Teaching Effective Troubleshooting In The Microprocessors Lab

Thomas E. Gendrachi, P.E.
Ward College of Technology, University of Hartford

Introduction

One of the most important functions of an engineering technologist, regardless of specific discipline, is her ability to solve real, practical problems. Most of the problems students solve are paper and pencil textbook problems written by the author of the textbook. Sometimes you will find troubleshooting problems included in the problem sets at the ends of chapters but, once again, these troubleshooting problems are paper and pencil textbook problems limited to creativity of the author. As good as some of these problems are, they are not “real” problems.

For example, a student might encounter a “real” problem in the laboratory setting up a circuit for a fundamental AC or DC course. The student realizes something is wrong when he does not get the expected results. The cause is usually a setup problem; something is not connected right. After checking the schematic, the student can determine the problem by inspection and correct the error. However, this technique does not work very well when the circuit the student has to construct involves a significant amount of wiring and components as in a memory circuit added to a computer. In the microprocessor, lab this is often the case.

The student cannot achieve the learning objectives of the lab if their circuit is not set up correctly. But instead of spending too much time checking every connection, the student can more quickly determine the problem if they are shown how to apply their knowledge of the operation of the circuit and apply some simple tests based on this knowledge. The reward for the student is that, in addition to achieving the learning objectives for the experiment, he also develops confidence in solving a “real” problem by applying his knowledge.

In this paper, I will discuss an example of a problem students usually encounter in the microprocessor lab. The problem demonstrated in the example is specific but the technique, identifying symptoms and performing appropriate tests, is general and can be applied to any problem.

A Common Laboratory Problem

We use a typical microcomputer in the laboratory component of our second microprocessor course. The operating system of the microcomputer is contained in a 1 kilobyte ROM chip. The primary input device is a multi-function keypad that enables the user to enter and execute programs. The programs can be executed without stopping or single-stepped. There is also a feature that causes a program to be halted at a user-specified "break" point. The programs are entered in machine code as hexadecimal values. The primary output device consists of six seven-segment displays. Using these displays and the keypad, the user can selectively examine the contents of a memory location or the contents of the microprocessor registers, change values in memory, and monitor program execution in single-step mode.

During a typical experiment, the students construct a circuit using some combination of integrated circuit (IC) chips that include standard logic gates, memory, input-output (I/O) buffers, and simple I/O devices such as pushbuttons and light emitting diodes (LEDs). The objectives of each experiment are to demonstrate progressively complicated concepts of interfacing a computer to external circuits. If there is a problem with the circuit the student has constructed or the software the student has entered, the demonstration fails and the student learns nothing and his confidence suffers. For both these factors it is important that the student have an easy, quick, and understandable method of finding and correcting the problem.

The experiments involve a substantial amount of wiring. Students connect the various components on a solder less circuit board using pieces of insulated, small gauge wire with the ends of each piece of wire stripped of insulation in order to make good electrical contact. In addition, the students must enter software via the hexadecimal keypad in a machine language format. Once these tasks are completed, the students are instructed to execute the program they entered and observe the results. These observations may include examining memory locations and/or noting information displayed on the built in seven-segment displays or the LEDs in the students' constructed circuit. Oftentimes students will make errors in wire connections or software entry or both. The result is the students' program does not run or produce the correct output in the circuit they constructed.

The challenge for the students is to determine what the cause of their problem is. Inexperienced students usually will completely re-wire their circuit and/or re-enter their software. This takes a lot of time and most times does not solve the problem or introduces new problems. They have not learned to apply what they learned in class. As the semester proceeds, they learn to pay attention to the results they are getting when they have trouble with their experiment working correctly. They learn to use this information and apply the theory to help them locate the problem. But it takes some instruction to get them to this point and to have them feel confident about what they are doing. Many times it is just a few wires or less that have been incorrectly connected.

