INTRODUCTION TO ARDUINO AND LEDS

Activity one: Make sure everybody is up and running

Arduino
Arduino

Physical Computing Essentials

- `pinMode(pinNumber, mode);` // declare a pin INPUT or OUTPUT
- `digitalRead(pinNumber);` // read the HIGH/LOW status of pin
- `digitalWrite(pinNumber, value);` // force HIGH/LOW voltage
- `analogWrite(pinNumber, value);` // PWM for intermediate vals
- `analogRead(pinNumber);` // read analog pin through ADC
Physical Computing Helpers

- `delay(ms);` // delay for ms milliseconds
- `millis();` // return total milliseconds since program start
- `Serial.begin(baud);` // set up serial communication to host
- `Serial.print(val);` // print on monitor (number, char, or string)
- `Serial.println(val);` // print with line feed
- `random(min, max);` // return random between min, max-1
- `map(val, fromLo, fromHi, toLo, toHi);` // interpolate to range
- `constrain(val, lo, hi);` // constrain value to a range

Digital Pins

- Each of the digital pins can be set to one of two values
  - High and Low (logic 1 (+5v) and logic 0 (0v))
  - `digitalWrite(<pin-number>, <value>);`

- `digitalWrite(13, HIGH);`
  - `digitalWrite(13, 1);`

- `digitalWrite(13, LOW);`
  - `digitalWrite(13, 0);`
Arduino Programming

Load Examples - Basics - Blink

Arduino SW system

- Two required functions

  - `void setup(){...}` // runs once at init time

  - `void loop(){...}` // loops forever after init
Arduino Programming

Verify, Upload, New, Open, Save

Blink Example

```c
int led = 13;

void setup() {
    pinMode(led, OUTPUT);
}

void loop() {
    digitalWrite(led, HIGH);
    delay(1000);
    digitalWrite(led, LOW);
    delay(1000);
}
```
Arduino

LED already connected to pin 13

Upload Blink to your Arduino

- Load Basics -> Blink into IDE
- Make sure you select the correct board
  - Tools -> Board -> (Uno or Duemilanove)
- Make sure you select the correct serial port
  - Not the bluetooth ports...
- Click on the upload button
  - Watch for blinky lights during upload
LEDs and Resistors

On LEDs, polarity matters. Shorter lead is "negative" side, goes to ground

Current flows from Anode to Cathode
Lights up when current flows
LEDs and Resistors

- On LEDs, polarity matters. Shorter lead is “negative” side, goes to ground.
- LED:
  - Anode +
  - Cathode -
  - Polarity doesn’t matter on resistors.
- Current flows from Anode to Cathode.
- Lights up when current flows.

Current Limiting Resistor

- Diodes have a “forward voltage” or “diode drop”:
  - Typically $V_f$ is around 0.7v for a diode, and 1.5v to 3.0v for an LED.
- Diodes also have a current limit:
  - Typically 20mA for an LED
  - If you don’t limit the current, they’ll burn out.
Current-Limiting Resistor

- Assume Pin10 can supply 5v
- Assume LED \( V_f \) is 2.0v
- \((5v - 2v) = 3v\) remaining for \(R1\)
- We want 20mA
- \( R = \frac{V}{I} = \frac{3v}{.020A} \)
- \( R = 150 \, \Omega \)

- In practice, 220 \( \Omega \) - 470 \( \Omega \) will work

Wiring an external LED
Solderless Breadboard
Wiring up an External LED

Wiring up an External LED
Blink Modifications

- Change numbers in the code
  - Then “upload” to Arduino – watch for blinky lights
- Change so that blink is on for 500msec and off for 100msec
  - What happens?
- Change so that blink is on for 50msec and off for 50msec
  - What happens?
- Change so that blink is on for 10ms and off for 10ms
  - What happens?

Aside: Multiple LEDs

Each LED needs its own resistor! You can’t share…
Multiple LED Example

Multiple LED Example
We just made LEDs blink... Big Deal?

- Most actuators are switched on and off with a digital output
  - The `digitalWrite(pin,value);` function is the software command that lets you control almost anything

- LEDs are easy!
  - Motors, servos, etc. are a little trickier, but not much
  - More on that later...

- Arduino has 14 digital pins (inputs or outputs)
  - can easily add more with external helper chips
  - More on that later...

Moving on...

Varying LED Brightness

Same circuit as Blink circuit but pin 9 instead of pin 13

The PWM pins work with the "`analogWrite(value)`" command where "value" ranges between 0 and 255.
To turn LED to half-bright, use `analogWrite(9, 128)`
Pulse Width Modulation

- `analogWrite(pin, value);`
- `value` can be 0 to 255
- Must be one of the “PWM pins” : pins 3, 5, 6, 9, 10, 11
- `output_voltage = (on_time / off_time) * max_voltage`

5 volts  
0 volts 
75% 25% 75% 25% 75% 25%  
3.75 Volts

5 volts  
0 volts 
50% 50% 50% 50% 50% 50%  
2.5 Volts

5 volts  
0 volts  
20% 80% 20% 80% 20% 80%  
1.0 Volts
Fading Program

```cpp
int ledPin = 9;    // LED connected to digital pin 9
void setup() {
    // nothing happens in setup (Why not?)
    // Answer: Pins default to being OUTPUT. You must define any that you want to use
    // as INPUT, but if you are using them as OUTPUT you can just assume the default.
    // Typically, though, it’s good practice to define the OUTPUT pins anyway…
}
void loop() {
    // fade in from min to max in increments of 5 points:
    for (int fadeValue = 0 ; fadeValue <= 255; fadeValue += 5) {
        analogWrite(ledPin, fadeValue);  // sets the value (range from 0 to 255):
        delay(30);                                  // wait for 30 milliseconds between brightness steps
    }
    // fade out from max to min in increments of 5 points:
    for (int fadeValue = 255 ; fadeValue >= 0; fadeValue -=5) {
        analogWrite(ledPin, fadeValue); // sets the value (range from 0 to 255):
        delay(30);                                // wait for 30 milliseconds between dimming steps
    }
}
```

