ELTN 117 – UNIT 10

Hardware interfacing with the Arduino / .h files

Aka – Bells and Whistles

Objectives

- Learn how to create a header file
- Understand what we can do besides read switches and light up LEDs.
- Understand how to interface with more complicated hardware
- Understand what libraries are and how they are used.
- Expanding I/O

Header (*.h) files

- One way to clean up / simplify our programs is to create header files to include our I/O definitions

- **Note:** This is not the same as a commented header!

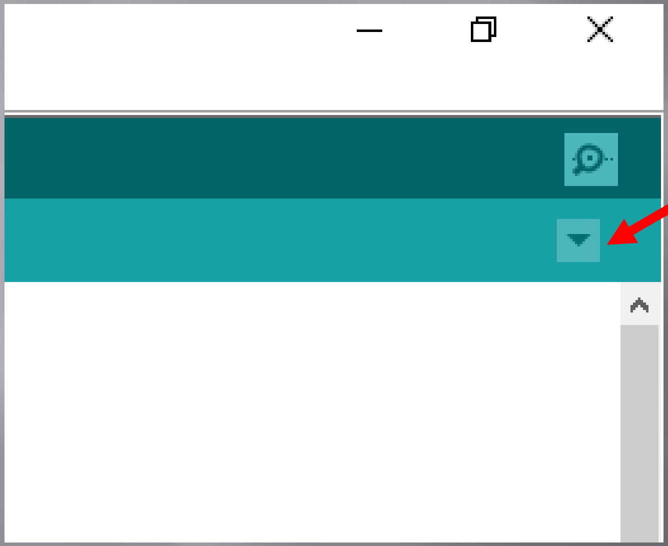
- The header file is a separate file that is added to the sketch folder.

- Once the header is created it can be used in other programs.

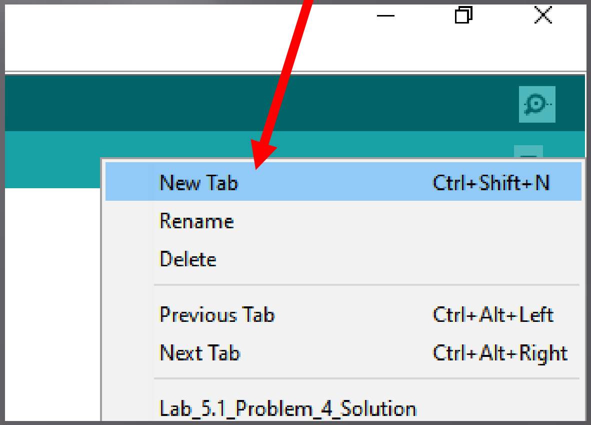
- The following screen shots show how this can be done…

Header (*.h) files

- To create a header file, we need to first add a new tab to the existing sketch:

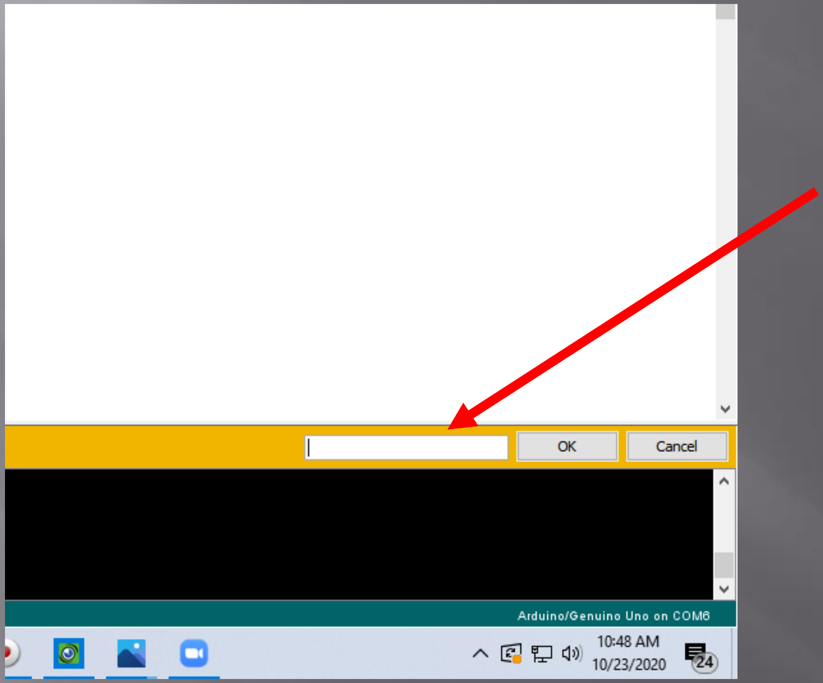


Click on this button, and select New Tab



Header (*.h) files

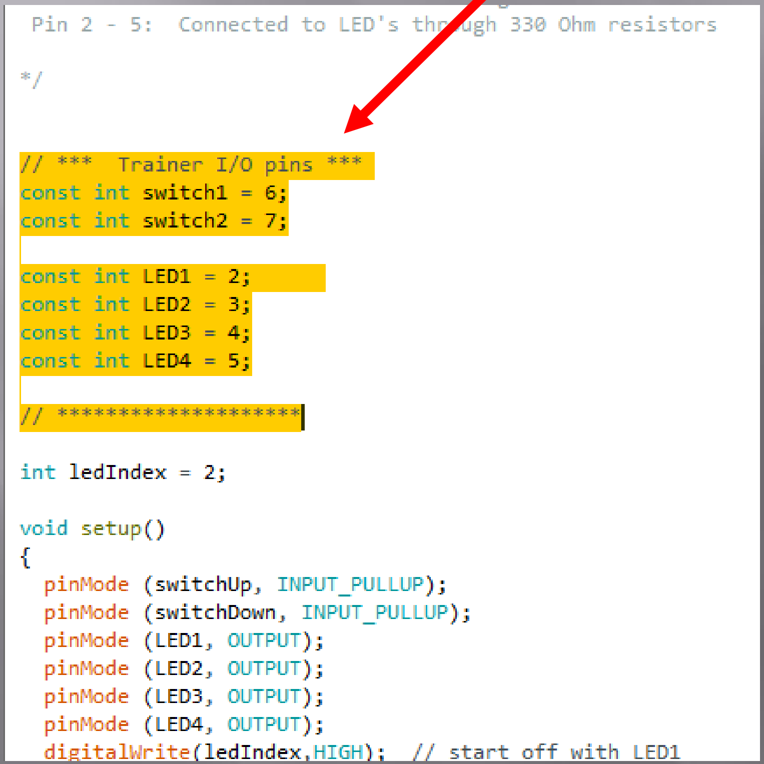
- Create a name for the new file:



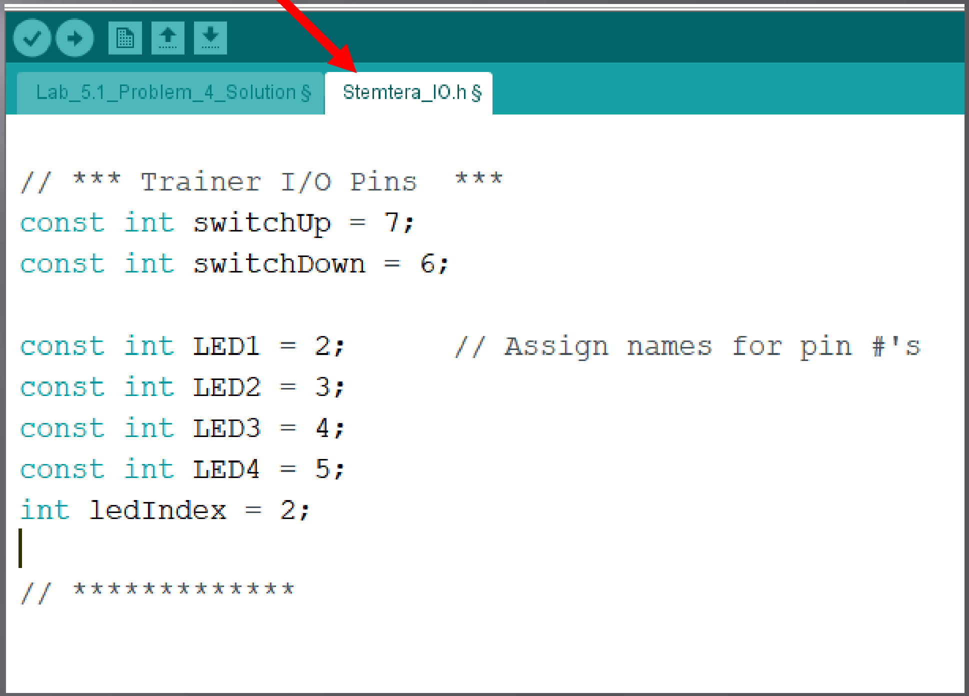
Type a file name here, for example: Stemtera_IO.h

