

# ELE700 Design Project 2000/20001

## Final Report: Palm OS<sup>TM</sup> GUI for the *eelab*

Andrew O'Malley  
aomalley@ee.ryerson.ca  
<http://www.ryerson.ca/~aomalley>  
961 504 685  
For Peter Hiscocks

15, December 2000

## 1 Project Description

### 1.1 Introduction

In an attempt to provide affordable lab instrumentation for Electrical Engineering students, faculty<sup>1</sup> at Ryerson Polytechnic University have developed an exciting solution, the *eelab*: a lab equipment kit that students assemble and interface to their personal computer (PC) for instrument controls and displays. The *eelab* currently consists of a power supply, function generator, and an oscilloscope; constructed on a collection of mid-sized circuit boards.

The kit remains within student budget by adhering to the following design principles: limiting expensive bandwidth to a few megahertz, well within the range of student lab work; replacing expensive hardware with a Graphical User Interface (GUI) on the host PC; and using development software available in the public domain, also encouraging further enhancements and modifications by users.[1]

### 1.2 Objective

The objective of this project is to further enhance the physical portability of the *eelab* by taking advantage of the recent popularity of Personal Data Assistants (PDA's) capable of serial communication. Used primarily to exchange data with host PC's for the purpose of data back-up and application installation, the serial communications available on these devices are equally suited for applications requiring data collection and display with simple computing<sup>2</sup> and communication capabilities.

The gaining popularity of PDA's makes them a viable alternative for user interfaces, whereby proprietary hardware solutions can be replaced by software to be installed on the end user's existing device.<sup>3</sup> In the case of this project, a Palm<sup>4</sup> PDA will be used in place of the PC for instrument display and control of the *eelab*, as illustrated in Figure 1, for which an appropriate GUI will be developed.

---

<sup>1</sup>See <http://www.ryerson.ca/~phiscock>

<sup>2</sup>The Motorola Dragonball microprocessor of the Palm, for instance, is not designed for heavy number-crunching.

<sup>3</sup>For example, a GPS receiver employing a LCD screen to visually display a user's whereabouts on a map could be reduced in size, and cost, by utilizing a PDA for the controls and display; a large portion of the hardware would thus be replaced by a software module.

<sup>4</sup>"Palm" and "Palm OS" are registered trademarks of *Palm, Inc.*

To effectively manage the project, it will be broken down into three modules, one for each component of the *eelab*, ordered in terms of increasing complexity: power supply, function generator, and oscilloscope. The modules will be implemented serially, beginning with the power supply. The goal of this project is to successfully create a working GUI for the power supply and function generator. Time permitting, the oscilloscope module may also be attempted. To maximize code re-use, programming modularity will be exercised wherever possible.

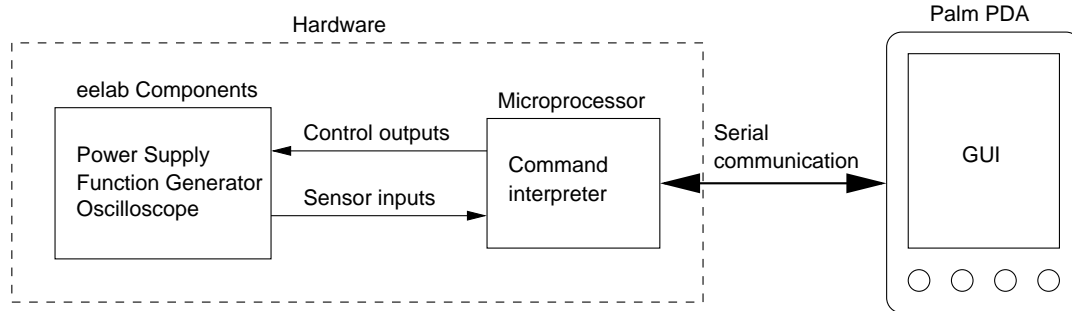


Figure 1: *eelab* / PDA Interface

## 2 Technical Specifications

Technical specifications for the *eelab*/Palm interface are as follows:

**PDA requirements** Palm Pilot Personal, Palm Pilot Professional, Palm III series, or equivalent with Palm OS version 2.0 or greater.