Fading Example
End of Activity One

- There are some additional slides that you can look at later
- There’s a summary at the end of the handout

Add a diffuser
Add a diffuser

http://www.ladyada.net/learn/arduino/lesson3.html
Source vs. Sink for current

- Arduino pins: source/sink a max of 40mA/pin
- Also 250mA total for ALL pins

Random Brightness

- Write a program to make the LED flicker like a flame
  - Choose a random intensity
  - For a random amount of time
- Use `analogWrite(ledPin, val)` to change brightness
- Main loop repeats itself forever...
  - Set the value of the brightness to a random value
  - Wait for a random amount of time
  - repeat
- The effect looks like flickering...
Flickering Pseudocode

1. Set the LED to a random brightness
2. Wait for a random amount of time
3. repeat

Flickering Pseudocode

1. Pick a random number between 100-255
2. Set LED to that brightness (use analogWrite)
3. Pick another random number between 10-150
4. Wait for that amount of time (in ms)
5. Repeat

int brightness;
brightness = random(100, 256);
Candle Program

- `random(min,max);` will return a random number between `min` and `(max-1)`
  - `randomSeed(int);` will initialize the random function
  - Not really needed...
  - `foo = random(10, 200);` // assign `foo` to random number between 10-199

- Remember `delay(val);` // waits for “val” milliseconds

Hints...

```c
int bright; // make a new variable called bright
bright = random(100, 256); // set “bright” to a random value between 100 and 255
Remember: analogWrite(pin,value); // sets a brightness on a pin
// “pin” is the pin number, “value” is between 0 – 255
```

Candle Program

```c
int ledPin = 9; // select pin for LED output
int bright = 0; // Variable to hold LED brightness
int time = 0; // variable to hold delay time

void setup () {
    randomSeed(0); // initialize the random function
    pinMode(ledPin, OUTPUT); // ledPin should be an output
}

void loop () {
    bright = random(100, 256); // random brightness value
    analogWrite(ledPin, bright); // set the LED brightness

    time = random(10,150); // random time in ms
    delay(time); // delay for that time
}
```

Load Sketchbook - DM - Candle
Candle Program (smaller)

```c
int ledPin = 9;  // select pin for LED output
void setup () {
    pinMode(ledPin, OUTPUT);  // ledPin should be output
}

void loop () {
    analogWrite(ledPin, random(100, 256));  // LED brightness
    delay(random(10,150)); // delay for random time
}
```

Silly LED Tricks
Summary – Whew!

- **Digital Pins**
  - Use `pinMode(<pin>, <INPUT/OUTPUT>)` for setting direction
    - Put these in the setup() function
    - `pinMode(13, OUTPUT); // set pin 13 as an output`
  - Use `digitalWrite(<pin>, <HIGH/LOW>)` for on/off
    - `digitalWrite(LEDpin, HIGH); // turn on pin “LEDpin”`
  - Use `analogWrite(<pin>, <val>)` for PWM dimming
    - Values from 0 – 255
    - PWM pins are 3, 5, 6, 9, 10, 11
    - `analogWrite(9, 235); // set LED on pin 9 to somewhat bright`

More Summary

- `delay(val)` delays for `val-number of milliseconds`
  - Milliseconds are thousandths of a sec (1000msec = 1 sec)
    - `delay(500); // delay for half a second`
  - `random(min,max)` returns a random number between `min` and `max`
    - You get a new random number each time you call the function
    - `foo = random(10, 255); // assign foo a random # from // 10 to 255`
More Summary

- Two required Arduino functions
  - void setup() { ... } // executes once at start for setup
  - void loop() { ... } // loops forever
    - statements execute one after the other inside loop, then repeat
      after you run out
- int i = 10; // define an int variable, initial value 10
- Other types of variables:
  - char – 8 bits
  - long - 32 bits
  - unsigned...
  - float – 32 bit floating point number

Still More Summary

- for (<start>; <stop>; <change>) { ... }
  - for (int i=0; i<8; i++) { ... } // loop 8 times
    // the value of i in each iteration is 0, 1, 2, 3, 4, 5, 6, 7
- if (<condition>) { ... }
  - if (foo < 10) { digitalWrite(ledPin, HIGH); }
- if (<condition>) {...} else { ... }
  - if (num == 10) { <do something> }
    else { <do something else> }
Last Summary (for now)

- LEDs – turn on when current flows from anode to cathode
  - Always use a current-limiting resistor!
  - Remember your resistor color codes
  - 220-470 ohm are good, general-purpose values for LEDs
  - Drive from Arduino on digital pins
  - Use PWM pins if you want to use analogWrite for dimming

![Diagram of LED connection](attachment:image.png)

Resources

- [http://www.ladyada.net/learn/arduino/index.html](http://www.ladyada.net/learn/arduino/index.html)
- [http://todbot.com/blog/bionicarduino/](http://todbot.com/blog/bionicarduino/)
- [http://todbot.com/blog/spookyarduino/](http://todbot.com/blog/spookyarduino/)
- [http://sheepdogguides.com/arduino/aht0led.htm](http://sheepdogguides.com/arduino/aht0led.htm)
Contact Information

- Erik Brunvand
  School of Computing
  University of Utah
  Salt Lake City, UT 84112

  elb@cs.utah.edu
  http://www.cs.utah.edu/~elb