M. David Johnson
http://www.bds-soft.com
info@bds-soft.com

A Minimally Invasive Trace

by M. David Johnson
2021/11/06

Abstract

A minimally invasive Assembly Language Trace Routine is presented. The TRACE itself only occupies 99 bytes and each call to it only occupies 3 bytes.

——

This paper and its associated code are available online at:

http://www.bds-soft.com/cocoPapers.php .

=====

Table of Contents

Introduction

EDTASM's ZBUG Debugger is a useful tool with several attractive features. But, sometimes, all you need is a trace of the Program Counter (PC) as it walks through your Assembly Language code.

TRACE.ASM is designed to be a minimally invasive solution to meet that requirement. It only occupies 99 bytes and can be moved more or less anywhere above &H1C00, although I like to keep it just below the code I'm testing.

In use, you **EXEC** the code you're testing from a BASIC Control Program and then **EXEC** TRACE.ASM's scroll routine one additional time before exiting the BASIC Control Program.

TRACE.ASM is then called from your Assembly Code by inserting the three-byte instruction "**JSR $1D00**" wherever you want a report of PC's location (or "**JSR $XXXX**" where "**XXXX**" is wherever you've relocated TRACE.ASM).

Each time TRACE is called, it displays PC's current location as four hexadecimal characters in the lower-right corner of the screen, after scrolling to the next line. Please note that, with this simple mechanism, if you call TRACE more than fourteen times in your code, earlier PC addresses will be lost off the top of the screen. Be especially aware of this if you're calling TRACE from inside any loops.

You can also temporarily disable any given TRACE call by replacing:

```
JSR $1D00 ($BD $1D $00)
```

with:

```
NOP ($12)
NOP ($12)
NOP ($12)
```

in the calling code.

Also, if you fail to **EXEC** TRACE,ASM's scroll routine at the end of your BASIC Control Program, the "OK" prompt will erase the final PC trace entry. You could also avoid this erase by adding something like "**1220 GOTO 1220**" at the end of your BASIC Control Program, or by adding something like "**LEXIT BRA LEXIT**" at the end of the Assembly Language code which you're testing; and then using the BREAK key to exit after the completion of the test run.

Once you've completed all your testing, you can eliminate the tracing from your code in one of three different ways:

1. Remove TRACE.ASM, remove all of it's "**JSR $1D00**" calls from the Assembly Language code you're testing, and reassemble your code without the trace calls.

2. Leave everything in place, but replace every "**JSR $1D00**" call with three **NOP**'s.

3. Leave everything in place, but add a "**POKE &H1D00, &H39**" instruction to each BASIC Control Program you subsequently use to call your code. That's the machine code for "**RTS**". If the code you're testing is located at **&H1D64** then the BASIC Control Program's EXEC sequence would look like:

```
1120 '
1130 LOADM "TRACE.BIN"
1135 POKE &H1D00, &H39
1140 LOADM "TTEST.BIN"
1150 '
1160 EXEC &H1D64
```

———

In works of this complexity (at least for me) typos and other errors are bound to sneak in. Please let me know about any you discover so I can note and correct them.

M.D.J. 2021/11/06
info@bds-soft.com

=====

TRACE.ASM

This is the TRACE routine itself. The code is relocatable and you should be able to move it more or less to wherever you want above $1C00 (to avoid conflict with low memory, Video RAM, and the first page of Graphics Screen RAM).

I like to keep it at $1D00 and place the code I'm testing immediately above it at $1D64. With TRACE at this location, the code being tested simply needs to use the call "**JSR $1D00**" wherever a report of the Program Counter's (PC) location is desired.

Within this routine, Line 450 calls the internal subroutine **TRCSCL**, at Line 1360, which scrolls the screen down one line.

Line 480 then gets the value of PC which was pushed to the stack by the code being tested, when it executed the **JSR $1D00** call.

Then Line 530 puts the high byte of PC, and Line 550 puts the low byte of PC, to the lower-right corner of the VIDRAM screen, via the internal **TRCBYT** subroutine at Line 610.

