DataGeneral

MICROPROGRAM LISTING

LISTING

077-000040-00

PROGRAM

MAPPED OCTAL DEBUG TOOL FOR MICRONOVA

TAPE

076-000040-00

ABSTRACT

SPECIFICATIONS FOR PROGRAMMABLE ROM TYPES
DGC 100-001791
DGC 100-001792

```
 0001 MODT      MACRO REV 06.30          00:05:07 02/10/79
01
02
03
04              ;*****************************************************************
05              ;
06              ; NAME: MODT.SR                          PART NUMBER:    67-40
07              ;
08              ; DESCRIPTION:  MAPPED OCTAL DEBUG TOOL FOR MICRO NOVA 602
09              ;
10              ; REVISION HISTORY
11              ;
12              ;       REV.            DATE
13              ;       00              02/12/79
14              ;
15              ;
16              ;
17              ;
18              ; COPYRIGHT (C) DATA GENERAL CORPORATION, 1976, 1977,1978,1979
19              ; ALL RIGHTS RESERVED.
20              ;*****************************************************************
21
22
23
```

```
!0002 MODT
                        .TITL   MODT
02
03      000000          .DO     0
04
05
06              1.      MAPPED OCTAL DEBUG TOOL (MODT) AND AUTOMATIC PROGRAM
07                      LOAD (APL).
08
09                      DESCRIPTION
10
11                      THIS ODT IS USED WITH THE 602 CPU CHIP WHICH ALLOWS
12                      MICRONOVA SYSTEMS TO USE UP TO 64K OF MEMORY.
13                      BIT 0 OF THE ADDRESS REFERS TO USER(0XXXXX) OR MAPPED
14                      (1XXXXX) MEMORY AND DOES NOT INDICATE INDIRECTION.
15                      THIS ODT ROM IS PHYSICALLY LOCATED AT 177000-177777
16                      WITH RESERVED RAM LOCATIONS AT 100000-100017.
17                      THE ENTIRE MEMORY IS ACCESSIBLE FROM THE ODT
18                      INSTRUCTION EMULATION IS USED IN ODT TO HANDLE THE
19                      NONDELETING BREAKPOINT.  THIS ALLOWS "ONE STEP" AND
20                      "QUASI SINGLE STEP" COMMANDS TO BE USED TO DEBUG USER
21                      PROGRAMS. THE "O" AND "Q" COMMANDS ARE DESCRIBED BELOW
22                      WITH THEIR LIMITATIONS. THE "Q" COMMAND MAY BE USED
23                      TO SINGLE STEP PROGRAMS IN ROM.
24
25                      1A.     AFTER POWER UP THE JUMPER WORD REGISTER (LOC
26                              177776) IS EXAMINED, IF DEVICE CODE 77 IS
27                              INSERTED THEN ODT IS ENTERED.  ANY OTHER DEVICE
28                              CODE WILL CAUSE AN AUTOMATIC PROGRAM LOAD.
29
30                      1B.     UPON ENTERING ODT THE BP (BREAKPOINT) & NMR
31                              NON MASKABLE INTERRUPT REQUEST) VECTOR IS
32                              LOADED INTO ADDRESS 100001.
33                              NOTE: IF ALTERED, THIS VECTOR MUST BE RELOADED
34                                    FOR PROPER HANDLING OF BP'S & NMR'S.
35
36                      1C.     THE CONTENTS OF ADDRESS 100000 IS PRINTED ON
37                              THE TERMINAL.  DURING POWER UP THIS IS USELESS.
38                              WHEN THE  ODT IS ENTERED VIA A BP OR HALT THE
39                              OUTPUT IS THE ADDRESS PLUS ONE OF THAT INSTRUC-
40                              TION IF AN NMR WAS PERFORMED THE OUTPUT IS PC.
41
42                      NOTE:  IN CASE OF RESET THE INTERNAL CPU REGISTERS ARE
43                      DESTROYED, SO TO CONTINUE THEY MUST BE REINITIALIZED
44                      WITH CORRECT DATA.
45
46              1.1     CONVENTIONS AND SYMBOLS
47                      THE FOLLOWING CONVENTIONS ARE USED BY THE ODT:
48                      ?       PRESSING ANY ILLEGAL KEY CAUSES THE ODT TO RES-
49                              POND WITH A "?".
50                      !       ODT IS READY AND AT YOUR SERVICE.
```

1.2 COMMAND STRUCTURE

AN ODT COMMAND HAS THE FOLLOWING FORMAT:

[ARGUMENT][COMMAND]

AN ARGUMENT MAY BE ONE OF THE FOLLOWING:

EXP — AN OCTAL EXPRESSION CONSISTING OF OCTAL NUMBERS SEPARATED BY PLUS (+) OR MINUS (-) SIGNS. LEADING ZEROS NEED NOT BE TYPED.

ADR — AN ADDRESS IS THE SAME AS AN EXPRESSION. BIT 0 DETERMINES USER (0) OR MAPPED (1) MEMORY.

A COMMAND IS A SINGLE TELETYPE CHARACTER

1.3 ODT COMMANDS

THE LOCATIONS THAT CAN BE EXAMINED AND MODIFIED BY THE USER ARE CALLED CELLS. THESE CELLS ARE OF TWO TYPE: INTERNAL CPU CELLS AND MEMORY LOCATIONS.

1.3.1 OPENING INTERNAL CELLS

THE COMMAND TO OPEN ONE OF THE INTERNAL REGISTERS IS OF THE FORM "NA" WHERE N IS ANY OCTAL EXPRESSION BETWEEN 0 AND 15.

0-3	TO LOOK AT ACCUMULATORS 0-3
4	PC OF BREAKPOINT OR HALT INSTRUCTION OR PC-1 IF NMR.
5	STACK POINTER
6	FRAME POINTER
7	CPU AND TTO STATUS
BIT	INTERPRETATION
15	1 IF TTO DONE WAS SET, 0 OTHERWISE
14	1 IF INTERRUPTS WERE ENABLED, 0 OTHERWISE
13	STATUS OF THE CARRY BIT
10	INSTRUCTION OF SET BREAKPOINT
11	ADDRESS OF THE LOCATION WHERE BREAKPOINT IS SET
12-15	TEMPORARY REGISTERS.

OTHER COMMANDS TO OPEN CELLS ARE:

ADR/	OPEN THE CELL AND PRINT ITS CONTENTS
./	OPEN THE CELL CURRENTLY POINTED BY THE POINTER AND PRINT ITS CONTENTS.
.+ADR/	ADD ADR TO THE POINTER, OPEN THE CELL AND PRINT ITS CONTENTS.
.-ADR/	SUBTRACT ADR FROM THE POINTER, OPEN THE CELL AND PRINT ITS CONTENTS.
"NL/LF"	THE NEW LINE OR LINE FEED KEY IS USED TO CLOSE THE OPEN CELL WITH OR WITHOUT MODIFICATION.
"CR"	RETURN IS USED TO CLOSE THE OPEN CELL WITH OR WITHOUT MODIFICATION AND TO OPEN THE SUCCEEDING CELL.
/	CLOSE THE OPEN CELL WITHOUT MODIFICATION, AND OPEN THE CELL POINTED BY ITS CONTENTS.
+ADR/	CLOSE THE OPEN CELL WITHOUT MODIFICATION, AND OPEN THE CELL POINTED BY ITS CONTENTS + ADR.
-ADR/	CLOSE THE OPEN CELL WITHOUT MODIFICATION, AND OPEN THE CELL POINTED BY ITS CONTENTS - ADR.