Header (*.h) files

- Next, cut and paste the following lines from the program into the new file tab (or you can just type in your own header file):



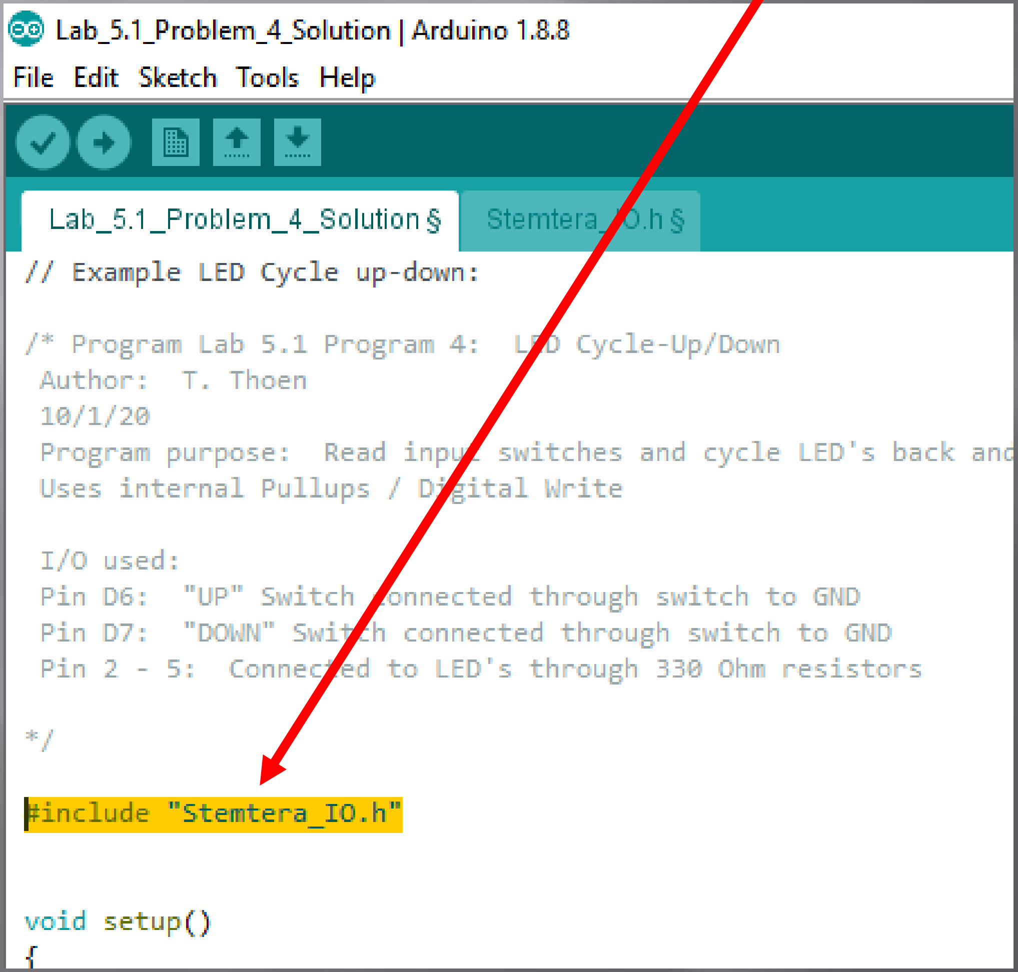
```
 Pin 2 - 5:  Connected to LED's through 330 Ohm resistors

*/


// ***  Trainer I/O pins ***
const int switch1 = 6;
const int switch2 = 7;

const int LED1 = 2;
const int LED2 = 3;
const int LED3 = 4;
const int LED4 = 5;

// ********************

int ledIndex = 2;

void setup()
{
  pinMode (switchUp, INPUT_PULLUP);
  pinMode (switchDown, INPUT_PULLUP);
  pinMode (LED1, OUTPUT);
  pinMode (LED2, OUTPUT);
  pinMode (LED3, OUTPUT);
  pinMode (LED4, OUTPUT);
  digitalWrite(ledIndex,HIGH);  // start off with LED1
```



```
// *** Trainer I/O Pins  ***
const int switchUp = 7;
const int switchDown = 6;

const int LED1 = 2;      // Assign names for pin #'s
const int LED2 = 3;
const int LED3 = 4;
const int LED4 = 5;
int ledIndex = 2;

// *************
```

Header (*.h) files

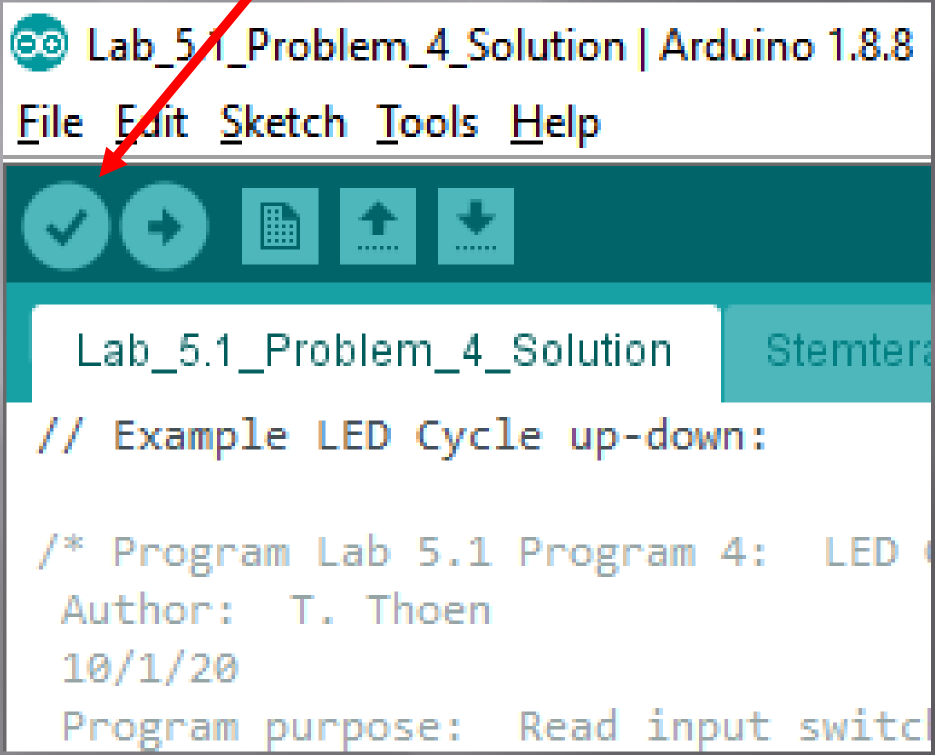
- Finally, create an #include statement in your main sketch tab. NOTE: The file name must EXACTLY match the header file name!


Lab_5.1_Problem_4_Solution | Arduino 1.8.8
File Edit Sketch Tools Help
Lab_5.1_Problem_4_Solution §
Stemtera_IO.h §

```
// Example LED Cycle up-down:

/* Program Lab 5.1 Program 4:  LED Cycle-Up/Down
 Author:  T. Thoen
 10/1/20
 Program purpose:  Read input switches and cycle LED's back and
 Uses internal Pullups / Digital Write

 I/O used:
 Pin D6:  "UP" Switch connected through switch to GND
 Pin D7:  "DOWN" Switch connected through switch to GND
 Pin 2 - 5:  Connected to LED's through 330 Ohm resistors

*/

#include "Stemtera_IO.h"


void setup()
{
```

Header (*.h) files

- Once done, click the Check button to make sure the program still compiles.