———

The Assembly Language TRACE Routine:

```
                00100 *****
                00110 *
                00120 * TRACE.ASM
                00130 * MDJ 2021/10/10
                00140 *
                00150 * PRINTS A TRACE OF
                00160 * THE PC REGISTER
                00170 * TO THE LOWER
                00180 * RIGHT CORNER OF
                00190 * THE TEXT SCREEN
                00200 *
                00210 * CALL WITH:
                00220 *   JSR  $1D00
                00230 *
                00240 *****
                00250
                00260 * SCRATCHPAD VARIABLES
                00270 * THE 8-BIT NUMBER
0076            00280 L0076   EQU     $0076
                00290
                00300 * THE HIGH NIBBLE
0077            00310 L0077   EQU     $0077
                00320
```

```
                    00330 * THE LOW NIBBLE
          00F3      00340 L00F3   EQU     $00F3
                    00350
                    00360 * SCREEN ADDRESSES
          0400      00370 VIDRAM  EQU     $0400
          0600      00380 VIDEND  EQU     $0600
          0420      00390 LINE2   EQU     $0420
                    00400
1D00                00410         ORG     $1D00
1D00 34   16        00420         PSHS    A,B,X
                    00430
                    00440 * SCROLL THE SCREEN
1D02 8D   41        00450         BSR     TRCSCL
                    00460
                    00470 * GET THE VALUE OF PC
1D04 EC   64        00480         LDD     4,S
                    00490
                    00500 * PUT PC VALUE
                    00510 * TO THE SCREEN
1D06 8E   05FC      00520         LDX     #$05FC
1D09 8D   07        00530         BSR     TRCBYT
1D0B 1F   98        00540         TFR     B,A
1D0D 8D   03        00550         BSR     TRCBYT
1D0F 35   16        00560         PULS    A,B,X
1D11 39             00570         RTS
                    00580
                    00590 * A CAPTIVE
                    00600 * VERSION OF PUTBYT
1D12 97   76        00610 TRCBYT  STA     L0076
                    00620
                    00630 * DIVIDE BY 16
1D14 44             00640         LSRA
1D15 44             00650         LSRA
1D16 44             00660         LSRA
1D17 44             00670         LSRA
                    00680
                    00690 * SAVE THE HIGH NIBBLE
1D18 97   77        00700         STA     L0077
                    00710
                    00720 * MULTIPLY BY 16
1D1A 48             00730         LSLA
1D1B 48             00740         LSLA
1D1C 48             00750         LSLA
1D1D 48             00760         LSLA
                    00770
                    00780 * SAVE TEMP RESULT
1D1E 97   F3        00790         STA     L00F3
```

```
                    00800
                    00810 * GET THE NUMBER AGAIN
1D20 96   76        00820         LDA     L0076
                    00830
                    00840 * SUBTRACT TEMP RESULT
1D22 90   F3        00850         SUBA    L00F3
                    00860
                    00870 * SAVE LOW NIBBLE
1D24 97   F3        00880         STA     L00F3
                    00890
                    00900 * IS LOW NIBBLE <= 9
1D26 81   09        00910         CMPA    #9
                    00920
                    00930 * GO IF NO
1D28 22   04        00940         BHI     L1CF7
                    00950
                    00960 * ADD ZERO OFFSET
1D2A 8B   70        00970         ADDA    #112
1D2C 20   02        00980         BRA     L1CF9
                    00990
                    01000 * ADD "A" OFFSET
1D2E 8B   37        01010 L1CF7   ADDA    #55
                    01020
                    01030 * SAVE LOW NIBBLE CHAR
1D30 97   F3        01040 L1CF9   STA     L00F3
                    01050
                    01060 * GET HIGH NIBBLE
1D32 96   77        01070         LDA     L0077
                    01080
                    01090 * IS HIGH NIBBLE <= 9
1D34 81   09        01100         CMPA    #9
                    01110
                    01120 * GO IF NO
1D36 22   04        01130         BHI     L1D07
                    01140
                    01150 * ADD ZERO OFFSET
1D38 8B   70        01160         ADDA    #112
1D3A 20   02        01170         BRA     L1D09
                    01180
                    01190 * ADD "A" OFFSET
1D3C 8B   37        01200 L1D07   ADDA    #55
                    01210
                    01220 * PUT HIGH NIBBLE CHAR
                    01230 * TO VIDRAM
1D3E A7   80        01240 L1D09   STA     ,X+
                    01250
                    01260 * GET LOW NIBBLE CHAR
```

```
1D40 96   F3       01270         LDA     L00F3
                   01280
                   01290 * PUT LOW NIBBLE CHAR
                   01300 * TO VIDRAM
1D42 A7   80       01310         STA     ,X+
1D44 39            01320         RTS
                   01330
                   01340 * A CAPTIVE
                   01350 * VERSION OF SCROLL
1D45 34   32       01360 TRCSCL  PSHS    A,X,Y
                   01370
                   01380 * Y = SOURCE POINTER
                   01390 * X = TARGET POINTER
                   01400
                   01410 * POINT X TO FIRST LINE
1D47 8E   0400     01420         LDX     #VIDRAM
                   01430
                   01440 * POINT Y TO SECOND LINE
1D4A 108E 0420     01450         LDY     #LINE2
                   01460
                   01470 * SCROLL THE SCREEN
1D4E A6   A0       01480 L1D20   LDA     ,Y+
1D50 A7   80       01490         STA     ,X+
                   01500
                   01510 * ARE WE DONE?
1D52 108C 0600     01520         CMPY    #VIDEND
                   01530
                   01540 * GO IF NO
1D56 25   F6       01550         BLO     L1D20
                   01560
                   01570 * CLEAR THE LAST LINE
                   01580 * LOAD BLANK GREEN CHAR
1D58 86   60       01590         LDA     #96
1D5A A7   80       01600 L1D2C   STA     ,X+
                   01610
                   01620 * ARE WE DONE?
1D5C 8C   0600     01630         CMPX    #VIDEND
                   01640
                   01650 * GO IF NO
1D5F 25   F9       01660         BLO     L1D2C
                   01670
                   01680 * EXIT
1D61 35   32       01690         PULS    A,X,Y
1D63 39            01700         RTS
                   01710
          1D64     01720 ENDADR  EQU     *
          0000     01730         END
```

