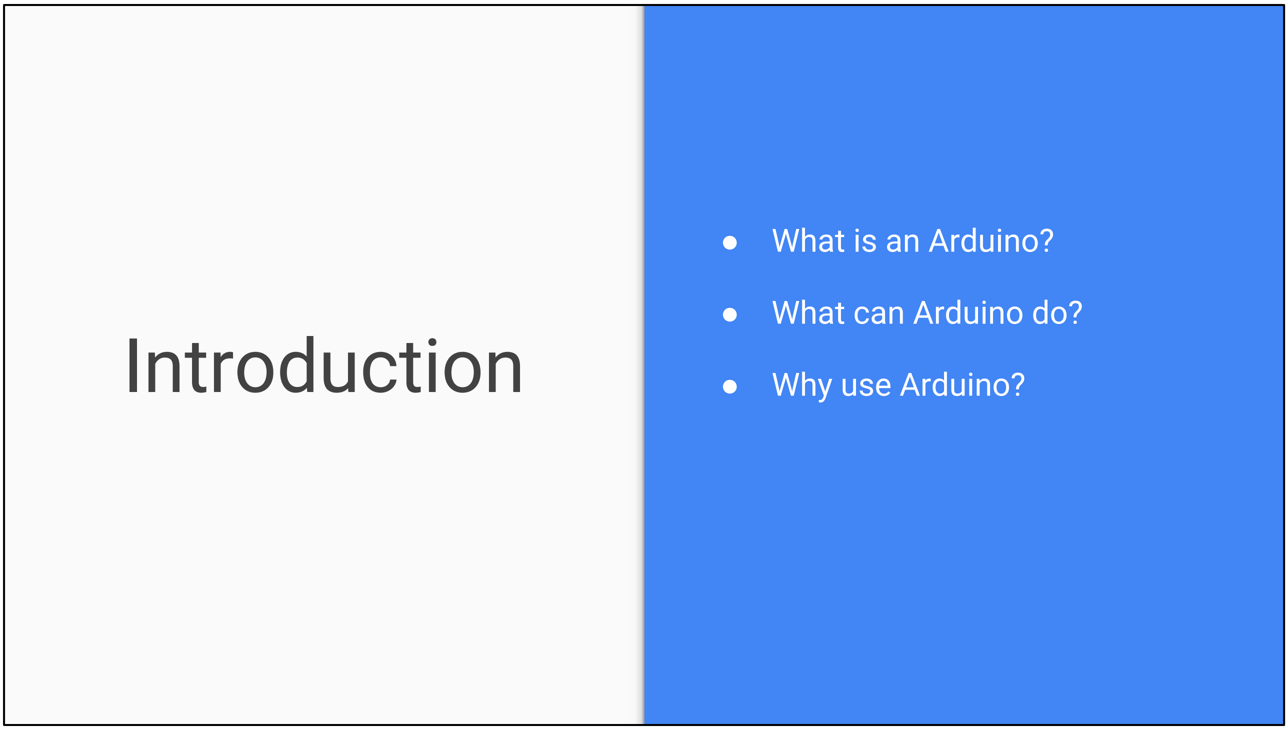
ARDUINO 101 - T. COWLES

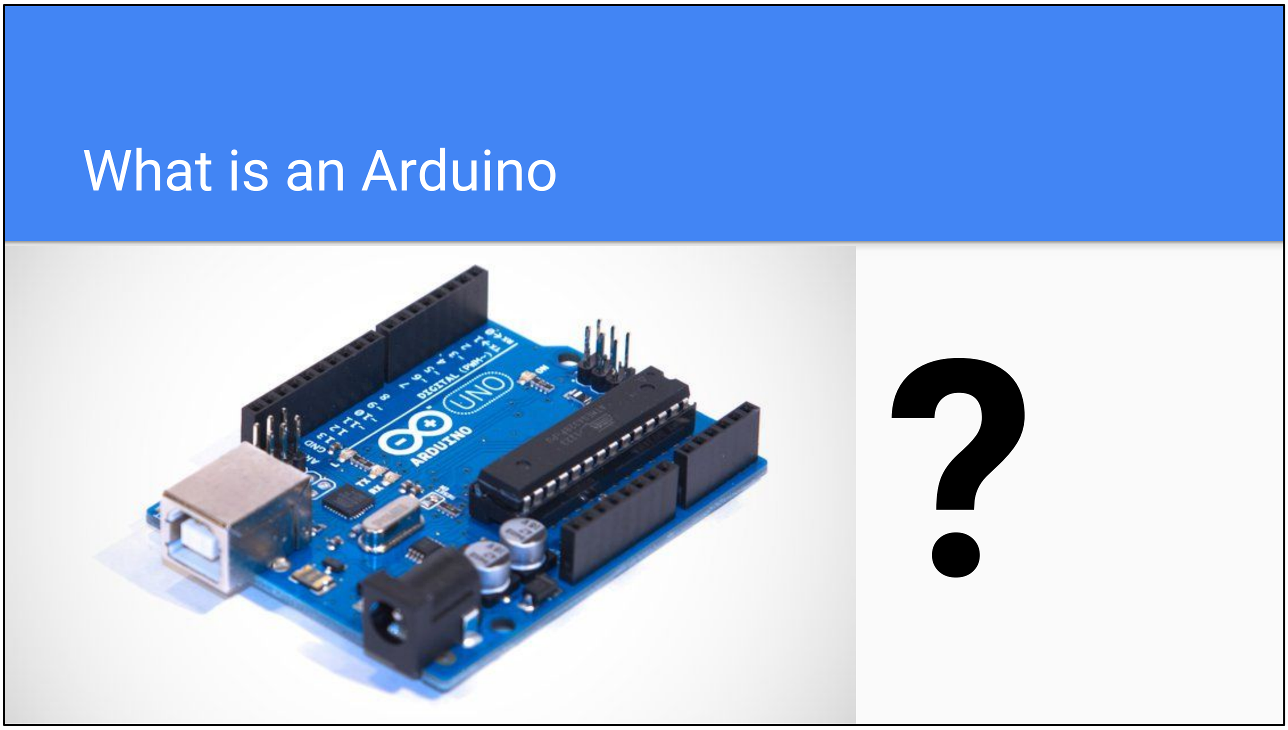
September 16, 2017



Introduction

- What is an Arduino?
- What can Arduino do?
- Why use Arduino?

What is an Arduino?

