18-549 Embedded System Design: Lab 3 Sensors and Actuators

Assigned: **02/20**
Due: **03/01 (In Lab Demo)**

The purpose of this lab is to add your custom sensors and actuators to your PCB and write firmware that demonstrates that you are able to successfully communicate with them.

Hand-in:

- This Lab is to be done by **your group**. There is no paper hand-in, you simply need to demonstrate your system to a TA during your Monday demo slot.

Grading:

- Your submissions will be graded on the following criteria:

Criteria	**Points**	**Score**
Working sensor hardware	4	
Sensor sending data over UART	1	
Working actuator hardware	4	
Controlling actuator over UART	1	
Final demo of sensor linked to actuator	5	
Bonus (anything cool)	3	
Total	15	

Lab Procedure

For this lab, you will need to install avr-gcc, avr-libc and supporting tools like make. In most cases, these are standard packages that you can install depending on your particular distribution. Make sure you can compile the example test UART program from lab 2.

1) Add your custom sensor to your lab 2 PCB
2) Write firmware to print out the sensors values over the UART terminal (see example program from lab 2)
3) Add your custom actuator to the PCB
4) Add to your demo application the ability to control your actuator over the UART terminal
5) Write a demo application where the sensor is linked to the actuator using open- or closed-loop control.

Demo all of the debugging apps to your TA. For full credit, all elements of the system (steps 2,4 and 5) should be shown from a single firmware image.

Installing the Toolchain

We briefly discussed this in lab 2, but it was not a requirement if the TA loaded the bootloader/program on your board. The command for programming the sample code using the bootloader (from lab 2) is:

```
$ avrdude -b 57600 -F -e -c arduino -P [port location] -p
atmega328p -U flash:w:[hex file path]
```

Linux

The packages can likely be found in your favorite package manager. For example, on a debian system, you would want to run:

```
$ sudo apt-get avr-gcc avr-libc avrdude
```

Mac OS X

On OS X, the tools can be found in the homebrew repositories. Since they are not in the main repository, we will have to 'tap' a github repository. You are, of course, welcome to use macports, install using crosspack, or build from source if you prefer those methods.

1) Install homebrew from http://brew.sh/
2) Tap the larsimmisch/homebrew-avr repository with formulas for avr-gcc and avr-lib
   ```
   $ brew tap larsimmisch/homebrew-avr
   ```

3) Install the required packages
```
$ brew install avr-gcc avr-libc avrdude
```

Windows

Atmel makes a very good software package called AVR Studio with support for programming, simulating, and debugging the ATmega chips we are using. It can be set up to program chips using avrdude and a bootloader by following this tutorial: http://www.jayconsystems.com/tutorial/atmerpt1/ (note that we have not tried this, but it looks right).

Example System

Lets take an example of a system that is comprised of a temperature sensor and an RGB LED. For the first part of the lab, we would expect a program that periodically reads the ADC on the ATmega and displays a raw ADC value over the serial port. The value does not need to be calibrated, but we will want to see that if you warm the sensors, the values change in order to demonstrate that the sensors is in fact working. For the actuator demo, it would be sufficient to have one or two keyboard values link to different colors on the LED. For example, if you send the character 'r' then the LED turns red or if you send the character 'g' the LED turns green. You need to show at least three different states to prove that it is more then a binary actuator (on, off, color). A good example for the final demo would be that if you then press 'a' (for auto), the board goes into a mode where it periodically reads the temperature sensor and maps it to a different LED color. A hotter temperature would turn increasingly red and a cooler temperature would turn bluer. This could be demonstrated with either the heat from your hand or something like a nearby soldering iron. Obviously your sensors and actuators could be different, so be creative.

Revision History