USB Logic Analyzer

Project Proposal

Shom Bandopadhaya
Advisor: Dr. James H. Irwin

December 8, 2005

Senior Capstone Project
Bradley University
Department of Electrical and Computer Engineering

Introduction

A logic analyzer displays logic level of digital signals. A logic analyzer differs from an oscilloscope which is a more powerful instrument but typically only has a few channels. This project seeks to continue work on a digital logic analyzer which has sixteen channels. The logic analyzer will display four logic levels: low, high, tri-state and indeterminate. The sixteen channels are sampled by an external conditioning hardware called a POD. The POD consists of an Opal Kelly XEM 3001 FPGA board and some external hardware. The POD interfaces to the computer using Universal Serial Bus [USB] protocol. The graphical user interface [GUI] on the computer will display one of the four possible logic levels sampled by the POD. The overall system block diagram can be seen in Figure 1.

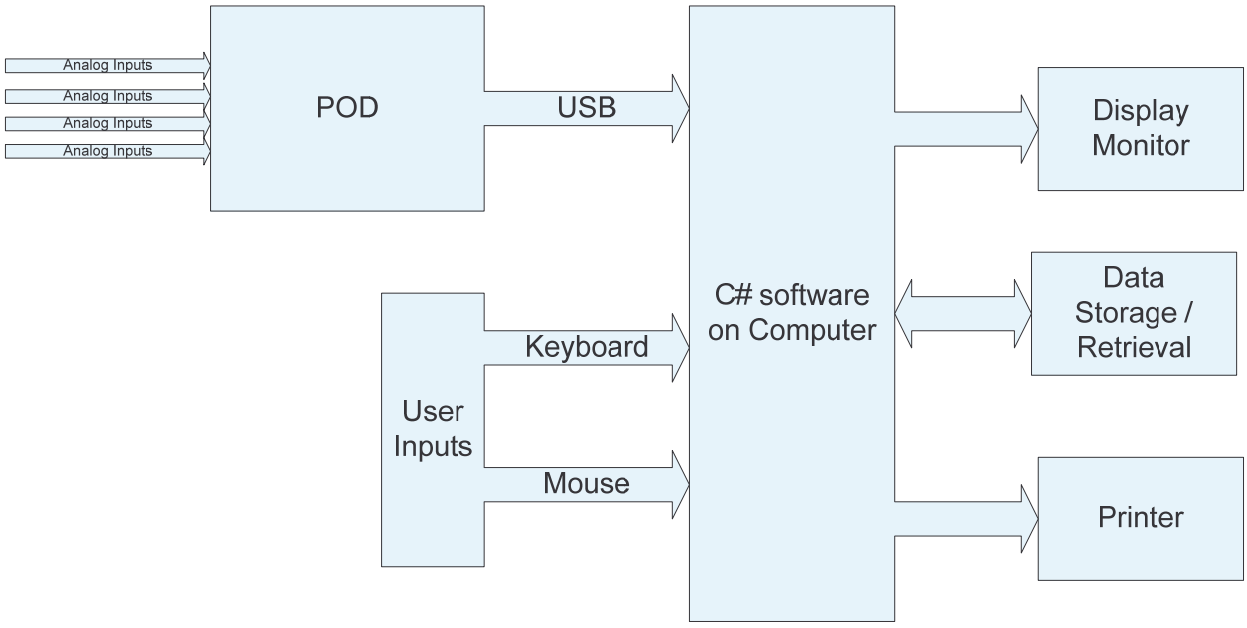

Figure 1: Overall system block diagram for USB Logic Analyzer

Inputs

Analog inputs:	These are the analog voltages on the probes of the POD. The POD handles CMOS or TTL levels.
User inputs:	The user interacts with the software GUI using the keyboard and mouse, and is able to select options, change display parameters, and access additional features like zoom, scroll and save.