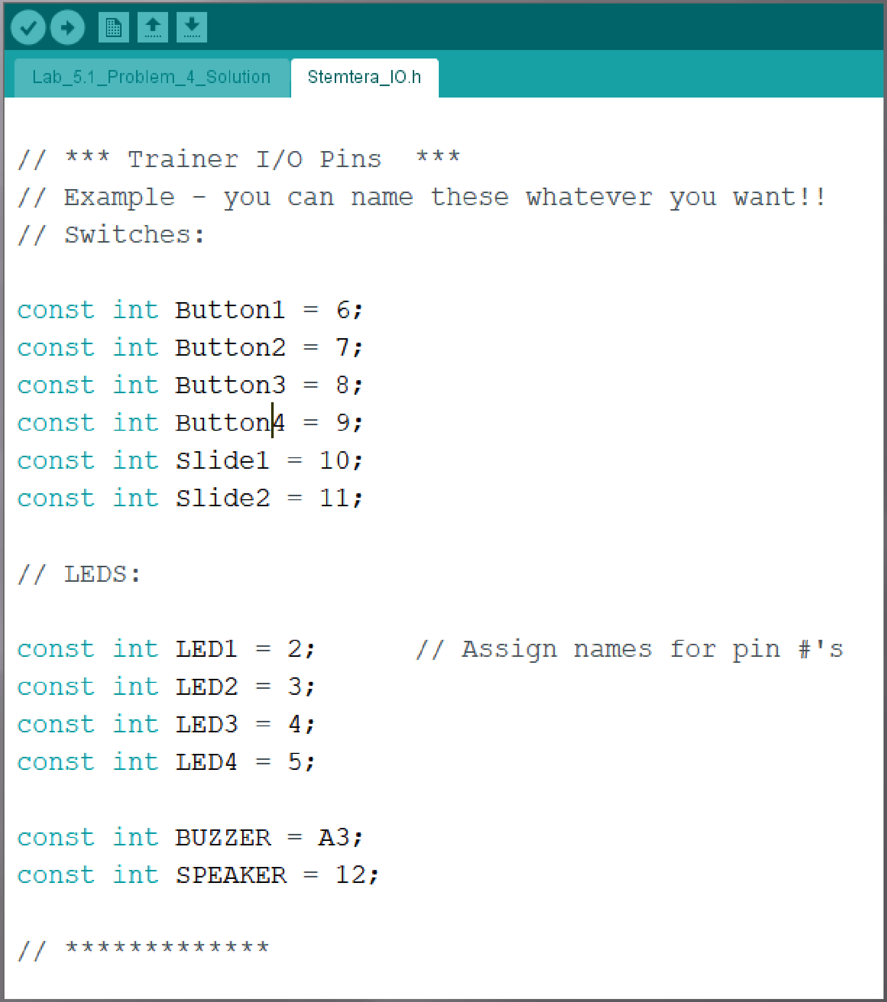


Lab_5.1_Problem_4_Solution | Arduino 1.8.8
File Edit Sketch Tools Help
Lab_5.1_Problem_4_Solution
Stemtera

```
// Example LED Cycle up-down:

/* Program Lab 5.1 Program 4:  LED
 Author:  T. Thoen
 10/1/20
 Program purpose:  Read input switc
```

Header (*.h) files

- I would highly recommend adding the following lines (or modify for your own naming preference) to your header file – that way you can use it in the rest of your programs!



```
// *** Trainer I/O Pins  ***
// Example - you can name these whatever you want!!
// Switches:

const int Button1 = 6;
const int Button2 = 7;
const int Button3 = 8;
const int Button4 = 9;
const int Slide1 = 10;
const int Slide2 = 11;

// LEDS:

const int LED1 = 2;      // Assign names for pin #'s
const int LED2 = 3;
const int LED3 = 4;
const int LED4 = 5;

const int BUZZER = A3;
const int SPEAKER = 12;

// *************
```

Digital outputs – more than LEDs!

“*If you can turn on an LED, you can turn on and off any electronic device*.” *

<u>However</u>, you are limited by the <u>current</u> and <u>voltage</u> levels from the Arduino.

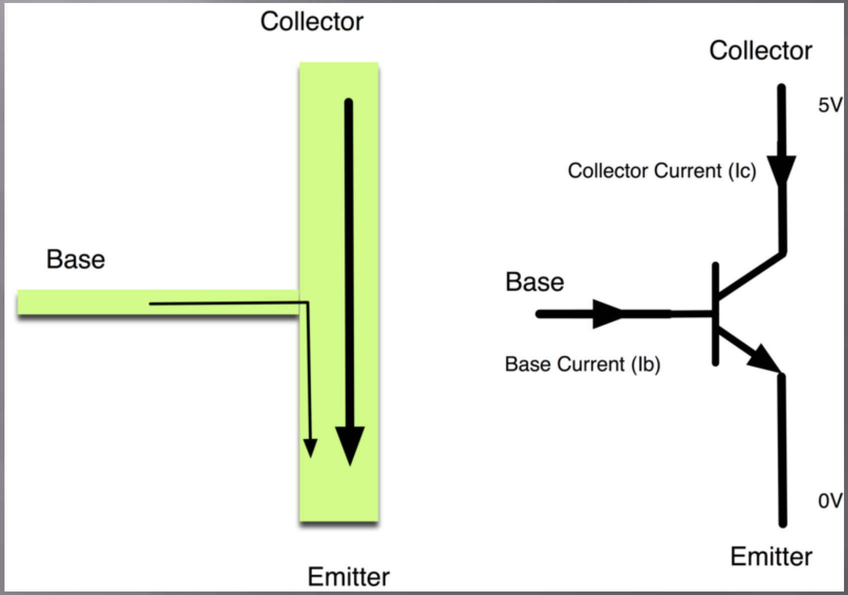
Typical digital outputs are limited to 5 volts at 30 milliAmps (mA).

To control devices that require more voltage or current we can use transistors or relays.

*T. Thoen

Transistor interfacing

Transistors can be used to switch higher voltage and current levels for D.C. circuits:

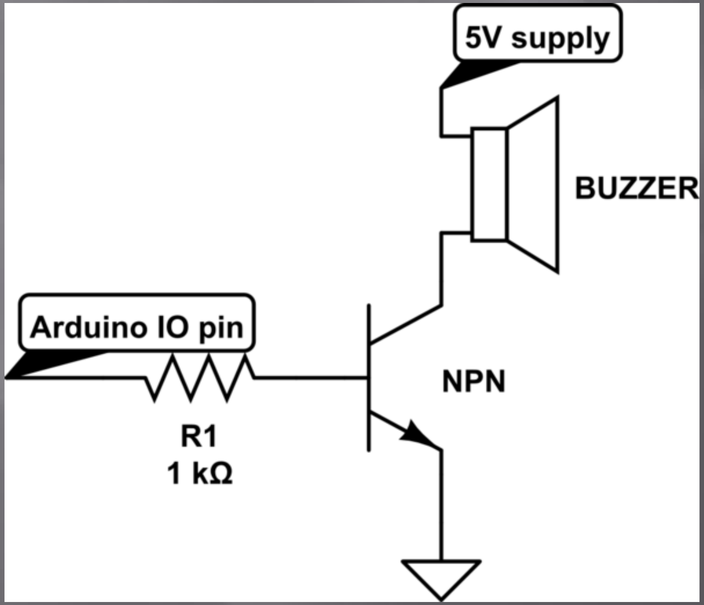


Basic concept – transistors can use a small current at the base to switch a large current at the collector.

How do we wire a transistor to control a buzzer?