00000 TOTAL ERRORS

=====

TTEST.ASM

Here, I present three pieces of code which use the TRACE to report the Program Counter's (PC) location during execution.

The first piece of code (TTEST.ASM) is just a bare, lone call to TRACE (at Line 250), followed by a holding loop at Line 260. TTEST.ASM simply serves as a proof of concept.

The second (TTEST02.ASM) is a portion of some other code I'm currently playing with. It has seven calls to TRACE (at Lines 1830, 2020, 2200, 2390, 2470, 2830, and 3010), followed by a holding loop at Line 3020.

The third (TTEST03.ASM) is the same as the second, except that it includes a Return to the BASIC Control Program instead of the holding loop.

———

TTEST.ASM:

```
                    00100 *****
                    00110 *
                    00120 * TTEST.ASM
                    00130 * MDJ 2021/10/11
                    00140 *
                    00150 * FIRST
                    00160 * TRACE TEST
                    00170 *
                    00180 * EXEC &H1D64
                    00190 *
                    00200 ****
                    00210
1D64                00220         ORG     $1D64
                    00230
                    00240 * DO TRACE
1D64 BD   1D00      00250         JSR     $1D00
1D67 20   FE        00260 L0003   BRA     L0003
                    00270
          1D69      00280 ENDADR  EQU     *
          0000      00290         END

00000 TOTAL ERRORS
```