Outputs

Display Monitor:	Displays the GUI. The user can see the current logic levels, triggering points and also observe the effects of their interactions with the GUI.
Data Storage/Retrieval:	Saves the captured data frame to disk. The GUI can also open and display previously saved captures.
Printer:	Prints the current screen.

Operation Modes

Option selection:	Allows the user to select from the available options, such as logic type (TTL or CMOS), display window range, etc.
Single Capture:	Captures and displays information from the time the capture was started till the time software buffer was filled.
Continuous Capture:	Continuously captures information into the buffer and displays the current buffer, so the logic levels are shown at “pseudo real-time”.

Goals

- Primary
    - New display engine
    - DLL interface for USB
    - Efficient data processing
    - Print/Save data capture
    - Continuous Update (Pseudo Real Time)
- Secondary
    - Remote Viewing
    - Reverse Assembly code decoding

Software

Conditioned Signals: Signals sent from the POD to the PC with all sampled data in packet form USB interface.

Keyboard/Mouse: Commands entered using keyboard and mouse to setup trigger conditions, change display format, save the waveform, setup, and hardcopies.

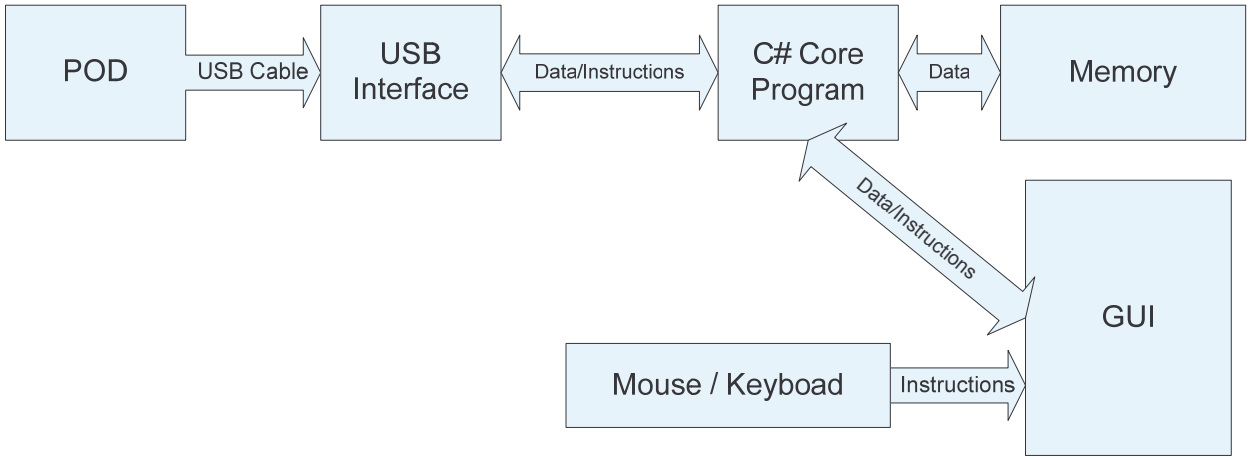

Figure 4.1 Software Subsystem Block Diagram

There are three major sub-blocks, the USB Interface, the C# Core Program, and the GUI as can be seen in Figure 4.1.

USB Interface

The USB interface uses the Opal Kelly provided library to connect to the FPGA using a USB 2.0 connection. It acquires data when requested by the C# core program and transmits the data back to the C# core program through the USB buffer. Figure 5.1 shows the process through which it is accomplished.

C# Core Program

The C# Core Program receives the instructions via the GUI, contacts the USB interface to start Data Acquisition, collects the data and then saves it to memory and sends it to the GUI to be displayed. Figure 5.2 describes the process.

