6800 Trace and Disassemble Program

This program puts you on the trail of runaway routines.

Richard Carickhoff
812 Pulaski Dr.
Lansdale PA 19446

Did you ever write a program that didn't work and then spend hours, or even days, debugging it? Did you ever wonder how the program got to that particular location? . . . why that compare instruction wasn't working as you thought it should? . . . why that multiply routine didn't work?

Well, I've been down that road many times myself, so I decided to write a program that would allow me to trace a program instruction by instruction while, at the same time, see exactly what was taking place before and after the execution of each instruction.

The 6800 Trace and Disassemble program does just that. The program enables the user to perform the following functions:

- Program trace function
- Go to user's program function
- Program disassemble function
- Memory examine and change function
- Register examine and change function

The detailed explanations, along with operating procedures for each of these functions, are described in the following paragraphs.

At the start of each function it is assumed that the last data character printed by the terminal is a colon (:), which is the program's prompt character. All values entered must be in hexadecimal format.

Program Trace Function

The program trace function will trace the user's program one instruction at a time, while outputting to the terminal the location, mnemonic, operand, contents of all MPU registers (CC, B, A, X, SP) and the next return address in the stack. The trace function will do this for each instruction prior to its execution.

The trace funtion can be performed by typing one of the following two responses:

: T nnnn
or
: T nnnn, mmmm

The first response must be terminated with a carriage return. The character T specifies a trace function. The four hexadecimal digits following T specify the starting address of the first instruction to be traced. This response instructs the program to trace only one instruction (see Example 1).

At this point the trace function waits for the operator to enter a character. If the character is any character other than the Escape (1B hex), the instruction displayed will be executed and the next instruction will be output to the terminal along with the contents of all the MPU registers (see Example 2).

The contents of the following MPU registers are printed along with each instruction:

cc—Condition code register
b—B register
a—A register
x—X register
sp—Stack pointer

```
: T 0103                    cc  b   a   x     sp    rtn
0103  JMP  0225             8F  19  FF  2242  A049  7B05
```

Example 1.

```
: T 0103                    cc  b   a   x     sp    rtn
0103  JMP  0225             8F  19  FF  2242  A049  7B05
0225  LDS  22               CF  19  FF  2242  A049  7B05
```

Example 2.

...—first return address at the top of the stack

The contents of the program counter is the location of the instruction to be executed.

With the use of the trace ...ction, the operator can step ...ough his program one in...uction at a time. The contents of all the MPU registers are always visible before and after the execution of each instruction. Also, the instruction is always printed before it is executed so the operator can decide whether to terminate the trace at that point (depressing Escape key) or to continue.

The second response to the prompt character is used to trace a program until the breakpoint address is reached. The first four hexadecimal digits define the starting address of the first instruction of the trace sequence. The second four hexadecimal digits following the comma define the breakpoint address. Once the last digit is entered, the program will immediately start tracing the program starting at the start address.

The output format is the ...e as the single trace func... except that the program ... continue outputting each instruction until the breakpoint address is reached. At that point the trace function operates in the same manner as the single trace function. That is, depressing the Escape key terminates the trace and depressing any other key executes the last instruction printed and outputs the next instruction. The Escape key is also used to terminate a trace sequence prior to reaching the breakpoint address.

Caution: The trace function traces a program with the use of the software interrupt (SWI). Always terminate any trace sequence using the Escape key. Using the system reset may leave a software interrupt in the user's program.

This method of tracing a program is normally used to determine how a program arrived at a particular location. If a CRT is ...sed for a terminal, the ... instructions executed will still appear on the screen (assuming the CRT has a minimum of 16 lines). The rate at which the program executes is controlled by the output rate of the terminal being used.

Program A shows an example of the trace function. The program selected is Tom Pittman's 6800 Tiny BASIC. I chose this program because it is well known and is an interesting program to trace. It also demonstrates the visibility of a program using the trace function.

The starting address was set at 0103 hex, which is Tiny BASIC's warm start address. The breakpoint address was set at an address that would not be reached. This allowed me to terminate the program at any point during the trace.

In Program A there are several instructions that are disassembled with asterisks (***) for the mnemonic and ROM for the operand. This alerts the operator that the trace function came upon a ROM address that could not be loaded with the software interrupt. The trace function in this case places the software interrupt at the return address. The trace function assumes that routines in ROM are functional and always return via the RTS (return subroutine) instruction.

The ROM address shown in Program A is the MIKBUG output routine (E1D1). Examining the contents of the A register prior to executing the output routine shows the character being output. Also, the output is reflected in the trace printout as indicated by the line feed following the first output by Tiny BASIC.

```
:T 0103,0FFF
0103  JMP    0225     C1 19 0D 2242 A07D 022A
0225  LDS    22       C1 19 0D 2242 A07D 022A
0227  JSR    062C     C9 19 0D 2242 A07F 0300
062C  LDA A  #0D      C9 19 0D 2242 A07D 022A
062E  BSR    0649     C1 19 0D 2242 A07D 022A
0649  CLR    00BF     C1 19 0D 2242 A07B 0630
064C  JMP    0598     C4 19 0D 2242 A07B 0630
0598  INC    00BF     C4 19 0D 2242 A07B 0630
059B  BMI    05A7     C0 19 0D 2242 A07B 0630
059D  STX    BA       C0 19 0D 2242 A07B 0630
059F  PSH B           C0 19 0D 2242 A07B 0630
05A0  JSR    0109     C0 19 0D 2242 A07A 1906
0109  JMP    E1D1     C0 19 0D 2242 A078 05A3
E1D1  ***    ROM

05A3  PUL B           C0 19 0D 2242 A07A 1906
05A4  LDX    BA       C0 19 0D 2242 A07B 0630
05A6  RTS             C0 19 0D 2242 A07B 0630
0630  LDA B  0111     C0 19 0D 2242 A07D 022A
0633  ASL B           C0 03 0D 2242 A07D 022A
0634  BEQ    063E     C0 06 0D 2242 A07D 022A
0636  PSH B           C0 06 0D 2242 A07D 022A
0637  BSR    0642     C0 06 0D 2242 A07C 0602
0642  CLR A           C0 06 0D 2242 A07A 0639
0643  TST    0111     C4 06 00 2242 A07A 0639
0646  BPL    0649     C0 06 00 2242 A07A 0639
0649  CLR    00BF     C0 06 00 2242 A07A 0639
064C  JMP    0598     C4 06 00 2242 A07A 0639
0598  INC    00BF     C4 06 00 2242 A07A 0639
059B  BMI    05A7     C0 06 00 2242 A07A 0639
059D  STX    BA       C0 06 00 2242 A07A 0639
059F  PSH B           C0 06 00 2242 A07A 0639
05A0  JSR    0109     C0 06 00 2242 A079 0606
0109  JMP    E1D1     C0 06 00 2242 A077 05A3
E1D1  ***    ROM
05A3  PUL B           C1 06 00 2242 A079 0606
05A4  LDX    BA       C1 06 00 2242 A07A 0639
05A6  RTS             C1 06 00 2242 A07A 0639
0639  PUL B           C1 06 00 2242 A07C 0602
063A  DEC B           C1 06 00 2242 A07D 022A
063B  DEC B           C1 05 00 2242 A07D 022A
063C  BNE    0636     C1 04 00 2242 A07D 022A
0636  PSH B           C1 04 00 2242 A07D 022A
0637  BSR    0642     C1 04 00 2242 A07C 0402
0642  CLR A           C1 04 00 2242 A07A 0639
0643  TST    0111     C4 04 00 2242 A07A 0639
0646  BPL    0649     C0 04 00 2242 A07A 0639
0649  CLR    00BF     C0 04 00 2242 A07A 0639
064C  JMP    0598     C4 04 00 2242 A07A 0639
0598  INC    00BF     C4 04 00 2242 A07A 0639
059B  BMI    05A7     C0 04 00 2242 A07A 0639
059D  STX    BA       C0 04 00 2242 A07A 0639
059F  PSH B           C0 04 00 2242 A07A 0639
05A0  JSR    0109     C0 04 00 2242 A079 0406
0109  JMP    E1D1     C0 04 00 2242 A077 05A3
E1D1  ***    ROM
```

Program A.

Trace Function Restrictions

There are only two restrictions on the trace function. The first is that it will not trace a program that uses a software interrupt, since the software interrupt interferes with the trace function's software interrupt. The second restriction is that the trace function cannot be used to trace itself.

Go to User's Program Function

This function allows the operator to execute his program. The operator may specify a breakpoint address in order to return to the trace program. This function can be performed by typing one of the following two responses:

: G nnnn

or

: G nnnn, mmmm

The first response must be terminated with a carriage return. The character G specifies a Go function. The four hexadecimal digits following G specify the starting address of the program to be executed (e.g., : G 0103).

The only way to return to the Trace and Disassemble program with this response is through the system monitor.

The second response is used to execute a user's program until the breakpoint address is reached. The first four hexadecimal digits define the starting address of the program to be executed. The second four hexadecimal digits following the comma define the breakpoint address. Once the last digit is entered, the MPU will start executing the user's program. Once the breakpoint address is reached, the control of the program is returned to the trace function (see Example 3).

The program can be traced from this point one instruction at a time by simply depressing any key other than the Escape key. The trace will operate in the same manner as if a trace function was being performed.

```
: G 0103.022F
022F   LDX   #0080   C1   00   00   07A1   0000
```

Example 3.

If the program does not reach the breakpoint address and the operator wishes to return to the trace and disassemble program, he must perform a system reset and return through the system monitor. However, the software interrupt still exists at the breakpoint address.

To remove the interrupt and replace it with the original instruction, the Go to User's Program function can be executed where the starting address is set to the breakpoint address. The program will immediately return, displaying the original instruction at the terminal. The operator can then terminate the trace function by depressing the Escape key.

Program Disassemble Function

This function allows the operator to disassemble any 6800 program including the Trace and Disassemble program itself. The disassemble function can disassemble one instruction at a time or a sequence of instructions, while outputting to the terminal the location, object code, mnemonic and operand for each instruction.

The disassemble function can be performed by typing one of the following two responses:

: D nnnn

or

: D nnnn, mmmm

```
: D 0225
0225   9E   22   LDS   22
```

Example 4.

```
: D 0225
0225   9E   22     LDS   22
0227   BD   062C   JSR   062C
```

Example 5.

The first response must be terminated with a carriage return. The character D specifies a disassemble function. The four hexadecimal digits following D specify the starting address of the instruction to be disassembled (see Example 4).

At this point the disassemble function waits for the operator to enter a character. If the character is any character other than an Escape, the next instruction in sequence will be disassembled (see Example 5). In doing so, the operator can step through a disassembly of a program one instruction at a time.

