQUAL
21171  31414        LDA    2,FLAG-B,3
21172 113213        ADDR#  0,2,SNC   ;HIT (IE. = IF S OR <> IF N) ?
21173    407        JMP    NXTSN     ;   NO
21174   5422        JSR    TCRLF-B,3;NEXT LINE - TYPE CR, LF
21175  25404        LDA    1,ADR1-B,3
21176   5442        JSR    TPADR-B,3;TYPE THE ADDRESS
21177   5472        JSR    TPCLN-B,3;TYPE A COLON
21200  27404        LDA    1,@ADR1-B,3
21201   5447        JSR    TPOCL-B,3;TYPE VALUE IN OCTAL FORM
21202  11404 NXTSN: ISZ    ADR1-B,3
21203    401        JMP    .+1
21204  11407        ISZ    COUNT-B,3
21205    754        JMP    .SN2
21206   1712        JMP    INCMD-B,3
```

```
; OVERLAY FOR D = DUMP MEMORY IN OCTAL
        21207 L.D:
         1364 D.D=   L.D-L.OVLAY
   21207    4775          JSR    GETA3+D.D;PICK UP B IN A3
   21210    5561          JSR    CNVRT-B,3;"D" = DUMP OCTAL - WORDS OR BYTES
   21211    5422 DLINE:JSR    TCRLF-B,3;TYPE CR
   21212   25400          LDA    1,OP1-B,3
   21213    5444          JSR    TPOCT-B,3;TYPE THE (WORD OR BYTE) ADDRESS
   21214    5472          JSR    TPCLN-B,3;TYPE A COLON
   21215   20414          LDA    0,CM10
   21216   41407          STA    0,COUNT-B,3
   21217   27404 DNEXT:LDA    1,@ADR1-B,3;TYPE NEXT WORD OR BYTE
   21220   21401          LDA    0,OP2-B,3
   21221  101015          SNZ    0,0       ;WORD MODE ?
   21222     406          JMP    DWORD     ;   YES
   21223   21400          LDA    0,OP1-B,3
   21224  101203          MOVR   0,0,SNC   ;LEFT BYTE ?
   21225  125300          MOVS   1,1       ;   YES
   21226   21542          LDA    0,C377-B,3
   21227  107402          AND    0,1,SZC
   21230   11404 DWORD:ISZ    ADR1-B,3
   21231  177770 CM10:  177770           ;NO-OP
   21232   11400          ISZ    OP1-B,3
   21233     401          JMP    .+1
   21234    5447          JSR    TPOCL-B,3;TYPE THE WORD OR BYTE IN OCTAL
   21235   11407          ISZ    COUNT-B,3;TYPED 8 ITEMS YET ?
   21236     761          JMP    DNEXT     ;   NO, TYPE ANOTHER NO.
   21237     752          JMP    DLINE     ;   YES, TYPE NEW LINE
```

```
; AUTO PROGRAM LOAD FROM DISC OR OTHER DMA DEVICE

        21240 L.IPL:
        21237 DEVCO=.-1        ;PROTECT AGAINST BEING OVERLAID BY BOOTSTRAP

21240   62677 IPL:   IORST
21241 102400         SUB     0,0
21242 126400         SUB     1,1
21243   60402        DIA     0,MEM
21244   34431        LDA     3,C.VIRGIN
21245 101213         SKO     0,0       ;EXTENDED MEMORY EXISTS,
21246 117414         AND#    0,3,SZR   ;AND VIRGIN ?
21247     416        JMP     P.IPL     ;  NO, THEN JUST IPL
21250   30426        LDA     2,C1000   ;YES, FILL CORE PAGE 2 WITH 400
21251 155220         MOVZR   2,3       ;FORM A3 = 400
21252   55000 P.ZER: STA     3,0,2
21253 151400         INC     2,2
21254 157415         AND#    2,3,SNR   ;HAS A2 BECOME = 1400 ?
21255     775        JMP     P.ZER     ;  NOT YET
21256   30420 P.CLM: LDA     2,C1000   ;YES
21257   62002        DOB     0,MEM     ;CLEAR ALL EXTENDED MEMORY
21260 125400         INC     1,1
21261 125113         SSN     1,1       ;PAST 16 MB,
21262 101212         SKE     0,0       ;  OR END OF EXTENDED MEMORY ?
21263     402        SKIP
21264     772        JMP     P.CLM     ;  NO
21265   20404 P.IPL: LDA     0,NIOSX   ;YES
21266   30751        LDA     2,DEVCO
21267 143000         ADD     2,0       ;CALCULATE THE NIOS DISC INSTR.
21270   40401        STA     0,NIOSX   ;STORE THE INSTRUCTION
21271   60100 NIOSX: NIOS    0
21272   63610 WT377: SKPDN   TTI       ;WILL BE OVERLAID BY BOOTSTRAP FROM DISK
21273     777        JMP     .-1
21274   63077        HALT              ;ALLOW TERMINAL INPUT TO RECOVER CONTROL

