OCTAL DEBUGGING PROGRAM (ODT-80) PART IV-LLL BASIC

By E. R. Fisher

FOREWARD

This article is the last part of a series of four articles covering the LLL 8080 BASIC Interpreter program released to the public domain by Lawrence Livermore Laboratories. This month we shall cover the description of the Octal Debugging Program ODT-80 and include the complete assembly listing of the ODT-80 program.

INTRODUCTION

ODT-80 is an octal debugging routine for use on the Intel 8080 microprocessor. This routine provides the capability to examine and modify all of the memory that is available to the microcomputer and transfer program control to the created program. ODT-80 makes use of simple keyboard commands from any terminal—such as a teletypewriter—that is attached to the system.

ODT-80 (Octal Debugging Technique) is a program written for the MCS-80 that allows the user to modify a program via a teletypewriter keyboard. The program occupies 400 octal words and must be located in the lowest memory page of the MCS-80 system, since the program uses the RESTART instructions.

ODT has been proved to be an effective aid to debugging on microprocessors. The first version,[1] for the Intel 8080,* has been "front panel" for virtually hundreds of microprocessor applications. The author submits this ODT for the 8080 in hopes that the tradition of soft panels may be perpetuated.

SYSTEM REQUIREMENTS

All addresses of memory locations and contents of memory locations are referred to in octal numbers. A question mark (?) will be typed for any illegal input.

The minimum system requirements for using ODT are as follows:

- MCS-80 computer set
- ODT programmable read only memory (PROM) at memory page $\emptyset\emptyset\emptyset_8$
- 256 word (RAM) at page $\emptyset 1\emptyset_8$
- Teletype interface with the following codes:

  OUT 2—SEND ASCII character

  IN 2—Input word from TTY

  IN 3—Read Flags (D_7 $D_\emptyset$) 11 111 111 (flag word)

  Sending Done

  Word Received

UTILITY ROUTINES

The following subroutines are available to the user as utility routines for other programs:

Address	Call	Routine
3Ø7	RST ODT	Restart ODT program. This is useful for error branching in program debugging.
367	RST SEND	Send the ASCII character presently in the "A" register.
315 333 Ø	CAL READ	Wait for a character to be received from the teletype and return with the ASCII character in the "A" register. The "A" and "B" registers are used in this routine.
315 370 Ø	CALL CRLF	Send a carriage return and a line feed to the teletypewriter. The "A" register is used in this routine.
315 301 Ø	CAL OCTALP	Send a space and type in Octal the three digit number in the "A" register. The A, B, and E registers are used in this routine.

COMMANDS

(n_8/) —The ASCII Slash (/) character is used to Open the n_8 address and type the contents in octal.

(LF) —The ASCII Line Feed (LF) character is used to close the currently open address, and open the next sequential address. The contents of the open register may be changed by typing the octal number to be input and then typing a (CR).

(.) —The ASCII Period (.) operator character may be used before the (/) operator to open the address last used.

(n_8S) —The SET command is used to set the H register to the memory page to be accessed.
(EX) 1ØS ;H = $\emptyset 1\emptyset_8$

(n_8R) —The READ operator is used to start a loader program in memory page No. 2, location Ø.

If this command is preceded by an "nS" command, the n value will be passed to the loader program in the H register.

(n_8G) —The GO routine is used to start a program in memory. The octal number typed before the "G" operator will set the starting address in memory. The "G" operator should be preceded by an "nS" command to select the desired page.

(CTRL-C) —The ASCII Control C character is used to command the type out of the top two locations in the stack. (See trap.) Leaves the stack pointer at its initial position -2.

(CR) —The ASCII Carriage Return (CR) character is used to close the currently open address.