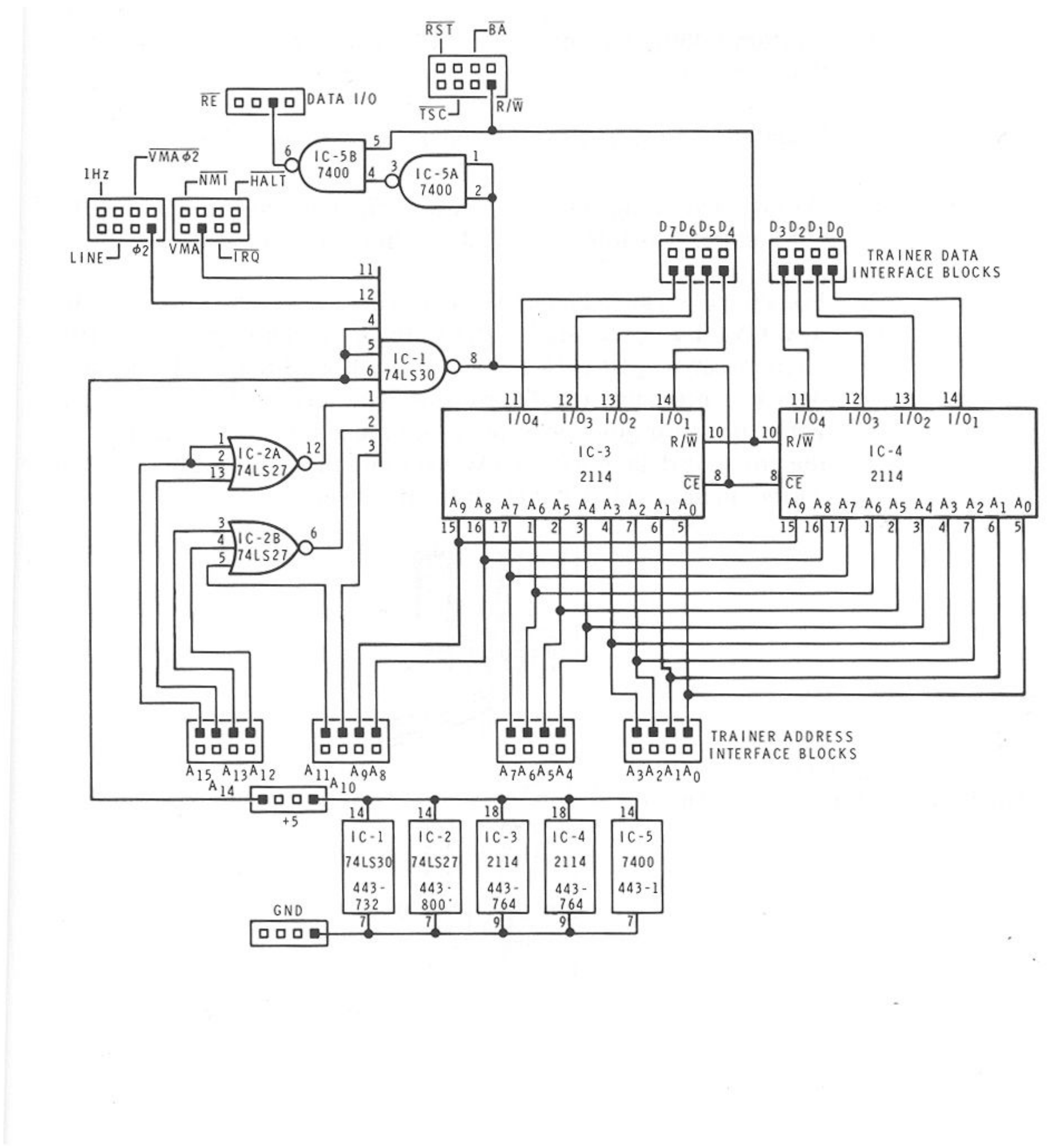

Figure 1 [1]

The circuit shown in Figure 1 above is part of an experiment where the students are adding 1 kilobyte of memory to the microcomputer. The circuit has 3 major parts-address decoder, control circuit, and memory. The address decoder is responsible for “turning on” the memory chips at the proper time. The control logic insures there is no “data collision” when data is transferred between the memory chips and the microprocessor chip. A common problem, because of a misplaced wire or two, is that the students are not able to save values in the memory chips of the circuit they just constructed. The students are taught some diagnostic techniques to help them more quickly determine what is causing the problem.

The first part is to write diagnostic software to test their circuit. For example, here is some software that a student might use to test the address-decoding portion of the circuit. (The $ indicates a hexadecimal value) This software is an endless loop designed to periodically assert the address $0400 on the address bus of the microcomputer.

```
X       LDAA        $0400         Load accumulator A with the
                                  contents of memory location $0400.

        JMP    X                  Jump back and repeat the load
                                  accumulator A
```

The students are then instructed to use a dual channel oscilloscope to monitor two signals, "RE not" (active low read enable) and A10 (bit 10 of the address bus) on the microcomputer. When the diagnostic software is executed, the signal patterns appearing on the oscilloscope screen should appear as shown in Figure 2 below, if the circuit is correctly wired and the software correctly entered.