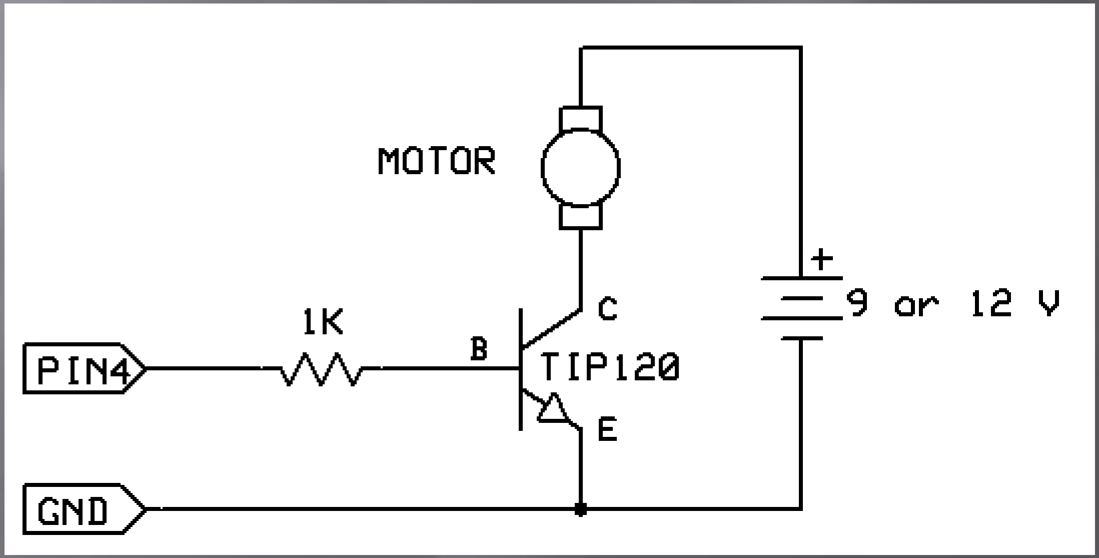
- For example, some buzzers require higher voltages or currents.

- We can connect a transistor to an Arduino output pin to increase the current and allow higher voltages.



How do we wire a transistor to control a motor?

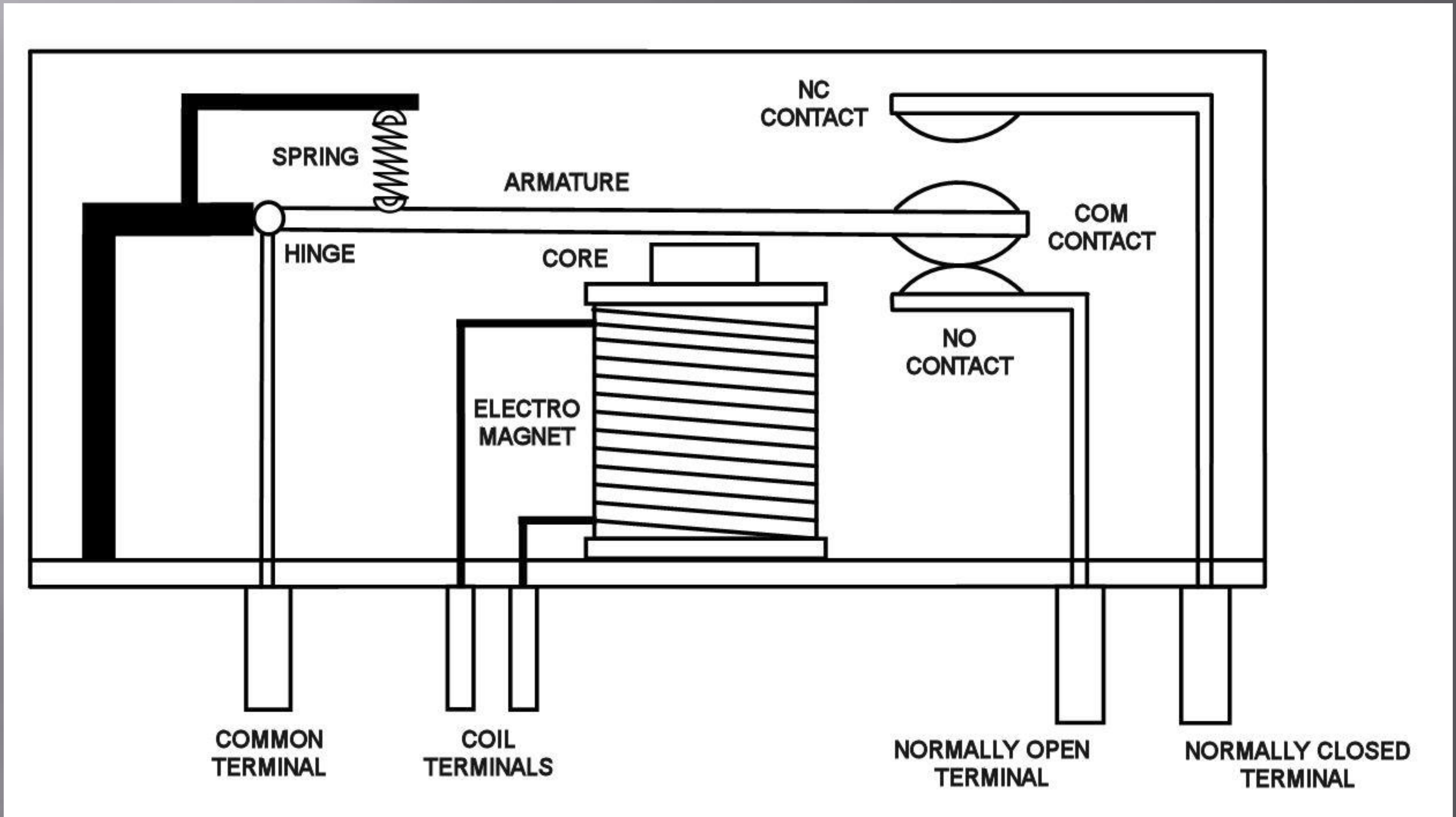
- We can use the same method to turn a motor on and off.
- However, we need to add a diode to prevent voltage damage on the output pin.
- Notice that the grounds MUST be connected!!



Using Relays

- Relays are electro-mechanical switches
- Relays can be used to control higher voltages or AC current.
- Relays also isolate the Arduino signals from the higher voltages and currents.
- This isolation provides more safety in the wiring.