The second response is used to disassemble a list of instructions. The first four hexadecimal digits specify the first instruction to be disassembled.

```
:D 0225,0280
0225 9E 22      LDS   22
0227 BD 062C    JSR   062C
022A FE 01FE    LDX   01FE
022D DF 2A      STX   2A
022F CE 0080    LDX   #0080
0232 DF C2      STX   C2
0234 CE 0030    LDX   #0030
0237 DF C0      STX   C0
0239 9F 26      STS   26
023B 8D B8      BSR   01F5
023D 8D 07      BSR   0246
023F 20 FA      BRA   023B
0241 8C 1066    CPX   #1066
0244 20 F3      BRA   0239
0246 CE 0117    LDX   #0117
0249 DF BC      STX   BC
024B 81 30      CMP A #30
024D 24 56      BCC   02A5
024F 81 08      CMP A #08
0251 25 91      BCS   01E4
0253 48         ASL A
0254 97 BD      STA A BD
0256 DE BC      LDX   BC
0258 EE 17      LDX   17,X
025A 6E 00      JMP   00,X
025C BD 062C    JSR   062C
025F 86 21      LDA A #21
0261 97 C1      STA A C1
0263 BD 0109    JSR   0109
0266 86 80      LDA A #80
0268 97 C3      STA A C3
026A D6 2B      LDA B 2B
026C 96 2A      LDA A 2A
026E F0 01FF    SUB B 01FF
0271 B2 01FE    SBC A 01FE
0274 BD 0542    JSR   0542
0277 96 C0      LDA A C0
0279 27 0F      BEQ   028A
027B CE 0293    LDX   #0293
027F DF 2A      STX   2A
0281 BD 05AD    JSR   05AD
```

Program B. Disassemble function.

The second four hexadecimal digits following the comma specify the last instruction to be disassembled.

Once the last digit is entered, the program will immediately list each instruction in sequence until the last address is reached. The last address specified must be on an instruction boundary. Otherwise, the disassembly will continue past the last address. The Escape key can be used to terminate any list sequence.

When the last address is reached, the disassembly will stop. The operator can continue the disassembly one instruction at a time by depressing any key other than Escape. Otherwise, the Escape key will terminate the disassembly and return control back to the contr

```
0960  7E E0 CC 7E E0 CA 7E E0 C8 7E E0 7E 7E E1 AC 7E
0970  E1 D1 A0 42 00 00 00 00 0A 7E 00 08 00 0A 00 00
0980  09 00 30 30 30 39 04 30 38 20 20 20 20 20 20 20
0990  20 49 4E 58 20 20 20 20 20 20 20 20 20 20 20 04
09A0  00 00 00 00 00 00 00 00 00 00 8E A0 42 BF C9 12
09B0  CE 0A E9 FF A0 14 BE 09 12 7F 09 21 CE 08 8A BD
09C0  09 09 BD 09 0C 16 BD 09 00 C1 44 27 18 C1 47 27
09D0  7B C1 54 27 27 C1 52 27 06 C1 4D 27 05 20 D7 7E
09E0  0A 11 7E 0A 17 BD 09 CE BD 0B A3 CE 09 22 BD 09
09F0  09 CE 0B 9C BD 09 09 BD 0A 97 20 EC BD 09 CE 30
0A00  B6 09 17 A7 05 B6 09 18 A7 06 BD 0B A3 CE 09 22
0A10  86 04 B7 09 26 BD 09 09 CE 09 2F BD 09 09 BD 0A
0A20  BE CE 0B 9C BD 09 09 BD 0A 97 BD 0B 11 3F BD 0A
0A30  65 FF 09 17 BD 09 0C 81 0D 27 0A BD 0A 59 FF 09
0A40  1F 7C 09 21 39 CE 0B 9C BD 09 09 39 BD 0A 65 FF
0A50  09 15 30 B6 09 15 A7 05 B6 09 16 A7 06 7F 09 1E
0A60  BD 09 0C 81 0D 27 09 BD 0A 59 FF 09 17 BD 0B 11
0A70  3B 30 FF 09 15 20 03 BD 0A 65 CE 0B 9C BD 09 09
0A80  CE 09 15 BD 09 06 FE 09 15 BD 09 03 FF 09 15 9D
0A90  09 0C 81 20 27 E4 81 08 26 0A FE 09 15 09 09 FF
0AA0  09 15 20 D6 BD 0A 81 BD 0A 75 09 A7 00 A1 00 27
0AB0  C9 86 3F BD 09 0F 7E 09 56 8D 0A CE 0B 9C BD 09
0AC0  09 FE 09 15 39 8D 0C B7 09 15 8D 07 B7 09 16 FE
0AD0  09 15 39 8D 09 48 48 48 16 8D 02 1F 39 BD 09
0AE0  4C 80 30 2B 0F 81 09 2F 0A 81 11 2B 07 81 16 2E
0AF0  03 80 07 39 7F 09 56 7D 09 21 27 0D FE 09 17 BC
0B00  09 1F 26 0A 7F 09 21 20 05 BD 09 0C 20 08 86 80
0B10  09 2A 0A B6 80 08 81 1B 26 03 7F 09 56 39 FE 09
0B20  12 08 BD 09 03 BD 09 03 BD 09 03 BD 09 06 08 FF
0B30  09 15 CE 09 15 BD 09 06 FE 09 15 08 BD 09 06 39
0B40  BF 09 12 30 6D 06 26 02 6A 05 6A 06 8D 09 30 EE
0B50  05 FF 09 17 7E 09 AA B6 09 1E 84 3C 81 08 26 08
0B60  FE 09 19 B6 09 14 A7 00 FE 09 17 B6 09 18 A7 00
0B70  39 B6 09 1E 84 1C 26 03 8D 32 39 81 04 26 08 FE
0B80  09 19 FF 09 17 20 F1 81 08 26 04 8D 53 20 E9 81
0B90  0C 26 11 4F F6 09 19 30 E8 06 A9 05 B7 09 17 F7
0BA0  09 18 20 04 30 EF 09 FF 09 17 20 CC FE 09 17 A6
0BB0  00 B7 09 1B 86 3F A7 00 A1 00 27 23 CE 09 17 FC
0BC0  09 06 CE 0B 92 BD 09 09 BD 0A 97 B6 09 1E 85 20
0BD0  27 05 FE 09 1C 20 03 30 EE 08 FF 09 17 20 CD 39
0BE0  FE 09 19 A6 00 B7 09 14 20 CA 0D 0A 00 00 00 00
0BF0  3A 04 20 2A 2A 2A 20 20 20 52 4F 4D 0D 0A 00 00
0C00  00 00 04 CE 09 22 C6 1D 86 20 A7 00 08 5A 26 FA
0C10  86 04 A7 00 FE 09 17 AC 00 B7 09 1B 48 CE 0D F4
0C20  FF 09 1C 24 03 7C 09 1C F6 09 1D 1B B7 09 1D 24
0C30  03 7C 09 1C CE 09 22 B6 09 17 BD 0C FF B6 09 18
0C40  BD 0C FF FE 09 1C A6 00 B7 09 1C E6 01 F7 09 1D
0C50  C4 03 FE 09 17 A6 01 B7 09 19 A6 02 B7 09 1A 37
0C60  CE 09 27 B6 09 1B BD 0C FF 08 5A 27 0F B6 09 19
0C70  BD 0C FF 5A 27 06 B6 09 1A BD 0C FF 33 FE 09 17
0C80  08 5A 26 FC FF 09 17 B6 09 1C 16 CE 0D 19 FF 09
0C90  1C 48 24 04 7C 09 1C 0C 1B 24 03 7C 09 1C 0C BB
0CA0  09 1D 24 03 7C 09 1C B7 09 1D FE 09 1C A6 00 B7
0CB0  09 31 A6 01 B7 09 32 A6 02 B7 09 33 CE 20 F7 09
0CC0  34 B6 09 1E 85 C0 27 08 2A 04 CE 41 20 02 C6 42
0CD0  F7 09 35 CE 09 37 B6 09 31 81 2A 26 08 B6 09 1B
0CE0  BD 0C FF 20 40 B6 09 1E 85 02 27 39 B6 09 1B 81

0C90  8D 27 12 84 F0 B7 09 1B 81 80 27 04 81 C0 26 05
0CA0  8E 23 A7 00 08 B6 09 1B 81 8D 27 38 84 F0 81 20
0CB0  27 32 B6 09 19 BD 0C FF B6 09 1E 85 01 27 06 B6
0CC0  09 1A BD 0C FF 06 09 19 81 60 27 08 81 A0 27 04
0CD0  81 E0 26 09 86 2C A7 00 08 86 58 A7 00 FE 09 17
0CE0  FF 09 1C 39 4F F6 09 19 2A 01 4A 0C FB 09 18 F7
0CF0  09 1A B9 09 17 B7 09 19 8D 05 17 8D 02 20 DE 3E
0D00  8D 06 32 08 8D 06 08 39 44 44 44 44 84 0F 8B 30
0D10  81 39 23 02 8B 07 A7 00 39 2A 2A 2A 4E 4F 50 54
0D20  41 50 54 50 41 49 4E 58 44 45 58 43 4C 56 53 45
0D30  56 43 4C 43 53 45 43 43 4C 49 53 45 49 53 42 41
0D40  43 42 41 54 41 42 54 42 41 44 41 41 41 42 41 42
0D50  52 41 42 48 49 42 4C 53 42 43 43 42 43 53 42 4E
0D60  45 42 45 51 42 5E 43 42 56 53 42 50 4C 42 4D 49
0D70  42 47 45 42 4C 54 42 47 54 42 4C 45 54 53 58 49
0D80  4E 53 50 55 4C 44 45 53 54 58 53 50 53 48 52 54
0D90  53 52 54 49 57 41 49 53 57 49 4E 45 47 43 4F 4D
0DA0  4C 53 52 52 4F 52 41 53 52 41 53 4C 52 4F 4C 44
0DB0  45 43 49 4E 43 54 53 54 43 4C 52 4A 4D 50 53 55
0DC0  42 43 4D 50 53 42 43 41 4E 44 42 49 54 4C 44 41
0DD0  45 4F 52 41 44 43 4F 52 41 41 44 44 43 50 52 42
0DE0  53 52 4C 44 53 53 54 41 53 54 53 4A 53 52 4C 44
0DF0  58 53 54 58 00 01 01 01 00 01 00 01 00 01 00 01
0E00  02 01 03 01 04 01 05 01 06 01 07 01 08 01 09 01
0E10  0A 01 0B 01 0C 01 0D 01 0D 01 00 01 00 01 11 01
0E20  0E 01 0F 01 00 01 10 01 00 01 11 01 00 01 00 01
0E30  00 01 00 01 12 06 00 01 13 0A 14 0A 15 0A 16 0A
0E40  17 0A 18 0A 19 0A 1A 0A 1B 0A 1C 0A 1D 0A 1E 0A
0E50  1F 0A 20 0A 21 01 22 01 23 81 23 41 24 01 25 01
0E60  26 81 26 41 00 01 27 11 00 01 28 11 00 01 00 01
0E70  29 01 2A 01 29 81 00 01 00 01 2C 81 2D 81 00 01
0E80  2E 81 2F 81 30 81 31 81 32 81 00 01 33 81 34 81
0E90  00 01 35 81 2B 41 00 01 00 01 2C 41 2D 41 00 01
0EA0  2E 41 2F 41 30 41 31 41 32 41 00 01 33 41 34 41
0EB0  00 01 35 41 2B 02 00 01 00 01 2C 02 2D 02 00 01
0EC0  2E 02 2F 02 30 02 31 02 32 02 00 01 33 02 34 02
0ED0  36 0E 35 02 2B 03 00 01 00 01 2C 03 2D 03 00 01
0EE0  2E 03 2F 03 30 03 31 03 32 03 00 01 33 03 34 03
0EF0  36 07 35 03 37 82 38 82 39 82 00 01 3A 82 3B 82
0F00  3C 82 00 01 3D 82 3E 82 3F 82 40 82 41 03 42 20
0F10  43 03 00 01 37 82 38 82 39 82 00 01 3A 82 3B 82
0F20  3C 82 44 82 3D 82 3E 82 3F 82 40 82 41 02 00 01
0F30  43 02 45 02 37 82 38 82 39 82 00 01 3A 82 3B 82
0F40  3C 82 44 82 3D 82 3E 82 3F 82 40 82 41 02 46 2E
0F50  43 02 45 02 37 83 38 83 39 83 00 01 3A 83 3B 83
0F60  3C 83 44 83 3D 83 3E 83 3F 83 40 83 41 03 46 27
0F70  43 03 45 03 37 42 38 42 39 42 00 01 3A 42 3B 42
0F80  3C 42 03 01 3D 42 3E 42 3F 42 40 42 00 01 00 01
0F90  47 03 00 01 37 42 38 42 39 42 00 01 3A 42 3B 42
0FA0  3C 42 44 42 3D 42 3E 42 3F 42 40 42 00 01 00 01
0FB0  47 02 48 02 37 42 38 42 39 42 00 01 3A 42 3B 42
0FC0  3C 42 44 42 30 42 3E 42 3F 42 40 42 00 01 00 01
0FD0  47 02 48 02 37 43 38 43 39 43 00 01 3A 43 3B 43
0FE0  3C 43 44 43 3D 43 3E 43 3F 43 40 43 00 01 3C 01
0FF0  47 03 48 03 00 00 00 00 00 00 00 00 00 00 00 00
>
```

