IEEE Micromouse Fall 2017

Lab 1: Soldering and Arduino Basics

- Soldering Tutorial
- Assemble PCB
- Blink LEDs with Arduino

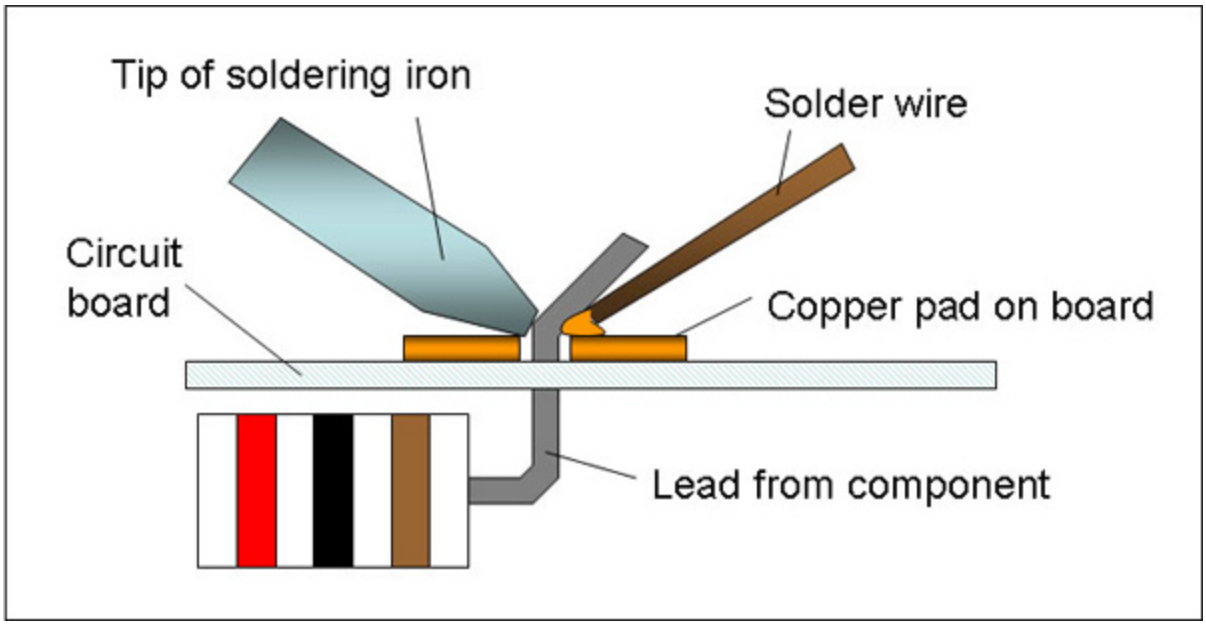
Soldering Tutorial

1) Turn on the soldering iron and wet the sponge -- you'll be using the sponge to remove excess solder from the soldering iron while soldering.

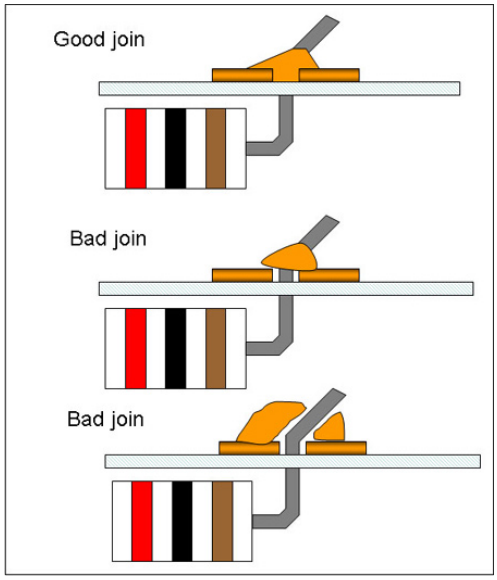
2) Wait 1~2 minutes for the iron to heat up, and clean off excess solder by wiping it on the sponge.

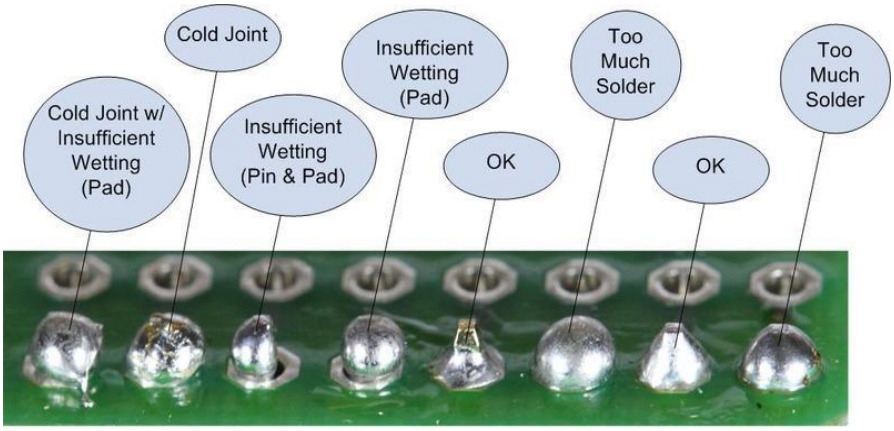
3) "Tin the iron" by touching the iron with solder. The solder should melt instantly and cover the tip with a protective layer that improves the iron's ability to transfer heat. As you're soldering, make sure that the tip is always shiny. If it is not, you can tin the iron again.