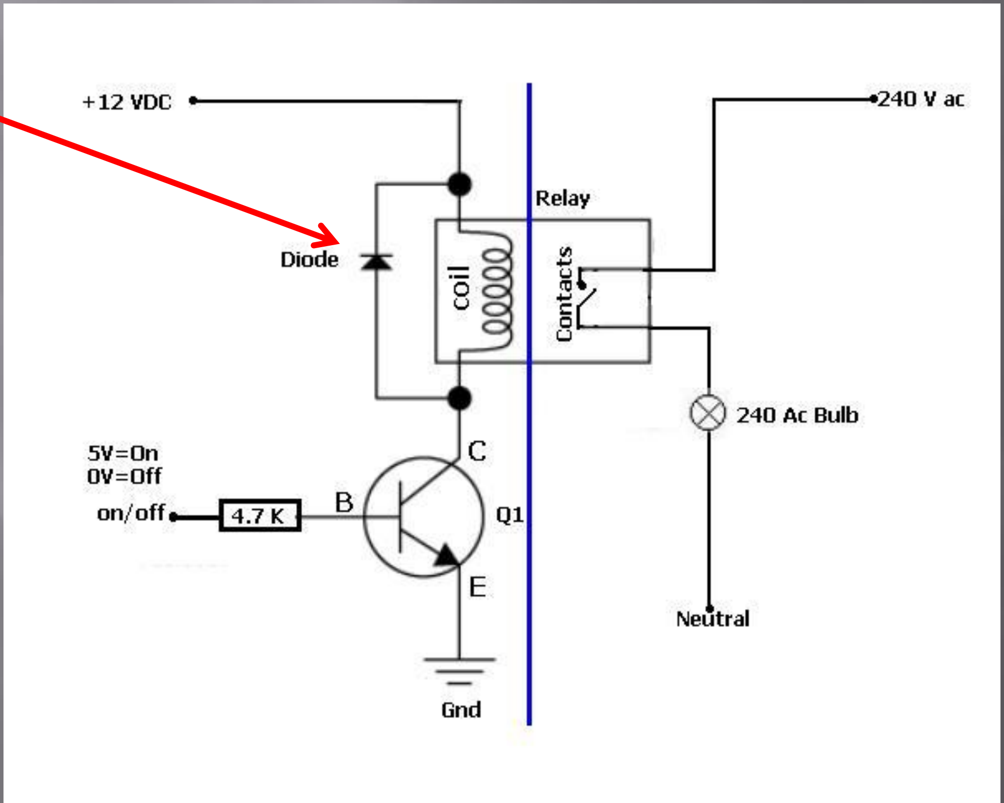
Using Relays



Using Relays

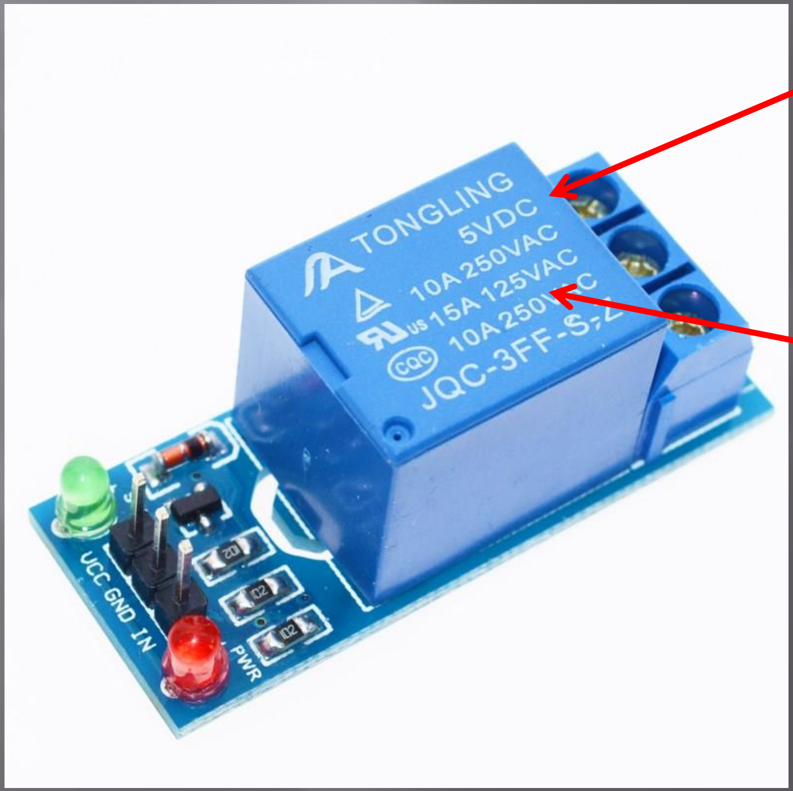
Just like motors, diodes should be used when connecting relay coils to an Arduino output.

Coil diode



Relay ratings

- Make sure to check the coil ratings – if greater than 5V at 30mA, use a transistor.
- Output ratings are often greater than 120V at 3 Amps!



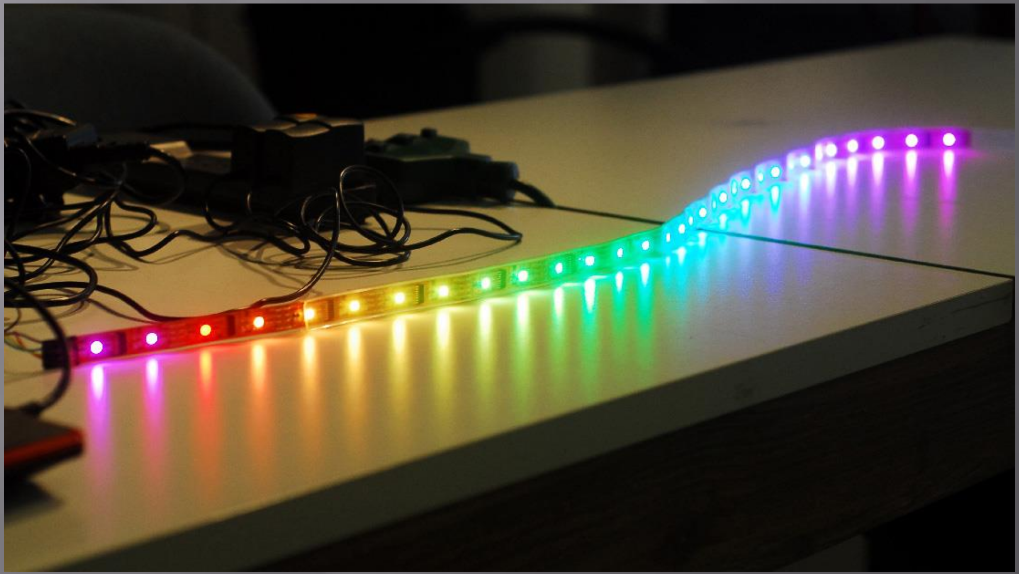
Coil voltage = 5V

Contact voltage: 125V @ 15 Amps!

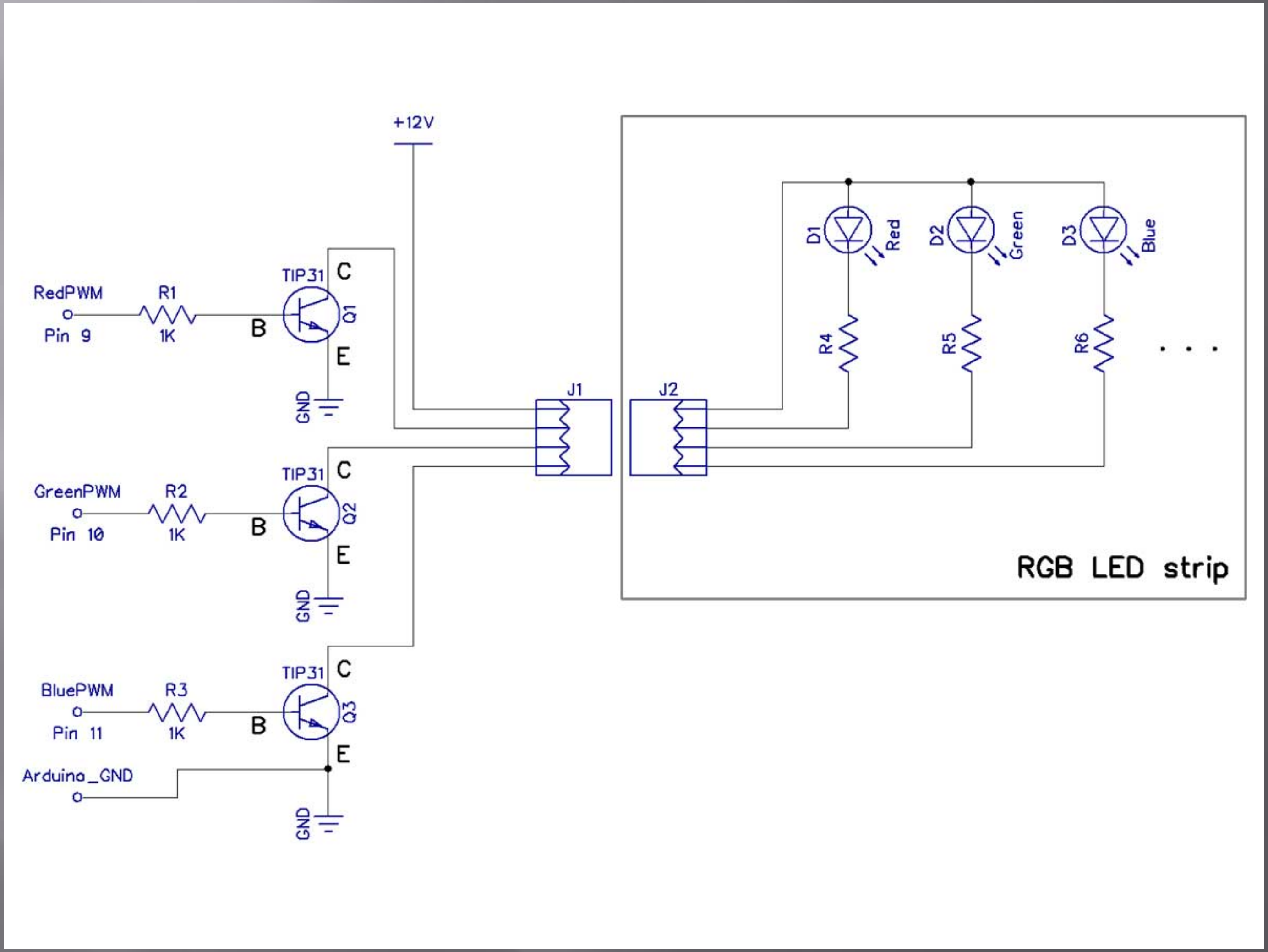
How do we wire a transistor to control an LED string?