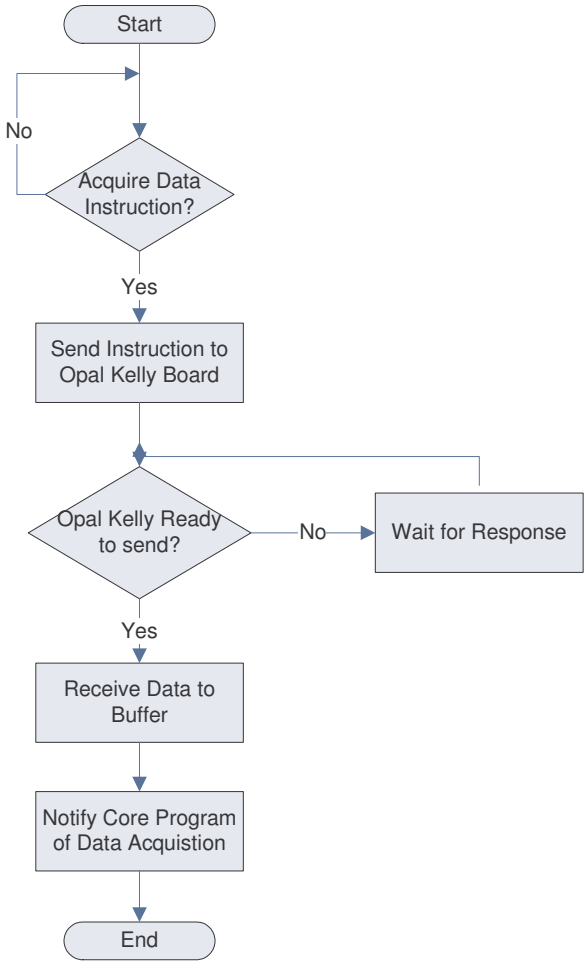

Figure 5.1 USB Interface Tasks

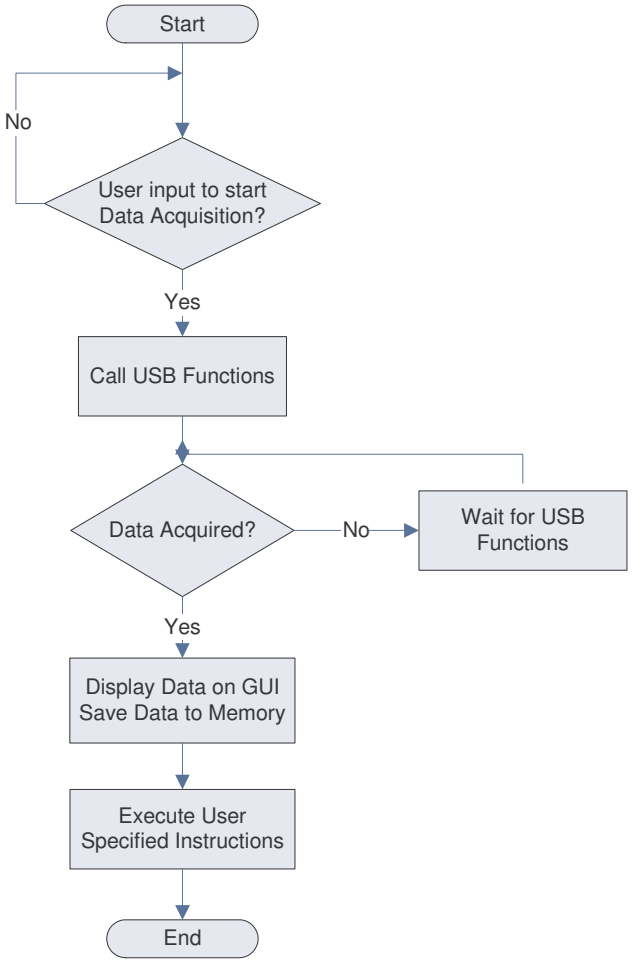

Figure 5.2 C# Core Program Tasks

GUI

The GUI interfaces the user to the C# core program. It receives instructions via the keyboard and mouse and displays the information on the screen. Once it receives any instructions it contacts the Core C# Program to perform those tasks. It idles till it receives the next set of instructions. If the user decides to terminate the GUI it terminates the whole software. Figure 6.1 shows this process.