4) Insert a part through the through-holes. Hold the iron to the joint, heading up both the lead and the copper pad on the board. Bring the tip of the solder wire and press it gently onto the now-heated copper pad. The solder should melt and connect the lead to the copper pad.
Note: Do ***not*** heat the solder and bring the melted solder onto the pad -- this creates an unreliable connection.



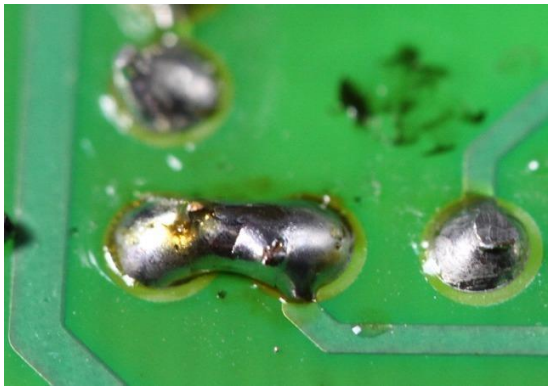
If done correctly, you should be able to see the solder “cling” to the pad and the component, as shown in the pictures below.





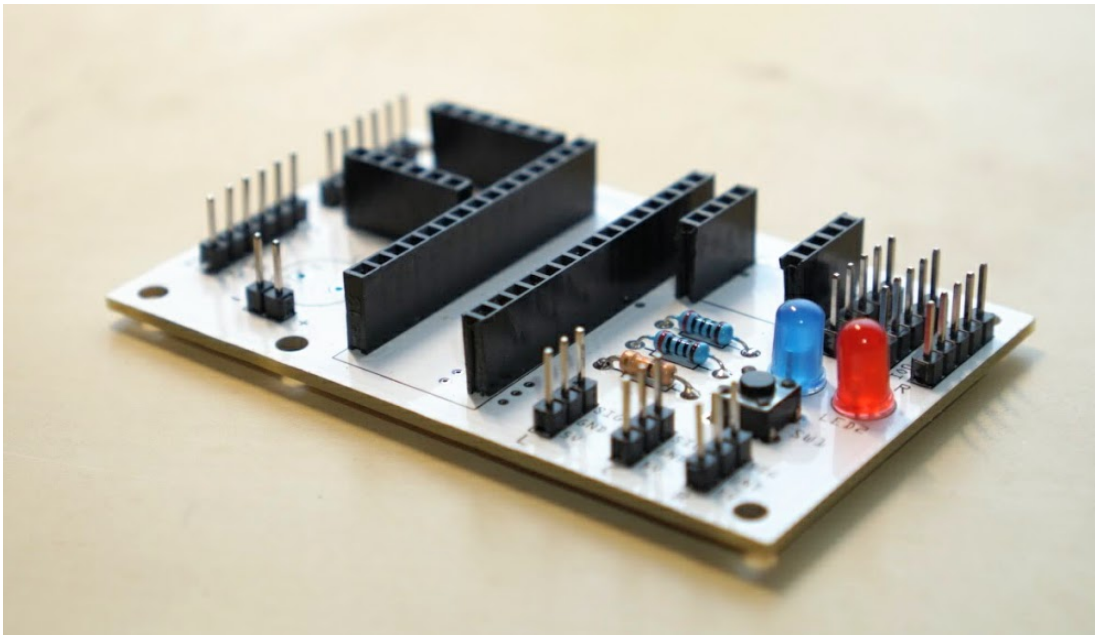
If you have created a bad joint, you can re-heat the joint and let the solder flow onto the pad. A list of common problems and their fixes can be found here: https://learn.adafruit.com/adafruit-guide-excellent-soldering/common-problems