- By replacing the motor with a string of LEDs we can control higher voltage (12V) LED strings.

- Again, Notice that the grounds MUST be connected!!



We can also use the analog outputs to control brightness:



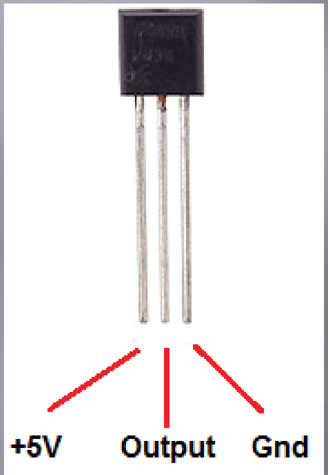
Analog inputs

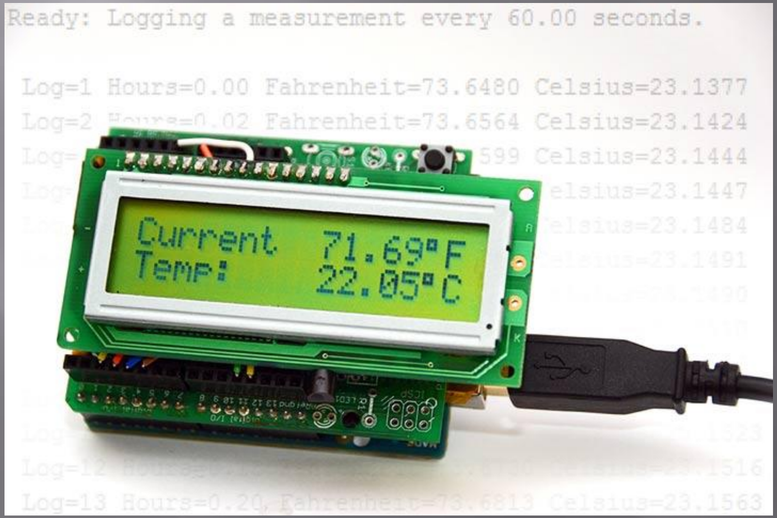
Analog inputs can be used to read sensors:

- **Temperature**
- **Pressure**
- **Light**
- **Sound**
- **Direction**
- **Acceleration**
- **Controller inputs**

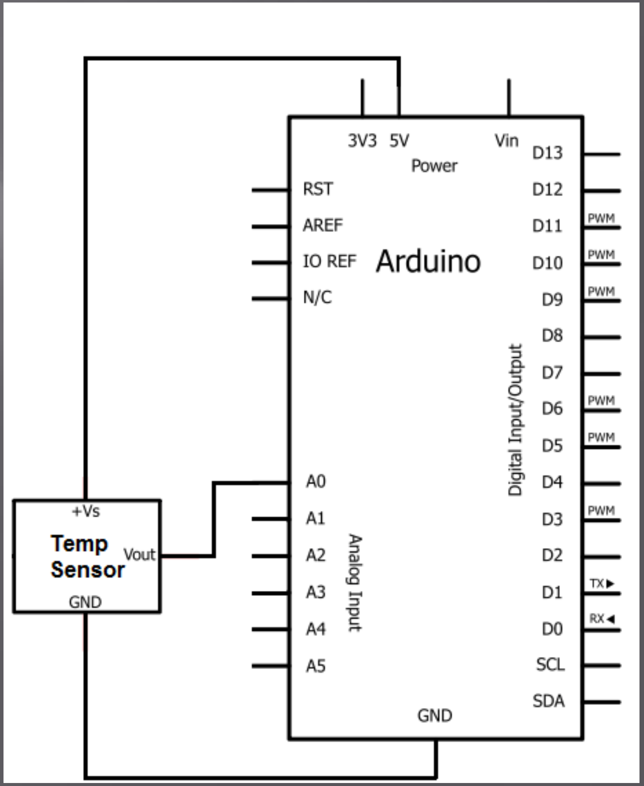
Temperature example

LM34 – analog temperature I.C.





```
Ready: Logging a measurement every 60.00 seconds.

Log=1 Hours=0.00 Fahrenheit=73.6480 Celsius=23.1377
Log=2 Hours=0.02 Fahrenheit=73.6564 Celsius=23.1424
Log=                               599 Celsius=23.1444
                                       elsius=23.1447
                                       elsius=23.1484
                                       elsius=23.1491
Log=12 Hours
Log=13 Hours=0.20 Fahrenheit=73.6813 Celsius=23.1563
```

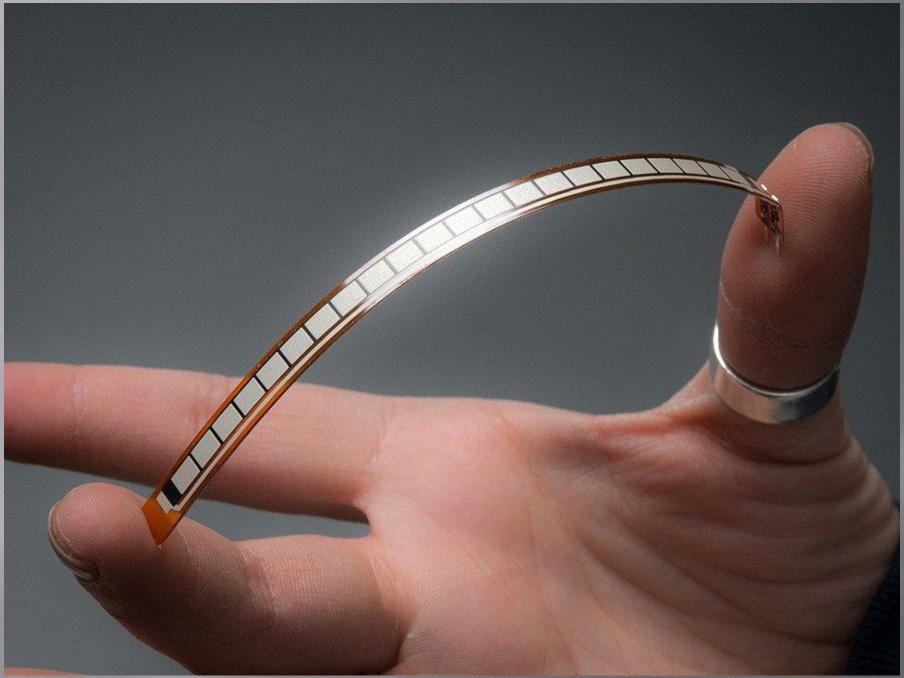


Analog inputs: Joystick

Two potentiometers for X-Y feedback

Since the values are only a fraction (approx. 90 degrees) of the rotation you can use the MAP function to scale the values.

Analog inputs: Flex sensors

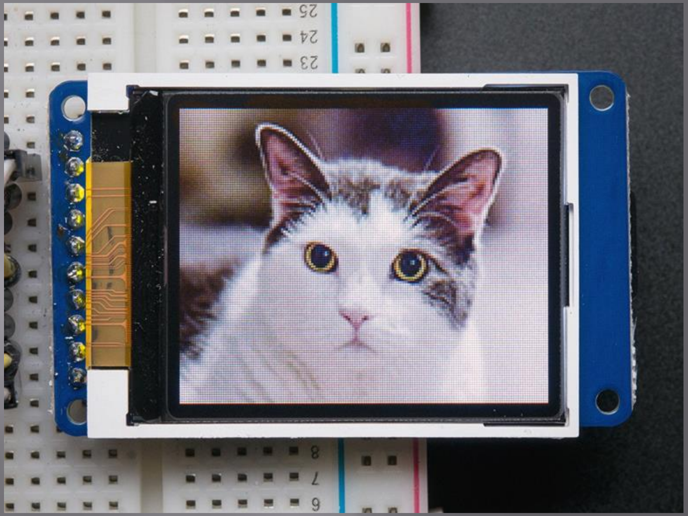


Flexing the material changes the resistance

Used for VR feedback with gloves, etc.

More complex devices…

LCD displays



Graphic Displays

RGB arrays

How do I program with these devices??