21275      20 C.VIRGIN:      20        ;0 MEANS MEMORY DOES NOT HOLD VALID DATA
21276    1000 C1000:         1000

        21277 E.IPL:

          344 PZ.DEVCO=      DEVCO+377-WT377

              .EOT  ;MISC.   MARK 12 MANIP ROUTINES
```

```
; MARK 12 PROHIBITED CODE TRAP HANDLER [MARK 12 ONLY]
; 23 OCT 85

; WHEN MK12-PROHIBITED CODE TRAPPING IS ENABLED, THE MARK 12 HARDWARE
; DETECTS ALL INDIRECT MEMORY REFERENCES THROUGH LOCATIONS 20-37 (AUTO-
; INCREMENT/DECREMENT), AND ALL STA, ISZ, AND DSZ INSTRUCTIONS WHOSE
; EFFECTIVE ADDRESS IS GREATER THAN THEIR OWN LOCATION BY 1-5 WORDS.
; WHEN ANY SUCH INSTRUCTION IS DETECTED, ITS LOCATION IS SAVED, CURRENT
; CPU STATUS IS SAVED, INTERRUPTS ARE DISABLED, AND CONTROL IS TURNED OVER
; TO A TRAP HANDLER ROUTINE. THE FOLLOWING IS AN APPROPRIATE TRAP HANDLER.

; THE TWO WORDS JUST BELOW THE INTERRUPT HANDLER, WHOSE ADDRESS IS IN
; LOCATION 1, ARE USED FOR THIS TRAP. (INTH-1) IS THE ADDRESS OF THE TRAP
; HANDLER, AND INTH-2 IS WHERE THE PROGRAM COUNTER IS SAVED. THE SAVED
; P.C. IS THE VALUE AFTER THE TRAPPED INSTRUCTION; I.E. WHERE TO RESUME.

; CPU STATUS IS SAVED SO THAT THE TRAP HANDLER CAN DETERMINE WHETHER
; INTERRUPTS WERE ENABLED ON ENTRY; IF SO THEY NEED TO BE RE-ENABLED ON
; EXIT. SAVED CPU STATUS IS READ WITH A DIAP AC,77 INSTRUCTION WITH AC
; CONTAINING A 1. THE 10-BIT IS THE ION BIT.

; NOTE: THE MK12 TRAP HANDLER MUST CONTAIN NO INSTRUCTION THAT CAN TRIGGER
; THE MK12 TRAP MECHANISM; I.E. NO AUTO INCREMENT/DECREMENT AND NO STA,
; ISZ, OR DSZ WITH (E.A. - LOC) <= 5.

                 1 .INTH=1        ;RESERVED LOCATION OF INTERRUPT HANDLER ADDRESS

   21277       301 .BLK  L.DBUG-200-.

   21600     40527 TRAPH:STA   0,AC0      ;SAVE ACCUMULATORS AND CARRY
   21601     44527       STA   1,AC1
   21602     50527       STA   2,AC2
   21603     54527       STA   3,AC3
   21604    101100       MOVL  0,0
   21605     40526       STA   0,CARRY
   21606     30001       LDA   2,.INTH
   21607     31376       LDA   2,-2,2     ;PC WHEN TRAP OCCURRED
   21610     50516       STA   2,RETURN
   21611     21377       LDA   0,-1,2     ;THE TRAPPED INSTRUCTION
   21612     34533       LDA   3,X377
   21613    117400       AND   0,3
   21614     24521       LDA   1,X1400
   21615    107705       ANDS  0,1,SNR    ;ADDRESSING MODE = PZ ?
   21616       414       JMP   AMPZ       ;  YES
   21617    150400       NEG   2,2        ;NO, DECREMENT SAVED PC, IN CASE REL.
   21620    150000       COM   2,2
   21621    125235       MOVZR# 1,1,SNR   ;ADDRESSING MODE = RELATIVE ?
   21622       404       JMP   AMREL      ;  YES
   21623     30506       LDA   2,AC2      ;NO, ASSUME BASE 2
   21624    125212       SKE   1,1        ;IS IT BASE 3 ?
   21625     30505       LDA   2,AC3      ;  YES
             21626 AMREL:
   21626     24516       LDA   1,X200     ;
   21627    167520       ANDZL 3,1        ;EXTRACT SIGN BIT
   21630    136400       SUB   1,3        ;EXTEND SIGN BIT
   21631    157000       ADD   2,3        ;A3 NOW = FIRST EFFECTIVE ADDRESS
```