**PDA development tools** GCC for Palm OS (C to .prc compiler), PilRC (Palm resource/form compiler), PRC-Tools (linker, debugger, and other miscellaneous development tools), POSE and Palm OS ROMs (Palm emulator).

**PDA development environment** Linux or Windows 95/98 operating system, capable of 'Hot-Sync'ing with the Palm for the purpose of transferring applications from the host PC to the Palm.

**Palm to *eelab* communications** Serial, RS232, 32.4 Kbaud, via Palm to DB9/DB25 connector or cradle.

## 3 Functional Specifications

The current functional specifications are based on the existing PC GUI; however, due to the limited screen size on the Palm, these may have to be reduced or modified. The following tables outline the current functionality provided by the controlling software, and the method of implementation, which this project will attempt to duplicate. The description of each *eelab* component is divided into two categories: display and control. Display elements are measurements of interest to the user from the hardware (such as power supply output, or frequency of the function generator) which require

queries to the hardware to determine their value. Control elements are hardware parameters that can be controlled by the user, whose values are indicated by the position or state of the on-screen control elements, and send an associated command to the hardware, but require no return message or confirmation.

### 3.1 Power Supply

**Display** The power supply display contains the following:

<i>Parameter</i>	<i>Method</i>
Variable $10[V_{dc}]$ output	Numerical text field
Fixed $5[V_{dc}]$ output	Numerical text field

**Control** The power supply controls are as follows:

<i>Parameter</i>	<i>Action</i>	<i>Method</i>
Variable output	Increase	Incremental pushbutton
Variable output	Decrease	Decremental pushbutton
Output	Update output	Pushbutton

### 3.2 Function Generator

**Display** The function generator display contains the following:

<i>Parameter</i>	<i>Method</i>
Frequency (new)	Numerical text field
Frequency (current)	Numerical text field
DC offset	Numerical text field
Amplitude	Numerical text field

**Control** The function generator controls are as follows:

<i>Parameter</i>	<i>Action</i>	<i>Method</i>
Waveform	Set waveform shape	Radio buttons
Frequency	Increase / decrease	Numeric keypad
Set frequency	Update output	Pushbutton
Frequency	Increase / decrease	Up / down slider
DC offset	Increase / decrease	Up / down slider
Amplitude	Increase / decrease	Up / down slider

### 3.3 Oscilloscope

**Display** The oscilloscope display contains the following:

<i>Parameter</i>	<i>Method</i>
Channel A or B waveform	Graphical window
Channel A and B $\Delta V$	Numerical text field
Channel A and B gain	Numerical text field
Common timebase period	Numerical text field
Common timebase frequency	Numerical text field

**Control** The oscilloscope controls are as follows:

<i>Parameter</i>	<i>Action</i>	<i>Method</i>
Channel A and B amplitude	Zoom in display	Incremental pushbutton
Channel A and B amplitude	Zoom out display	Decremental pushbutton
Channel A and B amplitude	ground signal display	Toggle button
Trigger	Mode select	Radio buttons
Trigger	Slope select	Radio buttons
Trigger	Source select	Radio buttons
Trigger	Level adjust	Up / down slider
Time Scale	Zoom in display	Incremental pushbutton
Time Scale	Zoom out display	Decremental pushbutton
Time Scale	Update measurement	Pushbutton

## 4 Design Issues

Although the GUI will be programmed in C, the Palm OS does not readily recognize common C functions, but instead supports a family of proprietary functions that will have to be studied for use during implementation. A large portion of project effort will be dedicated to learning these functions and the Palm OS in general, at the level of detail required to program the device.