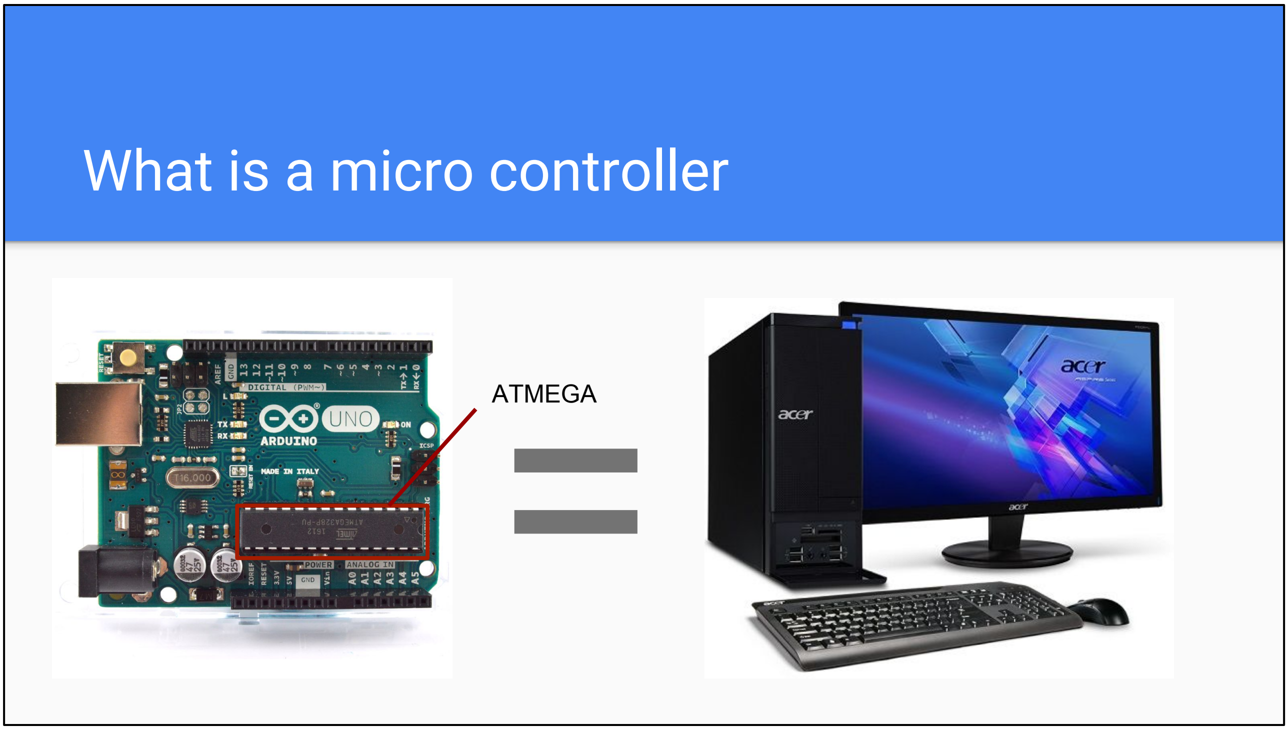


What is an Arduino

?

Simply put an Arduino is a 8 bit microcontroller, we will talk more about microcontrollers later
-There are many different types of arduino, the basic type we will be using