Hex listing of Trace and Disassemble program.

monitor.

Program B shows the disassembly of Tiny BASIC starting at address 0225 hex and finishing at 0280 hex. All values are in hexadecimal. Branch operands are the actual branch address. Direct addressing instructions are shown with two digit operands. If a location does not contain a valid op code, the disassembler will assume it is data and output asterisks (***) for the mnemonic.

Memory Examine and Change Function

This function can be used by the operator for inputting a program or making changes to an existing program. This function can be performed by typing in the following response:

: M nnnn

The character M specifies a memory change function. The four hexadecimal digits following M specify the address to be examined or changed. Once the last digit is entered, the program will respond with the address and its contents:

: M 0103
0103 7E

The operator must now decide whether to change memory, space to the next location, back space to the previous location or return to the control monitor.

If the contents of memory are to be changed, just enter the new value. The program will automatically output the next address and its contents. If the contents of memory cannot be changed, the program will output a (?) and return to the control monitor.

If the operator wishes to space to the next location, he'll just depress the space bar. The program will output the next location and its contents. For back spacing to the previous location, just depress the back space key (08 hex). The program will output the previous location and its contents. The back space function is useful for back spacing when an incorrect value is entered.

The memory change function can be terminated by depressing the Escape key or entering an invalid hex character (see Example 6).

Register Examine and Change Function

This function is used to ex-

```
: M 0103
0103  7E        (space)
0104  02        (back space)
0103  7E  BD
0104  02        (back space)
0103  BD  7E
0104  02        (space)
0105  25        (back space)
0104  02        (back space)
0103  7E        (escape)
```

Example 6.

```
MPU Register   : R
cc             A077 C1        (space)
B              A078 19   FE
A              A079 0D   A0
XH             A07A 22        (space)
XL             A07B 42        (space)
PCH            A07C 01        (space)
PCL            A07D 03        (space)
RTNH           A07E 02        (space)
RTNL           A07F 2A        (space)
               A080 FF        (escape)
               : T 0103
               0103   JMP   0225   C1   FE   A0   2242   A07D   022A
```

Example 7.

amine and change the contents of the MPU registers prior to executing the trace or Go to User's Program function. The trace and Go to User's Program functions use the return from interrupt (RTI) instruction to return to the user's program. The RTI instruction updates all the MPU registers with the values stored away in the stack.

The register examine and change function is initiated by entering the character R after the colon. The location of the first MPU register and its contents will be printed. The examining and changing of the data is done in the same manner as the M function (see Example 7).

Basic Memory Map

The 6800 Trace and Disassemble program resides in less than 2K of memory. The hex listing accompanies the article. The program uses some of the MIKBUG I/O routines. Table 1 lists I/O routines used by the program.

There are some parameters that may have to be changed depending on your particular machine. The stack pointer, for example, is initially loaded to $A042. If this value is changed, it should be set to at least ten locations down from the top of the stack.

The software interrupt vector is normally stored at location $FFFA. In my home-brew system the software interrupt vector points to a ROM subroutine that uses location $A014 as a programmable software interrupt vector. The Trace and Disassemble program initializes location $A014 to the return address of the trace function. This address ($A014) in the program will have to be changed to $FFFA (if programmable) or to whatever the programmable location is in your particular machine.

The Back Space and Escape Codes can be modified. They are presently set to 08 hex and 1B hex, respectively.

Break Test Routine

The break test is used by the program during a trace or disassemble program function. After each line of output the program jumps to the break test routine. The break test checks for a key being depressed. If one is not, the program returns normally. If a key is depressed, the character is input and tested for the Escape Code. If the character is not the Escape Code, the program exits from the routine normally. If the character is the Escape Code, the program returns to the control monitor.

Any changes to the break test must be made within the first three instructions. The remaining four are used by other routines within the program. There are some spare locations at the end of the program starting at $0FF4 for modifications to the break test (see Example 8).

Summary

The 6800 Trace and Disassemble program is an effective debugging tool. It requires no hardware changes, as long as your system has a programmable SWI vector. I've used it many times and so have other 6800 users. It allows you to trace your program instruction by instruction. You can make changes to your program, disassemble your patches and then trace them. You can make a listing of your program and even the trace of your program.

If you would like to get a copy of the listing of the program for relocation purposes or whatever, just send $5 with your name and address to:

Richard Carickhoff
812 Pulaski Drive
Lansdale PA 19446

If you have any problems with the program just send a self-addressed, stamped envelope to me and I'll try to answer any questions that you may have. ■

```
                JMP $E1AC     OUTPUT 2 HEX CHARS AND SPACE
BASIC MEMORY MAP
0900-0911       I/O ROUTINES
0912-0949       TEMPORARY STORAGE
094A            START OF PROGRAM
094A-0D18       EXECUTABLE PROGRAM
0B8A-0BA2       PROMPT, INVALID CODE AND CRLF MESSAGES
0D19-0FF3       MNEMONIC AND CODE TABLES
MIKBUG I/O ROUTINES
0900            JMP $E0CC     OUTPUT SPACE
0903            JMP $E0CA     OUTPUT 2 HEX CHARS AND SPACE
0906            JMP $E0C8     OUTPUT 4 HEX CHARS AND SPACE
0909            JMP $E07E     OUTPUT MESSAGE
090C            JMP $E1AC     INPUT A CHAR
090F            JMP $E1D1     OUTPUT A CHAR
PARAMETERS
094B-094C       $A042         MIDDLE OF STACK
0954-0955       $A014         SWI VECTOR (NORMALLY $FFFA)
0A37            $08           BACKSPACE CODE
0AB7            $1B           ESCAPE CODE
```

Table 1. Memory map of I/O routines and parameters.

T nnnn (CR)	Trace instruction at location nnnn.
T nnnn, mmmm	Trace program starting at location nnnn with breakpoint address set at mmmm.
G nnnn (CR)	Go to user's program starting at location nnnn.
G nnnn, mmmm	Go to user's program starting at location nnnn with breakpoint address set at mmmm.
D nnnn (CR)	Disassemble instruction at location nnnn.
D nnnn, mmmm	Disassemble instruction at location nnnn and ending at location mmmm.
M nnnn	Examine memory location nnnn.
R	Examine MPU registers starting with condition code.
(ESC)	Escape from present function and return to control monitor.

Table 2. Summary of control functions.

```
0AAE   BREAK   LDAA   $8009   PIA STATUS-KEY DEPRESSED?
0AB1           BPL    EXIT     NO
0AB3           LDAA   $8008   YES, INPUT CHAR
0AB6   CHECK   CMPA   #1B      ESCAPE CODE?
0AB8           BNE    EXIT     NO
0ABA           JMP    CONTROL  YES, RETURN TO CONTROL MONITOR
0ABD   EXIT    RTS             RETURN NORMAL
```

