Reducing the Trial-and-Error Design Method of Single Board Computer Projects via Automation of Timing Measurements

Charles Duvall, Joshua Galloway
cduvall@spsu.edu, jgallow2@spsu.edu
Southern Polytechnic State University
1100 S. Marietta Parkway
Marietta, Ga. 30062
678-915-3152

Reducing the Trial-and-Error Design Method of Single Board Computer Projects via Automation of Timing Measurements

Engineering-Technology students are challenged when first required to follow a complete industry-like project development cycle of designing, constructing, and testing a complex project. In single board computer based systems, the overall success of the students' efforts can be attributed to their ability to separate hardware problems from software problems when the synthesis phase begins. Unfortunately, the appropriate test definition and manual use of general purpose logic analyzers for hardware timing verification is complex. Subsequently, many students resort to a trial-and-error method of problem isolation and resolution by frequent changes to the system software. This paper will outline the automation of key hardware tests that will allow students to focus on the design, development, and analysis of the flow of information through the computer system.

A typical semester length project involves significant planning and subsystem development. As the subsystems are integrated into the complete system, testing, analysis, and modification of the behavior of the system are too frequently accomplished with regular reprogramming and changes to the system's software. Single board computer projects may be tested and analyzed with a more sound approach by leveraging debugging tools. Three of the most important tools used in debugging advanced systems are simulators, emulators, and logic analyzers. A simulator is software that allows the designer to imitate the architecture of the target system and to execute his or her program code on a development platform prior to implementing it in the objective machine. It allows the designer to set up various inputs and observe how the code reacts in a situation devoid of hardware influence. The simulator effectively allows the designer to remove the hardware as a variable and test the code in an isolated environment.

Emulators are devices that attempt to imitate various other devices in the target system by replacing the chosen processor with one that is under direct control of the development system. The emulator runs the designer's code in the target system while giving the designer the ability to start and stop the program at will. This is useful in that it permits the designer to take measurements or view the contents of allocated memory blocks in his or her code at any given point in the code's processes. Emulation allows the designer to introduce the hardware variable into the equation and closely observe how the

code reacts.

Once the hardware/software synthesis begins, the logic analyzer becomes an extremely useful tool. A logic analyzer allows the designer to view the numerous digital signals present in a given system simultaneously with relation to value (1 or 0) and time. This is useful because an emulator may have slightly different electrical characteristics than the actual device to be used in the system. These differences can cause the system to fault once the "real" device is implemented in the target board. The logic analyzer is a powerful computing platform and allows users to define test conditions where very specific events trigger measurements and the creation of a display like Figure 1. If the system does not function as intended, the visual verification of the signal's value and timing provided by the logic analyzer will allow the designer to start the trouble shooting process with enough information to pinpoint the problem as hardware, software, or both. Students are given an introduction to the instrument, but face a learning curve related to understanding the operation of the multi-buttoned front panel of the powerful and flexible logic analyzer, see Figure 2.

