Work in Progress - Using A Graphical Programming Language Teach to Microprocessor Interfacing

Gary Rommel
Department of Computer Science
Eastern Connecticut State University
Willimantic, CT 06226, USA
rommel@easternct.edu

Abstract – At Eastern Connecticut State University we have designed our computer science program around the ACM Computing Curricula 2001 Computer Science [1]. This is a fairly flexible guideline and permits distinctive solutions to the curriculum.

Because we are a liberal arts institution and teach students in a computer science program, hardware courses present themselves as challenges to instructors. On one hand hardware courses expand our students abilities while on the other hand they challenged them because they are very different than software courses.

We address this dilemma by using a graphical programming language. We use a similar approach developed by Jones, Lehrman, and Rogers [4].

Our situation differs from [4] in that we have a homogeneous student body of computer science majors. However, we have students that have both little and some hardware background. Hardware can be viewed as threatening for students with only a software background.

Like [4], we use sound and music to motivate difficult computer science principles such as Nyquist’s Sampling Theorem, Spectrum Analysis, and FFTs. By permitting students to use their intuition, they are able to understand and see these principles in a non-threatening way. In fact, our approach becomes a motivating method for the class.

Once we decided to use a graphical programming language, we needed to address four aspects of the course

1) The laboratories
2) The instructor’s teaching station
3) Course materials
4) Remote sharable stations

The paper describes these aspects of our course.

Introduction

Our goal in re-examining the microprocessor interfacing course was to make the major points of the class easier to understand and easier to implement as well as relevant to our twenty first century students. Among the major points of the class are an

1) Understanding of programming I/O devices
2) Understanding of latches and registers
3) Understanding of ADC and DAC
4) Understanding of serial and parallel communication
5) Understanding of status and data registers
6) Understanding of DMA

The microprocessor class has often been labeled by students as a difficult class. We immediately consider the graphical programming languages such as Labview [2] and Agilent’s VEE [5]. Because of the maturity [2] [3] its origins in 1984, and the ubiquitous usage, it can be found in the leading pharmaceutical industries. We wanted to eliminated work of wire warped circuits, proto-boards or intergraded circuit boards. However, we did not want special purpose units that could not be programmed. First, it would be expensive for students. Second, these units would not have a universal usage.

For the equipment required for our PCs, we choose the PC boards available. These included the NIC card, sound board, parallel port and the serial RS232 port. This gave us a diversity of interface approaches: serial, parallel, DMA. Certainly, a sound card has both ADC and DAC when we considered the microphone pickup and the speaker output. Most sound cards have multiple sampling rates. A typical card may have sampling rates of 8000, 11025, 22050, and 44100Hz.

We wanted to replace more than our special purpose cards with standard boards. We wanted to replace an old approach with a new approach. We also wanted to motivate our student as [4].

Once we decided to use a graphical programming language to teach PC interfacing, we had to address four aspects of the course:

1) The laboratories
2) The instructor’s teaching station
3) Course materials
4) Remote sharable station

Laboratories

Our course is structured on two pedagogical approaches:
1) Requirements
2) Imagination

We set the following requirements for each microprocessor interfacing laboratories

1) All laboratories must be self-motivating. By this we mean that the student should be motivated to do the laboratory for its inherited value.
2) All laboratories must be unique to each student. No two students can possibly duplicate the same laboratory.
3) All laboratories must be able to be done either in the department’s laboratory or in the student’s dormitory room.
4) Laboratories should build on the previous one.
5) Each laboratory should have at least one new theoretical contribution.
6) Each laboratory should have at least one new measurement contribution.

The natural consequence is to set up fixed requirements and a certain level of freedom for the students. To achieve this we incorporated:

1) 10 fixed laboratories
2) 1 creative laboratory

I. Fixed Laboratories

The motivation behind the fixed laboratories is that they are intended to give students confidence. These laboratories are:

1) Write a song with tones
2) Write a song with chords
3) Observe a song with your microphone.
4) Design a laboratory using UDP
5) Design a laboratory using TCP
6) Design a laboratory across the web
7) Design a home security system using the parallel port
8) Integrate the COM port with another computer
9) Integrate a storage device with any other device
10) Design a laboratory using a web server with an interface card

- **Tones** In the first laboratory each student must select a unique song to demonstrate. The students must go to the discussion board and find a song not selected by another student. Once they have selected a unique song, they list it at the discussion board. This reserves their song and prevents other students from using it. Once the song is selected, it must be played using tones. Labview comes with a sample that does this for a single note. However, the student must choose the method of implementation.