is called the ‘Uno’ but other common types include the nano and the mega
-Because Arduino is open source there are genuine and clone arduinos

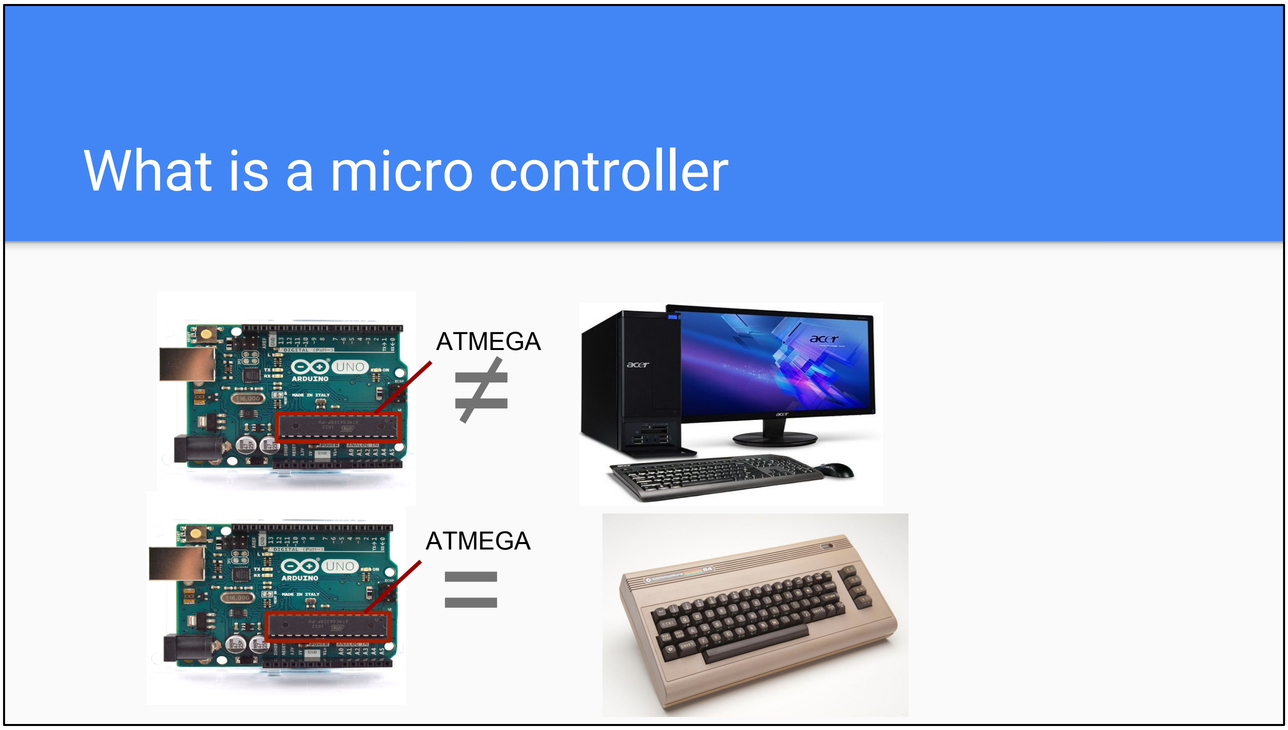
(ask does anyone know what a microcontroler is)