TRAP

The trap feature of ODT works as follows: When the central processor encounters a 377_8 as an instruction, the processor decodes this as a restart to location 70_8 in memory 0_8. An example of this is when a nonexistent memory is addressed, or when a 377_8 is placed in a program for a break point. At this time, ODT stores in the pushdown stack the address of the errant location and all of the registers. The trap indication is a "T" typed out on the teletypewriter. Control is now back in ODT and multiple Control-C's (CTRL-C) typed by the operator will yield, in sets of two (One set for each CTRL-C Typed), the trapped information in the following format:

ERROR PAGE NO.	ADDRESS +1 LOCATION	REGISTERS A	F	B	C	D	E	H	L
XXX	XXX	XXX	XXX*	XXX	XXX*	XXX	XXX	XXX	XXX

The "F" register is the status flags of the 8080, with the format shown in Fig. 1.

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
MSB (SIGN)	ZERO FLAG	Ø	CARRY 1	Ø	EVEN PARITY	1	CARRY 2

Fig. 1. "F" register format.

A CTRL-C typed at any time will type out the pushdown stack but *not* in the format shown above! The above format is only available immediately after a "T" has been sent by ODT.

If a continuous string of "T" are sent to the TTY without stopping, this is an indication that the stack pointer is pointing at nonexistent memory. It will be necessary to restart ODT.

RESTART/INTERRUPT LOCATIONS

There are five segments of memory reserved in ODT for INTERRUPT or RESTART operation. These locations, shown in Fig. 2, contain jumps to the scratch RAM memory #1Ø, used by ODT. This allows interrupt service to be handled even though memory #Ø is preprogrammed to contain ODT.

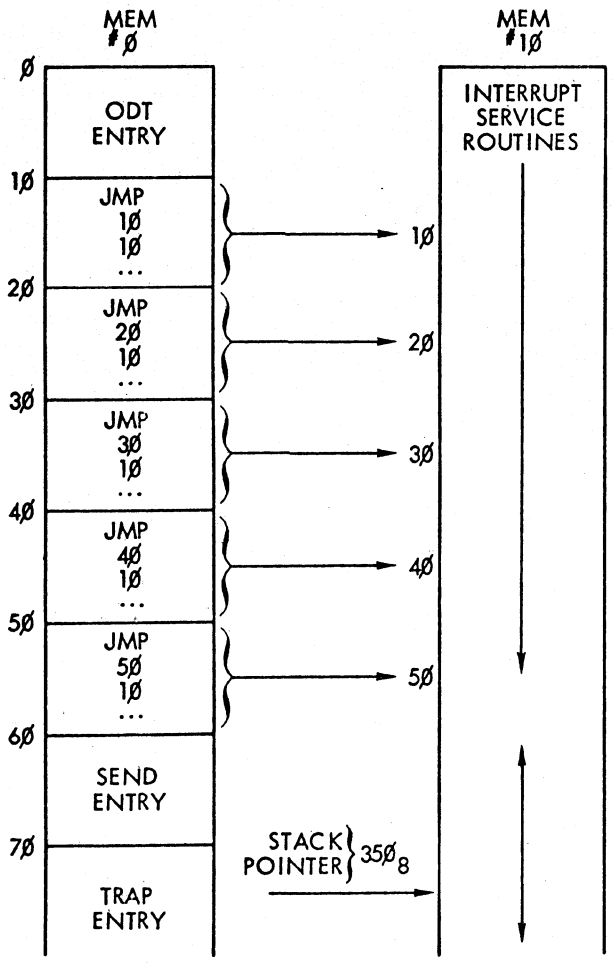

Figure 2. RESTART/INTERRUPT locations.

STACK POINTER

The stack pointer is reset to location $35Ø_8$ every time ODT is restarted via location Ø of ROM Ø—in other words, whenever a "?" is sent by ODT. ODT and other programs use the stack pointer; therefore, memory locations plus and minus this location are apt to be overwritten. However, the stack pointer moves down in memory (high addresses to low addresses), so normal operation will keep the stack pointer in RAM $1Ø_8$.

EXAMPLE OF TYPICAL DEBUGGING OPERATION

The following example illustrates a typical

debugging operation. It is assumed that a program has been assembled and that the program is to be loaded by a loader in PROM 2.