———

TTEST02.ASM:

```
                    00100 *****
                    00110 *
                    00120 * TTEST02.ASM
                    00130 * MDJ 2021/10/11
                    00140 *
                    00150 * SECOND
                    00160 * TRACE TEST
                    00170 *
                    00180 * EXEC &H1D64
                    00190 *
                    00200 ****
                    00210
                    00220 *****
                    00230 *
                    00240 * GENERAL EQUATES
                    00250 *
                    00260 *****
                    00270
                    00280 *YADA2 EQUATE
          1DAA      00290 OPJUMP  EQU     $1DAA
                    00300
                    00310 *YADA4 EQUATE
          3438      00320 P0JUMP  EQU     $3438
                    00330
                    00340 *YADA6 EQUATE
          3AAF      00350 P1JUMP  EQU     $3AAF
                    00360
                    00370 *YADA8 EQUATE
          3E95      00380 PBYTSL  EQU     $3E95
                    00390
                    00400 *YADAX EQUATES
          54DE      00410 GET2    EQU     $54DE
          5511      00420 GET4    EQU     $5511
          5548      00430 PUT2    EQU     $5548
          5571      00440 PUT4    EQU     $5571
          55C2      00450 GET2U   EQU     $55C2
          55EE      00460 GET4U   EQU     $55EE
          561A      00470 PUT2U   EQU     $561A
          5643      00480 PUT4U   EQU     $5643
                    00490
1D64                00500         ORG     $1D64
                    00510
1D64 34   76        00520         PSHS    A,B,X,Y,U
                    00530
1D66 16   004C      00540         LBRA    BEGIN
```

```
                    00550
                    00560 *****
                    00570 *
                    00580 * SYSTEM VARIABLES
                    00590 *
                    00600 *****
                    00610
                    00620 * THE CURRENT
                    00630 * FILE ADDRESS
                    00640 * BEING DISASSEMBLED
1D69                00650 CFADDR  RMB     2
                    00660
                    00670 * THE CURRENT
                    00680 * MEMORY ADDRESS
                    00690 * BEING DISASSEMBLED
1D6B                00700 CMADDR  RMB     2
                    00710
                    00720 * THE CURRENT
                    00730 * OBJECT CODE PREFIX
1D6D                00740 COBPRE  RMB     1
                    00750
                    00760 * THE CURRENT
                    00770 * OBJECT CODE
1D6E                00780 COBJCD  RMB     1
                    00790
                    00800 * THE CURRENT
                    00810 * POSTBYTE
1D6F                00820 CPOSTB  RMB     1
                    00830
                    00840 * THE CURRENT
                    00850 * DESTINATION
1D70                00860 CDESTN  RMB     2
                    00870
                    00880 * THE CURRENT
                    00890 * LINE NUMBER
                    00900 * DIVIDED BY 10
1D72                00910 CLINEN  RMB     2
                    00920
                    00930 *****
                    00940 *
                    00950 * OUTPUT STRING FIELDS
                    00960 *
                    00970 *****
                    00980
                    00990 * THE CURRENT
                    01000 * OUTPUT STRING'S
                    01010 * ADDRESS FIELD
```

```
                    01020 * (STRING POSITION 000)
                    01030 * (FOUR BYTES + SPACE)
1D74                01040 CSADDR  RMB     5
                    01050
                    01060 * THE CURRENT
                    01070 * OUTPUT STRING'S
                    01080 * OBJECT CODE PREFIX FIELD
                    01090 * (STRING POSITION 005)
                    01100 * (TWO BYTES)
1D79                01110 CSOBPR  RMB     2
                    01120
                    01130 * THE CURRENT
                    01140 * OUTPUT STRING'S
                    01150 * OBJECT CODE FIELD
                    01160 * (STRING POSITION 007)
                    01170 * (TWO BYTES + SPACE)
1D7B                01180 CSOBCD  RMB     3
                    01190
                    01200 * THE CURRENT
                    01210 * OUTPUT STRING'S
                    01220 * POSTBYTE FIELD
                    01230 * (STRING POSITION 010)
                    01240 * (TWO BYTES + SPACE)
1D7E                01250 CSPSTB  RMB     3
                    01260
                    01270 * THE CURRENT
                    01280 * OUTPUT STRING'S
                    01290 * DESTINATION FIELD
                    01300 * (STRING POSITION 013)
                    01310 * (FOUR BYTES + SPACE)
1D81                01320 CSDEST  RMB     5
                    01330
                    01340 * THE CURRENT
                    01350 * OUTPUT STRING'S
                    01360 * LINE NUMBER FIELD
                    01370 * (STRING POSITION 018)
                    01380 * (FOUR BYTES + ZERO
                    01390 * + SPACE)
1D86                01400 CSLINE  RMB     6
                    01410
                    01420 * THE CURRENT
                    01430 * OUTPUT STRING'S
                    01440 * LABEL FIELD
                    01450 * (STRING POSITION 024)
                    01460 * (EIGHT SPACES)
                    01470 * (I.E. NO LABELS)
1D8C                01480 CSLABL  RMB     8
```

```
                    01490
                    01500 * THE CURRENT
                    01510 * OUTPUT STRING'S
                    01520 * MNEMONIC FIELD
                    01530 * (STRING POSITION 032)
                    01540 * (FIVE BYTES + THREE SPACES)
                    01550 * (ANDCC IS LONGEST MNEMONIC)
1D94                01560 CSMNEM  RMB     8
                    01570