What is a micro controller

ATMEGA

An arduino is really a microcontroller board, the microcontroller the arduino is based on is called the ATMEGA
An ATMEGA like any microcontroller technically is a computer



What is a micro controller

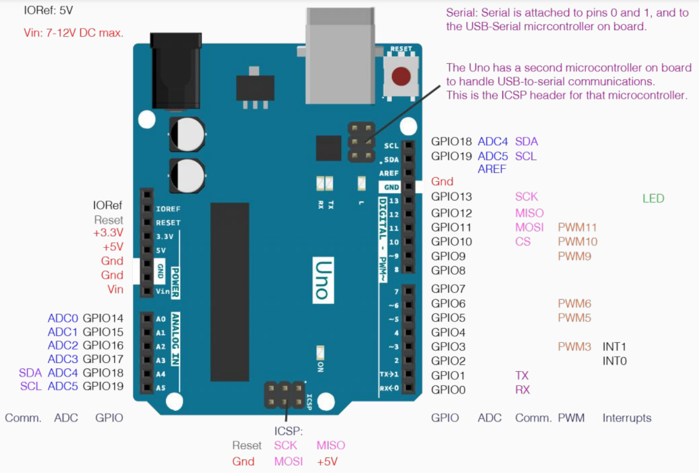
ATMEGA ≠

ATMEGA =

But the arduino is such a slow computer that it’s ATMEGA CPU is much closer to something like the C64 an 8 bit computer that came out in 1986 then any modern computing device

What can micro controllers do?

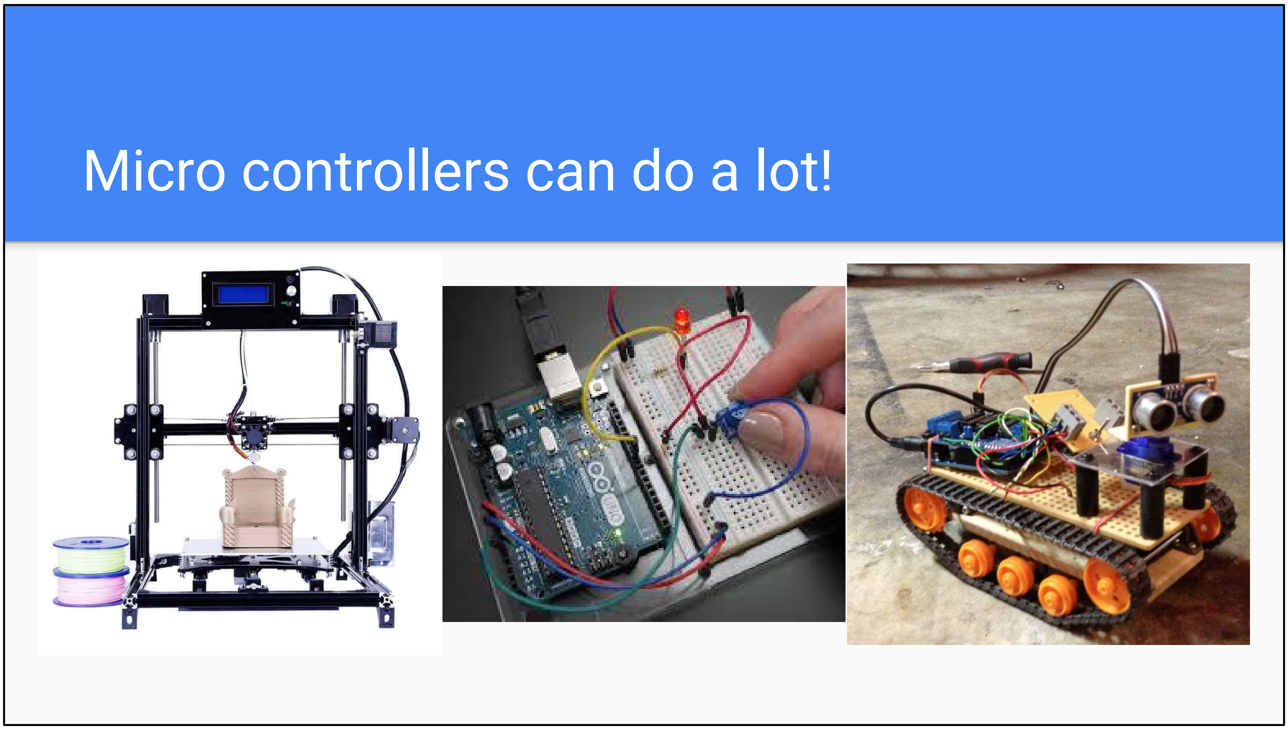
What can micro controllers do?