1.3.2 MODIFICATION OF A CELL

ONCE A CELL HAS BEEN OPENED ITS CONTENTS CAN BE MODIFIED BY TYPING THE NEW VALUE THE CELL IS TO CONTAIN IN THE FORM OF AN OCTAL EXPRESSION FOLLOWED BY "CR" OR "LF/NL". IF A + OR - IS TYPED AS THE FIRST CHARACTER OF THE EXPRESSION THEN THE VALUE OF THE EXPRESSION IS ADDED TO OR SUBTRACTED FROM THE OLD CONTENTS OF THE CELL. ADDRESS ITSELF OR AN EXPRESSION RELATIVE TO THE ADDRESS CAN BE DEPOSITED BY TYPING A "." OR ".+/-OCTAL EXPRESSION".

1.3.3 OTHER ODT COMMANDS

RUBOUT — THIS KEY IS USED TO DELETE ERRONEOUSLY TYPED DIGITS. EACH TIME THIS KEY IS PRESSED THE RIGHT MOST DIGIT IS DELETED AND ECHOED ON THE TERMINAL. IF THE RUBOUT KEY IS HIT IMMEDIATELY AFTER OPENING A CELL THE RIGHT MOST DIGIT OF ITS CONTENTS IS DELETED. THIS CELL CAN NOW BE MODIFIED AS IF ITS CONTENTS WERE TYPED-IN BY THE USER JUST BEFORE THE RUBOUT KEY WAS PRESSED.

DVCL — PERFORM A PROGAM LOAD WITH DVC EQUAL TO OCTAL CODE FOR I/O DEVICE
NOTE: DEVICE CODE 77 EQUALS CASSETTE

K — KILL THE STRING TYPED SO FAR. THE ODT RESPONDS WITH A "?" AND THE OPEN CELL IS CLOSED WITHOUT MODIFICATION.

ADRB — WILL SET A BREAKPOINT AT LOCATION ADR. IF A BREAKPOINT IS ALREADY SET THE ODT RESPONDS WITH A "?".

B — LIST ADDRESS OF EXISTING BREAKPOINT. IF BUFFER=0, THEN BREAKPOINT NOT SET. NO BREAKPOINTS MAY BE SET AT LOCATIONS 0 AND 100000.

D — DELETE EXISTING BREAKPOINT.

ADRR — START EXECUTING THE PROGRAM AT ADR.

R — START EXECUTING THE PROGRAM AT LOCATION (0)

P — RESTART THE EXECUTION OF THE PROGRAM AT LOCATION POINTED BY 4A (CURRENT PC) IF ODT ENTERED BY BREAKPOINT, OTHERWISE, RESTART AT PC+1

O — COMPUTE AND SET BP AT THE NEXT PC, EXECUTE PRESENT BP INSTRUCTION AND DELETE BP.

Q — OPERATION IS THE SAME AS THE "O" COMMAND EXCEPT INTERRUPTS ARE NEVER ENABLED BECAUSE ODT IS NEVER EXITED.

NOTE: EMULATION OF ANY STACK OPERATION IS NOT ALLOWED AND ODT WILL RESPOND WITH A "?". ODT MUST BE ENTERED BY A BP OR 4A MUST MATCH ACTUAL BP ADDRESS (NOT ADDRESS PRINTED BY MODT) SET FOR PROPER OPERATION OF "P","O",AND "Q" COMMANDS.