- Often there are libraries available from Sparkfun and Adafruit

- The libraries are pre-written functions that allow “easy” interfacing

- The Arduino library also has a number of libraries pre-defined for interfacing with devices

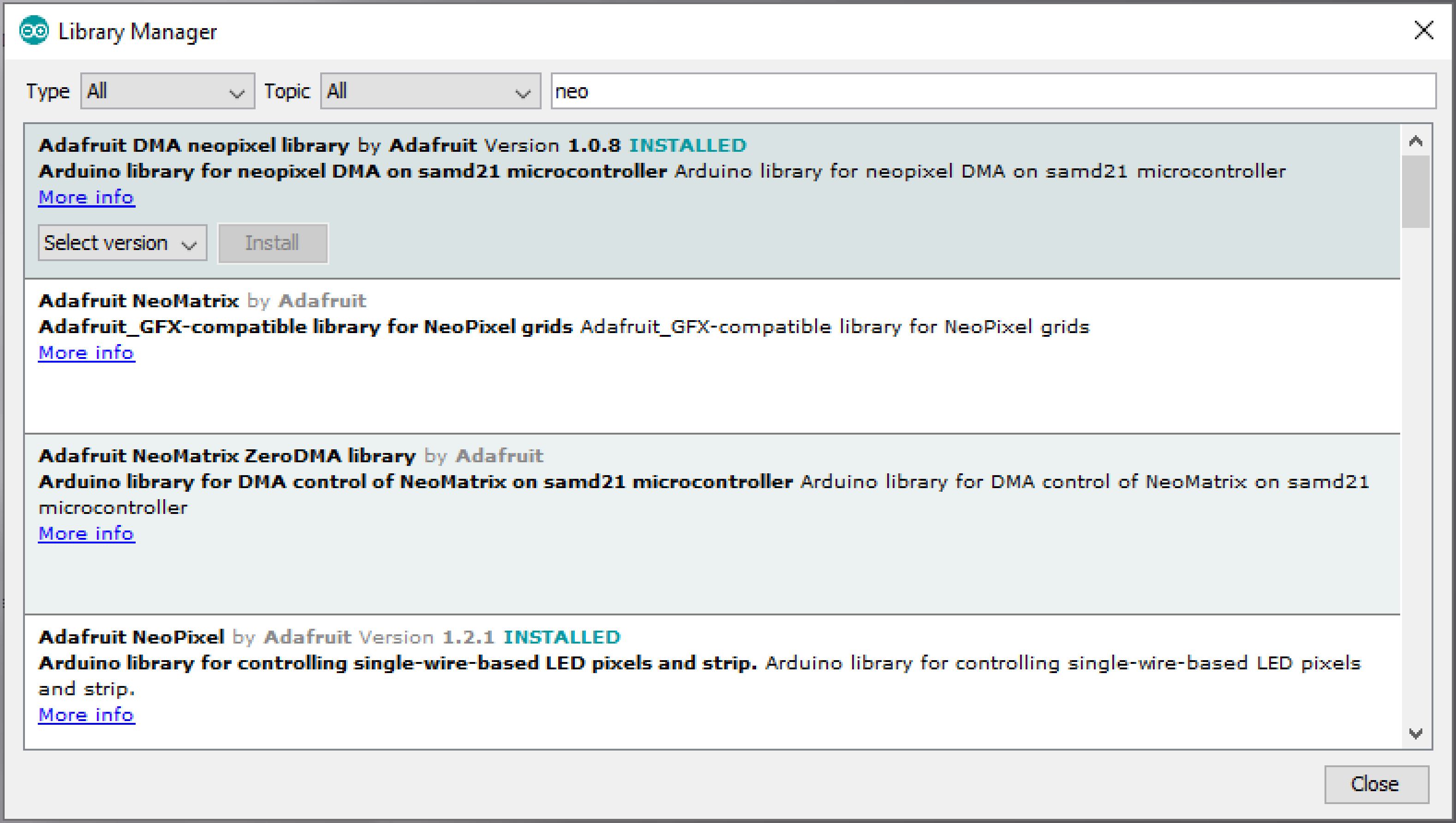
Libraries

- Many libraries are included with the Arduino program – however it requires them to be added

- Libraries are added through the ***Tools – Manage Libraries*** option

- For example, Neo-pixels:

Libraries



Type: All | Topic: All | neo

Adafruit DMA neopixel library by **Adafruit** Version **1.0.8** INSTALLED
Arduino library for neopixel DMA on samd21 microcontroller Arduino library for neopixel DMA on samd21 microcontroller
More info

Adafruit NeoMatrix by **Adafruit**
Adafruit_GFX-compatible library for NeoPixel grids Adafruit_GFX-compatible library for NeoPixel grids
More info

Adafruit NeoMatrix ZeroDMA library by **Adafruit**
Arduino library for DMA control of NeoMatrix on samd21 microcontroller Arduino library for DMA control of NeoMatrix on samd21 microcontroller
More info

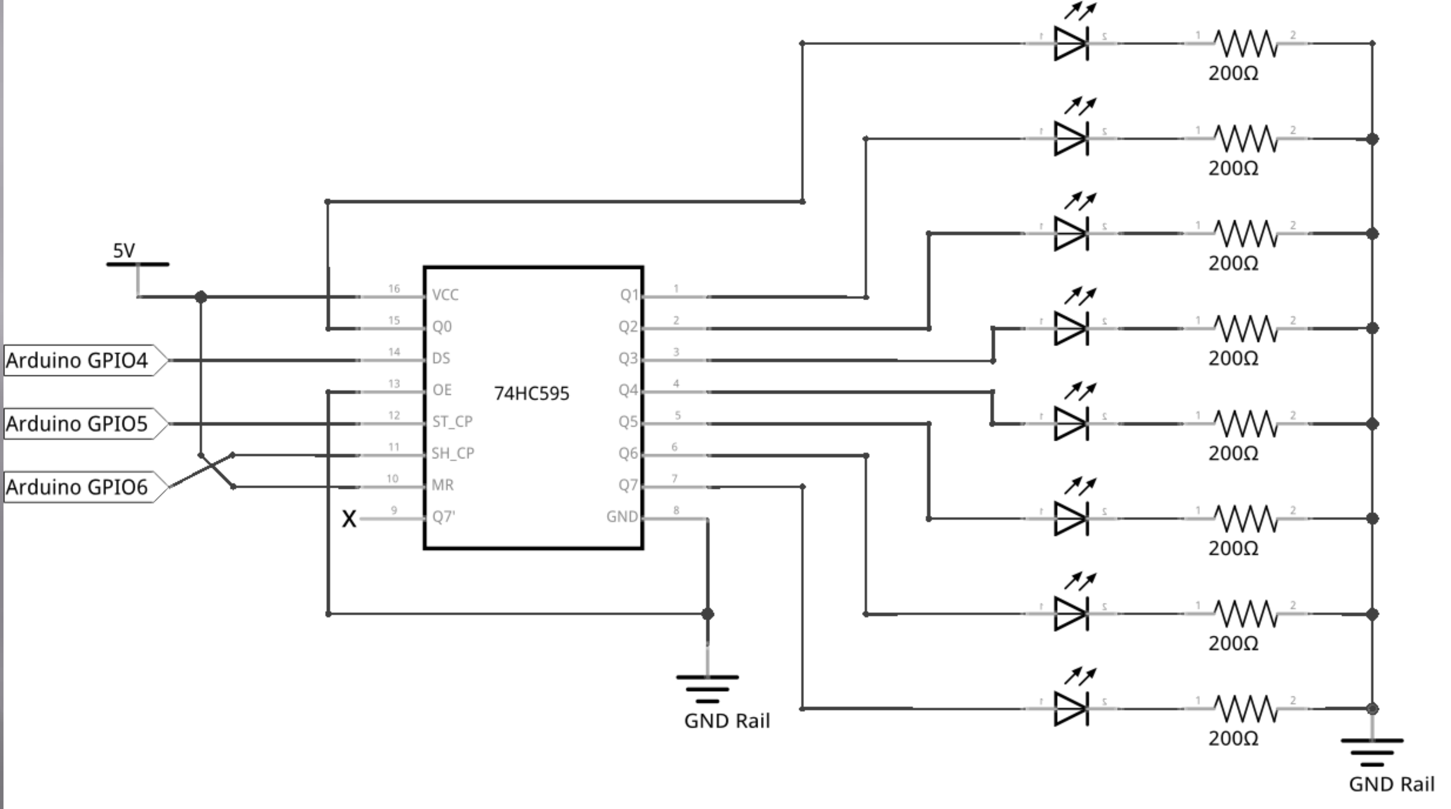
Adafruit NeoPixel by **Adafruit** Version **1.2.1** INSTALLED
Arduino library for controlling single-wire-based LED pixels and strip. Arduino library for controlling single-wire-based LED pixels and strip.
More info