Serial: Serial is attached to pins 0 and 1, and to the USB-Serial micrcontroller on board.

The Uno has a second microcontroller on board to handle USB-to-serial communications. This is the ICSP header for that microcontroller.

The tiny amount of computing power that microcontrolers have might make you beleive that they can’t do much but they have one saving grace; GPIO, GPIO or (ask) General Perpuse Input output, allows microcontrols to control all sorts of devices that normal computers can’t

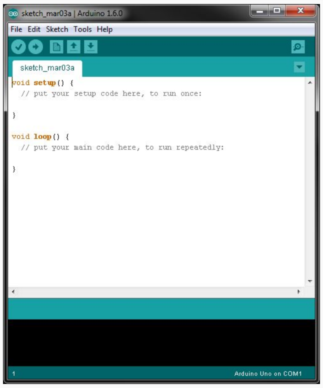


Micro controllers can do a lot!

Uses of this GPIO can be for simple digital on off devices and basic switches as we will focus on today or for much more complicated devices, most 3d printers including the library's new lulzbot and my two prusia clones at home use Arduino Megas and something called a Ramps 1.4 or Ramps 2 (Rambo) as there main board, Arduino's can also be used for many simple robotics projects

Why use Arduino over other microcontrollers?

Main advantages to Arduino



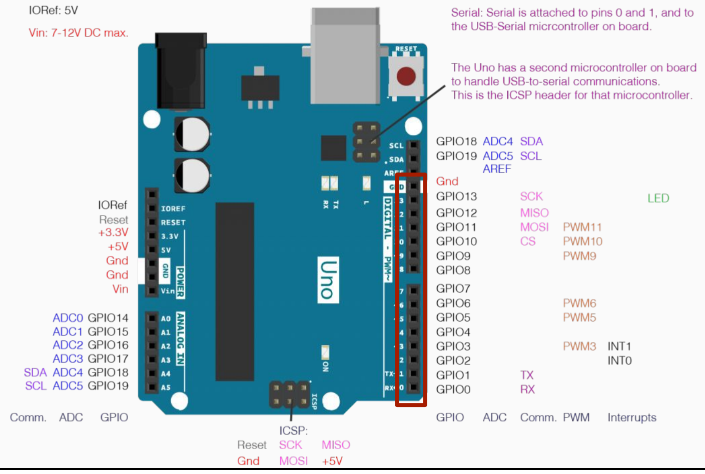
```
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

A great community, there comparatively low price and easy to use IDE make arduino the go to microcontroller for most electronic hobbyists

Main activities

- Intro to Arduino hardware with the blink example
- Intro to the Arduino IDE and flashing examples
- Arduino HW continued
- Intro to programming in the IDE
- Free Arduino projects

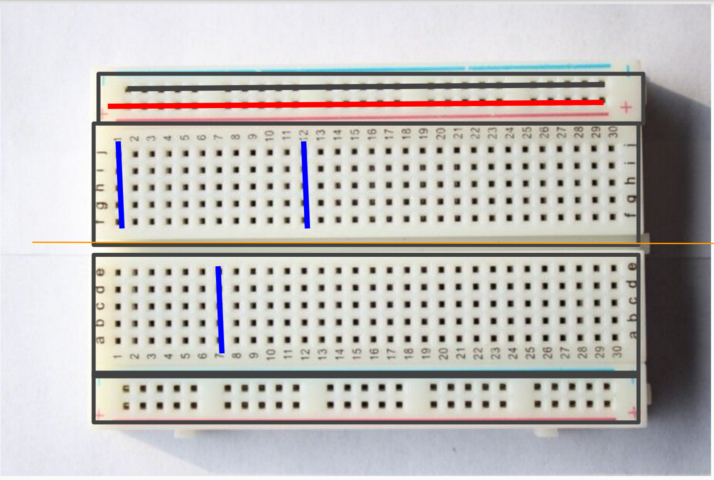
Arduino hardware step 1 GPIO


Serial: Serial is attached to pins 0 and 1, and to the USB-Serial micrcontroller on board.