                    01580 * THE CURRENT
                    01590 * OUTPUT STRING'S
                    01600 * OPERAND FIELD
                    01610 * (STRING POSITION 040)
                    01620 * (UP TO 24 BYTES)
1D9C                01630 CSOPRN  RMB     24
                    01640
                    01650 * THE CURRENT
                    01660 * OUTPUT STRING'S
                    01670 * TERMINATOR
                    01680 * (STRING POSITION 64)
                    01690 * (ONE BYTE)
1DB4      00        01700 CSTERM  FCB     0
                    01710
                    01720 *****
                    01730 *
                    01740 * INITIAL SETUP
                    01750 *
                    01760 *****
                    01770
                    01780 * BEGIN BY CLEARING THE
                    01790 * OUTPUT STRING TO
                    01800 * ALL SPACES (CODE 032)
                    01810
                    01820 * DO TRACE
1DB5 BD   1D00      01830 BEGIN   JSR     $1D00
                    01840
                    01850 * POINT TO STRING'S
                    01860 * POSITION 000
1DB8 8E   1D74      01870         LDX     #CSADDR
                    01880
                    01890 * LOAD THE SPACE CODE
1DBB 86   20        01900         LDA     #32
                    01910
                    01920 * STORE IT TO THE STRING
1DBD A7   80        01930 L0001   STA     ,X+
                    01940
                    01950 * ARE WE DONE?
```

```
1DBF 8C   1DB4     01960         CMPX    #CSTERM
                   01970
                   01980 * GO IF NO
1DC2 25   F9       01990         BLO     L0001
                   02000
                   02010 * DO TRACE
1DC4 BD   1D00     02020         JSR     $1D00
                   02030
                   02040 *****
                   02050 *
                   02060 * ADDRESS FIELD
                   02070 *
                   02080 *****
                   02090
                   02100 * LOAD THE CURRENT
                   02110 * FILE ADDRESS
1DC7 FC   1D69     02120         LDD     CFADDR
                   02130
                   02140 * POINT TO THE CURRENT
                   02150 * OUTPUT STRING'S
                   02160 * ADDRESS FIELD
1DCA 8E   1D74     02170         LDX     #CSADDR
                   02180
                   02190 * DO TRACE
1DCD BD   1D00     02200         JSR     $1D00
                   02210
                   02220 *****
                   02230 *
                   02240 * LINE NUMBER FIELD
                   02250 *
                   02260 *****
                   02270
                   02280 * LOAD THE CURRENT
                   02290 * LINE NUMBER
                   02300 * DIVIDED BY 10
1DD0 FC   1D72     02310         LDD     CLINEN
                   02320
                   02330 * POINT TO THE CURRENT
                   02340 * OUTPUT STRING'S
                   02350 * LINE NUMBER FIELD
1DD3 8E   1D86     02360         LDX     #CSLINE
                   02370
                   02380 * DO TRACE
1DD6 BD   1D00     02390         JSR     $1D00
                   02400
                   02410 * PUT FINAL ZERO OF
                   02420 * LINE NUMBER TO FIELD
```

```
1DD9 86   30       02430         LDA     #48
1DDB A7   80       02440         STA     ,X+
                   02450
                   02460 * DO TRACE
1DDD BD   1D00     02470         JSR     $1D00
                   02480
                   02490 *****
                   02500 *
                   02510 * OBJECT CODE OR PREFIX
                   02520 *
                   02530 *****
                   02540
                   02550 * POINT TO THE CURRENT
                   02560 * MEMORY ADDRESS
                   02570 * BEING DISASSEMBLED
1DE0 BE   1D6B     02580         LDX     CMADDR
                   02590
                   02600 * GET NEXT BYTE AND
                   02610 * INCREMENT ADDRESSES
1DE3 A6   80       02620         LDA     ,X+
1DE5 34   02       02630         PSHS    A
1DE7 FC   1D69     02640         LDD     CFADDR
1DEA C3   0001     02650         ADDD    #1
1DED FD   1D69     02660         STD     CFADDR
1DF0 FC   1D6B     02670         LDD     CMADDR
1DF3 C3   0001     02680         ADDD    #1
1DF6 FD   1D6B     02690         STD     CMADDR
1DF9 35   02       02700         PULS    A
                   02710
                   02720 * SAVE THE BYTE TO THE
                   02730 * OBJECT CODE VARIABLE
1DFB B7   1D6E     02740         STA     COBJCD
                   02750
                   02760 * PUT IT TO THE CURRENT
                   02770 * OUTPUT STRING'S
                   02780 * OBJECT CODE FIELD
1DFE 34   10       02790         PSHS    X
1E00 8E   1D7B     02800         LDX     #CSOBCD
                   02810
                   02820 * DO TRACE
1E03 BD   1D00     02830         JSR     $1D00
                   02840
1E06 35   10       02850         PULS    X
1E08 B6   1D6E     02860         LDA     COBJCD
                   02870
                   02880 * POINT TO THE OBJECT CODE
                   02890 * AND PREFIX JUMP TABLE
```

```
1E0B 108E 1DAA     02900          LDY     #OPJUMP
                   02910
                   02920 * ADD THE MEMORY BYTE THREE
                   02930 * TIMES TO OFFSET INTO THE
                   02940 * JUMP TABLE'S THREE-BYTE
                   02950 * INSTRUCTION
1E0F 31   A6       02960          LEAY    A,Y
1E11 31   A6       02970          LEAY    A,Y
1E13 31   A6       02980          LEAY    A,Y
                   02990
                   03000 * DO TRACE
1E15 BD   1D00     03010          JSR     $1D00
1E18 20   FE       03020 L0003    BRA     L0003
                   03030
          1E1A     03040 ENDADR   EQU     *
          0000     03050          END

00000 TOTAL ERRORS
```