.ENDC

```
 0005 MODT
                            .TITL   MODT
02
03          061001          .DIAC   MTSP    =061001
04          060001          .DIAC   MTFP    =060001
05          061201          .DIAC   MFSP    =061201
06          060201          .DIAC   MFFP    =060201
07          060377          .DIAC   MAP     =060377
08
09          000002 SAC0=    2               ;SAVE AC0
10          000003 SAC1=    SAC0+1          ;SAVE AC1
11          000004 SAC2=    SAC1+1          ;SAVE AC2
12          000005 SAC3=    SAC2+1          ;SAVE AC3
13          000006 SAVPC=   SAC3+1          ;SAVE PC
14          000007 SAVSP=   SAVPC+1         ;SAVE STACK POINTER
15          000010 SAVFP=   SAVSP+1         ;SAVE FRAME POINTER
16          000011 STATS=   SAVFP+1         ;SAVE CARRY AND FLAGS
17          000012 BPINS=   STATS+1         ;BP INSTRUCTION
18          000013 BPADR=   BPINS+1         ;BP ADDRESS
19          000014 EXIT=    BPADR+1         ;TEMPORARY REGISTER
20          000015 PCREL=   EXIT+1          ;TEMPORARY REGISTER
21          000016 MPFLG=   PCREL+1         ;MAPPED MEMORY FLAG
22          000017 EXINS=   MPFLG+1         ;TEMPORARY REGISTER
23
24          077777 TOP=     77777
```

```
 0006 MODT
01          0010                .RDX0   16
02          7E00        .LOC    TOP-777
03
04                  ;START OF OCTAL DEBUG TOOL (O.D.T.)
05
06
07 07E00    0122            JMP     RSTRT           ;RESTART AFTER PWR FAIL
08 07E01    4002            STA     0,SAC0          ;SAVE AC0
09 07E02    5805            STA     3,SAC3          ;SAVE AC3
10 07E03    212E            LDA     0,DCMSK         ;GET DEVICE CODE MASK
11 07E04    FC00            ADC     3,3             ;AC3=177777
12 07E05    38FF            LDA     3,-1,3          ;GET DEVICE CODE IN 77776'
13 07E06    E504            SUB     3,0,SZR         ;DEV CODE 77 SKIP INTO ODT
14 07E07    011E            JMP     APL             ;ANYTHING ELSE AUTO PROGRAM LOAD
15 07E08    6408            DOB     0,TTI           ;CLEAR CASSETTE
16 07E09    0920            JSR     @PHDL           ;STORE BP HANDLER ADDRESS
17 07E0A    5805 PENT:      STA     3,SAC3          ;SAVE AC3
18 07E0B    4002            STA     0,SAC0          ;SAVE AC0
19 07E0C    4803            STA     1,SAC1          ;SAVE AC1
20 07E0D    2800            LDA     1,0             ;AC1=CURRENT PC+1
21 07E0E    A100            NEG     1,0             ;DECREMENT TO ACTUAL
22 07E0F    8000            COM     0,0             ;PROGRAM COUNTER
23 07E10    4006            STA     0,SAVPC         ;SAVE CURRENT PC
24 07E11    5004            STA     2,SAC2          ;SAVE AC2
25 07E12    8570            SUBCL   0,0             ;SAVE THE CARRY
26 07E13    673F            SKPBN   CPU             ;SKIP IF INTERRUPTS ARE ENABLED
27 07E14    8251            MOVZL   0,0,SKP         ;DISABLED INTERRUPTS=0
28 07E15    8260            MOVOL   0,0             ;ENABLED INTERRUPTS=1
29 07E16    60BF            INTDS                   ;DISABLE INTERRUPTS
30 07E17    6789            SKPDN   TTO             ;SAVE THE TTO STATUS
31 07E18    8251            MOVZL   0,0,SKP         ;DEVICE OFF STATUS
32 07E19    8260            MOVOL   0,0             ;DEVICE ON STATUS
33 07E1A    4009            STA     0,STATS         ;AC0=CARRY,INTERRUPT,+TTO STATUS
34 07E1B    6281            MFSP    0               ;READ THE STACK POINTER
35 07E1C    4007            STA     0,SAVSP         ;AND SAVE IT
36 07E1D    6081            MFFP    0               ;READ THE FRAME POINTER
37 07E1E    4008            STA     0,SAVFP         ;AND SAVE IT
38 07E1F    60CA            NIOP    TTI             ;CLEAR BREAK LATCH
39 07E20    0008            JSR     @IPOCT          ;TYPE BP, HALT, OR NMR=1
40 07E21    050D            JMP     @ICLTTY         ;CLEAR INPUT DONE FLAG
41
42 07E22    68FF RSTRT:     MAP     1               ;MAP PEND
43 07E23    70FF            MAP     2               ;RESET
44 07E24    0400            JMP     @0              ;RETURN AFTER PWR FAIL
45
46 07E25    FED0 APL:       ADDZS   3,3             ;HIGH SPEED DEV IN CARRY
47 07E26    EA80            MOVR    3,1             ;CODE RIGHT BYTE, HSD IN BIT 0
48 07E27    D500            SUB     2,2
49 07E28    0508            JMP     @IJPLD          ;AUTOMATIC PROGRAM LOAD
50
51 07E29    5801 BPHDL:     STA     3,1             ;STORE PENT IN 1'
52 07E2A    0302            JMP     2,3             ;PRINT HALT LOCATION
53
54 07E2B    7F58 IPOCT:     POCT
55 07E2C    7F88 IPCMND:    PCMND
56 07E2D    0050 N120:      120
57 07E2E    7EED ICLTTY:    CLTTY
58 07E2F    0027 C47:       47
59 07E30    7FD2 IJPLD:     JPLD
60 07E31    3F01 DCMSK:     37401
```

```
 0007 MODT
01 07E32   3A09 STRUN:  LDA     3,STATS         ;MODIFY BIT STATUS
02 07E33   FA50         MOVZL   3,3             ;CLEAR BIT 0 OF STATUS
03 07E34   AA50         MOVZL   1,1             ;INSERT NEW MAP BIT
04 07E35   FAA0         MOVR    3,3             ;CLEAR BIT 0 OF STARTING ADDRESS
05 07E36   AAA0         MOVR    1,1             ;STORE NEW STATUS AND
06 07E37   5A09         STA     3,STATS         ;START EXIT PROCEDURE
07 07E38   0105         JMP     RUN+1           ;RETURN
08
09 07E39   2806         LDA     1,SAVPC         ;AC1=NEW PC AFTER BP
10 07E3A   AA00         INC     1,1             ;AC1=PC+1
11 07E3B   AA01         INC     1,1,SKP         ;AC1=PC+2
12 07E3C   01F6 RUN:    JMP     STRUN           ;MODIFY MAP BIT IF NEEDED
13 07E3D   4806         STA     1,SAVPC         ;RUN OR PROCEED COMMAND
14 07E3E   400C         STA     0,EXIT          ;EXECUTE INSTRUCTION
15 07E3F   2007         LDA     0,SAVSP
16 07E40   6201         MTSP    0               ;RESTORE STACK POINTER
17 07E41   2008         LDA     0,SAVFP
18 07E42   6001         MTFP    0               ;RESTORE FRAME POINTER
19 07E43   2809         LDA     1,STATS         ;RESTORE STATUS FLAGS
20 07E44   AA4A         MOVL#   1,1,SZC         ;IF MAP BIT = 1
21 07E45   8400         ADC     0,0             ;LOAD -1 FOR EXIT ROUTINE
22 07E46   4009         STA     0,STATS         ;LOAD BIT0=0 OTHERWISE