Example 8.

```
                                ********************************************************
                                *                                                      *
                                *       M6800 TRACE AND DISASSEMBLE PROGRAM            *
                                *                                                      *
                                *       R. C. CARICKHOFF - JAN,1978                    *
                                *                                                      *
                                *    COPYRIGHT RICHARD C. CARICKHOFF 1978              *
                                *                                                      *
                                ********************************************************
                                ********************************************************
                                *       MIKBUG I/O ROUTINES                            *
                                ********************************************************
                    0900                        ORG   $0900
0900 A  7E  E0CC F9B4   OUTS    JMP   $E0CC     OUTPUT SPACE
0903 A  7E  E0CA F812   OUT2HS  JMP   $E0CA     OUTPUT 2 HEX CHARS + SPACE
0906 A  7E  E0C8 F80F   OUT4HS  JMP   $E0C8     OUTPUT 4 HEX CHARS + SPACE
0909 A  7E  E07E F806   OUTM    JMP   $E07E     OUTPUT MESSAGE
090C A  7E  E1AC F800   INCH    JMP   $E1AC     INPUT A CHAR
090F A  7E  E1D1 OUTCH  F803    JMP   $E1D1     OUTPUT A CHAR
                                ********************************************************
                                *       PROGRAM VARIABLES                              *
                                ********************************************************
0912 A                          SP      RMB   2         STACK POINTER
0914 A                          SAVB    RMB   1         SAVE B
0915 A                          SAVX    RMB   2         SAVE X
0917 A                          OPERA   RMB   2         OPERATOR ADDRESS
0919 A                          OPER    RMB   2         OPERAND
091B A                          SAVA    RMB   1         SAVE A
091C A                          TMP1    RMB   2
091E A                          TYPE    RMB   1         INSTR TYPE
091F A                          TMP3    RMB   2         SAVE END ADDR
0921 A                          TMP4    RMB   1         END ADDR FLAG
0922 A                          LBUF    RMB   40        LINE BUFFER FOR DISASSEMBLY
                                ********************************************************
                                *       START OF PROGRAM                               *
                                ********************************************************
094A A  8E  A042            START   LDS   #$A042    MIDDLE OF STACK
094D A  BF  0912                    STS   SP        STACK POINTER
0950 A  CE  0AE0                    LDX   #SWI      SWI RETURN ADDRESS
0953 A  FF  A014                    STX   $A014     SWI VECTOR (NORMALLY $FFFA)
0956 A  8E  0912            CNTRL   LDS   SP
0959 A  7F  0921                    CLR   TMP4      CLEAR END ADDR FLAG
095C A  CE  0B6A                    LDX   #MCLP
095F A  BD  0909                    JSR   OUTM      OUTPUT CRLF AND PROMPT
0962 A  BD  090C                    JSR   INCH      INPUT CHAR
0965 A  16                          TAB
0966 A  BD  0900                    JSR   OUTS      OUTPUT SPACE
0969 A  C1  44                      CMPB  #'D
096B A  27  18                      BEQ   DFUNC     DISASSEMBLE FUNC
096D A  C1  47                      CMPB  #'G
096F A  27  78                      BEQ   GFUNC     GO TO USER PROG FUNC
0971 A  C1  54                      CMPB  #'T
0973 A  27  27                      BEQ   TFUNC     TRACE FUNC
0975 A  C1  52                      CMPB  #'R
0977 A  27  06                      BEQ   C1        REG CHANGE FUNC
0979 A  C1  4D                      CMPB  #'M
```

```
 1 0978 A  27       05                   BEQ  C2      MEMORY CHANGE FUNC
 2 097D A  20       D7                   BRA  CNTRL
 3 097F A  7E       0A11           C1    JMP  REFUNC
 4 0982 A  7E       0A17           C2    JMP  MEFUNC
 5                                ********************************************************
 6                                *       DISASSEMBLE FUNCTION   D XXXX,YYYY             *
 7                                ********************************************************
 8 0985 A  BD       09CE           DFUNC JSR  INPARM  INPUT ADDR PARAMETERS
 9 0988 A  BD       0BA3           DF1   JSR  DSMBL   DISASSEMBLE PRESENT LOCN
10 098B A  CE       0922                 LDX  #LBUF
11 098E A  BD       0909                 JSR  OUTM    OUTPUT PC,CODE,MNE,OPERAND
12 0991 A  CE       0B9C                 LDX  #MCL
13 0994 A  BD       0909                 JSR  OUTM    OUTPUT CRLF
14 0997 A  BD       0A97                 JSR  TEXIT   TEST FOR EXIT
15 099A A  20       EC                   BRA  DF1     NO,THEN CONTINUE
16                               ********************************************************
17                               *       TRACE FUNCTION   T XXXX,YYYY                   *
18                               ********************************************************
19 099C A  BD       09CE           TFUNC JSR  INPARM  INPUT ADDR PARAMETERS
20 099F A  30                            TSX
21 09A0 A  B6       0917                 LDAA OPERA   GET START ADDR
22 09A3 A  A7       05                   STAA 5,X     PUT IN STACK AT PC LOCN
23 09A5 A  B6       0918                 LDAA OPERA+1
24 09A8 A  A7       06                   STAA 6,X
25 09AA A  BD       0BA3           TF1   JSR  DSMBL   DISASSEMBLE PRESENT LOCN
26 09AD A  CE       0922                 LDX  #LBUF
27 09B0 A  B6       04                   LDAA #4      EOM
28 09B2 A  B7       0926                 STAA LBUF+4  PUT IN LINE BUFFER
29 09B5 A  BD       0909                 JSR  OUTM    OUTPUT PRESENT LOCN
30 09B8 A  CE       092F                 LDX  #LBUF+13
31 09BB A  BD       0909                 JSR  OUTM    OUTPUT MNE,OPERAND
32 09BE A  BD       0ABE                 JSR  OUTR    OUTPUT REG VALUES
33 09C1 A  CE       0B9C                 LDX  #MCL
34 09C4 A  BD       0909                 JSR  OUTM    OUTPUT CRLF
35 09C7 A  BD       0A97                 JSR  TEXIT   TEST FOR EXIT
36 09CA A  BD       0B11                 JSR  ISWI    NO,THEN INSERT SWI
37 09CD A  3B                            RTI          EXECUTE DISPLAYED INSTRUCTION
38                               ********************************************************
39                               *       INPUT PARAMETERS  XXXX,YYYY  OR  XXXX          *
40                               ********************************************************
41 09CE A  BD       0A65           INPARM JSR  BADDR   INPUT START ADDR
42 09D1 A  FF       0917                 STX  OPERA
43 09D4 A  BD       090C                 JSR  INCH    INPUT NEXT CHAR
44 09D7 A  81       0D                   CMPA #$0D    CR CODE?
45 09D9 A  27       0A                   BEQ  IP1     YES,THEN CRLF
46 09DB A  BD       0A59                 JSR  BADDCL  NO,INPUT END ADDR AND CRLF
47 09DE A  FF       091F                 STX  TMP3    SAVE VALUE
48 09E1 A  7C       0921                 INC  TMP4    SET END ADDR FLAG
49 09E4 A  39                            RTS          RETURN
50 09E5 A  CE       0B9C           IP1   LDX  #MCL
51 09E8 A  BD       0909                 JSR  OUTM    OUTPUT CRLF
52 09EB A  39                            RTS          RETURN
53                               ********************************************************
54                               *       GO EXECUTE USER PROGRAM   G XXXX,YYYY          *
55                               ********************************************************
56 09EC A  BD       0A65           GFUNC JSR  BADDR   INPUT START ADDR
57 09EF A  FF       0915                 STX  SAVX
```

```
 1 0912 A  30    .                       TSX
 2 0913 A  B6    0915                    LDAA SAVX
 3 0916 A  A7    05                      STAA 5,X        PUT IN STACK AT PC LOCN
 4 0918 A  B6    0916                    LDAA SAVX+1
 5 091B A  A7    06                      STAA 6,X
 6 091D A  7F    091E            .       CLR  TYPE
 7 0A00 A  BD    090C                    JSR  INCH       INPUT NEXT CHAR
 8 0A03 A  81    0D                      CMPA #$0D       CR CODE
 9 0A05 A  27    09                      BEQ  GF1        YES,THEN RETURN FROM INTERRUPT
10 0A07 A  BD    0A59                    JSR  BADDCL     NO,INPUT BREAKPOINT ADDR
11 0A0A A  FF    0917                    STX  OPERA
12 0A0D A  BD    0B11                    JSR  IS=I       INSERT SWI AT BREAKPOINT ADDR
13 0A10 A  3B                    GF1     RTI             RETURN FROM INTERRUPT
14                               ********************************************************
15                               *       REGISTER FUNCTION  R                           *
16                               ********************************************************
17 0A11 A  30                    RFUNC   TSX             USE STACK ADDR FOR
18 0A12 A  FF    0915                    STX  SAVX       MEMORY CHANGE START ADDR
19 0A15 A  20    03                      BRA  MF1        GO TO MEMORY CHANGE FUNC
20                               ********************************************************
21                               *       MEMORY CHANGE FUNCTION  M XXXX                 *
22                               ********************************************************
23 0A17 A  BD    0A65            MFUNC   JSR  BADDR      INPUT START ADDR
24 0A1A A  CE    0B9C            MF1     LDX  #MCL
25 0A1D A  BD    0909                    JSR  OUTM       OUTPUT CRLF
26 0A20 A  CE    0915                    LDX  #SAVX
27 0A23 A  BD    0906                    JSR  OUT4HS     OUTPUT START ADDR + SPACE
28 0A26 A  FE    0915                    LDX  SAVX
29 0A29 A  BD    0903                    JSR  OUT2HS     OUTPUT CONTENTS + SPACE
30 0A2C A  FF    0915                    STX  SAVX
31 0A2F A  BD    090C                    JSR  INCH       INPUT CHAR
32 0A32 A  81    20                      CMPA #$20       SPACE CODE?
33 0A34 A  27    E4                      BEQ  MF1              YES,THEN CONTINUE
34 0A36 A  81    08                      CMPA #$08       NO-BACKSPACE CODE?
35 0A38 A  26    0A                      BNE  MF2        NO, THEN INPUT DATA VALUE
36 0A3A A  FE    0915                    LDX  SAVX       YES
37 0A3D A  09                            DEX             DECR ADDR PTR TO PREVIOUS LOCN
38 0A3E A  09                            DEX
39 0A3F A  FF    0915                    STX  SAVX
40 0A42 A  20    D6                      BRA  MF1        OUTPUT PRESENT LOCN + CONTENTS
41 0A44 A  BD    0A81            MF2     JSR  INHEX+3    INPUT NEW DATA VALUE
42 0A47 A  BD    0A75                    JSR  BYTE+2
43 0A4A A  09                            DEX
44 0A4B A  A7    00                      STAA 0,X        PUT IN MEMORY
45 0A4D A  A1    00                      CMPA 0,X        CHECK FOR POSSIBLE LOAD ERROR
46 0A4F A  27    C9                      BEQ  MF1        NO,THEN OUTPUT NEXT LOCN + CONTENTS
47 0A51 A  86    3F                      LDAA #$3F
48 0A53 A  BD    090F                    JSR  OUTCH      YES, THEN OUTPUT '?'
49 0A56 A  7E    0956                    JMP  CNTRL      RETURN TO CONTROL MONITOR
50                               ********************************************************
51                               *       BUILD ADDRESS THEN CRLF                        *
52                               ********************************************************
53 0A59 A  BD    0A              BADDCL  BSR  BADDR      INPUT 4 HEX CHARS
54 0A5B A  CE    0B9C                    LDX  #MCL
55 0A5E A  BD    0909                    JSR  OUTM       OUTPUT CRLF
56 0A61 A  FE    0915                    LDX  SAVX
57 0A64 A  39                            [illegible]
```

```
                                ********************************************************
                                *       BUILD ADDRESS                                  *
                                ********************************************************
0A65 A  8D  0C          BADDR   BSR   BYTE      INPUT 2 HEX CHARS
0A67 A  B7  0915                STAA  SAVX
0A6A A  8D  07                  BSR   BYTE      INPUT NEXT 2 HEX CHARS
0A6C A  B7  0916                STAA  SAVX+1
0A6F A  FE  0915                LDX   SAVX
0A72 A  39                      RTS             RETURN
                                ********************************************************