The success of the project rests on proper communication between the Palm and the *eelab* hardware, so careful attention must be given to the use of serial communications; especially on the Palm side where an inefficient use of the serial port will quickly drain the batteries. Thankfully, the Palm OS emulator, running on the development PC, has serial port support, so the majority of development will occur there, using the physical Palm only for final testing.

## 5 Theory

Palm OS programming is *event* driven, and all elements (program, data, preferences, etc.) are saved in a proprietary database format. Users interact with the program through *forms*, which define the overall screen display, and controls (pushbuttons, text fields, etc.). Figure 2 shows a flow diagram of a typical Palm program.

During *Startup / Initialization*, the program is loaded into memory by the operating system, and any necessary databases are opened. Next, the initial form for the program is displayed on the screen. The program then enters its event loop, where it awaits events, which may come from user input through the form, or from the operating system. When an event has been detected, a specific handler will be called to perform the necessary tasks associated with the particular event. The event loop typically runs indefinitely<sup>5</sup> until a *stop* event is encountered, upon when the program should save any required preferences, close any opened databases, and return control to the operating system.

## 6 Milestones

The project has been decomposed into the following major ‘milestones’ of accomplishment for each module:

---

<sup>5</sup>Some applications might have specific time-out functions should there be no events within a pre-defined time limit.

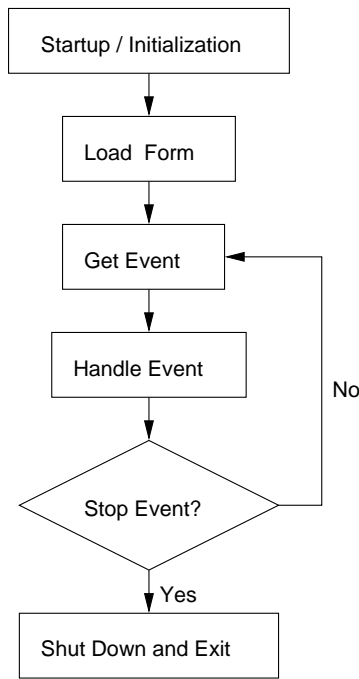


Figure 2: Palm Program Flow

**Form Design** involves mapping controls and displays to achieve best use of the limited Palm screen.

**Implement Form** program the form and test on the Palm OS emulator.

**Pseudo-code** conceptualize the code required for each module, including event handlers and serial communications.

**Implement Code** translate the pseudo-code model into actual C code, using Palm OS functions.

**Testing & Debugging** test and debug the code on the Palm OS emulator, first with a dumb terminal to demonstrate communications, then with the *eelab* hardware.

**Physical Testing** install the software on the Palm and verify operation with the *eelab* hardware.

The chart in Table 1 can be used to track these milestones, using each cell to record the date completed.