23 07E47   AA93         MOVZR   1,1,SNC         ;IS DONE BIT LEFT SET
24 07E48   6089         NIOC    TTO             ;NO
25 07E49   AA92         MOVZR   1,1,SZC         ;SHOULD WE ENABLE INTERRUPTS?
26 07E4A   607F         INTEN                   ;YES
27 07E4B   AA90         MOVZR   1,1             ;RESTORE CARRY
28 07E4C   2002         LDA     0,SAC0          ;RESTORE AC0
29 07E4D   2803         LDA     1,SAC1          ;RESTORE AC1
30 07E4E   3004         LDA     2,SAC2          ;RESTORE AC2
31 07E4F   3805         LDA     3,SAC3          ;RESTORE AC3
32 07E50   000C         JMP     EXIT            ;EXECUTE INSTRUCTION
33
34 07E51   8007 TRPM:   100007
35
36 07E52   39D8 OTHCM:  LDA     3,N120          ;AC3 = 120
37 07E53   F320         INCO    3,2             ;CARRY SET, AC2 = 121
38 07E54   C40C         ADC#    2,0,SZR         ;SKIP IF AC0 = 122 "R"
39 07E55   9435         ADCC    0,2,SNR         ;DON'T SKIP IF AC0 = 120 "P"
40 07E56   0104         JMP     .+4             ;CARRY IS SET FOR BOTH COMMANDS
41 07E57   F210         MOVZ    3,2             ;CLEAR CARRY, AC2 = 120
42 07E58   C40C         ADC#    2,0,SZR         ;SKIP IF AC0 = 121 "Q"
43 07E59   9435         ADCC    0,2,SNR         ;DON'T SKIP IF AC0 = 117 "O"
44 07E5A   0921         JSR     PRCD            ;RELOCATE EXIT ROUTINE
45 07E5B   05D1         JMP     @IPCMND         ;DECODE OTHER COMMANDS
46 07E5C   1806 OTODT:  DSZ     SAVPC           ;DECREMENT PC IF NO SKIP
47 07E5D   6000 C60K:   NIO     0               ;60000 SEPARATES MR & ALC INSTR
48 07E5E   0400         JMP     @0              ;RETURN TO ODT TO COMPLETE EXIT
49 07E5F   AA00 YSSKP:  MOV     1,1             ;END OF SUBROUTINE FLAG
50
```

```
:0008 MODT
01 07E60   7E3A IBRKPT: BRKPT
02 07E61   E07F STCKM:  160177
03 07E62   00FF C377:   377
04 07E63   7E61 WHAT1:  WHAT
05
06 07E64   4000 OEX:    STA     0,0             ;SAVE AC0
07 07E65   8400         ADC     0,0             ;THIS WILL FLAG
08 07E66   4008         STA     0,SAVFP         ;RE-ENTRY INTO ODT
09 07E67   2000         LDA     0,0             ;OR JMP TO BP
10 07E68   4002 OEX:    STA     0,SAC0          ;SAVE AC0
11 07E69   4803         STA     1,SAC1          ;SAVE AC1
12 07E6A   5004         STA     2,SAC2          ;SAVE AC2
13 07E6B   5805         STA     3,SAC3          ;SAVE AC3
14 07E6C   2806         LDA     1,SAVPC         ;NEXT ADDRESS TO GET BP
15 07E6D   0978         JSR     DELBP+2         ;DELETE OLD BP
16 07E6E   0DF2         JSR     @IBRKPT         ;UPDATE BREAKPOINT
17 07E6F   D300         INC     2,2             ;INCREMENT NEW BP ADDRESS
18 07E70   5000         STA     2,0             ;BREAKPOINT ADDRESS IN 0'
19 07E71   2002         LDA     0,SAC0          ;RESTORE AC0
20 07E72   2803         LDA     1,SAC1          ;RESTORE AC1
21 07E73   3004         LDA     2,SAC2          ;RESTORE AC2
22 07E74   3805         LDA     3,SAC3          ;RESTORE AC3
23 07E75   1008         ISZ     SAVFP           ;SKIP GOES THROUGH BREAKPOINT
24 07E76   0401         JMP     @1              ;STRAIGHT TO ODT AGAIN
25 07E77   1009 PREX:   ISZ     STATS           ;SKIP TO STAY IN MAPPED MEMORY
26 07E78   68FF         MAP     1               ;GO TO USERS MEMORY
27 07E79   70FF         MAP     2               ;RESET INTEL
28 07E7A   0406         JMP     @SAVPC          ;JUMP BACK TO USERS PROGRAM
29
```

```
10009 MODT
01 07E7A   FB00 PRCD:   INC     3,3          ;INCREMENT TO OTODT ADDRESS
02 07E7C   5800         STA     3,0          ;STORE ADDRESS OF EXIT ROUTINE
03 07E7D   FD30         SUBC    3,3          ;SAVE CARRY
04 07E7E   D206         MOV     2,2,SEZ      ;SKIP FIRST INSTRUCTION
05 07E7F   1000         ISZ     0            ;IF "R"
06 07E80   2400         LDA     0,@0         ;RELOCATE ROUTINE
07 07E81   4300         STA     0,PCREL,3    ;START AT PCREL
08 07E82   FB00         INC     3,3
09 07E83   8248         MOVL#   0,0,SNC      ;END OF EXIT ROUTINE = 125000
10 07E84   01FB         JMP     .-5          ;KEEP INCREMENTING
11 07E85   D203         MOV     2,2,SNC      ;CARRY = 1 = "P" OR "R"
12 07E86   0104         JMP     .+4
13 07E87   39F0         LDA     3,PREX       ;USE THIS RETURN TO
14 07E88   D100         NEG     2,2          ;COMPLETE EXIT RUN
15 07E89   0104         JMP     .+4          ;UPDATE RETURN ADDRESS
16 07E8A   39DA         LDA     3,OEX        ;THIS RETURN FOR "O"
17 07E8B   D204         MOV     2,2,SZR
18 07E8C   39DC         LDA     3,QEX        ;THIS RETURN FOR "Q"
19 07E8D   1000         ISZ     0            ;INCREMENT ADDRESS 0
20 07E8E   3400         LDA     2,@0         ;UNTIL WE FIND THE
21 07E8F   DD0C         SUB#    2,3,SZR      ;THE CORRECT RETURN
22 07E90   01FD         JMP     .-3          ;NOT YET
23 07E91   8202         MOV     0,0,SZC      ;SKIP IF NOT "R"
24 07E92   01AA         JMP     RUN          ;START RUN PROCEDURE
25 07E93   2808         LDA     1,BPADR      ;GET A BP ADDRESS
26 07E94   3806         LDA     3,SAVPC      ;GET CURRENT PC
27 07E95   BD0C         SUB#    1,3,SZR      ;CHECK ADDRESS MATCHES
28 07E96   01A3         JMP     RUN-3        ;NOT BP START +1
29 07E97   200A         LDA     0,BPINS      ;GET THE INSTRUCTION
30 07E98   3989         LDA     3,TRPM       ;AC3=100007, TRAP MASK
31 07E99   F300         INC     3,2          ;AC2=100010
32 07E9A   9F5D         ANDZL#  0,3,SNR      ;SKIP IF TRAP INSTRUCTION
33 07E9B   975F         ANDZL#  0,2,SBN
34 07E9C   0106         JMP     NOTRP        ;ADDRESS OF TRAP HANDLER
35 07E9D   3192         LDA     2,C47
36 07E9E   68FF         MAP     1            ;STORE TRAP ADDRESS IN 46
37 07E9F   4AFF         STA     1,-1,2       ;GET FINAL DESTINATION
38 07EA0   0937         JSR     TRPEN        ;EXIT TRAP SIMILAR TO JMP
39 07EA1   0116         JMP     IJMP+1       ;AC3=60000
40 07EA2   39BB NOTRP:  LDA     3,C60K       ;AC2 = 160177
41 07EA3   318E         LDA     2,STCKM      ;ISOLATE STACK INSTRUCTION
42 07EA4   9700         AND     0,2          ;SKIP IF NOT STACK OPERATION
43 07EA5   F40D         ADC#    3,2,SNR      ;DON'T ALLOW STACK OPERRATION
44 07EA6   058D         JMP     @WHAT1       ;AC0 = INSTRUCTION