```
           21632 AMPZ:
  21632  24514         LDA    1,X2000
  21633 107415         AND#   0,1,SNR   ;INDIRECT ADDRESSING ?
  21634    410         JMP    CKSMC     ;  NO, CHECK IF SELF-MODIFYING CODE
  21635  20505         LDA    0,X20     ;YES, CHECK IF AUTO-INDEX
  21636  24505         LDA    1,X40
  21637 162033         SLS    3,0       ;DOING @ USING 20-37 ?
  21640 166033         SLS    3,1
  21641    402         JMP    NOAX      ;  NO, NOT AUTO-INDEX
  21642    436         JMP    VIOLATION

  21643  35400 NOAX:  LDA    3,0,3     ;  YES

; NOW A3 CONTAINS THE EFFECTIVE ADDRESS OF THE TRAPPED INSTRUCTION

  21644  30462 CKSMC: LDA    2,RETURN
  21645 156415         SNE    2,3       ;IS EA = THE VERY NEXT LOCATION ?
  21646    432         JMP    VIOLATION;  YES, VIOLATION !
  21647  24470         LDA    1,X4      ;NO
  21650 147000         ADD    2,1
  21651 166433         SLE    3,1       ;IS EA BEYOND DANGER ZONE ?
  21652  63077         HALT             ;  YES, HARDWARE ERROR !!
  21653 102400         SUB    0,0
  21654  40460         STA    0,XFLAG   ;INITIALIZE SKIP-FLAG TO 0
           21655 LOOP:                  ;CHECK IF THERE IS AN INTERVENING JMP OR JSR
  21655  21000         LDA    0,0,2
  21656  24460         LDA    1,X10K
  21657 106033         SLS    0,1       ;JMP OR JSR INSTRUCTION,
  21660  14454         DSZ    XFLAG     ;  AND SKIP-FLAG NOT = 1 ?
  21661    426         JMP    NOSMC     ;     YES, THEN THERE'S NO VIOLATION
  21662 106032         SGE    0,1       ;NO, IS IT AN ISZ/DSZ INSTRUCTION ?
  21663    404         JMP    NIDSZ     ;  NO
  21664 125120         MOVZL  1,1       ;MAYBE
  21665 106032         SGE    0,1       ;IS IT ?
  21666    405         JMP    SETSK     ;  YES, SET SKIP-FLAG = 1
  21667  24451 NIDSZ: LDA    1,X7      ;NO
  21670 101112         SSP    0,0       ;ALU INSTRUCTION,
  21671 107415         AND#   0,1,SNR   ;  AND POSSIBLE SKIP ?
  21672 126401         SUB    1,1,SKP   ;    NO, SET SKIP-FLAG = 0
  21673 126520 SETSK: SUBZL  1,1       ;    YES, SET SKIP-FLAG = 1
  21674  44440         STA    1,XFLAG
  21675 151400         INC    2,2       ;STEP TO NEXT INSTRUCTION
  21676 156414         SEQ    2,3       ;HAVE WE CHECKED UP TO THE MODIFIED INSTR. ?
  21677    756         JMP    LOOP      ;  NO, KEEP CHECKING
           21700 VIOLATION:             ;YES, THEN IT'S A VIOLATION
  21700  30426         LDA    2,RETURN
  21701 140400         NEG    2,0
  21702 100000         COM    0,0       ;A0 = LOCATION OF S-M CODE
  21703  25377         LDA    1,-1,2    ;A1 = SELF-MOD. INSTRUCTION
  21704  30425         LDA    2,AC2     ;A2 = AS WE FOUND IT
  21705  34425         LDA    3,AC3     ;A3 = AS WE FOUND IT
  21706  63077         HALT             ;<<--- OR INSERT CUSTOM PROCESSING HERE
```