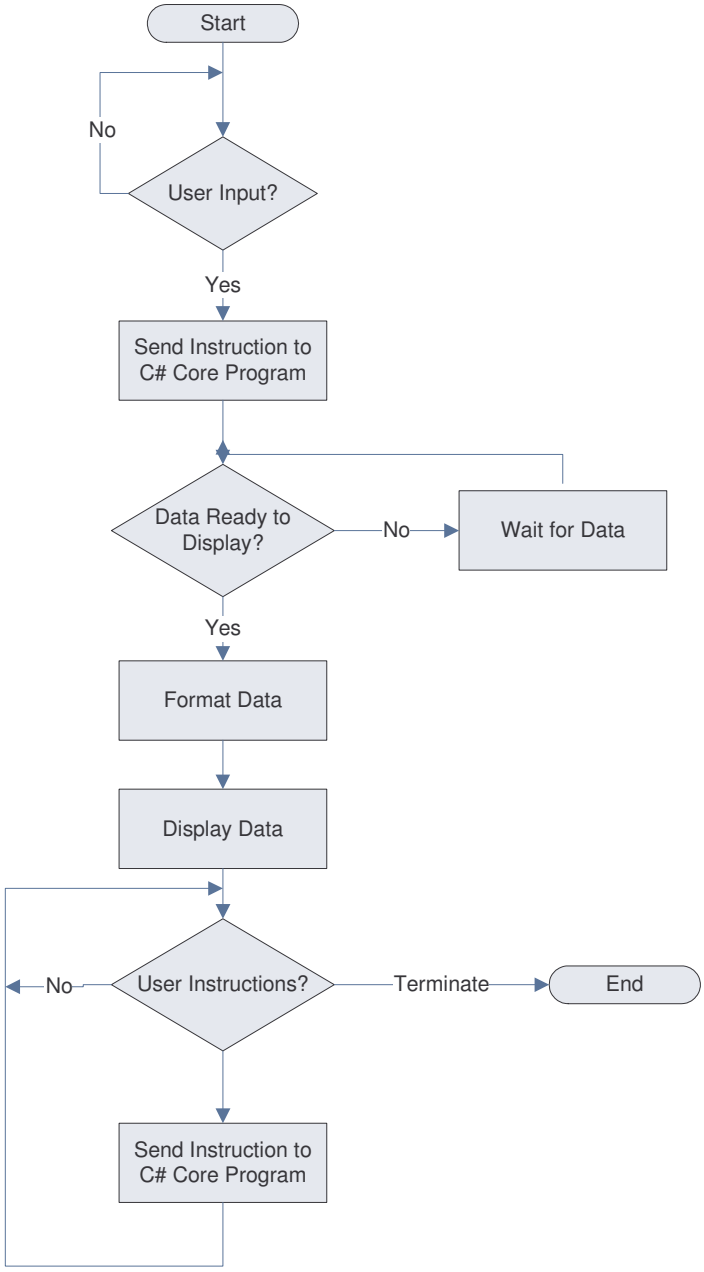

Figure 6.1 GUI Tasks

Goals

The following are the primary goals of the project:

- New display engine for NPlot
- Combine lines into busses
- DLL interface for USB
- Data processing through memory instead of hard drive
- Print/Save data capture
- Continuous Update (Pseudo Real Time)

These goals and their dependencies are reflected in the timeline in the next section.

Schedule

The GANTT charts show the schedule for the project. The start and stop dates, the duration of each task, the percentage of completion of each task, and the inter dependencies of the tasks can be seen in the GANTT charts.

The Fall semester schedule can be seen in Figure 7.1. As can be seen all the tasks scheduled for this semester have been completed, except for "Develop New Graphics Library" which will be completed over Winter Break as denoted by the schedule slip block.