The Uno has a second microcontroller on board to handle USB-to-serial communications. This is the ICSP header for that microcontroller.

We will use the digital GPIO to accept inputs from buttons or sensors and to output to LEDs Buzzers etc.

HW step 2 Breadboard

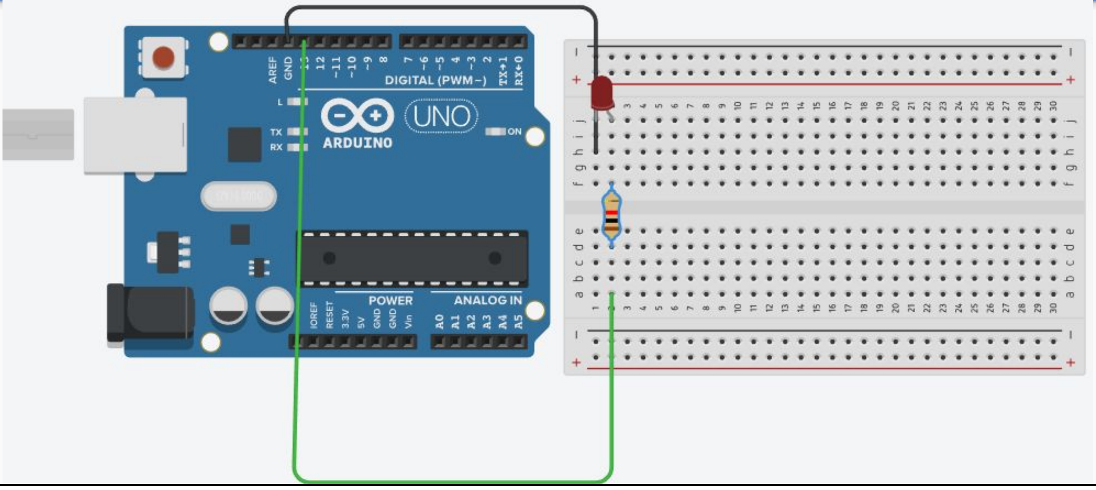


The breadboard comes in 4 sections,

2 horizontal sections and 2 vertical sections, in these sections there are many sub sections

The sections are mirrored over the orange line

Add an LED

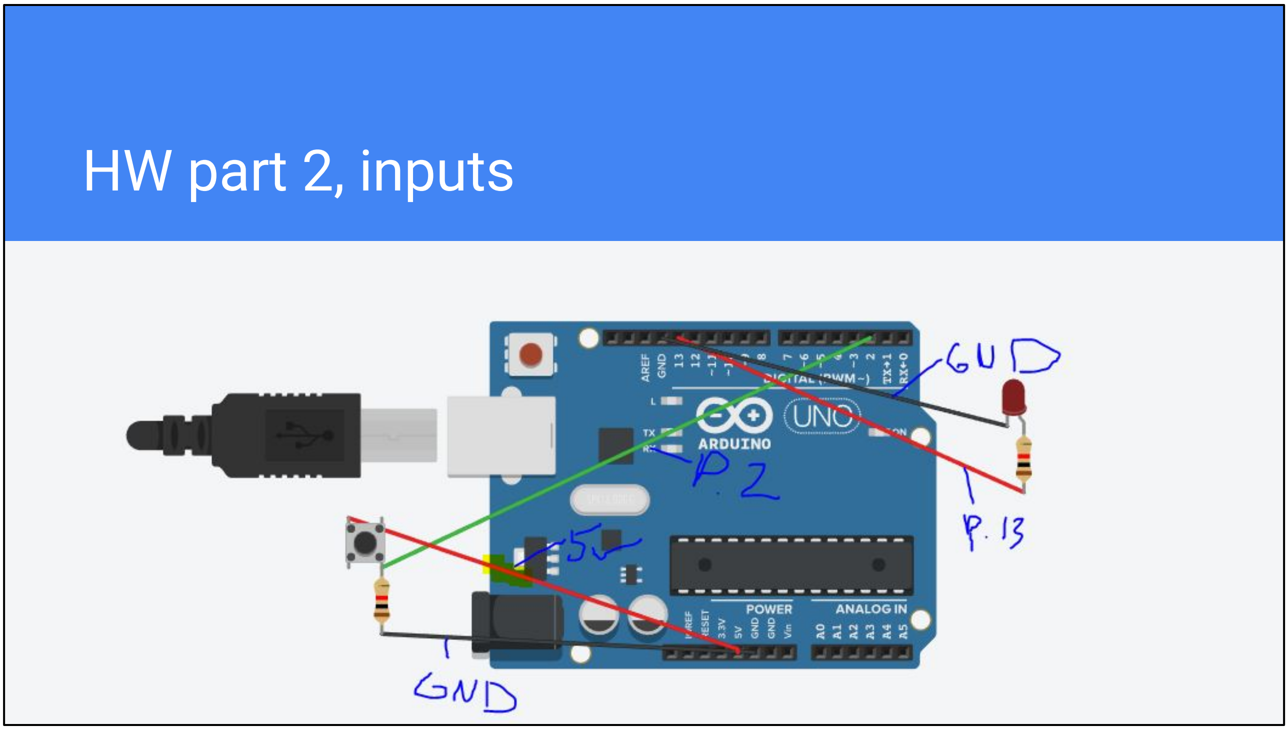

AREF
GND
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX→1
RX←0
DIGITAL (PWM~)
L
TX
RX
UNO
ARDUINO
ON
POWER
IOREF
RESET
3.3V
5V
GND
GND
Vin
ANALOG IN
A0
A1
A2
A3
A4
A5
a b c d e
f g h i j
–
+

Now we move to the IDE

Visit arduino.cowles.cf to download the switch example,

Open the Arduino IDE

Follow instructions to load/flash the example



HW part 2, inputs