In general, students used two methods to implement their designs. The first approach uses a series of case statements to determine the volume and pitch of each note in the song. While the other approach uses an array to hold the volume and pitch of each note in the song. In both cases, they were very motivated. The key was to design an electronic metronome for their application.

In Figure 1 we give the control panel for our program that plays a simple sound. We implement this program using a case statement. In Figure 2 we give the block diagram of the song program. Writing a simple song requires that the student understands frequencies of pitch. It also requires that the student understand timing and duration. Let’s not forget about volume.

- **Chords** Let’s consider the chord laboratory. Students were instructed on the proper method to make a chord. This laboratory motivated our students to understand important mathematical concepts such as the Fourier Series and to a lesser extent the Fourier Transform. Because Labview contains these as modules, the student was able to focus on the data acquisition.

The real data acquisition issue in this laboratory is sampling rate. At a sampling rate of 8000 Hz a C major chord is lost. The students learn that sampling rate selection determines if the chord can be heard. Figure 3 shows the three frequencies that make up the C major chord. Students are also asked to listen to the chord and watch its power spectrum.

- **Microphone** In the next laboratory, the microphone laboratory, students are again asked to select an instrument. Here we consider the transient generator of the instrument. As before the student is asked to find the power spectrum of the music from the instrument. The student now is able to see a closed loop system. The first part of the laboratory as before sends out an audio set of notes, their song, while the new part of the laboratory receives the audio using the microphone. The audio information picked up by the microphone is displayed in both the time and frequency domains.

- **UDP and TCP** The UDP and TCP laboratories are demonstrated at the instructor’s station using two laptops.

Fig. 1. Front Panel

Fig. 2. Block Diagram

Fig. 3. Chords
and a hub. The hub is a simple 4 port unit that fits in a PC case. Using the freeware [7], [6] the instructor can easily demonstrate a TCP session by following the SYN packet from the client, the SYN A CK from the server, and the client’s ACK. Along the same lines the DHCP, DSN, and HTTP protocols can be addressed. Because it is done with the hub, the promiscuous sniffer picks up all communication and displays it on the projector. Figure 4 demonstrates an actual connection to a web server. Here we can see the request for the DNS server. The demonstration easily enforces a quick understanding of each protocol. The students are then asked to design a laboratory with UDP and TCP protocols on their own.

- **Web Server** The web laboratory is generally viewed by students as the most amazing. In the demonstration phase, the instructor sets up a laboratory. We use a temperature probe. Next, we set up the web server with the temperature laboratory. Then we browse to it with the second laptop. Next, we control the experiment with the client browser. Finally, we have the server take back control of the experiment. Students are then asked to consider their own experiment. During the experiment they are asked to monitor the network traffic.

- **Parallel Port** The parallel port laboratory gives an understanding of parallel interfacing. Here we introduce latches and registers. We also explain LEDs and switches. We center parallel interfacing around the home security system. Students are again permitted to use their creativity.

- **Serial Port** The serial communication laboratory takes advantage of the serial port of the PC. Here we introduce the crossover cable. We demonstrate that we can easily connect two computers together using such a cable. Students are then asked to design a laboratory interfacing two computers with a shared experiment.

- **Storage** The storage laboratory gives the student some basic knowledge of both reading and writing to secondary storage. Again the student is asked to develop their own laboratory.

- **Creative** The last laboratory uses all the knowledge of the other laboratories.

Yes, we consider typical topics such as optional amplifiers, register, latches and gates.

### II. Creative Laboratory

It is important to understand that the second approach is purely creative. We require that each student develop one laboratory that is unique to the student. Perhaps, a more enlightened way to say this is that our class is both structured and free form.

A course must allow a student to use the material of that course to develop material that is unique to the student but related to the course.

As with nearly all our assignments student must first go to a discussion board to sign up for a topic.

### Instructor’s Station

We set the following requirements for the microprocessor interfacing instructor’s station:

1. The instructor’s station must be composed of one to two laptops.
2. The instructor’s station must have access to the internet.
3. The instructor’s station is connected to either an LCD or a digital light processing projector.
4. The two laptops are connected together through a simple hub.
5. All laboratories could be performed at the instructor’s station.

The instructor’s station includes a copy of Labview, a promiscuous sniffer, and a browser. We use a hub to communicate between laptops and the web.

### Course Materials

We set the following requirements for the microprocessor interfacing materials

1. All laboratories are accessed through the internet.
2. All software is available to students either as downloadable freeware [7], [6] or student licensed software [2].
3. All student selections are based on queries to a discussion board.
4. All student grades are immediately available to students.