TTEST03.ASM:

```
                   00100 *****
                   00110 *
                   00120 * TTEST03.ASM
                   00130 * MDJ 2021/10/11
                   00140 *
                   00150 * THIRD
                   00160 * TRACE TEST
                   00170 *
                   00180 * EXEC &H1D64
                   00190 *
                   00200 ****
                   00210
                   00220 *****
                   00230 *
                   00240 * GENERAL EQUATES
                   00250 *
                   00260 *****
                   00270
                   00280 *YADA2 EQUATE
          1DAA     00290 OPJUMP   EQU     $1DAA
                   00300
                   00310 *YADA4 EQUATE
          3438     00320 P0JUMP   EQU     $3438
                   00330
```

```
                    00340 *YADA6 EQUATE
          3AAF      00350 P1JUMP  EQU     $3AAF
                    00360
                    00370 *YADA8 EQUATE
          3E95      00380 PBYTSL  EQU     $3E95
                    00390
                    00400 *YADAX EQUATES
          54DE      00410 GET2    EQU     $54DE
          5511      00420 GET4    EQU     $5511
          5548      00430 PUT2    EQU     $5548
          5571      00440 PUT4    EQU     $5571
          55C2      00450 GET2U   EQU     $55C2
          55EE      00460 GET4U   EQU     $55EE
          561A      00470 PUT2U   EQU     $561A
          5643      00480 PUT4U   EQU     $5643
                    00490
1D64                00500         ORG     $1D64
                    00510
1D64 34   76        00520         PSHS    A,B,X,Y,U
                    00530
1D66 16   004C      00540         LBRA    BEGIN
                    00550
                    00560 *****
                    00570 *
                    00580 * SYSTEM VARIABLES
                    00590 *
                    00600 *****
                    00610
                    00620 * THE CURRENT
                    00630 * FILE ADDRESS
                    00640 * BEING DISASSEMBLED
1D69                00650 CFADDR  RMB     2
                    00660
                    00670 * THE CURRENT
                    00680 * MEMORY ADDRESS
                    00690 * BEING DISASSEMBLED
1D6B                00700 CMADDR  RMB     2
                    00710
                    00720 * THE CURRENT
                    00730 * OBJECT CODE PREFIX
1D6D                00740 COBPRE  RMB     1
                    00750
                    00760 * THE CURRENT
                    00770 * OBJECT CODE
1D6E                00780 COBJCD  RMB     1
                    00790
                    00800 * THE CURRENT
```

```
                    00810 * POSTBYTE
1D6F                00820 CPOSTB  RMB     1
                    00830
                    00840 * THE CURRENT
                    00850 * DESTINATION
1D70                00860 CDESTN  RMB     2
                    00870
                    00880 * THE CURRENT
                    00890 * LINE NUMBER
                    00900 * DIVIDED BY 10
1D72                00910 CLINEN  RMB     2
                    00920
                    00930 *****
                    00940 *
                    00950 * OUTPUT STRING FIELDS
                    00960 *
                    00970 *****
                    00980
                    00990 * THE CURRENT
                    01000 * OUTPUT STRING'S
                    01010 * ADDRESS FIELD
                    01020 * (STRING POSITION 000)
                    01030 * (FOUR BYTES + SPACE)
1D74                01040 CSADDR  RMB     5
                    01050
                    01060 * THE CURRENT
                    01070 * OUTPUT STRING'S
                    01080 * OBJECT CODE PREFIX FIELD
                    01090 * (STRING POSITION 005)
                    01100 * (TWO BYTES)
1D79                01110 CSOBPR  RMB     2
                    01120
                    01130 * THE CURRENT
                    01140 * OUTPUT STRING'S
                    01150 * OBJECT CODE FIELD
                    01160 * (STRING POSITION 007)
                    01170 * (TWO BYTES + SPACE)
1D7B                01180 CSOBCD  RMB     3
                    01190
                    01200 * THE CURRENT
                    01210 * OUTPUT STRING'S
                    01220 * POSTBYTE FIELD
                    01230 * (STRING POSITION 010)
                    01240 * (TWO BYTES + SPACE)
1D7E                01250 CSPSTB  RMB     3
                    01260
                    01270 * THE CURRENT
```

```
                    01280 * OUTPUT STRING'S
                    01290 * DESTINATION FIELD
                    01300 * (STRING POSITION 013)
                    01310 * (FOUR BYTES + SPACE)
1D81                01320 CSDEST  RMB     5
                    01330
                    01340 * THE CURRENT
                    01350 * OUTPUT STRING'S
                    01360 * LINE NUMBER FIELD
                    01370 * (STRING POSITION 018)
                    01380 * (FOUR BYTES + ZERO
                    01390 * + SPACE)
1D86                01400 CSLINE  RMB     6
                    01410
                    01420 * THE CURRENT
                    01430 * OUTPUT STRING'S
                    01440 * LABEL FIELD
                    01450 * (STRING POSITION 024)
                    01460 * (EIGHT SPACES)
                    01470 * (I.E. NO LABELS)
1D8C                01480 CSLABL  RMB     8
                    01490
                    01500 * THE CURRENT
                    01510 * OUTPUT STRING'S
                    01520 * MNEMONIC FIELD
                    01530 * (STRING POSITION 032)
                    01540 * (FIVE BYTES + THREE SPACES)
                    01550 * (ANDCC IS LONGEST MNEMONIC)
1D94                01560 CSMNEM  RMB     8
                    01570
                    01580 * THE CURRENT
                    01590 * OUTPUT STRING'S