45 07EA7   9000         COM     0,2          ;IS IT INDEX INSTRUCTION
46 07EA8   E51A         SUBZ#   3,0,SZC
```

```
10010 MODT
01 07EA9   0190 JMPO:   JMP     RUN-3        ;NO, EXECUTE INSTRUCTION
02 07EAA   2934         LDA     1,C4         ;AC1 = 40000
03 07EAB   CF95         ANDZR   2,1,SNR      ;IS IT STA
04 07EAC   015B         JMP     ISTA         ;YES
05 07EAD   CF95         ANDZR   2,1,SNR      ;IS IT LDA
06 07EAE   0151         JMP     ILDA         ;YES
07 07EAF   3931         LDA     3,C14K       ;NO: AC3=14000
08 07EB0   E51A         SUBZ#   3,0,SZC      ;IS IT DSZ
09 07EB1   013E         JMP     IDSZ         ;YES
10 07EB2   CF95         ANDZR   2,1,SNR      ;IS IT ISZ
11 07EB3   0142         JMP     IISZ         ;YES
12 07EB4   CF95         ANDZR   2,1,SNR      ;IS IT JSR
13 07EB5   0106         JMP     IJSR         ;YES
14 07EB6   090A IJMP:   JSR     EFADD        ;AC2=EFFECTIVE ADDRESS
15 07EB7   D140         NEGL    2,2          ;MAKE BIT 0 = 0 AND
16 07EB8   D090         COMZR   2,2          ;DECREMENT NEW PC
17 07EB9   5006         STA     2,SAVPC      ;STORE NEW PC
18 07EBA   014D         JMP     ISTA+3       ;GET OUT OF O.D.T.
19
20 07EBB   0905 IJSR:   JSR     EFADD        ;AC2=EFFECTIVE ADDRESS
21 07EBC   2806         LDA     1,SAVPC      ;GET INSTR PC
22 07EBD   AB00         INC     1,1          ;UPDATE FOR AC3
23 07EBE   4805         STA     1,SAC3       ;AFTER JSR
24 07EBF   01F8         JMP     IJMP+1       ;GET OUT OF ODT
25
26 07EC0   29A2 EFADD:  LDA     1,C377       ;AC1=377
27 07EC1   9200         MOV     0,2          ;INSTRUCTION IN AC2
28 07EC2   B700         AND     1,2          ;AC2 = DISPLACEMENT
29 07EC3   2918         LDA     1,DRCT       ;AC1 = 1400
30 07EC4   A7C5         ANDS    1,0,SNR      ;SAVE FULL DISPLACEMENT
31 07EC5   010E         JMP     DIIND        ;IF PAGE 0
32 07EC6   299C         LDA     1,C377
33 07EC7   AB90         INCZR   1,1          ;AC1 = 200
34 07EC8   CF54         ANDZL   2,1,SZR      ;NEG OR POS DISPLACEMENT?
35 07EC9   B500         SUB     1,2          ;NEGATIVE
36 07ECA   8296         MOVZR   0,0,SEZ      ;AC0 = 1,2,OR3
37 07ECB   0106         JMP     AC3RL        ;AC3 RELATIVE
38 07ECC   2804         LDA     1,SAC2       ;LOAD AC2 IT MUST
39 07ECD   8203         MOV     0,0,SNC      ;ONLY 1 OR 2 LEFT
40 07ECE   0104         JMP     AC3RL+1      ;AC2 RELATIVE
41 07ECF   2806         LDA     1,SAVPC      ;AC1=CURRENT PC
42 07ED0   0102         JMP     .+2          ;PC RELATIVE
43 07ED1   2805 AC3RL:  LDA     1,SAC3       ;AC1=AC3
44 07ED2   B600         ADD     1,2          ;AC2=DISPLACEMENT
45 07ED3   210C DIIND:  LDA     0,C2000      ;AC0=2000
46 07ED4   280A         LDA     1,BPINS      ;AC1=INSTRUCTION
47 07ED5   8F05         AND     0,1,SNR      ;@BIT
48 07ED6   0300         JMP     0,3          ;DIRECT
```

```
!0011 MODT
01 07ED7   0143 TRPEN:  JMP     AID             ;CHECK FOR AUTO INC/DEC
02 07ED8   68FF         MAP     1               ;GO TO USERS MEMORY
03 07ED9   3200         LDA     2,0,2           ;AC2=INDIRECT ADDRESS
04 07EDA   D248         MOVL#   2,2,SNC         ;IS IT DIRECT OR INDIRECT
05 07EDB   0300 DRCT:   JMP     0,3             ;DIRECT
06 07EDC   D6A0         ADDOR   2,2             ;CLEAR @ BIT
07 07EDD   01FA         JMP     TRPEN           ;GET NEW ADDRESS
08
09 07EDE   4000 C4:     40000
10 07EDF   0400 C2000:  2000
11 07EE0   1A00 C14K:   14000
12 07EE1   7E52 IOTHCM: OTHCM
13 07EE2   60FF BPINT:  MAP     0
14
15 07EE3   FBA4 DELBP:  INCOR   3,3,SZR         ;SKIP IF IT IS A "D"
16 07EE4   05FD         JMP     @IOTHCM         ;TO DELETE BREAKPOINT
17 07EE5   200A         LDA     0,BPINS         ;AC1 = BP INSTRUCTION
18 07EE6   3008         LDA     2,BPADR         ;AC2 = ADDRESS OF BP
19 07EE7   68FF         MAP     1               ;GO TO USERS MEMORY
20 07EE8   4200         STA     0,0,2           ;RESTORE INST. AT BP
21 07EE9   A530         SUBC    0,0             ;AC0 = 0
22 07EEA   400B         STA     0,BPADR         ;DELETE BP IN BUFFER
23 07EEB   FA04         MOV     3,3,SZR
24 07EEC   0300         JMP     0,3
25 07EED   608A CLTTY:  NIOC    TTI             ;CLEAR TTY INPUT DONE FLAG
26 07EEE   0160         JMP     WAIT1           ;DO CR LF AND PERFORM
27                              ;OTHER COMMANDS
28 07EEF   09D1 IDSZ:   JSR     EFADD           ;AC2=EFFECTIVE ADDRESS
29 07EF0   68FF         MAP     1               ;GO TO USERS MEMORY
30 07EF1   2A00         LDA     1,0,2           ;AC1=CONTENTS
31 07EF2   8550         SUBZL   0,0             ;AC0=1
32 07EF3   8D00         SUB     0,1             ;DECREMENT CONTENTS
33 07EF4   0105         JMP     .+5             ;NO SKIP
34 07EF5   09CB IISZ:   JSR     EFADD           ;AC2=EFFECTIVE ADDRESS
35 07EF6   68FF         MAP     1               ;GO TO USERS MEMORY
36 07EF7   2A00         LDA     1,0,2           ;AC1=CONTENTS
37 07EF8   AB00         INC     1,1             ;SKIP IF RESULT=0
38 07EF9   251F         LDA     0,@IYSSKP       ;NO SKIP =MOV 1,1
39 07EFA   AA05         MOV     1,1,SNR         ;SHOULD WE ALTER ODT RETURN?
40 07EFB   8300         INC     0,0             ;SKIP =MOV 1,1 SKP
41 07EFC   68FF         MAP     1               ;GO TO USERS MEMORY
42 07EFD   4A00         STA     1,0,2           ;RESTORE IN MEMORY
43 07EFE   01AB JMP01:  JMP     JMP0            ;GET OUT OF O.D.T.
```

```
!0012 MODT
01 07EFF   090A ILDA:   JSR     DOALL           ;DETERMINE EFF. ADDRESS
02 07F00   68FF         MAP     1               ;GO TO USERS MEMORY
03 07F01   2A00         LDA     1,0,2           ;AC1=CONTENTS OF EFF. ADDRESS
04 07F02   4B02         STA     1,SAC0,3        ;LOAD UNKOWN AC
05 07F03   0104         JMP     .+4             ;GET OUT OF O.D.T.
06 07F04   0905 ISTA:   JSR     DOALL           ;DETERMINE EFF. ADDRESS,
07 07F05   68FF         MAP     1               ;AC AND CONTENTS OF AC
08 07F06   4A00         STA     1,0,2           ;STORE AC IN USERS MEMORY
09 07F07   2511         LDA     0,@IYSSKP       ;PC + 1
10 07F08   01F6         JMP     JMP01           ;GET OUT OF O.D.T.
11