### Remote Station

We set the following requirements for the microprocessor interfacing remote station

1. This facility is accessible through the internet.
2. This facility address instrument needs beyond those of the student facilities
3. This facility has at least
   a) 16 - 16 bit ADC,
   b) 2 -12-bit analog outputs;
   c) 8 digital I/O lines;
   d) two 24-bit counters.

This facility acts as a sharable resource which can be used on high-end problems.

### Impact on Students’ Learning

It is most important to have students self motivated. To do this we present simple but important laboratories to our students.

A chord is easy to understand, but extremely complex to appreciate. Why is the C major chord acceptable in the western society, but not in all societies? We can introduce the
harmonic scale with the chord. Likewise, we can introduce the mathematics of the scale by showing that the $n^{th}$ note in the A scale is

$$note_n = 440(2)^{(1/12)n}$$

For example, if we consider prefect $A = 440\text{Hz}$, then the next half is $A_2 = 466.16\text{Hz}$?

Our Internet laboratories were equally motivated. When we demonstrated TCP/IP, we carefully motivated the students. We first showed a simple UDP exchanged. Then, we followed it by a TCP session. We could easily show the sequence numbers and the corresponding acknowledgements using both Labview and Snort. However, the clear winner was the remote instrument web server. Having the students see the exchange of command between the network browser and the instrument web server, clearly motivated the educational process. Seeing it both at a high level, browser and server, and at the packet level, gave our students a unique window into the internet.

Students could easily measure the value of the fixed laboratories. With that measure in mind they could see what was needed in the creative laboratory.

We believe that it is very important for students to have access to this material in a reliable fashion. To this end, we have decided to use Course Compass. We have made heavy usage of the discussion boards and the external links.

**Metrics of Success**

There are several measures of success. The simplest is student retention. Because this was a required course, retention could not be used a fair metric.

As we have said throughout the paper, the real measure is self motivation. Each laboratory posed a slightly different problem as we stated. However, no problem was very developed. If we look back to Figure 1 and Figure 2, we see that the solution is not complex. However, it does require a certain amount of self-motivation. Visual programming is new for most students.

We see that the metric of success is really the number of laboratories completed. If we examine the first and second assignments, we see that most of the students understood the laboratories. Several of the students extended the assignments.

**Future Directions**

There are several directions that we can take in the future.

1) The first is to extend this course to non-majors. We may want to invite graphic arts majors and communication majors into this class. We see the value of programs such as the one at Tufts University[4]. Perhaps, the best way to broaden the enrollment is to write a class similar to [4].

2) We will approach Fourier series and spectrum analysis in a way that is easily understandable. We have started a series of lectures based on Euler’s Equation to explain Fourier series and power spectrum.

3) Another goal is to extend this course to the sciences. We have considered joint work with the astronomy department at one of our other campuses. In this project we will develop a remote laboratories centered around a laser radar (lidar) atmospheric sensing laboratory and an industrial instrumentation laboratory.

However, we believe the key extension is simply to teach the course so that any major can understand the issues of the class. Consider the problem of explaining the power spectrum. Likewise, consider the problem that Nyquist’s Sampling Theorem places on a non-major.

**Conclusion**

Our basic conclusion is that students are really motivated to work on laboratories that they can grasped. Perhaps, this needs real explanation. A simple song can be grasp immediately. However, what is a song? It is extremely complex.

This is the point. A song is a series of tones at different volumes and pitches and durations. The student knows that the program to do this must be clocked, must have different volumes, and must have different pitches.

With a visual language our students quickly grasped the concept of timing. With a front panel with charts, indicator, and controls, students were able to interact with the equipment.

Consider the laboratory centered around creating a chord. We asked the students to listen to the chord and then to see the chord. We gave the students the mathematical relationship among the notes of a chord. Clearly, we motivated the need for the Fourier series and spectrum analysis.

We believe that seeing packets as with Snort, is motivating. Our students immediately captured the major points of the internet. They quickly understood the need and the approach used in ARP. The explanation of the DHCP protocol was apparent. DNS was a snap. TCP and IP protocol was made simple. The Ethernet was clear. The octets of IP and the 48 bits of the MAC address were forever understood.

The ease at which we could make a web server for instrumentation problems motivated our students to design their own server.

In summary, we strongly feel that the graphical programming language with our use of

1) The laboratories
2) The instructor’s teaching station
3) Course materials
4) Remote sharable station

improve the understanding and performance of our students.

**Acknowledgment**

The author would like to thank Kevin Shirey from PVI Systems Inc, Chris Rogers of Tufts University and Jesse Jones of Tufts University who got me started with graphical programming languages.

**References**


October 19 – 22, 2005, Indianapolis, IN

Session S2G

0-7803-9077-6/05/$20.00 © 2005 IEEE