                    01600 * OPERAND FIELD
                    01610 * (STRING POSITION 040)
                    01620 * (UP TO 24 BYTES)
1D9C                01630 CSOPRN  RMB     24
                    01640
                    01650 * THE CURRENT
                    01660 * OUTPUT STRING'S
                    01670 * TERMINATOR
                    01680 * (STRING POSITION 64)
                    01690 * (ONE BYTE)
1DB4      00        01700 CSTERM  FCB     0
                    01710
                    01720 *****
                    01730 *
                    01740 * INITIAL SETUP
```

```
                    01750 *
                    01760 *****
                    01770
                    01780 * BEGIN BY CLEARING THE
                    01790 * OUTPUT STRING TO
                    01800 * ALL SPACES (CODE 032)
                    01810
                    01820 * DO TRACE
1DB5 BD   1D00      01830 BEGIN   JSR     $1D00
                    01840
                    01850 * POINT TO STRING'S
                    01860 * POSITION 000
1DB8 8E   1D74      01870         LDX     #CSADDR
                    01880
                    01890 * LOAD THE SPACE CODE
1DBB 86   20        01900         LDA     #32
                    01910
                    01920 * STORE IT TO THE STRING
1DBD A7   80        01930 L0001   STA     ,X+
                    01940
                    01950 * ARE WE DONE?
1DBF 8C   1DB4      01960         CMPX    #CSTERM
                    01970
                    01980 * GO IF NO
1DC2 25   F9        01990         BLO     L0001
                    02000
                    02010 * DO TRACE
1DC4 BD   1D00      02020         JSR     $1D00
                    02030
                    02040 *****
                    02050 *
                    02060 * ADDRESS FIELD
                    02070 *
                    02080 *****
                    02090
                    02100 * LOAD THE CURRENT
                    02110 * FILE ADDRESS
1DC7 FC   1D69      02120         LDD     CFADDR
                    02130
                    02140 * POINT TO THE CURRENT
                    02150 * OUTPUT STRING'S
                    02160 * ADDRESS FIELD
1DCA 8E   1D74      02170         LDX     #CSADDR
                    02180
                    02190 * DO TRACE
1DCD BD   1D00      02200         JSR     $1D00
                    02210
```

```
                    02220 *****
                    02230 *
                    02240 * LINE NUMBER FIELD
                    02250 *
                    02260 *****
                    02270
                    02280 * LOAD THE CURRENT
                    02290 * LINE NUMBER
                    02300 * DIVIDED BY 10
1DD0 FC   1D72      02310         LDD     CLINEN
                    02320
                    02330 * POINT TO THE CURRENT
                    02340 * OUTPUT STRING'S
                    02350 * LINE NUMBER FIELD
1DD3 8E   1D86      02360         LDX     #CSLINE
                    02370
                    02380 * DO TRACE
1DD6 BD   1D00      02390         JSR     $1D00
                    02400
                    02410 * PUT FINAL ZERO OF
                    02420 * LINE NUMBER TO FIELD
1DD9 86   30        02430         LDA     #48
1DDB A7   80        02440         STA     ,X+
                    02450
                    02460 * DO TRACE
1DDD BD   1D00      02470         JSR     $1D00
                    02480
                    02490 *****
                    02500 *
                    02510 * OBJECT CODE OR PREFIX
                    02520 *
                    02530 *****
                    02540
                    02550 * POINT TO THE CURRENT
                    02560 * MEMORY ADDRESS
                    02570 * BEING DISASSEMBLED
1DE0 BE   1D6B      02580         LDX     CMADDR
                    02590
                    02600 * GET NEXT BYTE AND
                    02610 * INCREMENT ADDRESSES
1DE3 A6   80        02620         LDA     ,X+
1DE5 34   02        02630         PSHS    A
1DE7 FC   1D69      02640         LDD     CFADDR
1DEA C3   0001      02650         ADDD    #1
1DED FD   1D69      02660         STD     CFADDR
1DF0 FC   1D6B      02670         LDD     CMADDR
1DF3 C3   0001      02680         ADDD    #1
```

```
1DF6 FD   1D6B     02690          STD     CMADDR
1DF9 35   02       02700          PULS    A
                   02710
                   02720 * SAVE THE BYTE TO THE
                   02730 * OBJECT CODE VARIABLE
1DFB B7   1D6E     02740          STA     COBJCD
                   02750
                   02760 * PUT IT TO THE CURRENT
                   02770 * OUTPUT STRING'S
                   02780 * OBJECT CODE FIELD
1DFE 34   10       02790          PSHS    X
1E00 8E   1D7B     02800          LDX     #CSOBCD
                   02810
                   02820 * DO TRACE
1E03 BD   1D00     02830          JSR     $1D00
                   02840
1E06 35   10       02850          PULS    X
1E08 B6   1D6E     02860          LDA     COBJCD
                   02870
                   02880 * POINT TO THE OBJECT CODE
                   02890 * AND PREFIX JUMP TABLE
1E0B 108E 1DAA     02900          LDY     #OPJUMP
                   02910
                   02920 * ADD THE MEMORY BYTE THREE
                   02930 * TIMES TO OFFSET INTO THE
                   02940 * JUMP TABLE'S THREE-BYTE
                   02950 * INSTRUCTION
1E0F 31   A6       02960          LEAY    A,Y
1E11 31   A6       02970          LEAY    A,Y
1E13 31   A6       02980          LEAY    A,Y
                   02990
                   03000 * DO TRACE
1E15 BD   1D00     03010          JSR     $1D00
                   03020
                   03030 * RETURN TO BASIC
1E18 35   76       03040          PULS    A,B,X,Y,U
1E1A 39            03050          RTS
                   03060
          1E1B     03070 ENDADR   EQU     *
          0000     03080          END

00000 TOTAL ERRORS
```