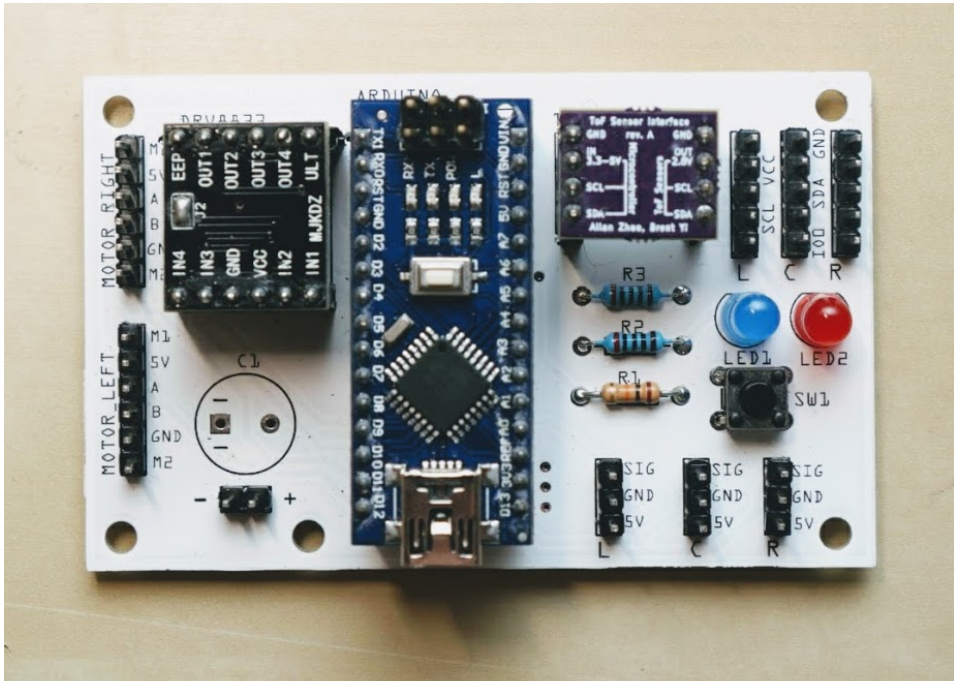
If there is excess solder on the board (e.g., a solder bridge, shown on the right image below), you can remove the excess solder with a solder pump.

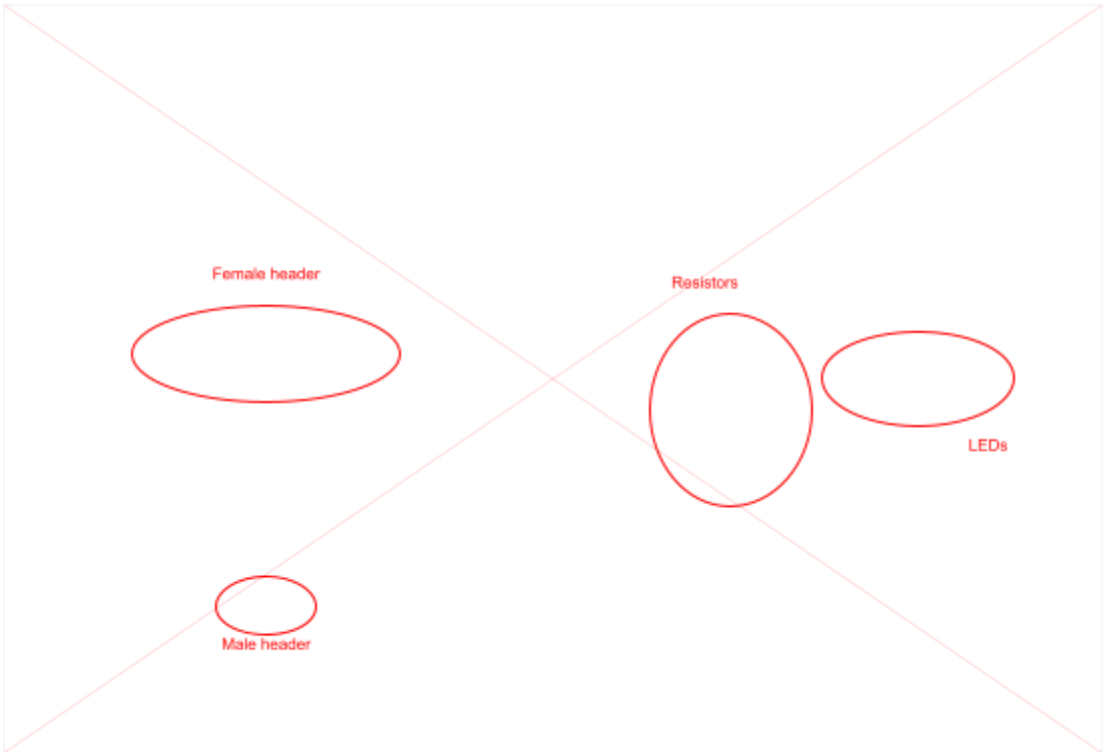


Assemble PCB

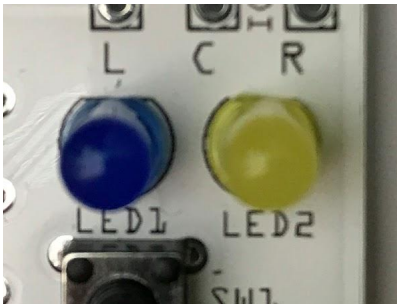
Once you're ready to solder, it's time to assemble your printed circuit board (PCB), which connects the electrical components on our mouse together. Here's what it'll look like when you're done:







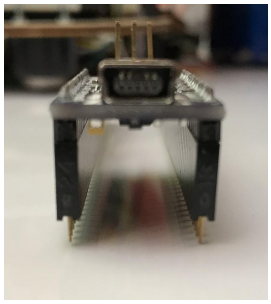
Solder each component on, paying extra attention to the color of the resistors and direction of the LEDs. The flat side of the LEDs should face away from the button:



LED direction

Tips

- You'll have to cut the headers to the right length.
- To make sure the headers for the various modules on the board are aligned, you should insert the module into the headers before soldering.



Checkoff #1

1. Show your mentor your assembled PCB!

Arduino Basics

We'll be using the Arduino IDE to program our mouse:
https://www.arduino.cc/en/Main/Software

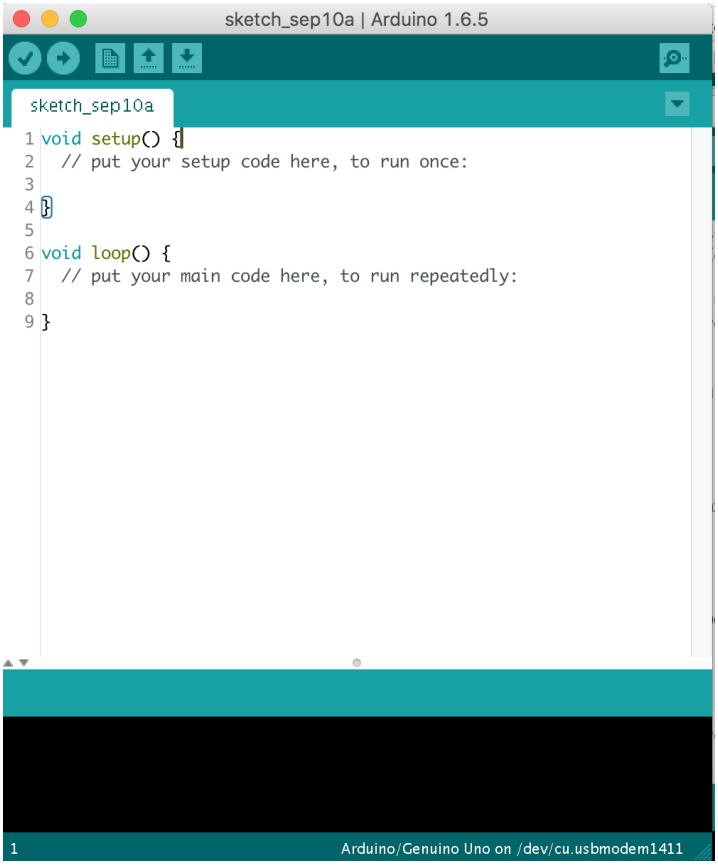
Setup

- Connect your Nano to the computer using Mini-B USB cable in your bag
- Pick your board type and port
    - Tools >> Board >> Arduino Nano
    - Tools >> Processor >> ATmega328p
    - Tools >> Port >> [Correct COM port]

Understanding the IDE Layout

Things you care about:

- Upload button (➔) - sends code to Arduino
- `setup()` method - everything here runs once at the beginning
- `loop()` method - everything here repeats continuously after setup



Let's first start with an example:

```
#define PIN_LED1 A3
#define PIN_LED2 A2
#define PIN_BUTTON A1

void setup() {
  // Set the LED pin as an output
  // This lets us "write" the voltage
  pinMode(PIN_LED1, OUTPUT);
}

void loop() {
  // Turn the LED on
  digitalWrite(PIN_LED1, HIGH);

  // Wait one second
  delay(1000);

  // Turn the LED off
  digitalWrite(PIN_LED1, LOW);

  // Wait one second
  delay(1000);
}
```

What does this do?

Defines which pins the LEDs are connecting to

- We want the microcontroller to know which pins are LEDs.

Setting up the pins

- In our `setup()` method, we use a function called `pinMode()`. This method lets us set a pin to either INPUT (read) or OUTPUT (write) mode. Since we want to output something to our LED pin, we pick OUTPUT.

Writing to the pins

- Here is where we tell the LEDs what to do. We use the method `digitalWrite()` that takes in the pin number and writes HIGH or LOW to it. HIGH indicates the LED is on and LOW indicates the LED is off. In the example blink LED1.

Checkoff #2

1. Turn the LEDs on and off in response to a button press.
    a. Hint: use `digitalRead(PIN_BUTTON)` to check whether the button is being pressed or not.