```
           21707 NOSMC:                         ;NO SELF-MODIFYING CODE VIOLATION
    21707 102520         SUBZL   0,0
    21710  60777         DIAP    0,CPU          ;READ SAVED CPU STATUS
    21711  24430         LDA     1,X10
    21712 107400         AND     0,1            ;INDICATES INTERRUPTS WERE ENABLED
    21713  44421         STA     1,XFLAG        ;SAVE FOR LATER
    21714  20417         LDA     0,CARRY        ;RESTORE CARRY AND ACCUMULATORS
    21715 101200         MOVR    0,0
    21716  20411         LDA     0,AC0
    21717  24411         LDA     1,AC1
    21720  30411         LDA     2,AC2
    21721  34411         LDA     3,AC3
    21722  10412         ISZ     XFLAG          ;WERE INTERRUPTS ENABLED AT TRAP TIME ?
    21723  14411         DSZ     XFLAG
    21724  60177         INTEN                  ;    YES, THEN RE-ENABLE THEM
    21725   2401         JMP     @RETURN

    21726      0 RETURN:         0
    21727      0 AC0:            0
    21730      0 AC1:            0
    21731      0 AC2:            0
    21732      0 AC3:            0
    21733      0 CARRY:          0
    21734      0 XFLAG:          0
    21735   1400 X1400:          1400
    21736  10000 X10K:           10000

; THE FOLLOWING CONSTANTS ARE IN IRIS PAGE ZERO, AND ARE NOT NEEDED HERE
; IF THE TRAP HANDLER IS ASSEMBLED WITH IRIS PZ SOURCE.

    21737      4 X4:             4
    21740      7 X7:             7
    21741     10 X10:            10
    21742     20 X20:            20
    21743     40 X40:            40
    21744    200 X200:           200
    21745    377 X377:           377
    21746   2000 X2000:          2000

                 .EOT  ;MARK 12 PROHIBITED CODE TRAP HANDLER
```

Appendix A
CPU STATUS WORD

TABLE A-1. MARK 6/12 CPU STATUS WORD (1 OF 2)

Bit	Symbol	Name and Significance
0*	CY	Carry flag in saved CPU status word. Always 0 in current status. If set by software to a 1, along with MSK=0, will set 64KW-addressing mode and enable EIS (if implemented), ignoring all other bits in the NIOP instruction.
1	MSK	Mask. If set to a 0 by software in a NIOP, bits 2-14 will be ignored. If set to a 1, enable bits 2-14. Always set to 1 during a read.
2	AUT	Auto Restart flag. Set by hardware to a 1, in saved CPU status word, on power up, if front panel keywsitch is in the AUTO position. General purpose flag otherwise.
3	-	Reserved.
4	-	Reserved.
5	-	Reserved.
6	-	Reserved.
7	TSC	Trap Self-Modifying Code control bit. Set by software to a 1 to enable.
*Most significant bit		

TABLE A-1. MARK 6/12 CPU STATUS WORD (2 OF 2)

Bit	Symbol	Name and Significance
8	PEL	Front panel Parity Error Light control bit. Set to a 1/0 by software to turn parity error light on/off.
9	M12	MARK 12 Mode control bit. If set by software to a 1, along with 64K=1, will enable MARK 12 mode.
10	-	Reserved.
11	RNL	Run flag. Set to a 1/0 by hardware to indicate Run/Halt state. Also set by hardware to a 0, in saved CPU status word, at power up.
12	ION	Interrupts Enabled flag. Set by hardware only to indicate interrupts enabled.
13	DME	Data Channel Map Enable flag. Set by software to a 1 to enable data channel mapping.
14	-	Reserved for EIS enablement.
15	64K	64KW-addressing mode control bit. Must set to a 1 to enable M12 mode or TSC control bit.

COMMENT SHEET

MANUAL TITLE: MARK 6/12 CPU Self-Test/MANIP Manual

PUBLICATION NO. HM-0812-0064 REVISION A

FROM: NAME/COMPANY: ________________

BUSINESS ADDRESS: ________________

CITY/STATE/ZIP: ________________

COMMENTS: Your evaluation of this manual will be appreciated by POINT 4 Data Corporation. Notation of any errors, suggested additions or deletions, or general comments may be made below. Please include page number references where appropriate.

NO POSTAGE NECESSARY IF MAILED IN U.S.A.
Fold on Dotted Lines and Tape

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 1458 TUSTIN, CA

POSTAGE WILL BE PAID BY ADDRESSEE

POINT 4 Data Corporation
PUBLICATIONS DEPARTMENT
15442 Del Amo Avenue
Tustin, CA 92680

CUT ON THIS LINE