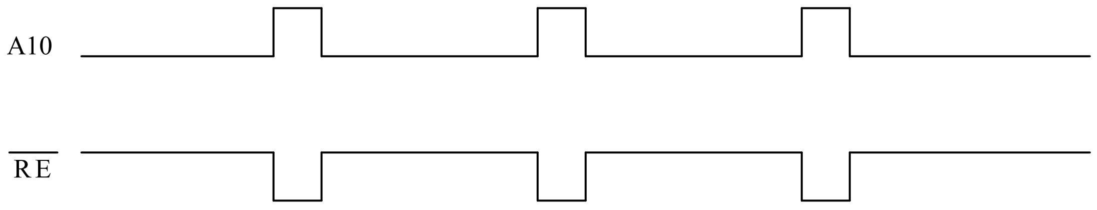

Figure 2

The reason for these patterns is as follows. The value $0400 appears on the address bus only when the LDAA $0400 instruction is executed. This causes A10 to be logically high. Since the program is an endless loop, A10 goes high each time the LDAA $0400 instruction is executed. A10 is low at all other times because none of the other locations appearing on the address bus during program execution cause A10 to be logically high. The read-write memory available to the user for her programs on the microcomputer resides in locations $0000 through $01FF.

When there is a wiring problem with the address decoder, signal "RE not" is usually corrupted because the student constructed circuit is responsible for producing this signal. This signal is responsible for enabling an I/O buffer that allows data to pass from the memory chips of the student constructed circuit onto the data bus of the microcomputer. The A10 signal, on the other hand, is generated by the microprocessor as it executes the diagnostic software. This is where the students get a chance to apply their digital knowledge and learn a lesson about the dynamic operation of the system that includes the circuit they have just built. It adds a very important, practical, unintended and potentially valuable dimension to the experiment by allowing the students the opportunity to solve a complicated, practical problem.

The students are asked to examine the circuit schematic and to find the logic gate in their circuit producing the "RE not" signal and trace the signal back to its sources. Looking at Figure 1, the students can see the signal "RE not" is the output of a NAND gate (IC-5B). The inputs to this NAND gate are the R/W not signal from the microprocessor and the output signal of another NAND gate IC-5A, which is functioning as an INVERTER since its inputs are tied together. They continue to follow the circuit back and find that the signal "RE not" is produced by various address bus and control signals from the microprocessor applied to their constructed circuit. The students are then asked to determine the logic state for every output and input of their circuit that will produce a logic low for "RE not". They can use this information to look for a signal that is incorrect.

For example, the students know, from basic digital logic theory, that in order for "RE not" to be low, the inputs to IC-5B, R/W not and the output of IC-5A must both be high simultaneously. They can look at these signals with their oscilloscope. If they find that both of the input signals are correct there probably is a problem with gate IC-5B. Maybe they used the wrong chip or maybe the NAND gate is bad. The important thing is that they have found an inconsistency that is most likely the cause of their problem. The output signal of the NAND gate (IC-5B) is incorrect given its input signals. On the other hand, if one or both of the input signals is incorrect the students are instructed to look further for problems. Maybe one of the input wires to IC-5B is misconnected? The important thing is the students use their knowledge to locate and correct their problem. The circuit operation "comes alive" for them.

As another example, if the student finds that the output of IC-1 is high when it should be low. The student should look at the inputs of IC-1 to see if any of them were low causing the incorrect output. If an input were low, this signal would have to be investigated further. As simple as this action might seem most students lack the confidence to take this action. However, once the students have successfully corrected a problem using this approach, they find that they have a new tool to solve their problems, one that makes use of the theory in a practical way that is an important component of their intended profession, independent problem solving based on application of knowledge using available data. The lab experiments can go well beyond their primary intent by building student confidence in solving "real" problems.

Conclusion

We need to give the students the opportunity and experience of solving "real" problems in their technical courses. By requiring students to physically build complex circuits in laboratory experiments, they usually get that opportunity and experience because they have to correct a misconstructed circuit.

We also need to give them a systematic approach for correcting their problem. Since this systematic approach makes use of their theoretical knowledge, it reinforces their learning and builds self-confidence.

The result is that the students are better prepared to enter the workforce with skills that they need to succeed.

[1] Microprocessors Workbook, Heathkit Educational Systems, page 15

THOMAS E. GENDRACHI P.E., Associate Professor Electronic Engineering Technology