ID	Task Name	Start	Finish	Duration	% Complete
1	Functional Description	10/6/2005	10/26/2005	3w	100%
2	System Block Diagram	10/27/2005	11/23/2005	4w	100%
3	Project Proposal	11/29/2005	12/8/2005	1.6w	100%
4	Notebook/Website	10/6/2005	10/26/2005	3w	100%
5	Learn C#	10/13/2005	11/2/2005	3w	100%
6	Using Graphics Library	11/3/2005	11/9/2005	1w	100%
7	Develop New Graphics Library	11/10/2005	12/7/2005	4w	50%
8	Research Data Structures for Graphing	11/10/2005	11/23/2005	2w	100%

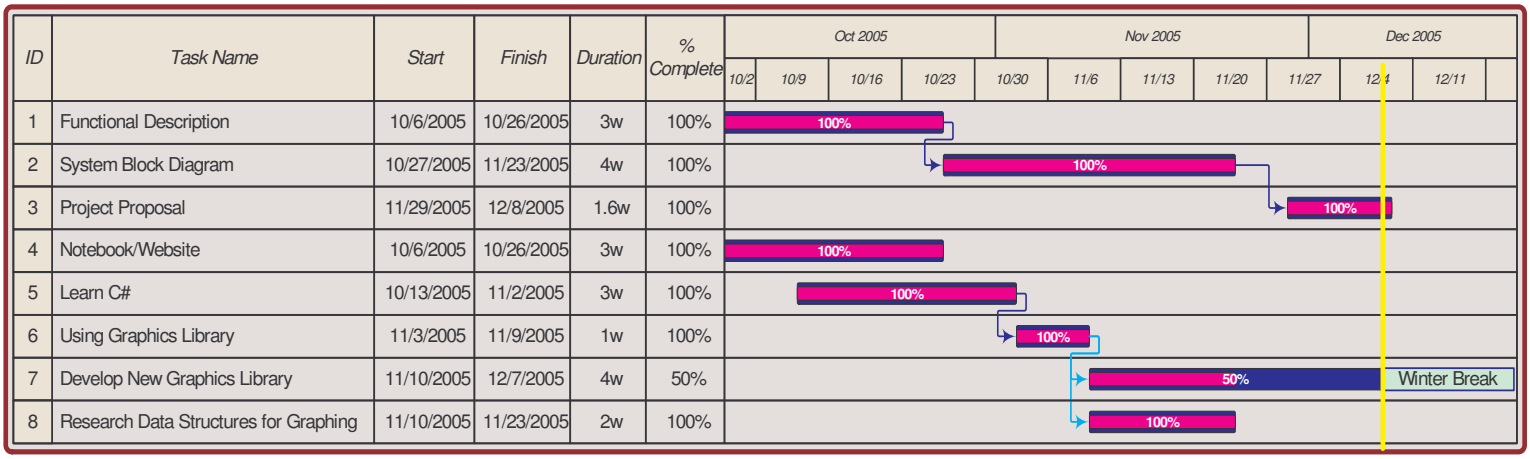

Figure 7.1 Fall Semester Schedule

Not a lot of time was initially delegated to the deliverables part of the project, which caused some delay in other parts. In addition to that there were some tasks were parallel in nature, which also caused some schedule slip. These factors were considered in developing the Spring semester schedule which can be seen in figure 8.1.