12 07F09   580C DOALL:  STA     3,EXIT          ;SAVE RETURN ADDRESS
13 07F0A   0986         JSR     EFADD           ;AC2 = EFF. ADDRESS
14 07F0B   380A         LDA     3,BPINS         ;AC3 = INSTRUCTION
15 07F0C   29CF         LDA     1,DRCT          ;AC1 = 1400
16 07F0D   FA80         MOVR    3,3             ;SHIFT RIGHT
17 07F0E   FA80         MOVR    3,3             ;SHIFT RIGHT
18 07F0F   FA80         MOVR    3,3             ;SHIFT RIGHT
19 07F10   8FC0         ANDS    1,3             ;AC3 = AC
20 07F11   2806         LDA     1,SAVPC         ;IS THE ORIGINAL PC THE
21 07F12   850C         SUB#    1,2,SZR         ;SAME AS THE NEW PC?
22 07F13   0103         JMP     .+3             ;NO
23 07F14   3158         LDA     2,N12           ;THEN LOAD OR STORE BPINS
24 07F15   100C         ISZ     EXIT            ;RETURN TO SKIP MAP PEND INSTR
25 07F16   2802         LDA     1,SAC0,3        ;AC1 = CONTENTS OF AC
26 07F17   040C         JMP     @EXIT           ;RETURN
27 07F18   7E5F IYSSKP: YSSKP
28 07F19   0041 N101:   101
29
30 07F1A   2954 AID:    LDA     1,N41
31 07F1B   CC1B         ADCZ#   2,1,SNC         ;SKIP IF AC2 < 41
32 07F1C   01BC         JMP     TRPEN+1         ;NOT AUTO INC/DEC
33 07F1D   CA60         MOVOL   2,1             ;AC1 NEVER = 0
34 07F1E   AEC0         ADDS    1,1             ;SHIFT BITS 11 & 12
35 07F1F   AE53         ADDZL   1,1,SNC         ;TO CARRY & 0
36 07F20   0188         JMP     TRPEN+1         ;< 20
37 07F21   68FF         MAP     1               ;AND SKIP IF ON
38 07F22   2200         LDA     0,0,2           ;GET ADDRESS TO BE MODIFIED
39 07F23   68FF         MAP     1               ;IF NECESSARY
40 07F24   AA42         MOVL    1,1,SZC         ;BIT 0 DETERMINES
41 07F25   8101         NEG     0,0,SKP         ;AUTO DECREMENT OR
42 07F26   8301         INC     0,0,SKP         ;AUTO INCREMENT LOCATION
43 07F27   8000         COM     0,0             ;FINISH DECREMENT
44 07F28   4200         STA     0,0,2           ;RESTORE MODIFIED ADDRESS
45 07F29   9200         MOV     0,2             ;AC2 = NEW ADDRESS
46 07F2A   0180         JMP     DRCT+1
47
```

```
10013 MODT
01 07F25   100F DIGIT:  ISZ     EXINS
02 07F2C   AA51         MOVZL   1,1,SKP
03 07F20   AD20 N40:    SUBO    1,1
04 07F2E   AE50         ADDZL   1,1             ;SHIFT AC1 TO LEFT BY 3 PLACES
05 07F2F   EC00         ADC     3,1             ;AC1=OCTAL WORD BEING INPUTED BY
06                                              ;THE OPERATOR
07 07F30   F860         COMOL   3,3             ;AC3=1 IF INPUT=0
08 07F31   580F         STA     3,EXINS         ;SAVE AC3
09 07F32   0922         JSR     TPCHR
10 07F33   0921 DCRST:  JSR     TPCHR           ;ECHO EVERY OTHER CHARACTER
11 07F34   0155         JMP     OPRIN
12 07F35   39E4         LDA     3,N101          ;AC3=101
13 07F36   9D05         SUB     0,3,SNR         ;IF AN "A" HAS BEEN TYPED
14 07F37   013A         JMP     ACC             ;SERVICE ACCUMULATOR ROUTINE
15 07F38   FBA4         INCOR   3,3,SZR         ;AC3=0 IF "B"
16 07F39   01AA         JMP     DELBP           ;KEEP DECODING
17 07F3A   B240 BRKPT:  MOVL    1,2             ;ADDRESS 0 IN AC1 THEN
18 07F3B   0295         MOVZR   2,2,SNR         ;NO MAP BP
19 07F3C   010E         JMP     LSTBP           ;LIST BREAKPOINTS
20 07F3D   2008         LDA     0,BPADR         ;IF A PREVIOUS BP
21 07F3F   8204         MOV     0,0,SZR         ;WAS ALREADY SET THEN
22 07F3F   0142         JMP     WHAT            ;DON'T ALLOW NEW BP
23 07F40   68FF         MAP     1               ;MAP CHANGE PENDING
24 07F41   2200         LDA     0,0,2           ;DO NOT ALLOW A
25 07F42   400A         STA     0,BPINS         ;STORE IN INSTRUCTION BUFFER
26 07F43   219F         LDA     0,BPINT         ;GET BREAKPOINT INSTR. (MAP 0)
27 07F44   68FF         MAP     1               ;STORE IT IN
28 07F45   4200         STA     0,0,2           ;PROPER MEMORY LOCATION
29 07F46   5008         STA     2,BPADR         ;STORE BP ADDRESS IN BUFFER
30 07F47   FA04         MOV     3,3,SZR         ;IF THIS IS A SUB-
31 07F48   0300         JMP     0,3             ;ROUTINE, THEN RETURN
32 07F49   01A4         JMP     CLTTY           ;CLEAR TTI
33 07F4A   0909 LSTBP:  JSR     TYSP            ;TYPE A SPACE
34 07F4B   002F N57:    57
35 07F4C   2808         LDA     1,BPADR         ;AC1=ADDRESS OF BP'S
36 07F4D   0908         JSR     POCT            ;PRINT OUT BP ADDRESS
37 07F4E   0135 WAIT1:  JMP     WAIT            ;YES, OUTPUT CR, LF
```

```
10014 MODT
01 07F4F   2300 PAC0:   LDA     0,0,3           ;AC0=CHARACTER TO BE PRINTED
02 07F50   AA90 SHIFT:  MOVZR   1,1
03 07F51   AA90         MOVZR   1,1
04 07F52   AA91         MOVZR   1,1,SKP
05 07F53   21DA TYSP:   LDA     0,N40           ;PREPARE TO TYPE A SPACE
06 07F54   6249 TPCHR:  DOAS    0,TTO           ;TYPE AC0
07 07F55   6749         SKPBZ   TTO             ;WAIT FOR THE PRINTER
08 07F56   01FF         JMP     .-1             ;KEEP ON WAITING
09 07F57   0301         JMP     1,3             ;RETURN TO AC3+1
10
11 07F58   580D POCT:   STA     3,PCREL         ;SAVE THE RETURN ADDRESS
12 07F59   480F         STA     1,EXINS         ;SAVE AC1
13 07F5A   B200         MOV     1,2             ;ANY # TO BE ECHOED
14 07F5B   AD91         SUBZR   1,1,SKP         ;AC1=100000
15 07F5C   B501         SUB     1,2,SKP
16 07F5D   21EE         LDA     0,N57           ;AC0=57
17 07F5E   8300         INC     0,0
18 07F5F   B52A FRMDGT: SUBO#   1,2,SZC         ;IF AC2 IS LESS THAN AC1 THEN
19                                              ;GO TO PRINT THE DIGIT
20 07F60   09F0 PRIDGT: JSR     SHIFT
21 07F61   01F8         JMP     .-5
22 07F62   AA04         MOV     1,1,SZR
23 07F63   01FA         JMP     FRMDGT-2
24 07F64   280F         LDA     1,EXINS         ;RESTORE AC1
25 07F65   09EE         JSR     TYSP            ;TYPE SPACES
26 07F66   007F N177:   177
27 07F67   040D         JMP     @PCREL          ;PCREL = RETURN ADDRESS
28
29
30 07F68   580D CRLF:   STA     3,PCREL         ;SAVE THE RETURN ADDRESS
31 07F69   09E6         JSR     PAC0            ;TYPE A "CR"
32 07F6A   000D N15:    15
