Willis Allstead
CPE 301-1001
Assignment #4
September 28, 2016

# Assignment Description:

Today we will be doing the LED lab. In this lab we will be learning about how an LED is configured and how to properly hook it up to a circuit without making it explode. Basically an LED has an anode and a cathode. We will be applying a positive current to the anode, and connecting the cathode to ground. We cannot apply the full 5V to the LED as that will make it explode, so we need to first implement a 330 Ohm resistor in the circuit, then connect the anode of the LED to the output of the resistor. We will be implementing a couple different circuits to solidify the concepts of using an LED correctly in combination with a resistor.

# Problems Encountered:

The problem that persisted throughout the seemingly simple lab was that sometimes I couldn't keep track of the wires so I wasn't hooking up the transistor properly each time. I progressed through the lab slowly due to this reason. I also had issues with the initial setup of the Arduino IDE at the beginning of the 1st part.

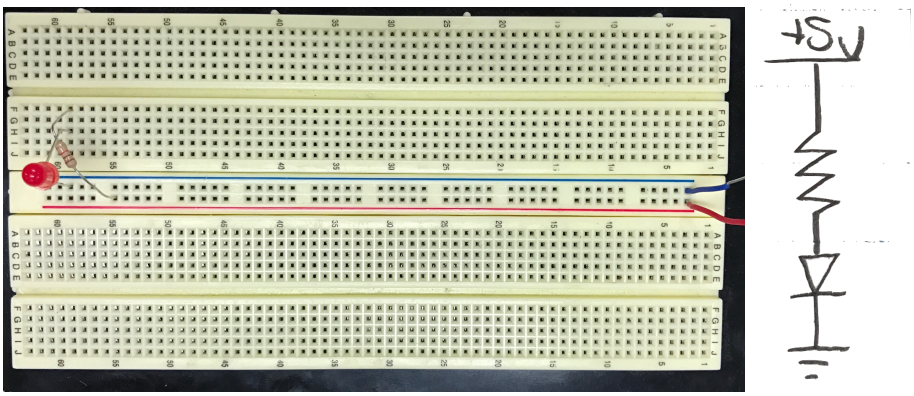
# Lessons Learned:

I learned how to actually use the Arduino IDE which felt nice since I thought it would take a lot more time to set up. I learned how to correctly implement transistors which has been something I've wanted to learn for a long time now. I also learned that I need to continue to develop my organizational skills when it comes to the breadboard, and that I need to more thoroughly check each circuit before I hook it up to power and ground.

# Description of Completed Lab:

## Part 1)

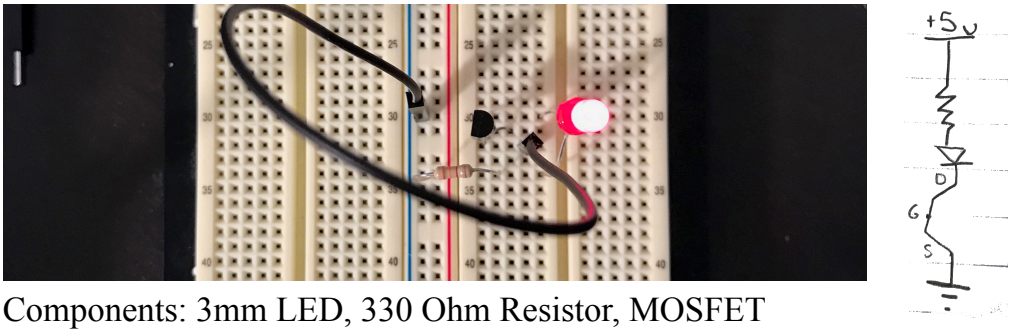
1) This circuit was simple and I implemented it as shown below:




Components: 3mm LED, 330 Ohm Resistor

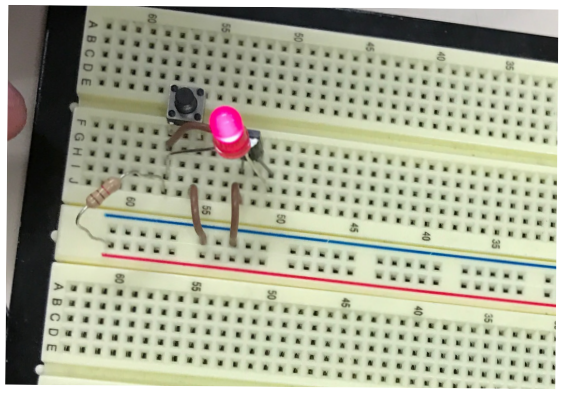
A. The voltage was 5V then dropped by 3.05V to 1.95V across the resistor.
B. If increased to 1k ohms, the LED is less bright since less voltage is getting to it.
C. If the LED is reversed it simply does not work.

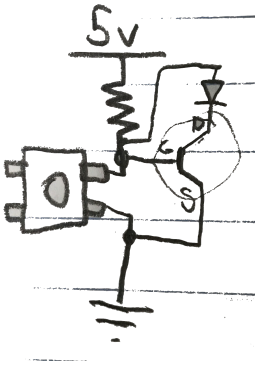
2) We then used a transistor as a switch in this circuit




Components: 3mm LED, 330 Ohm Resistor, MOSFET

3) We then implemented a circuit using a transistor and a button.



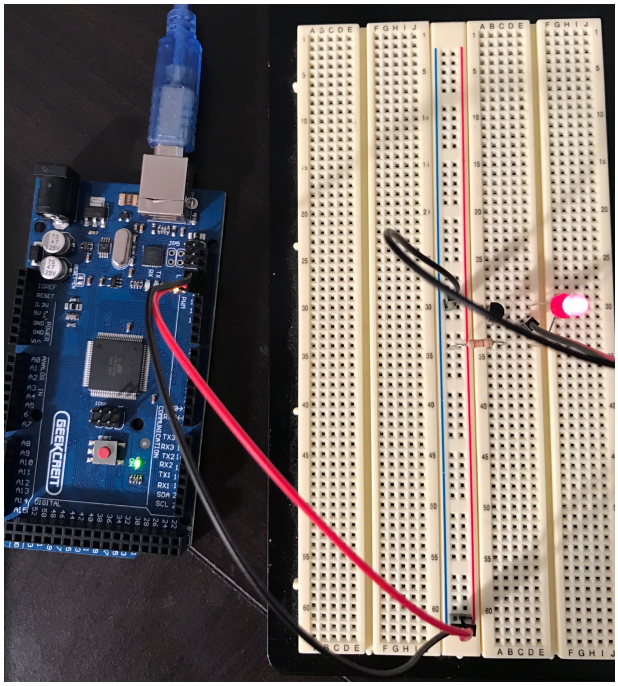


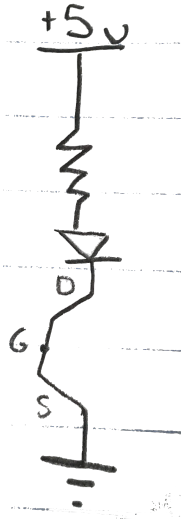

Components: 3mm LED, 330 Ohm Resistor, MOSFET, Button

A. The transistor here serves as a switch.

**Part 2)**

3) This circuit was implemented using the same configuration as in step 2 of part 1, but instead blinking by using pin13 of the Arduino as power:



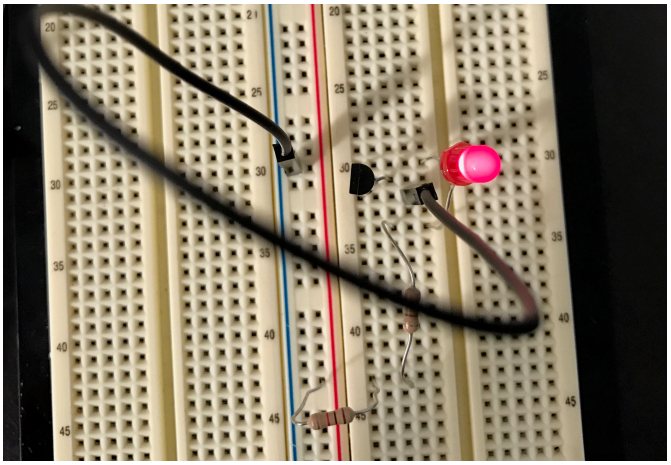



Components: 3mm LED, 330 Ohm Resistor, 1k Ohm resistor, MOSFET

A. 40ma per pin is the max current.
B. Out of these two resistors the smallest one I can use is the 330 ohm resistor.

C. The LED starts flickering instead of flashing in a controlled way.

4) We then implemented a circuit implementing one more resistor



Components: 3mm LED, 330 Ohm Resistor, 1k Ohm resistor, MOSFET

A. Now two resistors are limiting the current to the LED.
B. This circuit will now make sure that the LED does not have too much current flowing into it.

**Part 3)**

1) I opened the source files

A. Preprocessor definitions like HIGH, LOW make coding programs for the Arduino simple, and subtract from the time wasted finding each exact address of something when you only need to perform a simple action (such as turning on an LED).
B. BitRead and bitWrite are simple ways of changing bits without having to use masks, which can get complicated if you don't know what you're doing.
C. Below is pinMode

```
void pinMode(uint8_t pin, uint8_t mode)
{
     uint8_t bit = digitalPinToBitMask(pin); // setting bit
masks of bit
     uint8_t port = digitalPinToPort(pin);
     volatile uint8_t *reg, *out;

     if (port == NOT_A_PIN) return; // cancel if port is
not a pin

     // JWS: can I let the optimizer do this?
```

```
     reg = portModeRegister(port);
     out = portOutputRegister(port); // both configure the
port for either input or output

     if (mode == INPUT) { // if input
          uint8_t oldSREG = SREG;
                cli(); // get command line arguments
          *reg &= ~bit; // set mask of register
          *out &= ~bit; // set output
          SREG = oldSREG;
     } else if (mode == INPUT_PULLUP) { // if input pullup
is enabled
          uint8_t oldSREG = SREG;
                cli();
          *reg &= ~bit;
          *out |= bit;
          SREG = oldSREG;
     } else { // if it wasn't an input and the input pullup
wasn't implemented
          uint8_t oldSREG = SREG;
                cli();
          *reg |= bit;
          SREG = oldSREG;
     }
}
```

D) Below is digitalWrite

```
void digitalWrite(uint8_t pin, uint8_t val)
{
     uint8_t timer = digitalPinToTimer(pin); // set a timer
on the pin
     uint8_t bit = digitalPinToBitMask(pin); // set a mask
for pin
     uint8_t port = digitalPinToPort(pin); // set a port
from supplied pin
     volatile uint8_t *out; // prepare for output

     if (port == NOT_A_PIN) return; // stop if not a pin

     // If the pin that support PWM output, we need to turn
it off
     // before doing a digital write.
     if (timer != NOT_ON_TIMER) turnOffPWM(timer); // turn
off plus width modulation if not a timer
```

```
	out = portOutputRegister(port);

	uint8_t oldSREG = SREG;
	cli();

	if (val == LOW) {
		*out &= ~bit; // set output if value is Low
	} else {
		*out |= bit;
	}

	SREG = oldSREG;
}
```

E) Below is digitalRead

```
int digitalRead(uint8_t pin)
{
	uint8_t timer = digitalPinToTimer(pin); // setting a
pin as a timer by default
	uint8_t bit = digitalPinToBitMask(pin); // pin to
bitmask
	uint8_t port = digitalPinToPort(pin); // supplied pin
to port

	if (port == NOT_A_PIN) return LOW; // Return low if
the port isn't a pin

	// If the pin that support PWM output, we need to turn
it off
	// before getting a digital reading.
	if (timer != NOT_ON_TIMER) turnOffPWM(timer);

	if (*portInputRegister(port) & bit) return HIGH; // if
the port input register ANDED with bit returns 1, return a
High
	return LOW; // else return low.
}
```

2) A. In main.cpp, setup() is called directly after the USB device is attached. loop() is called from a simple for (;;) loop directly after setup.

3) He said we can't use them unless specifically stated. Anything we use them on is a definite ZERO.