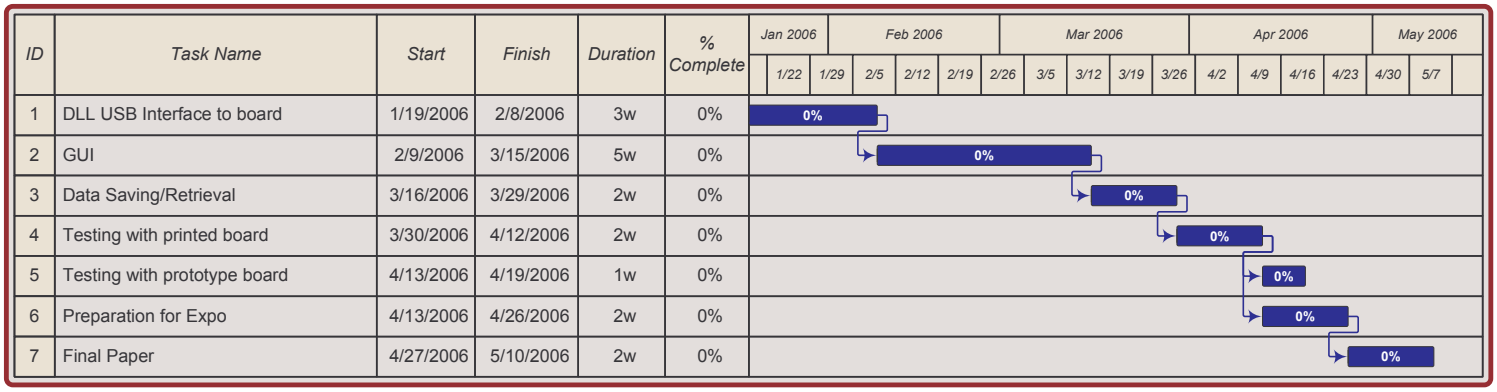

Figure 8.1 Spring semester schedule

Cost Justification

The Opal Kelly XEM 3001 FPGA board is a development platform with a Xilinx Spartan 3 FPGA onboard. The development platform was used to speed up the development time since it implements USB protocol natively, and the hardware part of the project can just be concerned with the external circuitry and VHDL development. The cost of the development platform was $200.00, whereas the Spartan 3 FPGA single unit costs $32.00. If the development board was not used there would be additional costs associated with the fabrication of the external circuitry along with the cost of the signal conditioning circuitry. The development board however can be used in Lab as a test board for other projects as well. If this project was ever to go to production, then the cost of the product is going to be significantly cheaper than equitable commercial solutions that sell for $400.00.

Progress

As the schedule demonstrates a lot of tasks were accomplished this semester. The deliverables have culminated in this project proposal and the research done will be briefly described. After researching and learning C#, which was a significant time investment, the NPlot graphics library was tackled. Test values were then plotted and some user interaction was added to the plot, specifically, the ability to select a region and zoom in on it. After understanding the fundamentals of the design a better GUI can be developed during the Spring semester. Figure 9.1 shows four lines of data being plotted together.