33 07F6B   09E4         JSR     PAC0            ;TYPE A "LF"
34 07F6C   000A N12:    12
35 07F6D   09E2         JSR     PAC0            ;TYPE A "!"
36 07F6E   0021 N41:    41
37 07F6F   AD00         SUB     1,1             ;AC1 = 0
38 07F70   040D         JMP     @PCREL          ;PCREL = RETURN ADDRESS
39
40 07F71   09E2 ACC:    JSR     TYSP            ;TYPE A SPACE
41 07F72   0037 N67:    67
42 07F73   313F         LDA     2,ISAC0         ;AC2 = 100002
43 07F74   CE00         ADD     2,1             ;RESTORE ADDRESS
44 07F75   480E         STA     1,MPFLG         ;STORE ADDRESS OF CELL
45 07F76   0136         JMP     ACCLOC          ;INTERNAL CELL IS OPENED
```

```
!0015 MODT
01 07F77   39D4 CONTU:  LDA     3,N57           ;AC3 = 57
02 07F78   E505         SUB     3,0,SNR         ;SKIP IF AC0 IS NOT "/"
03 07F79   0131         JMP     OPNLOC
04 07F7A   280E         LDA     1,MPFLG         ;GET ORIGINAL ADDRESS
05 07F7B   8305         INC     0,0,SNR         ;SKIP IF IT IS NOT A "."
06 07F7C   0108         JMP     CNTWT           ;WAIT FOR MORE INPUT
07 07F7D   AD00         SUB     1,1             ;AC1=0
08 07F7E   9B44         INCOR   0,3,SZR         ;SKIP IF IT WAS A "-"
09 07F7F   E385         INCCR   3,0,SNR         ;SKIP IF IT WAS NOT A "+"
10 07F80   0107         JMP     OPRIN-2
11 07F81   09CE WHAT:   JSR     PAC0            ;TYPE A "?"
12 07F82   003F         77
13 07F83   09E5 WAIT:   JSR     CRLF
14 07F84   FD20 CNTWT:  SUB0    3,3             ;AC3=0
15 07F85   5800         STA     3,0             ;0 = 0
16 07F86   FC00         ADC     3,3             ;AC3 = 177777
17 07F87   580C         STA     3,EXIT          ;EXIT=177777
18 07F88   580F         STA     3,EXINS
19 07F89   6788 OPRIN:  SKPDN   TTI             ;WAIT FOR OPERATOR INPUT
20 07F8A   01FF         JMP     .-1
21 07F8B   6188         DIAC    0,TTI           ;INPUT A CHARACTER
22 07F8C   39DA         LDA     3,N177          ;AC3=177
23 07F8D   E700         AND     3,0             ;GET RID OF THE PARITY BIT
24 07F8E   9D05         SUB     0,3,SNR         ;SKIP IF AC0 IS NOT 177
25 07F8F   09C0         JSR     PAC0            ;ECHO A '' FOR EACH DIGIT
26                                              ;DELETED
27 07F90   FC5F         ADCZL#  3,3,SKP         ;RIGHT BYTE = 137; SKIP THE NEXT
28                                              ;INSTRUCTION
29 07F91   01F7         JMP     OPRIN-1
30 07F92   39E0         LDA     3,N67           ;AC3=67
31 07F93   9D2A         SUB0#   0,3,SZC         ;IF THE ASCII VALUE IS >
32 07F94   0104         JMP     WHERE           ;67 THEN GO TO "WHERE"
33 07F95   3986         LDA     3,N57           ;AC3 = 57
34 07F96   9D22         SUB0    0,3,SZC         ;SKIP IF AC0 IS MORE THAN 57
35 07F97   0194         JMP     DIGIT
36 07F98   100C WHERE:  ISZ     EXIT            ;SKIP IF THE PREVIOUS SIGN WAS
37                                              ;A "+"
38 07F99   A900         NEG     1,1             ;NEGATE IF IT WAS A "-"
39 07F9A   3800         LDA     3,0             ;AC3=0 IF + OR - NOT USED
40 07F9B   EE00         ADD     3,1             ;UPDATE ADDRESS
41 07F9C   4800         STA     1,0             ;STORE THE NEW VALUE
42 07F9D   39CF         LDA     3,N12           ;LINE FEED CODE
43 07F9E   9D0C         SUB#    0,3,SZR         ;DON'T ECHO A LF OR CR
44 07F9F   39C8         LDA     3,N15           ;CARRIAGE RETURN CODE
45 07FA0   9D0C         SUB#    0,3,SZR
46 07FA1   0192         JMP     DCRST           ;DECODE REST OF THE COMMANDS
47 07FA2   0911         JSR     MPCLR           ;LOC IN MAP SKIPS NEXT INSTR
48 07FA3   68FF         MAP     1               ;GO TO USERS MEMORY
49 07FA4   4A00 NXTLOC: STA     1,0,2           ;RESTORE THE OPEN LOCATION
50 07FA5   829B         MOVZR#  0,0,SNC         ;SKIP IF IT WAS NOT A "NL/LF"
51 07FA6   01DD         JMP     WAIT
52 07FA7   09C1         JSR     CRLF            ;TYPE A CR, LF
53 07FA8   280E         LDA     1,MPFLG         ;GET PREVIOUS ADDRESS
54 07FA9   AB00         INC     1,1             ;AND OPEN NEXT LOC
```

```
!0016 MODT
01 07FAA   480E OPNLOC: STA     1,MPFLG         ;STORE NEW ADDRESS
02 07FAB   09AD NTACC:  JSR     POCT            ;TYPE AC1
03 07FAC   0907 ACCLOC: JSR     MPCLR           ;LOC IN MAP SKIPS NEXT INST
04 07FAD   68FF         MAP     1
05 07FAE   2A00         LDA     1,0,2           ;CONTENTS = AC1
06 07FAF   09A9         JSR     POCT            ;TYPE THE OPEN LOCATION
07 07FB0   01D4         JMP     CNTWT           ;GET ORIGINAL PC
08 07FB1   004C N114:   114
09 07FB2   8002 ISAC0:  100002
10
11 07FB3   300E MPCLR:  LDA     2,MPFLG         ;AC2 = PRESENT ADDRESS
12 07FB4   D25B         MOVZL#  2,2,SNC         ;NO MAP BIT SET
13 07FB5   0300         JMP     0,3             ;RETURNS CALL+1
14 07FB6   D6A0         ADDOR   2,2             ;CLEAR MAP BIT FROM ADDRESS
15 07FB7   0301         JMP     1,3             ;AND RETURN CALL+2
16
17 07FB8   31F9 PCMND:  LDA     2,N114
18 07FB9   9524         SUB0    0,2,SZR         ;SKIP IF IT IS AN "L"
19 07FBA   018D         JMP     CONTU           ;NOT AN "L"
20 07FBB   500B         STA     2,BPADR         ;DELETE BREAK POINT
21 07FBC   3928         LDA     3,N77           ;DEVICE 77 IS CASSETTE
22 07FBD   8D0C         SUB#    1,3,SZR         ;SPECIAL PROGRAM LOAD
23 07FBE   0114         JMP     JPLD            ;NOI NORMAL DEVICE
24 07FBF   092E         JSR     LOAD            ;USE FOLLOWING LOADER
25
26 07FC0   D520 CASS:   SUB0    2,2             ;CLEAR ADDRESS COUNTER
27 07FC1   7408         DOB     2,TTI           ;2 DOB'S
28 07FC2   7408         DOB     2,TTI           ;START TAPE
29 07FC3   FD20         SUB0    3,3             ;CLEAR WORD REGISTER
30 07FC4   2902         LDA     1,ETTE          ;LOAD CONSTANT THIS CPU
31 07FC5   AB01         INC     1,1,SKP         ;INC TIMER
32 07FC6   FFE3 ETTE:   177743                  ;MICRO NOVA CONSTANT
33 07FC7   6788         SKPDN   TTI             ;DONE YET?
34 07FCA   01FD         JMP     ETTE-1          ;LOOP
35 07FC9   628F         DOAC    0,CPU           ;CLEAR DONE