                                *       BYTE                                           *
                                ********************************************************
0A73 A  8D  09          BYTE    BSR   INHEX     INPUT HEX CHAR
0A75 A  48                      ASLA
0A76 A  48                      ASLA
0A77 A  48                      ASLA
0A78 A  48                      ASLA
0A79 A  16                      TAB
0A7A A  8D  02                  BSR   INHEX     INPUT NEXT HEX CHAR
0A7C A  1B                      ABA
0A7D A  39                      RTS             RETURN
                                ********************************************************
                                *       INPUT HEX CHAR                                 *
                                ********************************************************
0A7E A  8D  090C        INHEX   JSR   INCH      INPUT CHAR
0A81 A  80  30                  SUBA  #$30
0A83 A  2B  0F                  BMI   H2        NOT HEX CHAR,EXIT
0A85 A  81  09                  CMPA  #$09
0A87 A  2F  0A                  BLE   H1        HEX CHAR,RETURN
0A89 A  81  11                  CMPA  #$11
0A8B A  2B  07                  BMI   H2        NOT HEX CHAR,EXIT
0A8D A  81  16                  CMPA  #$16
0A8F A  2E  03                  BGT   H2        NOT HEX CHAR,EXIT
0A91 A  80  07                  SUBA  #7
0A93 A  39              H1      RTS             RETURN NORMAL
0A94 A  7E  0956        H2      JMP   CNTRL     RETURN TO CONTROL MONITOR
                                ********************************************************
                                *       TEST HERE FOR EXIT                             *
                                ********************************************************
0A97 A  7D  0921        TEXIT   TST   TMP4      END ADDR FLAG SET
0A9A A  27  0D                  BEQ   T1        NO,THEN WAIT FOR INPUT
0A9C A  FE  0917                LDX   OPERA     GET PRESENT ADDR
0A9F A  BC  091F                CPX   TMP3      COMPARE WITH END ADDR
0AA2 A  26  0A                  BNE   BRLAK     NO,CHECK FOR KEY DEPRESSED
0AA4 A  7F  0921                CLR   TMP4      YES,CLEAR FLAG
0AA7 A  20  05                  BRA   BREAK
0AA9 A  8D  090C        T1      JSR   INCH      INPUT CHAR
0AAC A  20  08                  BRA   CHECK
0AAE A  B6  8009        BREAK   LDAA  $8009     PIA STATUS-KEY DEPRESSED?
0AB1 A  2A  0A                  BPL   EXIT      NO
0AB3 A  B6  8008                LDAA  $8008     YES,INPUT CHAR
0AB6 A  81  1B          CHECK   CMPA  #$1B      ESCAPE CODE?
0AB8 A  26  03                  BNE   EXIT      NO
0ABA A  7E  0956                JMP   CNTRL     YES,RETURN TO CONTROL MONITOR
0ABD A  39              EXIT    RTS             RETURN NORMAL
                                ********************************************************
                                *       OUTPUT INTERNAL REGISTERS                      *
```

```
                                ********************************************************
0ABE A FE      0912     OUTR    LDX  SP
0AC1 A 08               *       INX
0AC2 A BD      0903             JSR  OUT2HS  CC
0AC5 A BD      0903             JSR  OUT2HS  B
0AC8 A BD      0903             JSR  OUT2HS  A
0ACB A BD      0906             JSR  OUT4HS  X
0ACE A 08               *       INX
0ACF A FF      0915             STX  SAVX
0AD2 A CE      0915             LDX  #SAVX
0AD5 A BD      0906             JSR  OUT4HS  SP
0AD8 A FE      0915             LDX  SAVX
0ADB A 08               *       INX
0ADC A BD      0906             JSR  OUT4HS  RTN
0ADF A 39               *       RTS
                                ********************************************************
                                *       RETURN HERE FROM SOFTWARE INTERRUPT            *
                                ********************************************************
0AE0 A BF      0912     SWI     STS  SP      SAVE STACK POINTER
0AE3 A 30               *       TSX
0AE4 A 6D      06               TST  6,X     DECR PC IN STACK
0AE6 A 26      02               BNE  S1
0AE8 A 6A      05               DEC  5,X
0AEA A 6A      06      S1       DEC  6,X
0AEC A 8D      09               BSR  RSWI    REMOVE SWI FROM USER PROGRAM
0AEE A 30               *       TSX
0AEF A EE      05               LDX  5,X     GET PC FROM STACK
0AF1 A FF      0917             STX  OPERA   PUT IN OPERATOR ADDR
0AF4 A 7E      09AA             JMP  TF1     GO TO TRACE FUNC
                                ********************************************************
                                *       REMOVE SWI FROM  USER PROGRAM                  *
                                ********************************************************
0AF7 A B6      091E     RSWI    LDAA TYPE    TYPE OF INSTR
0AFA A 84      3C               ANDA #$3C    MASK TYPE CODE
0AFC A 81      08               CMPA #$08    CHECK IF COND BRA
0AFE A 26      08               BNE  R1
0B00 A FE      0919              LDX  OPER    YES,THEN GET FIRST LOCN
0B03 A B6      0914             LDAA SAVB
0B06 A A7      00               STAA 0,X     PUT BACK USER INSTR
0B08 A FE      0917     R1      LDX  OPERA   GET SECOND LOCN
0B0B A B6      0918             LDAA SAVA
0B0E A A7      00               STAA 0,X     PUT BACK USER INSTR
0B10 A 39               *       RTS          RETURN
                                ********************************************************
                                *       INSERT SWI  INTO USER PROGRAM                  *
                                ********************************************************
0B11 A B6      091E     ISWI    LDAA TYPE    GET INSTR TYPE
0B14 A 84      1C               ANDA #$1C    MASK TYPE CODE
0B16 A 26      03               BNE  I2      CHECK IF SEQUENTIAL
0B18 A 8D      32      I1       BSR  I6      YES,THEN INSERT SWI
0B1A A 39               *       RTS          RETURN
0B1B A 81      04       I2      CMPA #4      CHECK IF UNC JMP OR BRA
0B1D A 26      08               BNE  I3
0B1F A FE      0919              LDX  OPER    YES,THEN PUT JUMP ADDR
0B22 A FF      0917             STX  OPERA   INTO CURRENT OPER ADDR
0B25 A 20      F1               BRA  I1      BEFORE CONTINUING
0B27 A 81      08       I3      CMPA #8      CHECK IF COND BRA
```

```
 1 0829 A 26 04              BNE  I4
 2 082B A 8D 53              BSR  I11      YES,THEN INSERT SWI IN BRA ADDR
 3 082D A 20 E9              BRA  I1       BEFORE CONTINUING
 4 082F A 81 0C        I4    CMPA #$C      CHECK IF JMP INDEX
 5 0831 A 26 11              BNE  I5
 6 0833 A 4F                 CLRA          YES,THEN GET INDEX
 7 0834 A F6 0919            LDAB OPER
 8 0837 A 30                 TSX
 9 0838 A EB 06              ADDB 6,X      ADD TO X VALUE IN STACK
10 083A A A9 05              ADCA 5,X
11 083C A B7 0917            STAA OPERA    PUT IN CURRENT OPER ADDR
12 083F A F7 0918            STAB OPERA+1
13 0842 A 20 D4              BRA  I1       CONTINUE
14 0844 A 30                 TSX           RTS TYPE CODE
15 0845 A EE 09              LDX  9,X      GET RETURN ADDR FROM STACK
16 0847 A FF 0917            STX  OPERA    PUT IN CURRENT OPER ADDR
17 084A A 20 CC              BRA  I1       CONTINUE
18 084C A FE 0917      I6    LDX  OPERA    GET CURRENT OPER ADDR
19 084F A A6 00              LDAA 0,X
20 0851 A B7 091B            STAA SAVA     SAVE INSTR
21 0854 A 86 3F        I7    LDAA #$3F     SWI CODE
22 0856 A A7 00              STAA 0,X      PUT IN CURRENT OPER ADDR
23 0858 A A1 00              CMPA 0,X      CHECK FOR ROM ADDR
24 085A A 27 23              BEQ  I10      NO,THEN RETURN
25 085C A CE 0917            LDX  #OPERA   YES,THEN OUTPUT CURRENT ADDR
26 085F A BD 0906            JSR  OUT4HS   AND '***   ROM' MESSAGE
27 0862 A CE 0892            LDX  #RMES
28 0865 A BD 0909            JSR  OUTM
29 0868 A BD 0A97            JSR  TEXIT    TEST FOR EXIT FROM CURRENT FUNC
30 086B A B6 091E            LDAA TYPE     NO, THEN GET INSTR TYPE
31 086E A 85 20              BITA #$20     CHECK FOR BSR OR JSR
32 0870 A 27 05              BEQ  I8
33 0872 A FE 091C            LDX  TMP1     YES,THEN GET NEXT SEQ INSTR ADDR
34 0875 A 20 03              BRA  I9
35 0877 A 30                 TSX           NO,THEN GET RETURN ADDR FROM STACK
36 0878 A EE 0B              LDX  11,X
37 087A A FF 0917      I9    STX  OPERA    PUT IN CURRENT OPER ADDR
38 087D A 20 CD              BRA  I6
39 087F A 39           I10   RTS           RETURN
40 0880 A FE 0919      I11   LDX  OPER     GET BRA ADDR
41 0883 A A6 00              LDAA 0,X
42 0885 A B7 0914            STAA SAVB     SAVE INSTR
43 0888 A 20 CA              BRA  I7       INSERT SWI AT BRA ADDR
44 088A A 0D 0A 00     MCLP  FCB  $D,$A,0,0,0,0,$3A,4
   088D A 00 00 00
   0890 A 3A 04
45 0892 A 20 2A 2A     RMES  FCC  / ***   ROM/
   0895 A 2A 20 20
   0898 A 20 52 4F
   089B A 4D
46 089C A 0D 0A 00     MCL   FCB  $D,$A,0,0,0,0,4
   089F A 00 00 00
   08A2 A 04
47                          ********************************************
48                          *      DISASSEMBLER                        *
49                          ********************************************
50 08A3 A CE 0922      DSMBL LDX  #[illegible]   START OF [illegible]
```

```
 1 0BA6 A  C6      10                    LDAB #$1D     BUFFER LENGTH
 2 0BA8 A  86      20                    LDAA #$20     SPACE CODE
 3 0BAA A  A7      00                    STAA 0,X      PUT IN BUFFER
 4 0BAC A  08                            INX           INCR POINTER
 5 0BAD A  5A                            DECB          DECR LENGTH
 6 0BAE A  26      FA                    BNE  *-4      IS BUFFER FILLED WITH SPACES
 7 0BB0 A  86      04                    LDAA #4       YES,THEN PUT EOM
 8 0BB2 A  A7      00                    STAA 0,X      INTO NEXT BUFFER LOCN
 9 0BB4 A  FE      0917                  LDX  OPERA    GET OPERATOR ADDR
10 0BB7 A  A6      00                    LDAA 0,X      GET INSTR TO BE DSMBLD
11 0BB9 A  B7      091B                  STAA SAVA     SAVE IN A
12 0BBC A  48                            ASLA          MULTIPLY BY 2
13 0BBD A  CE      0DF4                  LDX  #TBL2    GET START OF TABLE2
14 0BC0 A  FF      091C                  STX  TMP1
15 0BC3 A  24      03                    BCC  *+5
16 0BC5 A  7C      091C                  INC  TMP1     ADD OFFSET
17 0BC8 A  F6      091D                  LDAB TMP1+1
18 0BCB A  1B                            ABA
19 0BCC A  B7      091D                  STAA TMP1+1
20 0BCF A  24      03                    BCC  *+5
21 0BD1 A  7C      091C                  INC  TMP1
22                                  ******************************************************
23                                  *     OUTPUT LOCATION                                *
24                                  ******************************************************
25 0BD4 A  CE      0922                  LDX  #LBUF    START OF LINE BUFFER
26 0BD7 A  B6      0917                  LDAA OPERA    CONVERT OPER ADDR INTO
27 0BDA A  BD      0CFF                  JSR  D11      4 HEX CHARS AND PUT
28 0BDD A  B6      0918                  LDAA OPERA+1  IN LINE BUFFER
29 0BE0 A  BD      0CFF                  JSR  D11
30 0BE3 A  FE      091C                  LDX  TMP1     GET TABLE 2 OFFSET
31 0BE6 A  A6      00                    LDAA 0,X      GET TABLE 1 OFFSET
32 0BE8 A  B7      091C                  STAA TMP1     SAVE IN TMP
33 0BEB A  E6      01                    LDAB 1,X      GET CODE TYPE
34 0BED A  F7      091E                  STAB TYPE     SAVE IN TYPE
35 0BF0 A  C4      03                    ANDB #3       SAVE BYTE COUNT IN B REG
36 0BF2 A  FE      0917                  LDX  OPERA    PUT OPER ADDR IN X REG
37 0BF5 A  A6      01                    LDAA 1,X      GET 2 BYTE OPERAND
38 0BF7 A  B7      0919                  STAA OPER     SAVE VALUE
39 0BFA A  A6      02                    LDAA 2,X
40 0BFC A  B7      091A                  STAA OPER+1
41                                  ******************************************************
42                                  *     OUTPUT CODE AND OPERAND                        *
43                                  ******************************************************
44 0BFF A  37                            PSHB          SAVE BYTE COUNT
45 0C00 A  CE      0927                  LDX  #LBUF+5  6TH POSITION IN LINE BUFFER
46 0C03 A  B6      0918                  LDAA SAVA     CONVERT OPERATOR INTO
47 0C06 A  BD      0CFF                  JSR  D11      2 HEX CHARS AND PUT
48 0C09 A  08                            INX           INTO LINE BUFFER AND SPACE
49 0C0A A  5A                            DECB          DECR BYTE COUNT