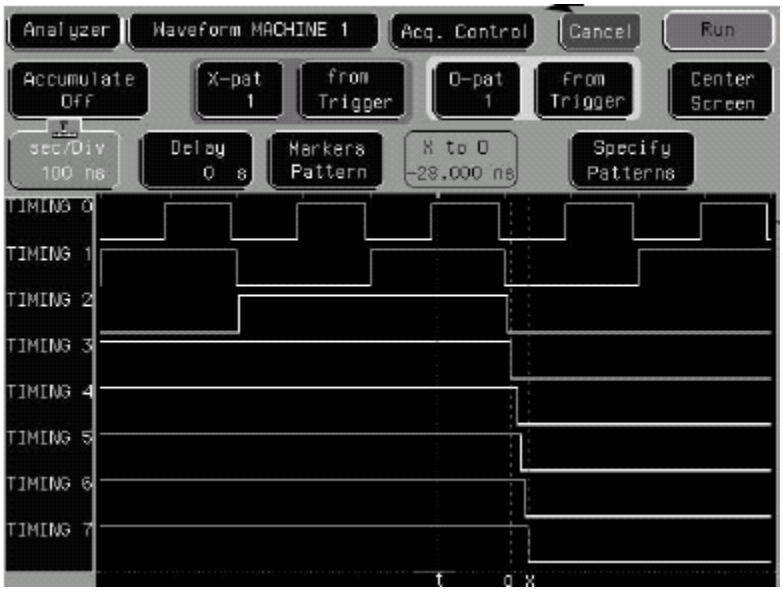

Figure 1 – Timing Measurement from Logic Analyzer

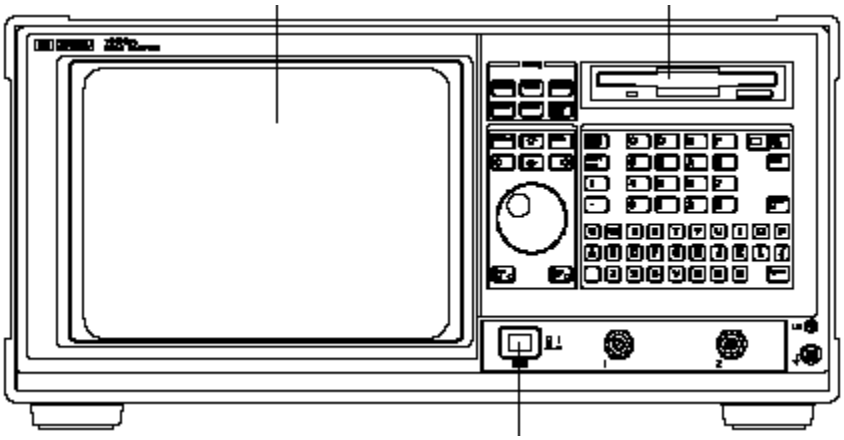

Figure 2 – Logic Analyzer Front Panel

The test definition, configuration, and verification of timing signals becomes more complex in single board computer systems when many devices or additional memory beyond the central processing units capabilities are required. Typical connections in and out of the computing platform are sensors, motors, and displays for the users to understand the state of the system. Two approaches are used to make the connections: one where each device has its own dedicated address and data connections (Harvard architecture), and another where there is a single shared address and data connection used by all devices (Von Neuman architecture). The shared, or bus, approach provides the most extensibility and is the preferred physical approach (less wiring, less space….), but challenges the students' understanding of the general purpose logic analyzer to configure, test, and analyze the performance of the address and the data signals. When faced with the choice of understanding a flexible general purpose test platform or frequent reprogramming of the code for their project, too many students resort to a trial-and-error design method of changing the code repeatedly. Depending on the nature of the system, the reprogramming could lead to problem resolution, but a much more efficient method is to leverage the logic analyzer to delineate hardware from software problems. To reach this goal, the test definition and analysis of extended memory systems has been automated using a test and measurement configuration via a

LabView program, which gives the students a much simpler interface for testing.

The logic analyzer has been interfaced to a personal computer with the front panel buttons being automated using LabView, a program that can be used to simplify the use of general purpose test equipment for a specific task. The complex interface of the logic analyzer front panel has been simplified for the students to an input, the desired address on which to begin (a.k.a. "trigger") a timing measurement, and an output, the time delay results (Figure 3).

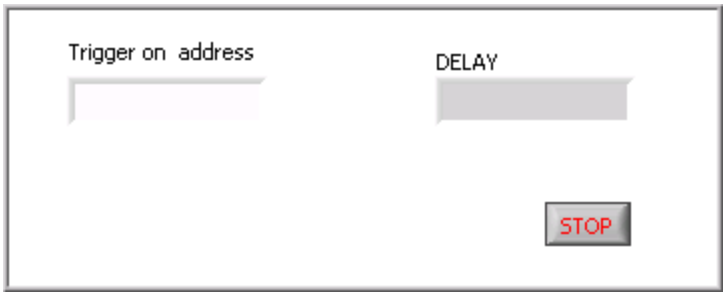

Figure 3 – LabView based user interface for timing measurement

The student's proposed project was to build a password protected entry system for a door that could also provide verbal prompts to the user. The main input device for the system was to be a four by four keypad, and the main outputs were to be a 20 character by 4 line liquid quartz display (LCD) and a speech synthesizer. The system was also to allow for the editing of usernames and their individual passwords by way of an administrative mode, and provide a manual override switch to allow entry through the use of a key. In the event of a fire, the system was to immediately open the door and provide audio prompts to elicit any personnel in the room to exit.

Various integrated circuits and devices were used to implement the project. The processor used was a compulsory 8051 core microcontroller, specifically the Atmel AT89C51. The microcontroller's on-chip random access memory (RAM) was inadequate

for storing a reasonable number of usernames and passwords, so the RAM was extended off-chip with a 2 kilobyte by 8-bit static RAM chip and a battery back-up. To realize the temperature sensor, an ADC80804 analog-to-digital converter (ADC) was used along with an LM34 precision Fahrenheit temperature sensor. Speech was synthesized with an addressable digital voice recorder.

The overall system flow was as follows. The System in idle mode would display the current temperature until a user was present. Once a user was detected, the system would ascertain through the administrative switch which mode to enter—administrative or normal user. In the normal user mode, the user was to be prompted to select his or her user name form a list; following which, he or she would need to enter their five digit code to gain access to the room. The system was to allow three tries to enter the correct password before defaulting back to the temperature display.

When an administrative user was present, the system was to ask the administrator to select whether he or she would like to add or delete a user. If he or she wishes to enter a user, the system was to prompt the administrator for the new username and password, which would then be stored in the RAM. In the event that the administrator wanted to delete a user, he or she would have to select the user, press delete, and then confirm the deletion. The administrator would then be asked whether he or she would like to delete another user or exit the administrative mode.

Impact of the automation on student design practices will be discussed with a specific example of a semester length single board computer project that is currently in progress will be discussed at the presentation of this paper.