36 07FCA   FA83         MOVR    3,3,SNC         ;MOVE BIT FOUND
37 07FCB   01F9         JMP     ETTE-2          ;RELOAD CONSTANT
38 07FCC   5A7F         STA     3,177,2         ;STORE WORD START AT 177
39 07FCD   D300         INC     2,2             ;INC ADDRESS POINTER
40 07FCE   187F         DSZ     177             ;DECREMENT WORD COUNTER
41 07FCF   01F4         JMP     ETTE-3          ;GO BACK FOR NEXT WORD
42 07FD0   0880         JSR     200             ;START AT 200
43
44 07FD1   F47F ENDCH:  172177
45
46 07FD2   0918 JPLD:   JSR     LOAD            ;YES, LOAD FROM DEVICE
47                                              ;# TYPED IN BEFORE "L"
```

```
10017 MODT
01 07FD3   100C LOADER: ISZ     14,0          ;COUNT DEVICE CODE
02 07FD4   1018         ISZ     30,0          ;INTO ALL IO
03 07FD5   101A         ISZ     32,0          ;INSTRUCTIONS
04 07FD6   A804         INC     1,1,SZR       ;DONE ?
05 07FD7   0005 ADDRV:  JMP     5,0           ;STAY IN LOOP AND KEEP
06                                            ;INCREMENTING DEVICE CODES
07
08 07FD8   300E         LDA     2,16          ;YES, PUT JMP 377 AT
09 07FD9   50FF         STA     2,377,0       ;LOCATION 377
10 07FDA   603F         60077                 ;START THE DEVICE
11 07FDB   8242         MOVL    0,0,SZC       ;TEST BIT 0 OF THE "L"
12                                            ;COMMAND AND SKIP IF IT
13                                            ;IS 0 (LOW SPEED DEVICE)
14 07FDC   00FF         JMP     377,0         ;FOR HIGH SPEED DEVICE GO
15                                            ;AND WAIT AT LOCATION 377
16 07FDD   0818         JSR     30,0          ;GET A FRAME
17 07FDE   8235         MOVC    0,0,SNR       ;IS IT NONZERO ?
18 07FDF   000F         JMP     17,0          ;NO, IGNORE AND GET ANOTHER
19
20 07FE0   0817         JSR     27,0          ;YES, GET FULL WORD
21 07FE1   4C16         STA     1,@26         ;STORE STARTING AT 100
22                                            ;(AUTOINCREMENT)
23 07FE2   1040         ISZ     100           ;COUNT WORD = DONE ?
24 07FE3   0012         JMP     22
25 07FE4   003F N77:    JMP     77            ;YES, = LOCATION COUNTER AND
26                                            ;JUMP TO LAST WORD
27 07FE5   AD10         SUBZ    1,1           ;CLEAR AC1 AND SET CARRY
28
29 07FE6   677F         63577                 ;DONE ?
30 07FE7   0018         JMP     30            ;NO, WAIT
31 07FE8   613F         60477                 ;YES, READ IN AC0 (DIAS 0,-)
32 07FE9   8EF3         ADDCS   0,1,SNC       ;ADD 2 FRAMES SWAPPED. GOT 2ND ?
33 07FEA   0018         JMP     30            ;NO, GO BACK AFTER IT
34 07FEB   AAC0         MOVS    1,1           ;YES, SWAP THEM
35 07FEC   0300         JMP     0,3           ;RETURN WITH FULL WORD
36
37 07FED   A210 LOAD:   MOVZ    1,0           ;AC0 = DEVICE CODE
38 07FEE   5800         STA     3,0           ;ADDRESS OF LOAD PROGRAM
39 07FEF   39E2         LDA     3,ENDCH       ;LAST INSTRUCTION IN PROG
40 07FF0   2C00 STORE:  LDA     1,@0          ;AC1 = CONTENTS TO BE MOVED
41 07FF1   68FF         MAP     1
42 07FF2   4A05         STA     1,5,2         ;STORE AC1 STARTING AT LOC. 5
43 07FF3   1000         ISZ     0             ;INCREMENT PROG ADDRESS
44 07FF4   D300         INC     2,2
45 07FF5   8F0C         AND#    1,3,SZR       ;CONTINUE UNTIL THE LAST
46                                            ;WORD HAS BEEN MOVED
47 07FF6   01FA         JMP     STORE
48 07FF7   628F         DOAC    0,77          ;IO-RESET
49 07FF8   29EC         LDA     1,N77         ;GET DEVICE MASK
50 07FF9   8F00         AND     0,1           ;ISOLATE DEVICE CODE
51 07FFA   A800         COM     1,1           ;-DEVICE CODE -1
52 07FFB   68FF         MAP     1
53 07FFC   70FF         MAP     2
54 07FFD   05DA         JMP     @ADDRV        ;START AT 5
55
56 07FFE   FFFF         -1
57 07FFF   7E01         TOP-776
58
59                      .END

**00000 TOTAL ERRORS, 00000 PASS 1 ERRORS
```

```
0018 MODT

AC3KL  007ED1   10/37   10/40   10/43
ACC    007F71   13/14   14/40
ACCLO  007FAC   14/45   16/03
ADDRV  007FD7   17/05   17/54
AID    007F1A   11/01   12/30
APL    007E25    6/14    6/46
BPADR  000008    5/18    5/19    9/25   11/18   11/22   13/20   13/29
                13/35   16/20
BPHDL  007E29    6/16    6/51
BPINS  00000A    5/17    5/18    9/29   10/46   11/17   12/14   13/25
BPINT  007EE2   11/13   13/26
BRKPT  007F3A    8/01   13/17
C14K   007EE0   10/07   11/11
C2000  007EDF   10/45   11/10
C377   007E62    8/03   10/26   10/32
C4     007EDE   10/02   11/09
C47    007E2F    6/58    9/35
C60K   007E5D    7/47    9/40
CASS   007FC0   16/26
CLTTY  007EED    6/57   11/25   13/32
CNTWT  007F84   15/06   15/14   16/07
CONTU  007F77   15/01   16/19
CRLF   007F68   14/30   15/13   15/52
DCMSK  007E31    6/10    6/60
DCRST  007F33   13/10   15/46
DELBP  007EE3    8/15   11/15   13/16
DIGIT  007F2B   13/01   15/35
DIIND  007ED3   10/31   10/45
DOALL  007F09   12/01   12/06   12/12
DRCT   007EDB   10/29   11/05   12/15   12/46
EFADD  007EC0   10/14   10/20   10/26   11/28   11/34   12/13
ENDCH  007FD1   16/44   17/39
ETTE   007FC6   16/30   16/32   16/34   16/37   16/41
EXINS  00000F    5/22   13/01   13/08   14/12   14/24   15/18
EXIT   00000C    5/19    5/20    7/14    7/32   12/12   12/24   12/26
                15/17   15/36
FRMDG  007F5F   14/18   14/23
IBRKP  007E60    8/01    8/16
ICLTT  007E2E    6/40    6/57
IDSZ   007EEF   10/09   11/28
IISZ   007EF5   10/11   11/34
IJMP   007E86    9/39   10/14   10/24
IJPLD  007E30    6/49    6/59
IJSR   007E88   10/13   10/20
ILDA   007EFF   10/06   12/01
IOTHC  007EE1   11/12   11/16
IPCMN  007E2C    6/55    7/45
IPOCT  007E28    6/39    6/54
ISAC0  007FB2   14/42   16/09
ISTA   007F04   10/04   10/18   12/06
IYSSK  007F18   11/38   12/09   12/27
JMP0   007EA9   10/01   11/43
JMP01  007EFE   11/43   12/10
JPLD   007FD2    6/59   16/23   16/46
LOAD   007FED   16/24   16/46   17/37
LOADE  007FD3   17/01
LSTBP  007F4A   13/19   13/33
MPCLR  007FB3   15/47   16/03   16/11
MPFLG  00000E    5/21    5/22   14/44   15/04   15/53   16/01   16/11
```