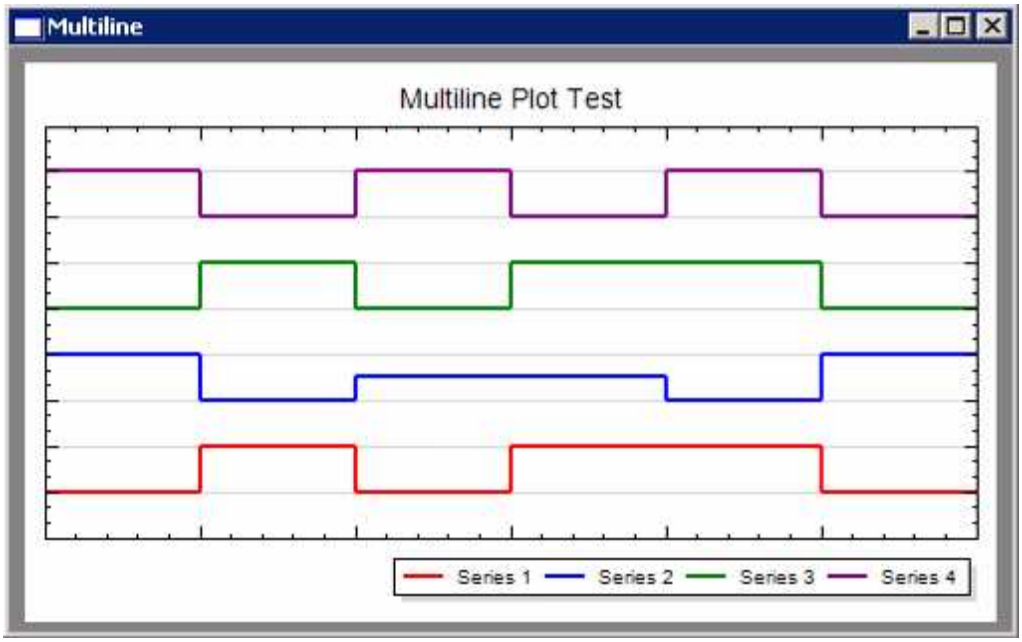

Figure 9.1 Multiple Lines being plotted on one surface

Eventually, the GUI will be modified to hold more information, to resemble Figure 9.2.

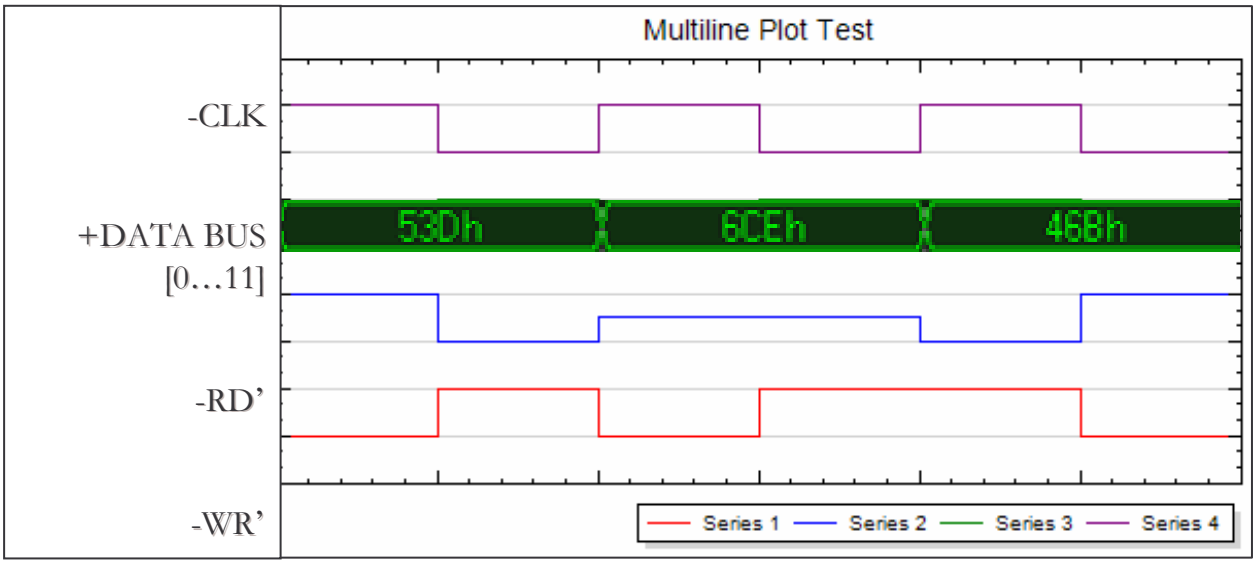

Figure 9.2 Possible future look of the GUI

In order to hold all the required information, two different data structures will be required. Table 10.1 lists the parameters of each of the data structures.

Line Class	Group Class
Data to be plotted	Internal Group Name
Internal Signal Name	User Defined Group Name
User Defined Signal Name	Number of total lines
Line Color	Lines part of this Group
Part of Group	Number Base (bin, dec, hex)*
Visible/Invisible	*defaults to hex

Table 10.1 Data structures and their contents

Finally, the project website tracks the project's progress. The website is located under Bradley University Electrical Engineering project pages at the following location: http://cegt201.bradley.edu/projects/proj2006/usbla/. Figure 10.1 shows a snapshot of the website for reference.

Figure 10.1 Project website snapshot