50 0C0B A  27      0F                    BEQ  D1       IS BYTE COUNT ZERO?
51 0C0D A  B6      0919                  LDAA OPER     NO,CONVERT MSB OF OPERAND
52 0C10 A  BD      0CFF                  JSR  D11      INTO 2 HEX CHARS AND PUT IN LB
53 0C13 A  5A                            DECB          DECR BYTE COUNT
54 0C14 A  27      06                    BEQ  D1       IS BYTE COUNT ZERO?
55 0C16 A  B6      091A                  LDAA OPER+1   NO, CONVERT LSB OF OPERAND
56 0C19 A  BD      0CFF                  JSR  D11      INTO  2HEX CHARS AND PUT IN LB
57 0C1C A  33                       D1   PULB          RESTORE BYTE COUNT IN B REG
```

```
                                ********************************************************
                                *       OUTPUT MNEMONIC AND OPERAND                    *
                                ********************************************************
0C1D A FE      0917             D1      LDX   OPERA    PUT OPER ADDR INTO X REG
0C20 A 08                               INX            ADVANCE PTR TO NEXT INSTR
0C21 A 5A                               DECB           IN USER MEMORY
0C22 A 26      FC                       BNE   *-2
0C24 A FF      0917                     STX   OPERA    SAVE IN OPER ADDR
0C27 A B6      091C                     LDAA  TMP1     GET TABLE 1 OFFSET
0C2A A 16                               TAB            SAVE IN B
0C2B A CE      0D19                     LDX   #TBL1    GET TABLE1 START ADDR
0C2E A FF      091C                     STX   TMP1     SAVE IN TMP
0C31 A 48                               ASLA           MULTIPLY TABLE1 OFFSET BY 2
0C32 A 24      04                       BCC   *+6
0C34 A 7C      091C                     INC   TMP1     ADD TO START OF TABLE1
0C37 A 0C                               CLC
0C38 A 1B                               ABA
0C39 A 24      03                       BCC   *+5
0C3B A 7C      091C                     INC   TMP1
0C3E A 0C                               CLC
0C3F A BB      091D                     ADDA  TMP1+1
0C42 A 24      03                       BCC   *+5
0C44 A 7C      091C                     INC   TMP1
0C47 A B7      091D                     STAA  TMP1+1
0C4A A FE      091C                     LDX   TMP1
0C4D A A6      00                       LDAA  0,X      GET MNEMONIC FOUND THERE
0C4F A B7      0931                     STAA  LBUF+15  PUT INTO LINE BUFFER
0C52 A A6      01                       LDAA  1,X      STARTING AT 16TH POSITION
0C54 A B7      0932                     STAA  LBUF+16
0C57 A A6      02                       LDAA  2,X
0C59 A B7      0933                     STAA  LBUF+17
0C5C A C6      20                       LDAB  #$20
0C5E A F7      0934                     STAB  LBUF+18  PUT SPACE IN 19TH POS
0C61 A B6      091E                     LDAA  TYPE     GET CODE TYPE
0C64 A 85      C0                       BITA  #$C0     CHECK IF INSTR REFERENCES
0C66 A 27      08                       BEQ   D3       A OR B REGISTERS
0C68 A 2A      04                       BPL   D2       YES,
0C6A A C6      41                       LDAB  #'A      A REG
0C6C A 20      02                       BRA   D3
0C6E A C6      42             D2        LDAB  #'B      B REG
0C70 A F7      0935           D3        STAB  LBUF+19  PUT A OR B IN LINE BUFFER POS 20
0C73 A CE      0937                     LDX   #LBUF+21 ADVANCE X-REG TO POS 22
0C76 A B6      0931                     LDAA  LBUF+15  CHECK IF MNE IS ***
0C79 A 81      2A                       CMPA  #'*
0C7B A 26      08                       BNE   D4       NO
0C7D A B6      0918                     LDAA  SAVA     YES,THEN CONVERT OPER INTO
0C80 A BD      0CFF                     JSR   D11      2 HEX CHARS AND PUT LINE BUFFER
0C83 A 20      58                       BRA   D9       AT POS 22 AND 23.GO TO EXIT
0C85 A B6      091E           D4        LDAA  TYPE     GET TYPE
0C88 A 85      02                       BITA  #2       CHECK INHERENT TYPE CODE
0C8A A 27      51                       BEQ   D9       YES,THEN FINISHED
0C8C A B6      0918                     LDAA  SAVA     NO, GET INSTR CODE
0C8F A 81      8D                       CMPA  #$8D     CHECK IF BSR
0C91 A 27      12                       BEQ   D6       YES
0C93 A 84      F0                       ANDA  #$F0     NO,
0C95 A B7      0918                     STAA  SAVA
0C98 A 81      80                       CMPA  #$80     CHECK IF IMMED ADDR
```

```
 1 0C9A A  27      04              BEQ  D5       YES
 2 0C9C A  81      C0              CMPA #$C0
 3 0C9E A  26      05              BNE  D6       NO
 4 0CA0 A  86      23        D5    LDAA #'#
 5 0CA2 A  A7      00              STAA 0,X      PUT'#'INTO LINE BUFFER
 6 0CA4 A  08                      INX           ADVANCE POINTER
 7                                 ********************************************
 8                                 *       CHECK FOR RELATIVE ADDRESS         *
 9                                 ********************************************
10 0CA5 A  B6      0918     D6    LDAA SAVA
11 0CA8 A  81      8D              CMPA #$8D     CHECK IF BSR
12 0CAA A  27      38              BEQ  D10      YES,CALC REL ADDR
13 0CAC A  84      F0              ANDA #$F0     NO
14 0CAE A  81      20              CMPA #$20     CHECK IF OTHER REL
15 0CB0 A  27      32              BEQ  D10      YES,CALC REL ADDR
16 0CB2 A  B6      0919            LDAA OPER     NO,CONVERT OPERAND MSB
17 0CB5 A  BD      0CFF            JSR  D11      INTO 2 HEX CHARS AND PUT IN LB
18 0CB8 A  B6      091E            LDAA TYPE     CHECK FOR BYTE COUNT OF 2 OR 3
19 0CBB A  85      01              BITA #1
20 0CBD A  27      06              BEQ  D7       BYTE COUNT OF2, CHK FOR INDEXED
21 0CBF A  B6      091A            LDAA OPER+1   BYTE COUNT OF 3
22 0CC2 A  BD      0CFF            JSR  D11      OUTPUT LSB OF OPERAND
23 0CC5 A  B6      0918     D7     LDAA SAVA     GET MASKED INSTR CODE
24 0CC8 A  81      60              CMPA #$60     CHECK IF INDEXED TYPE
25 0CCA A  27      08              BEQ  D8       YES
26 0CCC A  81      A0              CMPA #$A0
27 0CCE A  27      04              BEQ  D8       YES
28 0CD0 A  81      E0              CMPA #$E0
29 0CD2 A  26      09              BNE  D9       NO
30                                ********************************************
31                                *       PUT ',X' IN LINE BUFFER            *
32                                ********************************************
33 0CD4 A  86      2C        D8    LDAA #',      OUTPUT ',' INTO LINE BUFFER
34 0CD6 A  A7      00              STAA 0,X
35 0CD8 A  08                      INX
36 0CD9 A  86      58              LDAA #'X      OUTPUT 'X' INTO LINE BUFFER
37 0CDB A  A7      00              STAA 0,X
38 0CDD A  FE      0917     D9     LDX  OPERA    GET NEXT OPER ADDR
39 0CE0 A  FF      091C            STX  TMP1     SAVE IN TMP 1
40 0CE3 A  39                      RTS           RETURN
41                                ********************************************
42                                *       CALCULATE RELATIVE ADDRESS         *
43                                ********************************************
44 0CE4 A  4F        D10   CLRA
45 0CE5 A  F6      0919            LDAB OPER     OPERAND OFFSET
46 0CE8 A  2A      01              BPL  *+3
47 0CEA A  4A                      DECA
48 0CEB A  0C                      CLC
49 0CEC A  FB      0918            ADDB OPERA+1  ADD TO OPERATOR ADDR
50 0CEF A  F7      091A            STAB OPER+1
51 0CF2 A  B9      0917            ADCA OPERA
52 0CF5 A  B7      0919            STAA OPER
53 0CF8 A  8D      05              BSR  D11      OUTPUT BRANCH ADDR TO
54 0CFA A  17                      TBA           LINE BUFFER
55 0CFB A  8D      02              BSR  D11
56 0CFD A  20      DE              BRA  D9
57                                ********************************************
```

```
                                        *       CONVERT BINARY NUMBER IN A-REG TO 2 HEX     *
                                        *       CHARS AND OUTPUT TO LINE BUFFER             *
                                        ***********************************************
0CFF A  36              D11     PSHA            SAVE A
0D00 A  8D      06              BSR   D12      OUTPUT FIRST HEX CHAR TO
0D02 A  32                      PULA           LINE BUFFER
0D03 A  08                      INX            ADVANCE PTR
0D04 A  8D      06              BSR   D13      OUTPUT SECOND HEX CHAR TO
0D06 A  08                      INX            LINE BUFFER
0D07 A  39                      RTS            ADVANCE PTR AND RETURN
0D08 A  44              D12     LSRA
0D09 A  44                      LSRA
0D0A A  44                      LSRA
0D0B A  44                      LSRA
0D0C A  84      0F      D13     ANDA #$F
0D0E A  8B      30              ADDA #$30
0D10 A  81      39              CMPA #$39
0D12 A  23      02              BLS  *+4
0D14 A  8B      07              ADDA #7
0D16 A  A7      00              STAA 0,X
0D18 A  39                      RTS
                                ***********************************************
                                *       MNEMONIC TABLE                        *
                                ***********************************************
0D19 A  2A      2A      2A      TBL1    FCC  /***/
0D1C A  4E      4F      50              FCC  /NOP/
0D1F A  54      41      50              FCC  /TAP/
0D22 A  54      50      41              FCC  /TPA/
0D25 A  49      4E      58              FCC  /INX/
0D28 A  44      45      58              FCC  /DEX/
0D2B A  43      4C      56              FCC  /CLV/
0D2E A  53      45      56              FCC  /SEV/
0D31 A  43      4C      43              FCC  /CLC/
0D34 A  53      45      43              FCC  /SEC/
0D37 A  43      4C      49              FCC  /CLI/
0D3A A  53      45      49              FCC  /SEI/
0D3D A  53      42      41              FCC  /SBA/
0D40 A  43      42      41              FCC  /CBA/
0D43 A  54      41      42              FCC  /TAB/
0D46 A  54      42      41              FCC  /TBA/
0D49 A  44      41      41              FCC  /DAA/
0D4C A  41      42      41              FCC  /ABA/
0D4F A  42      52      41              FCC  /BRA/
0D52 A  42      48      49              FCC  /BHI/
0D55 A  42      4C      53              FCC  /BLS/
0D58 A  42      43      43              FCC  /BCC/
0D5B A  42      43      53              FCC  /BCS/
0D5E A  42      4E      45              FCC  /BNE/
0D61 A  42      45      51              FCC  /BEQ/
0D64 A  42      56      43              FCC  /BVC/
0D67 A  42      56      53              FCC  /BVS/
0D6A A  42      50      4C              FCC  /BPL/
0D6D A  42      4D      49              FCC  /BMI/
0D70 A  42      47      45              FCC  /BGE/
0D73 A  42      4C      54              FCC  /BLT/
0D76 A  42      47      54              FCC  /BGT/
0D79 A  42      4C      45              FCC  /BLE/
```

```
 1 0D7C A  54       53       58              FCC  /TSX/
 2 0D7F A  49       4E       53              FCC  /INS/
 3 0D82 A  50       55       4C              FCC  /PUL/
 4 0D85 A  44       45       53              FCC  /DES/
 5 0D88 A  54       58       53              FCC  /TXS/
 6 0D8B A  50       53       48              FCC  /PSH/
 7 0D8E A  52       54       53              FCC  /RTS/
 8 0D91 A  52       54       49              FCC  /RTI/
 9 0D94 A  57       41       49              FCC  /WAI/
10 0D97 A  53       57       49              FCC  /SWI/
11 0D9A A  4E       45       47              FCC  /NEG/
12 0D9D A  43       4F       4D              FCC  /COM/
13 0DA0 A  4C       53       52              FCC  /LSR/
14 0DA3 A  52       4F       52              FCC  /ROR/
15 0DA6 A  41       53       52              FCC  /ASR/
16 0DA9 A  41       53       4C              FCC  /ASL/
17 0DAC A  52       4F       4C              FCC  /ROL/
18 0DAF A  44       45       43              FCC  /DEC/
19 0DB2 A  49       4E       43              FCC  /INC/
20 0DB5 A  54       53       54              FCC  /TST/
21 0DB8 A  43       4C       52              FCC  /CLR/
22 0DBB A  4A       4D       50              FCC  /JMP/
23 0DBE A  53       55       42              FCC  /SUB/
24 0DC1 A  43       4D       50              FCC  /CMP/
25 0DC4 A  53       42       43              FCC  /SBC/
26 0DC7 A  41       4E       44              FCC  /AND/
27 0DCA A  42       49       54              FCC  /BIT/
28 0DCD A  4C       44       41              FCC  /LDA/
29 0DD0 A  45       4F       52              FCC  /EOR/
30 0DD3 A  41       44       43              FCC  /ADC/
31 0DD6 A  4F       52       41              FCC  /ORA/