1. Set the memory field with the S command, and read the tape with the R command.

```
1ØS*          ;SET MEMORY FIELD TO 1Ø
ØR            ;START READER ON TTY
 ?            ;WHEN TAPE IS READ IN ODT
              RESTARTS (startup of ODT de-
              pends upon the reader program
              in PROM 2)
```

2. Start the program with the G command.

```
1ØS           ;START PROGRAM AT LOCATION
              Ø

ØG
```

TEST PROGRAM; TYPED BY PROGRAM WITH AN ERROR

3. Change location (30) to correct output. Restart ODT.

```
 ?
1ØS
30/327 322)   ;CHANGE "W" TO AN "R"
ØG

 TEST PROGRAM  ; CORRECTED OUTPUT
```

EXAMPLE PROGRAM LISTING

```
1
 8080 MACRO ASSEMBLER, VER 1.1  ERRORS = 0 PAGE 1

                                ;ODT TEST PROGRAM
                                ;10-21-74
 000006                 SEND EQU 6
 000370                 CRLF EQU 370Q
                                ;
                                ;PROGRAM PROPER
                                ;
 004000                         ORG 4000Q
 004000   041 022 010          LXI H TABLE          ;SET UP ADDRESS
 004003   176           LOOP:   MOV A,M              ;GET ENTRY
 004004   376 000                CPI 0
 004006   312 016 010            JZ ND               ;IF 0 END OF LIST
 004011   367                    RST SEND            ;TYPE CHARACTER
 004012   043                    INX H
 004013   303 003 010            JMP LOOP            ;LOOP


 004016   315 370 000 ND:        CALL CRLF           ;SEND CRLF

 004021   166                    HLT
                                ;
 004022   324           TABLE:   DB 324Q               ;T
 004023   305                    DB 305Q               ;E
 004024   323                    DB 323Q               ;S
 004025   324                    DB 324Q               ;T
 004026   340                    DB 240Q               ;SPACE
 004027   240                    DB 320Q               ;P
 004030   327                    DB 327Q               ;W FOR ERROR
 004031   317                    DB 317Q               ;O
 004032   307                    DB 307Q               ;G
 004033   322                    DB 322Q               ;R
 004034   301                    DB 301Q               ;A
 004035   315                    DB 315Q               ;M
 004036     Ø                    DB Ø
                                END

NO PROGRAM ERRORS

1
 8080 MACRO ASSEMBLER, VER 1.1  ERRORS = 0 PAGE 2

                        SYMBOL TABLE

  * 01

  A       000007    B       000000    C       000001    CRLF    000370
  D       000002    E       000003    H       000004    L       000005
  LOOP    004003    M       000006    ND      004015    PSW     000006
  SEND    000006    SP      000006    TABLE   004021
```

ACKNOWLEDGEMENTS

I wish to acknowledge the work of Jim English, who coauthored with me the ODT for the 8008, from which a large part of this program was derived, and Walt Binge and Mike Maples, who helped on the preparation of this paper.

