CMSC 328 Embedded Operating Systems

LED: Getting to know your Pi

The purpose of this lab is to get you and your Raspberry Pi up and running. Rather than the traditional "Hello, World" we'll be writing three different programs to blink an LED — the embedded systems "Hello, World."

Step 0 - Installing the OS

To start, we will use the Raspian OS for the Raspberry PI. You can download the NOOB software here: http://www.raspberrypi.org/downloads [a copy will also be available via flash drive]

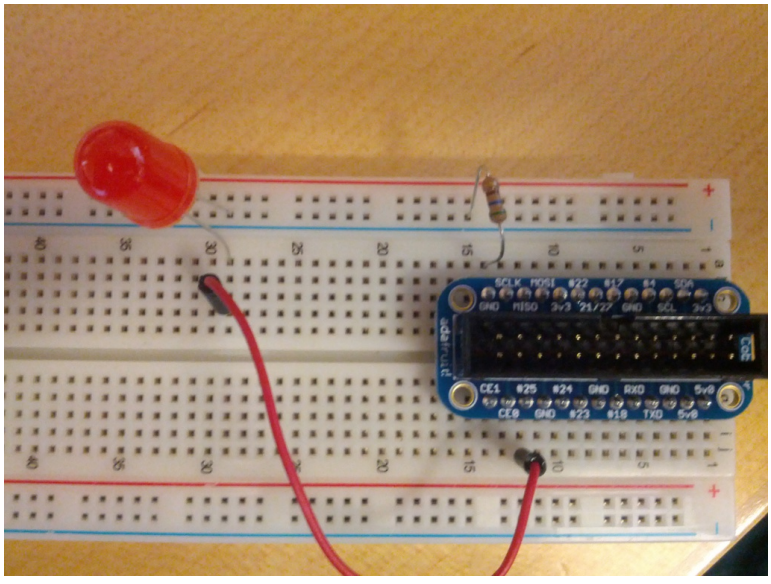
Before unzipping the NOOB.zip make sure you format the SD card appropriately:
http://elinux.org/RPi_Easy_SD_Card_Setup

Once you insert the SD card, attach a keyboard, mouse, display, and finally boot up, you should see a login prompt:

username: *pi*
password: *raspberry*

Step 1 - Wire Up the LED

Let's wire up an LED to GPIO25 pin on the Raspberry PI. From your kit, you will need:



1. 1x 560 ohm resistor
2. 1x LED
3. 1 wire
4. The "Pi cobbler" breakout

First, connect the breakout adapter to the breadboard and the Pi. Make sure the red dot is aligned with Pi 1 closest to the SD card. Then using the resistor, connect GND on the "Pi Cobbler" breakout adaptor to the ground line of your breadboard. Next connect the short end of the LED (cathode, -) to the the ground line of your breadboard. (Unlike the resistor, it matters which end of the LED goes where.) Lastly, connect the long leg of the LED (anode, +) to GPIO25 using the wire.

Step 2: Blinking with Bash

```
sudo -i
echo 25 > /sys/class/gpio/export
echo out > /sys/class/gpio/gpio25/direction
echo 1 > /sys/class/gpio/gpio25/value
echo 0 > /sys/class/gpio/gpio25/value
```

Step 3: Blinking with Python

Write a script to continually blink a LED connected to GPIO 25. You will first have to install the rPI.GPIO python module:

```
sudo apt-get install python-rpi.gpio
```

https://code.google.com/p/raspberry-gpio-python/

Step 4: Controlling the LED with C

Using the BMC2835 library, write a program to control the LED connected to GPIO 25. The user should be able to specify the pin number and the strings “on” and “off” using the command-line. The C functions `atoi` and `strncmp` will be useful.

http://www.airspayce.com/mikem/bcm2835/

```
./rpi_led 25 on
./rpi_led 25 off
```

Questions

1. Why is the first command in Step 2 `sudo -i` needed?
2. What is the purpose of this command in Step 2:
   `echo out > /sys/class/gpio/gpio25/direction`
3. What is the difference between the two pin numbering systems (e.g. pin 22 vs GPIO25)?
4. How could you blink the LED with python without using the “rPI.GPIO” module?

Deliverables (due Sep 16 2013)

A zip file or tarball that expands to a directory named `lab1_lastname/` with the following files:

1. A picture of your completed circuit
2. Answers to the four questions (`answers.txt`)
3. Python script (`lastname_lab1.py`)
4. C Program (`lastname_lab1.c`)

Resources

The following datasheet will be useful as the semester progresses:

http://www.raspberrypi.org/wp-content/uploads/2012/02/BCM2835-ARM-Peripherals.pdf

http://elinux.org/File:GPIOs.png