32 0DD9 A  41       44       44              FCC  /ADD/
33 0DDC A  43       50       58              FCC  /CPX/
34 0DDF A  42       53       52              FCC  /BSR/
35 0DE2 A  4C       44       53              FCC  /LDS/
36 0DE5 A  53       54       41              FCC  /STA/
37 0DE8 A  53       54       53              FCC  /STS/
38 0DEB A  4A       53       52              FCC  /JSR/
39 0DEE A  4C       44       58              FCC  /LDX/
40 0DF1 A  53       54       58              FCC  /STX/
41                                  ********************************************************
42                                  *      TABLE 2 CONTAINS 2 BYTES FOR EVERY              *
43                                  *      6800 INSTR CODE. THE MSB IS THE                 *
44                                  *      OFFSET FOR TABLE1. THE LSB PROVIDES             *
45                                  *      INFORMATION ABOUT THE INSTR. THE LSB            *
46                                  *      IS DEFINED AS FOLLOWS:                          *
47                                  *            BITS 0-1 :BYTE COUNT (1,2 OR 3)           *
48                                  *            BITS 2-5 :CODE TYPE USED FOR TRACE        *
49                                  *                      0- NO JMP OR SEQUENTIAL         *
50                                  *                      1- JMP,JSR,BRA,BSR(UNC)         *
51                                  *                      2- BRA (CONDITIONAL)            *
52                                  *                      3- JMP,JSR(INDEXED)             *
53                                  *                      4- RTS                          *
54                                  *            BIT 6: REFERENCES B REG                   *
55                                  *            BIT 7: REFERENCES A REG                   *
56                                  ********************************************************
57 0DF4 A  0001                     TAL2    EQU  *-TAL1
```

```
    1 0DF6 A  0101                FDB  $0101
    2 0DF8 A  0001                FDB  $0001
    3 0DFA A  0001                FDB  $0001
    4 0DFC A  0001                FDB  $0001
    5 0DFE A  0001                FDB  $0001
    6 0E00 A  0201                FDB  $0201
    7 0E02 A  0301                FDB  $0301
    8 0E04 A  0401                FDB  $0401
    9 0E06 A  0501                FDB  $0501
   10 0E08 A  0601                FDB  $0601
   11 0E0A A  0701                FDB  $0701
   12 0E0C A  0801                FDB  $0801
   13 0E0E A  0901                FDB  $0901
   14 0E10 A  0A01                FDB  $0A01
   15 0E12 A  0B01                FDB  $0B01
   16 0E14 A  0C01                FDB  $0C01
   17 0E16 A  0D01                FDB  $0D01
   18 0E18 A  0001                FDB  $0001
   19 0E1A A  0001                FDB  $0001
   20 0E1C A  0001                FDB  $0001
   21 0E1E A  0001                FDB  $0001
   22 0E20 A  0E01                FDB  $0E01
   23 0E22 A  0F01                FDB  $0F01
   24 0E24 A  0001                FDB  $0001
   25 0E26 A  1001                FDB  $1001
   26 0E28 A  0001                FDB  $0001
   27 0E2A A  1101                FDB  $1101
   28 0E2C A  0001                FDB  $0001
   29 0E2E A  0001                FDB  $0001
   30 0E30 A  0001                FDB  $0001
   31 0E32 A  0001                FDB  $0001
   32 0E34 A  1206                FDB  $1206
   33 0E36 A  0001                FDB  $0001
   34 0E38 A  130A                FDB  $130A
   35 0E3A A  140A                FDB  $140A
   36 0E3C A  150A                FDB  $150A
   37 0E3E A  160A                FDB  $160A
   38 0E40 A  170A                FDB  $170A
   39 0E42 A  180A                FDB  $180A
   40 0E44 A  190A                FDB  $190A
   41 0E46 A  1A0A                FDB  $1A0A
   42 0E48 A  1B0A                FDB  $1B0A
   43 0E4A A  1C0A                FDB  $1C0A
   44 0E4C A  1D0A                FDB  $1D0A
   45 0E4E A  1E0A                FDB  $1E0A
   46 0E50 A  1F0A                FDB  $1F0A
   47 0E52 A  200A                FDB  $200A
   48 0E54 A  2101                FDB  $2101
   49 0E56 A  2201                FDB  $2201
   50 0E58 A  2381                FDB  $2381
   51 0E5A A  2341                FDB  $2341
   52 0E5C A  2401                FDB  $2401
   53 0E5E A  2501                FDB  $2501
   54 0E60 A  2681                FDB  $2681
   55 0E62 A  2641                FDB  $2641
   56 0E64 A  0001                FDB  $0001
   57 0E66 A  2711                FDB  $2711
```

```
    1 0E68 A  0001          FDB   $0001
    2 0E6A A  2811          FDB   $2811
    3 0E6C A  0001          FDB   $0001
    4 0E6E A  0001          FDB   $0001
    5 0E70 A  2901          FDB   $2901
    6 0E72 A  2A01          FDB   $2A01
    7 0E74 A  2B81          FDB   $2B81
    8 0E76 A  0001          FDB   $0001
    9 0E78 A  0001          FDB   $0001
   10 0E7A A  2C81          FDB   $2C81
   11 0E7C A  2D81          FDB   $2D81
   12 0E7E A  0001          FDB   $0001
   13 0E80 A  2E81          FDB   $2E81
   14 0E82 A  2F81          FDB   $2F81
   15 0E84 A  3081          FDB   $3081
   16 0E86 A  3181          FDB   $3181
   17 0E88 A  3281          FDB   $3281
   18 0E8A A  0001          FDB   $0001
   19 0E8C A  3381          FDB   $3381
   20 0E8E A  3481          FDB   $3481
   21 0E90 A  0001          FDB   $0001
   22 0E92 A  3581          FDB   $3581
   23 0E94 A  2B41          FDB   $2B41
   24 0E96 A  0001          FDB   $0001
   25 0E98 A  0001          FDB   $0001
   26 0E9A A  2C41          FDB   $2C41
   27 0E9C A  2D41          FDB   $2D41
   28 0E9E A  0001          FDB   $0001
   29 0EA0 A  2E41          FDB   $2E41
   30 0EA2 A  2F41          FDB   $2F41
   31 0EA4 A  3041          FDB   $3041
   32 0EA6 A  3141          FDB   $3141
   33 0EA8 A  3241          FDB   $3241
   34 0EAA A  0001          FDB   $0001
   35 0EAC A  3341          FDB   $3341
   36 0EAE A  3441          FDB   $3441
   37 0EB0 A  0001          FDB   $0001
   38 0EB2 A  3541          FDB   $3541
   39 0EB4 A  2B02          FDB   $2B02
   40 0EB6 A  0001          FDB   $0001
   41 0EB8 A  0001          FDB   $0001
   42 0EBA A  2C02          FDB   $2C02
   43 0EBC A  2D02          FDB   $2D02
   44 0EBE A  0001          FDB   $0001
   45 0EC0 A  2E02          FDB   $2E02
   46 0EC2 A  2F02          FDB   $2F02
   47 0EC4 A  3002          FDB   $3002
   48 0EC6 A  3102          FDB   $3102
   49 0EC8 A  3202          FDB   $3202
   50 0ECA A  0001          FDB   $0001
   51 0ECC A  3302          FDB   $3302
   52 0ECE A  3402          FDB   $3402
   53 0ED0 A  360E          FDB   $360E
   54 0ED2 A  3502          FDB   $3502
   55 0ED4 A  2B03          FDB   $2B03
   56 0ED6 A  0001          FDB   $0001
   57 0ED8 A  0001          FDB   $0001
```

```
    1 0EDA A  2C03          FDB   $2C03
    2 0EDC A  2D03          FDB   $2D03
    3 0EDE A  0001          FDB   $0001
    4 0EE0 A  2E03          FDB   $2E03
    5 0EE2 A  2F03          FDB   $2F03
    6 0EE4 A  3003          FDB   $3003
    7 0EE6 A  3103          FDB   $3103
    8 0EE8 A  3203          FDB   $3203
    9 0EEA A  0001          FDB   $0001
   10 0EEC A  3303          FDB   $3303
   11 0EEE A  3403          FDB   $3403
   12 0EF0 A  3607          FDB   $3607
   13 0EF2 A  3503          FDB   $3503
   14 0EF4 A  3782          FDB   $3782
   15 0EF6 A  3882          FDB   $3882
   16 0EF8 A  3982          FDB   $3982
   17 0EFA A  0001          FDB   $0001
   18 0EFC A  3A82          FDB   $3A82
   19 0EFE A  3B82          FDB   $3B82
   20 0F00 A  3C82          FDB   $3C82
   21 0F02 A  0001          FDB   $0001
   22 0F04 A  3D82          FDB   $3D82
   23 0F06 A  3E82          FDB   $3E82
   24 0F08 A  3F82          FDB   $3F82
   25 0F0A A  4082          FDB   $4082
   26 0F0C A  4103          FDB   $4103
   27 0F0E A  4226          FDB   $4226
   28 0F10 A  4303          FDB   $4303
   29 0F12 A  0001          FDB   $0001
   30 0F14 A  3782          FDB   $3782
   31 0F16 A  3882          FDB   $3882
   32 0F18 A  3982          FDB   $3982
   33 0F1A A  0001          FDB   $0001
   34 0F1C A  3A82          FDB   $3A82
   35 0F1E A  3B82          FDB   $3B82
   36 0F20 A  3C82          FDB   $3C82
   37 0F22 A  4482          FDB   $4482
   38 0F24 A  3D82          FDB   $3D82
   39 0F26 A  3E82          FDB   $3E82
   40 0F28 A  3F82          FDB   $3F82
   41 0F2A A  4082          FDB   $4082
   42 0F2C A  4102          FDB   $4102
   43 0F2E A  0001          FDB   $0001
   44 0F30 A  4302          FDB   $4302
   45 0F32 A  4502          FDB   $4502
   46 0F34 A  3782          FDB   $3782
   47 0F36 A  3882          FDB   $3882
   48 0F38 A  3982          FDB   $3982
   49 0F3A A  0001          FDB   $0001
   50 0F3C A  3A82          FDB   $3A82
   51 0F3E A  3B82          FDB   $3B82
   52 0F40 A  3C82          FDB   $3C82
   53 0F42 A  4482          FDB   $4482
   54 0F44 A  3D82          FDB   $3D82
   55 0F46 A  3E82          FDB   $3E82
   56 0F48 A  3F82          FDB   $3F82
   57 0F4A A  4082          FDB   $4082
```

```
 1 0F4C A  4102                FDB  $4102
 2 0F4E A  462E                FDB  $462E
 3 0F50 A  4302                FDB  $4302
 4 0F52 A  4502                FDB  $4502
 5 0F54 A  3783                FDB  $3783
 6 0F56 A  3883                FDB  $3883
 7 0F58 A  3983                FDB  $3983
 8 0F5A A  0001                FDB  $0001
 9 0F5C A  3A83                FDB  $3A83
10 0F5E A  3B83                FDB  $3B83
11 0F60 A  3C83                FDB  $3C83
12 0F62 A  4483                FDB  $4483
13 0F64 A  3D83                FDB  $3D83
14 0F66 A  3E83                FDB  $3E83
15 0F68 A  3F83                FDB  $3F83
16 0F6A A  4083                FDB  $4083
17 0F6C A  4103                FDB  $4103
18 0F6E A  4627                FDB  $4627
19 0F70 A  4303                FDB  $4303
20 0F72 A  4503                FDB  $4503
21 0F74 A  3742                FDB  $3742
22 0F76 A  3842                FDB  $3842
23 0F78 A  3942                FDB  $3942
24 0F7A A  0001                FDB  $0001
25 0F7C A  3A42                FDB  $3A42
26 0F7E A  3B42                FDB  $3B42
27 0F80 A  3C42                FDB  $3C42
28 0F82 A  0001                FDB  $0001
29 0F84 A  3D42                FDB  $3D42
30 0F86 A  3E42                FDB  $3E42
31 0F88 A  3F42                FDB  $3F42
32 0F8A A  4042                FDB  $4042
33 0F8C A  0001                FDB  $0001
34 0F8E A  0001                FDB  $0001
35 0F90 A  4703                FDB  $4703
36 0F92 A  0001                FDB  $0001
37 0F94 A  3742                FDB  $3742
38 0F96 A  3842                FDB  $3842
39 0F98 A  3942                FDB  $3942
40 0F9A A  0001                FDB  $0001
41 0F9C A  3A42                FDB  $3A42
42 0F9E A  3B42                FDB  $3B42
43 0FA0 A  3C42                FDB  $3C42
44 0FA2 A  4442                FDB  $4442
45 0FA4 A  3D42                FDB  $3D42
46 0FA6 A  3E42                FDB  $3E42
47 0FA8 A  3F42                FDB  $3F42
48 0FAA A  4042                FDB  $4042
49 0FAC A  0001                FDB  $0001
50 0FAE A  0001                FDB  $0001
51 0FB0 A  4702                FDB  $4702
52 0FB2 A  4802                FDB  $4802
53 0FB4 A  3742                FDB  $3742
54 0FB6 A  3842                FDB  $3842
55 0FB8 A  3942                FDB  $3942
56 0FBA A  0001                FDB  $0001
57 0FBC A  3A42                FDB  $3A42
```

```
 1 OFBE A  3B42                FDB  $3B42
 2 OFC0 A  3C42                FDB  $3C42
 3 OFC2 A  4442                FDB  $4442
 4 OFC4 A  3D42                FDB  $3D42
 5 OFC6 A  3E42                FDB  $3E42
 6 OFC8 A  3F42                FDB  $3F42
 7 OFCA A  4042                FDB  $4042
 8 OFCC A  0001                FDB  $0001
 9 OFCE A  0001                FDB  $0001
10 OFD0 A  4702                FDB  $4702
11 OFD2 A  4802                FDB  $4802
12 OFD4 A  3743                FDB  $3743
13 OFD6 A  3843                FDB  $3843
14 OFD8 A  3943                FDB  $3943
15 OFDA A  0001                FDB  $0001
16 OFDC A  3A43                FDB  $3A43
17 OFDE A  3B43                FDB  $3B43
18 OFE0 A  3C43                FDB  $3C43
19 OFE2 A  4443                FDB  $4443
20 OFE4 A  3D43                FDB  $3D43
21 OFE6 A  3E43                FDB  $3E43
22 OFE8 A  3F43                FDB  $3F43
23 OFEA A  4043                FDB  $4043
24 OFEC A  0001                FDB  $0001
25 OFEE A  0001                FDB  $0001
26 OFF0 A  4703                FDB  $4703
27 OFF2 A  4803                FDB  $4803
28                                END
```