ODT-80 PROGRAM LISTING

```
8080 MACRO ASSEMBLER, VER 1.1  ERRORS = 0 PAGE 1

                                ;ODT PROGRAM FOR THE MCS-80
                                ;**********AUTHOR EUGENE R FISHER**********
                                  4-20-74
 000000                         ORG 0
 000002                 TTY EQU 2
 000006                 SND EQU 6
 000003                 FLAG EQU 3
                                ;;
 000000   061 350 010 ERROR:    LXI SP,4350Q        ;SET STACK POINTER,PAGE10,LOC 350
 000003   076 277               MVI A,277Q          ;SEND A ?
 000005   303 077 000           JMP ERR
 000010   303 010 010           JMP 4010Q           ;FIRST RESTART LOCATION
 000013   000                   NOP
 000014   000                   NOP
 000015   000                   NOP
 000016   000                   NOP
 000017   000                   NOP
 000020   303 020 010           JMP 4020Q
 000023   000                   NOP
 000024   000                   NOP
 000025   000                   NOP
 000026   000                   NOP
 000027   000                   NOP
 000030   303 030 010           JMP 4030Q
 000033   000                   NOP
 000034   000                   NOP
 000035   000                   NOP
 000036   000                   NOP
 000037   000                   NOP
 000040   303 040 010           JMP 4040Q
 000043   000                   NOP
 000044   000                   NOP
 000045   000                   NOP
 000046   000                   NOP
 000047   000                   NOP
 000050   303 050 010           JMP 4050Q
 000053   000                   NOP
 000054   115           PER:    MOV C,L             ;ROUTINE TO MAKE PERIOD CURRENT LOCATION
 000055   303 107 000           JMP NEXTC
 000060   303 344 000   SEND:   JMP SEN              ;ENTRY POINT TO SEND ROUTINE
 000063   315 370 000   GO:     CALL CRLF           ;SEND A CRLF
 000066   151                   MOV L,C             ;SET THE L REG
 000067   351                    PCHL       ;JMP TO STARTING ADDRESS
 000070   343           TRAP:    XTHL               ;CHANGE ORDER OF STACK H AND L LAST OUT
 000071   325                    PUSH D             ;FORTH OUT
 000072   305                    PUSH B              :THIRD OUT
 000073   365                    PUSH     PSW           :SECOND OUT
 000074   345                    PUSH H             ;FIRST OUT IS THE ADR WHENCE WE CAME
 000075   076 324                MVI A,324Q  ;SEND A T
                                ;
                                ;THE STACK MAY BE DUMPED BY HITTING A CTRL-C TO GET THE FOLLOWING
                                ; SP+1  SP A F B C D E H L
                                ;
 000077   367           ERR:    RST SND
 00C100   315 370 000           CALL CRLF
 000103   257           BEGIN:  XRA A               ;CLEAR THE AC
 000104   026 004       BEG:    MVI D,4
 000106   117           SAV:    MOV C,A
 000107   315 333 000   NEXTC:  CALL READ
 000112   170                   MOV A,B
 000113   326 270               SUI 270Q            ;IS IT A NUMBER
 000115   362 224 000           JP  TERM            ;270 MUST BE TERMINATOR OR ILLEGAL DIGIT
 000120   170                   MOV A,B
 000121   326 260               SUI 260Q
 000123   372 224 000           JM  TERM            ;YES
 000126   025                   DCR D               ;>260 MUST BE DIGIT BUMP BUFFER CNT
 000127   312 000 000           JZ  ERROR           ;BUFFER OVERFLOW
 000132   107                   MOV B,A
 000133   171                   MOV A,C             ;GET PREVIOUS INPUTS
 000134   027                   RAL
 000135   027                   RAL
 000136   027                   RAL
 000137   332 000 000           JC  ERROR           ;IF CARRY NUMBER WAS TOO BIG
 000142   200                   ADD B
 000143   303 106 000           JMP SAV
 000146   151           SLASH:  MOV L,C             ;C CONTAINS ADDRESS
 000147   176           GETCON: MOV A,M             ;GET CONTENTS
 000150   315 301 000           CALL OCTALP
 000153   076 240               MVI A,240Q
 000155   367                   RST SND
 000156   303 103 000           JMP BEGIN
 000161   172           LF:     MOV A,D
 000162   326 004               SUI 4
 000164   312 170 000           JZ  NINP            ;IF CNT STILL 4 NO INPUT WAS RECEIVED