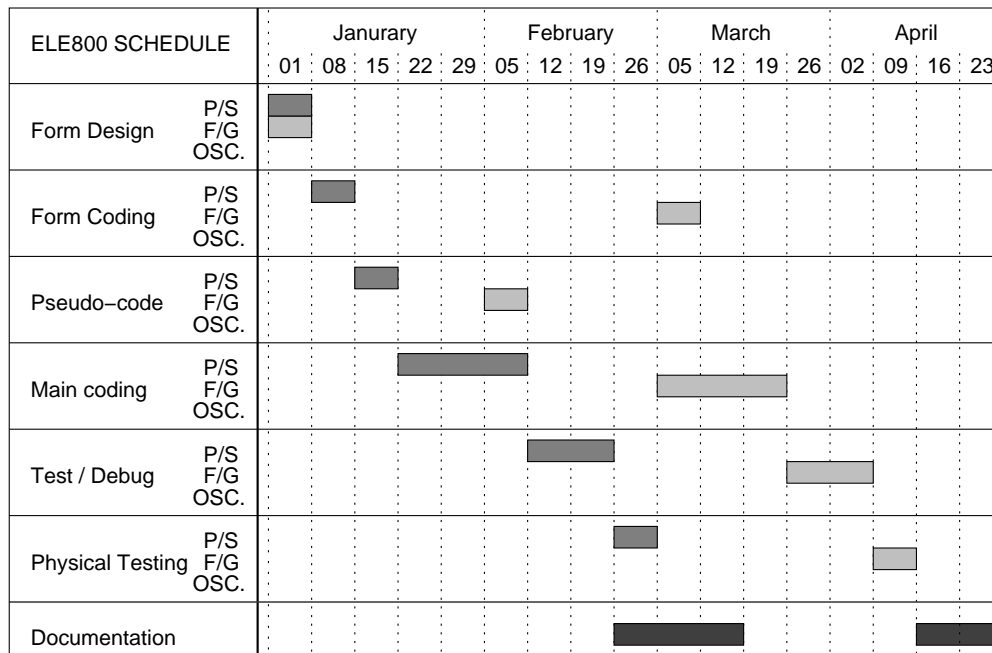
## 7 Schedule

Based on the milestones outlined in Section 6, a schedule has been created for the second semester of the design project, ELE800. The timeline for project progress can be seen in the Gantt Chart of

<i>Milestones</i>	Power Supply	Function Generator	Oscilloscope
Form Design	00/12/03		
Implement Form			
Pseudo-code			
Implement Code			
Testing & Debugging			
Physical Testing			

Table 1: Project Milestones

Figure 3. Each time block represents one week, and the time columns are labeled with the calendar date starting each week. As much as possible, effort expended on the power supply module will be “re-used,” allowing for some parallel activities nearing the completion of this first module. It can be seen that there will be no time available for any efforts toward the oscilloscope, unless project progress goes significantly faster than anticipated — which is an unrealistic assumption given that the nature of the work includes a formidable learning curve.



P/S – Power Supply  
 F/G – Function Generator  
 OSC. – Oscilloscope

Figure 3: Project Schedule

## 8 Design Progress

Thus far, project effort has been committed to general research of Palm OS programming, and the required development tools and environment. Several options were available, and these had to be evaluated against both cost and functionality. The best solution was determined to be the proven collection of public domain development tools (as listed in Section 2) that many private developers have used successfully in lieu of Metroworks' commercial solution, CodeWarrior<sup>6</sup>, for a broad range of Palm applications. Currently, I am in the process of obtaining the necessary development tools and evaluating them on both Linux and Windows 95/98 systems.

In parallel to creating the development environment, I have started giving thought to form design. Presently, the power supply PC GUI is the only one small enough to be directly ported to the Palm screen, and a possible design based on that GUI, is presented in Figure 4. The 'eelab' header is typical of the Palm application format, and the triangle beside the 'Power Supply' label indicates a pull-down menu used to navigate between different forms, in this case the other two *eelab* modules.

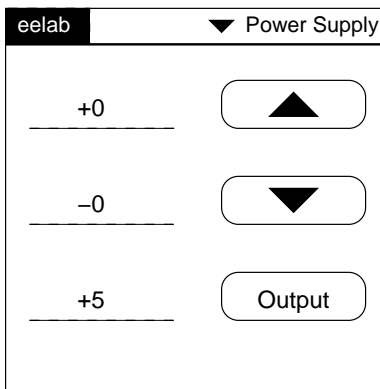


Figure 4: Power Supply Palm Form

## 9 Long Lead-Time Concerns

Since this is a software project dealing exclusively with existing hardware, there are few lead-time concerns for the required resources. By choosing development software available in the public domain, the tools are widely available on the World Wide Web; which also serves as an excellent source of information and documentation, available in the form of newsgroups, articles, and web pages. The biggest concern is hardcopy materials, which may take up to five weeks to obtain from publishers if not immediately stocked by distributors.

## References

- [1] *eelab Lab Equipment For Electrical Engineering Students*, Peter D. Hiscocks, Ryerson Polytechnic University, 1999.

---

<sup>6</sup>See <http://www.metroworks.com>