SYMBOL TABLE

BADDCL	0A59	BADDR	0A65	BREAK	0AAE	BYTE	0A73	CHECK	0AB6
CNTRL	0956	C1	097F	C2	0982	DFUNC	0985	DF1	0988
DSMBL	0BA3	D1	0C1C	D10	0CE4	D11	0CFF	D12	0D08
D13	0D0C	D2	0C6L	D3	0C70	D4	0C85	D5	0CA0
D6	0CA5	D7	0CC5	D8	0CD4	D9	0CDD	EXIT	0ABD
GFUNC	09EC	GF1	0A10	H1	0A93	H2	0A94	INCH	090C
INHEX	0A7L	INPARM	09CE	IP1	09E5	ISWI	0B11	I1	0B18
I10	0B7F	I11	0B80	I2	0B1B	I3	0B27	I4	0B2F
I5	0B44	I6	0B4C	I7	0B54	I8	0B77	I9	0B7A
LBUF	0922	MCL	0B9C	MCLP	0B8A	MFUNC	0A17	MF1	0A1A
MF2	0A44	OPER	0919	OPERA	0917	OUTCH	090F	OUTM	0909
OUTR	0ABE	OUTS	0900	OUT2HS	0903	OUT4HS	0906	RFUNC	0A11
RMES	0B92	RSWI	0AF7	R1	0B08	SAVA	091B	SAVB	0914
SAVX	0915	SP	0912	START	094A	SWI	0AE0	S1	0AEA
TBL1	0D19	TBL2	0DF4	TEXIT	0A97	TFUNC	099C	TF1	09AA
TMP1	091C	TMP3	091F	TMP4	0921	TYPE	091E	T1	0AA9

. ASCT	0FF4	00
. BSCT	0000	01
. PSCT	0000	02
. DSCT	0000	03
. CSCT	0000	04

ERRORS DETECTED: 0
FREE CORE: 12587. WORDS

TRACEC.OBJ,TRACEC.LST/C=TRACEC.MAC