 000167   161                   MOV M,C             ;IF CNT <4 DEPOSIT ONPUT IN MEMORY
 000170   076 215       NINP:   MVI A,215Q
 000172   367                   RST SND
 000173   054                   INR L               ;BUMP ADDRESS
 000174   174                   MOV A,H             ;GET HI PART
 000175   315 301 000           CALL OCTALP         ;TYPE HI ADDRESS
 000200   175                   MOV A,L             ;GET LOW PART
 000201   315 301 000           CALL OCTALP         ;TYPE LOW ORDER ADDRESS
 000204   303 147 000           JMP GETCON
 000207   315 370 000   CR:     CALL CRLF           ;SEND A CRLF
 000212   172                   MOV A,D
 000213   326 004               SUI 4               ;BUFCNT =4 ?
 000215   312 103 000           JZ  BEGIN           ;YES NO INPUT SINCE LAST TERMINATOR
 000220   161                   MOV M,C             ;LOAD MEMORY WITH INPUT
 000221   303 103 000           JMP BEGIN           ;ERF
 000224   170           TERM:   MOV A,B
 000225   376 212               CPI 212Q            ;IS IT A LF
 000227   312 161 000           JZ  LF
 000232   376 215               CPI 215Q            ;IS IT RETURN
 000234   312 207 000           JZ  CR
 000237   376 322               CPI 322Q            ;R
 000241   312 000 002           JZ  1000Q           ;START READER PROGRAM
 000244   376 257               CPI 257Q
 000246   312 146 000           JZ  SLASH
 000251   376 307               CPI 307Q
 000253   312 063 000           JZ  GO              ;G
 000256   376 256               CPI 256Q            ;PERIOD
 000260   312 054 000           JZ  PER
 000263   376 203               CPI 203Q
 000265   312 357 000           JZ  CTRLC           ;CONTROL C
 000270   376 323               CPI 323Q            ;S
 000272   302 000 000           JNZ ERROR           ;FOUND AN ERROR
 000275   141           SETX:   MOV H,C             ;SAME HI ADDRESS
 000276   303 100 000           JMP ERR+1
 000301   006 004       OCTALP: MVI B,4             ;SET CNTR
 000303   007                   RLC
```

```
000304  007             RLC
000305  137             MOV E,A         ;SAVE AC
000306  076 240         MVI A,240Q      ;SEND A SPACE
000310  367             RST SND
000311  173             MOV A,E         ;GET THE AC
000312  346 003         ANI 3           ;MASK
000314  005       DECR: DCR B           ;DECR
000315  310             RZ              ;DONE ?
000316  306 260         ADI 260Q        ;NO
000320  367             RST SND     ;SEND A CHARAACTER
000321  173             MOV A,E         ;GET AC
000322  007             RLC
000323  007             RLC
000324  007             RLC
000325  137             MOV E,A         ;SAVE AC
000326  346 007         ANI 7           ;MASK
000330  303 314 000     JMP DECR
000333  333 003   READ: IN FLAG         ;ROUTINE TO READ ONE CHAR FROM TTY
000335  037             RAR
000336  322 333 000     JNC READ
000341  333 002         IN TTY
000343  107             MOV B,A
000344  323 002    SEN: OUT TTY         ;ROUTINE TO OUTPUT AN ASCII CHAR
000346  333 003   SEN1: IN FLAG
000350  037             RAR
000351  037             RAR
000352  170             MOV A,B           ;RESTORE A REG FROM READ
000353  330             RC                ;DONE?
000354  303 346 000     JMP SEN1          ;NO
000357  341       CTRLC: POP H
000360  174             MOV A,H
000361  315 301 000     CALL OCTALP
000364  175             MOV A,L
000365  303 150 000     JMP GETCON+1
000370  076 215    CRLF: MVI A,215Q      ;SEND A CRLF
000372  367             RST SND
000373  076 212         MVI A,212Q
000375  303 344 000     JMP SEN           ;RETURN VIA SEND
                        END

NO PROGRAM ERRORS

                    SYMBOL TABLE

* 01

A       000007   B       000000   BEG     000104 *  BEGIN   000103
C       000001   CR      000207   CRLF    000370    CTRLC   000357
D       000002   DECR    000314   E       000003    ERR     000077
ERROR   000000   FLAG    000003   GETCO   001147    GO      000063
H       000004   L       000005   LF      000161    M       000006
NEXTC   000107   NINP    000170   OCTAL   000301    PER     000054
PSW     000006   READ    000333   SAV     000106    SEN     000344
SEN1    000346   SEND    000060 *  SETX    000275 *  SLASH   000146
SND     000006   SP      000006   TERM    000224    TRAP    000070 *
TTY     000002
```