Expanding I/O

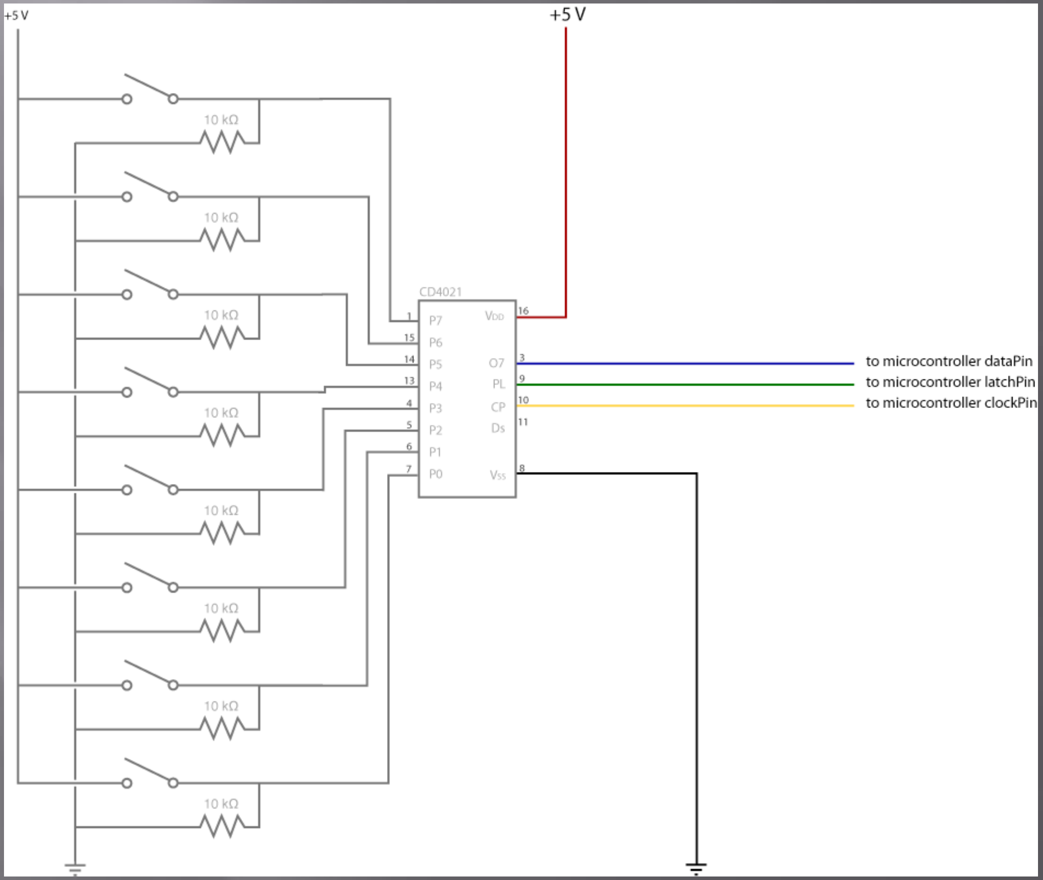
- The Arduino has limited I/O
- You can move up to a Mega board
- 54 I/O vs. 14!!
- Another way is using shift registers

Output Shift Registers

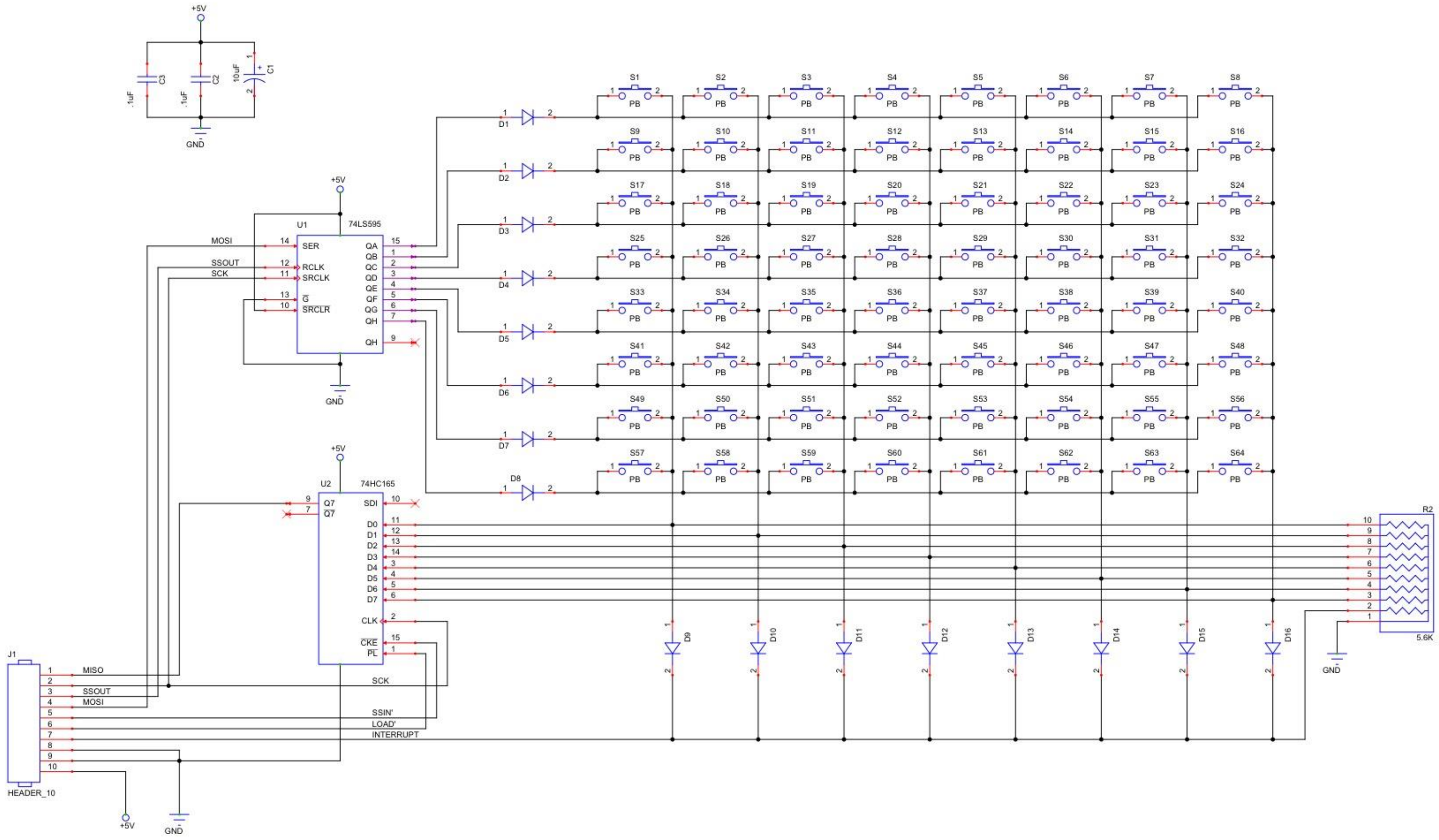
- 74595 Outputs:



Input shift registers: 74165 or 4021 Shift in



Creating a keyboard matrix



Stepper motors

- Stepper motors are driven by electronic pulses
- Each pulse moves the motor shaft a fixed rotary distance (i.e. 1.8 degrees)
- They can also be used with “microsteppers” to move fractions of a degree for more precision.
- Transistors or IC’s are required to interface between the Arduino and Stepper to increase the current output

https://www.arduino.cc/en/reference/stepper

Example: Stepper motors