Physics 335, Spring Quarter 2012
The Common Project for the lab 15-17 May
Pulse-Width Modulation

INTRODUCTION

Once you can communicate with and program a microcontroller, you are only limited by your time and imagination. Simple controllers like the PIC are also limited by their spare instruction set, small program and data memory and restricted range of peripherals. However, even this small PIC controller is very useful. It is used, for example, for lighting controls, relay drivers, motor controllers, and consumer electronics.

This lab is meant to demonstrate some capabilities of the PIC processor and get you more used to PIC programming. In this lab, you'll build the hardware and develop a computer program that performs the following task:

PULSE-WIDTH MODULATION OF THE BRIGHTNESS OF A LED

You have a job: You're the chief engineer for Apple Computer. Your boss wants you to design and build a digital circuit that varies the brightness of an LED on some consumer device.

The device has two buttons. One button makes the LED brighter, the other makes it dimmer. You need the apparent-brightness change to be approximately linear in time: the longer the button is pushed, the brighter or dimmer the LED appears.

There are many ways to realize this outcome. In this lab, you'll be using pulse-width modulation. The LED is connected to the digital output of the PIC processor. The LED brightness or dimness is set by the duty-factor of the PIC output pulse; that is, the brightness is controlled by the fraction of time the LED on is relative to the full pulse period; 100% duty-factor means the LED is on continuously, while 5% duty-factor means the LED is on only 5% of the time. This is a very common scheme to modulate brightness and it's used, for instance, in your common wall-switch light dimmer.

You can use any of the small components you find in your parts-drawers, but you might want to use the following parts from your to build this device. Note: as a successful researcher, you should be able to build what you need with what you have.

- PIC microcontroller
- Resistors
- LEDs
- Push-buttons or switches

Here's how your software architecture could be organized:

- Initialize hardware and software for the reset/start
- Define one counter for the number of time-intervals on, and another counter for time-intervals off; true pulse-width modulation would have their sum a constant time.
- Light the LED for the on time-interval, off for the off interval.
- Check switches and increment intervals appropriately.
- Loop.

You'll likely need to code appropriate time-delays and you may find that the apparent LED brightness is not linear in duty-factor; you may want to explore first this issue with the function-generator.

BONUS SECTION

If you've finished the LED display above and have some extra time, then move on to this bonus section. Here, you'll connect a 7-segment LED display to indicate the apparent brightness of the LED. The 7-segment display should indicate a number from 0-9 that corresponds to the apparent brightness of the LED. Your PIC will need to convert a binary display value to the 7-segment display code.

HOW THIS IS GRADED

You and your lab partner need to demonstrate operability of your common project to your TA. You'll likely need to consult data sheets for the processor and perhaps the assembler as well. You're responsible for debugging your code on time, so code carefully and have a code-development plan.

You and your lab partner submit a common lab writeup at the end of the lab. For your writeup, include a circuit diagram and brief description of how your circuit and software works. Also include a printout of your assembly code. The TA will quickly look at this printout and see if it's a mess, or if it's understandable on its own. The TA may grant a bonus of 10% for particularly tidy and readable code, or subtract 10% for particularly obscure and unreadable code. Messy scrawl will be heavily downgraded; imagine you're showing your work to your boss at Apple. Full credit is given only to fully functional circuits.

Finally, you should come prepared. Come to lab with a detailed wiring diagram with all parts shown and connected pins shown. Also arrive with an outline of the program code structure. The TAs will likely note their absence and without preparing them you're unlikely to finish.

[ver 07may12 10:30]