=====

TTEST.BAS

This is the BASIC Control Program I used with TTEST.ASM, TTEST02.ASM, and TTEST03.ASM (adjusting Line 1140 below accordingly).

Lines 1100 and 1110 make room for all the assembly code.

Line 1160 executes the code to be tested.

Line 1220 executes the final scroll to protect the last report from being erased by the OK prompt.

———

TTEST.BAS:

```
1000 '*****
1010 '*
1020 '* TTEST.BAS
1030 '* MDJ 2021/10/11
1040 '*
1050 '* FIRST
1060 '* TRACE TEST
1070 '*
1080 '****
1090 '
1100 PCLEAR 1
1110 CLEAR 200,&H1D00
1120 '
1130 LOADM"TRACE.BIN"
1140 LOADM"TTEST03.BIN"
1150 '
1160 EXEC &H1D64
1170 '
1180 'DO FINAL SCROLL
1190 'BY GOING INTO
1200 'THE MIDDLE
1210 'OF TRACE.BIN
1220 EXEC &H1D45
1230'
1240 END
```

———

Result of the TTEST03.ASM run:


VCC 2.1.0d Tandy Color Computer 3 Emulator
File Edit Configuration Cartridge Help
TTEST03 ASN 3 A 3
TTEST03 BIN 2 B 1
OK
LOAD"TTEST.BAS"
OK
RUN
1DB8
1DC7
1DD0
1DD9
1DE0
1E06
1E18
OK
Skip:01 | FPS: 60 | MC5809 @ 0.89Mhz FD-502:Idle

As expected.

=====

Appendix A: New BDS Software License

This New Software License applies to all software found on the BDS Software site, and supersedes all previous copyright notices and licensing provisions which may appear in the software itself or in any documentation therefor.

All software which has previously been placed in the public domain remains in the public domain.

All other software, programs, experiments and reports, documentation, and any other material on this site (other than that attributed to outside sources) is hereby copyright © 2018 (or later if so marked) by M. David Johnson.

All software, documentation, and other information on the BDS Software site is available for you to freely download without cost.

Whether you downloaded such items directly from this site, or you obtained them by any other means, you are hereby licensed to copy them, to sell or give away such copies, to use them, and to excerpt from them, in any way whatsoever, so long as nothing you do with them would denigrate the name of our Lord and Savior, Jesus Christ.

I make absolutely no warranty whatsoever for any of these items. You use them entirely at your own risk.

If they don't work for you, I commiserate.

If they crash your system, I sympathize.

But I accept no responsibility whatsoever for any such consequences. Under no circumstances will BDS Software or M. David Johnson be liable for any negative results of any kind which you may experience from downloading or using these items.

BDS Software's former mail address at P.O. Box 485 in Glenview, IL is no longer valid. Any mail sent to that address will be rejected by the U.S. Postal Service. See my Contact page.

M.D.J. 2018/06/08

=====