Here is a free form picture, sometimes you will see these online, remember how bread board is wired and try to translate this to it

If you need help then please raise your hand

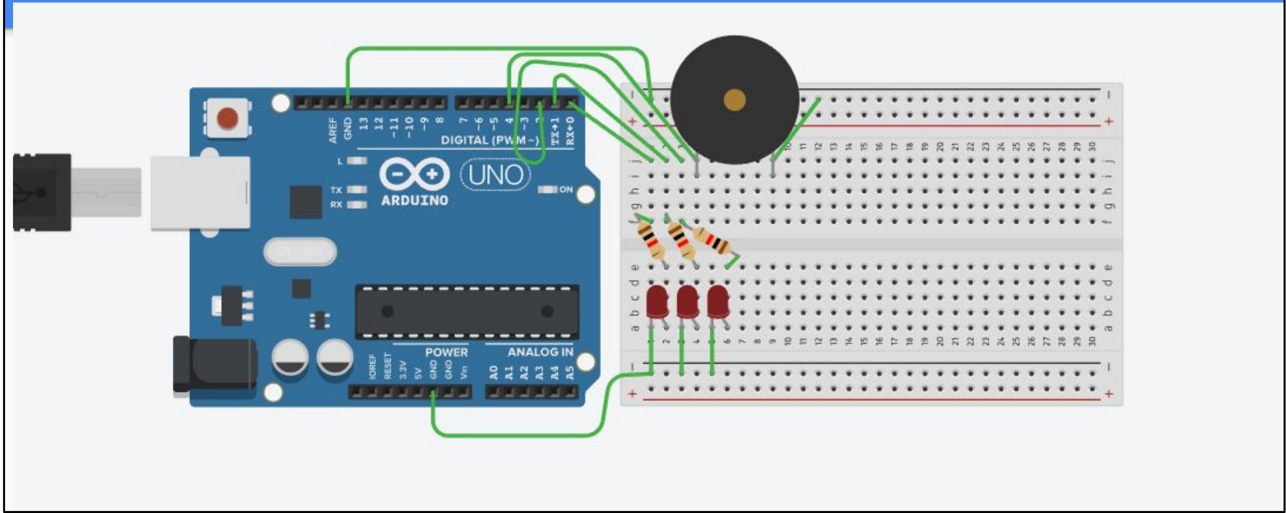
Arduino programing!

We are going to create an alarm that flashes some lights and rings a buzzer when a button is pushed, later if we wanted to we could replace this button with a pressure plate or a circuit that completes when a door is opened etc.

Optional Challenges: -include an RGB status indicator using the RGB LEDs
-Use variables to make the alarm ring until a deactivation button is pressed

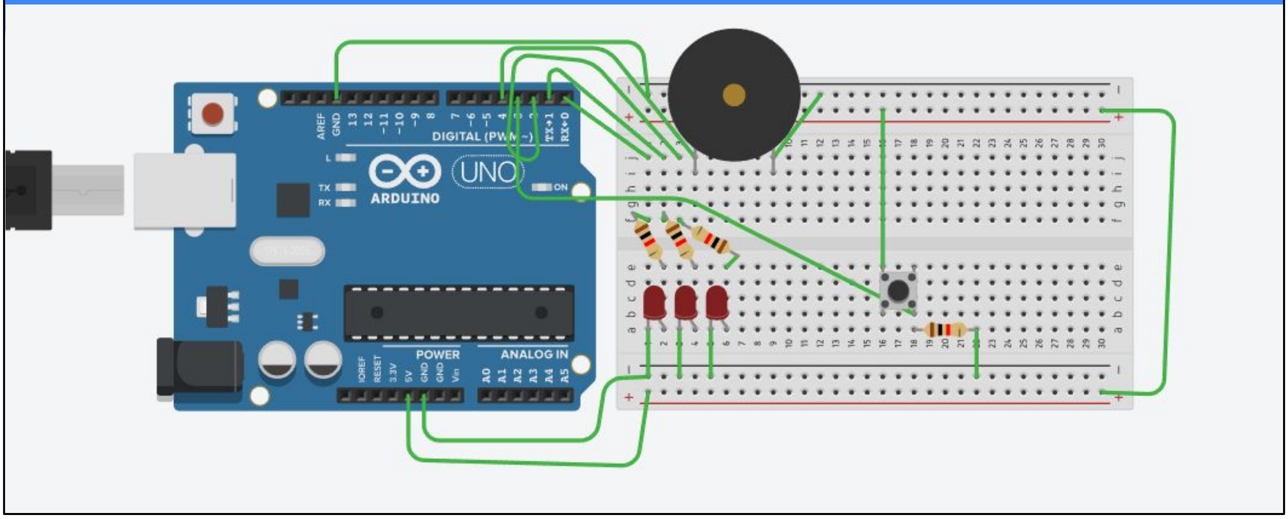
Step 1- Choice what pins will be used for what, consider how many LEDs you want, etc
I recommend writing these pins down in a document that you keep in the same folder as your code

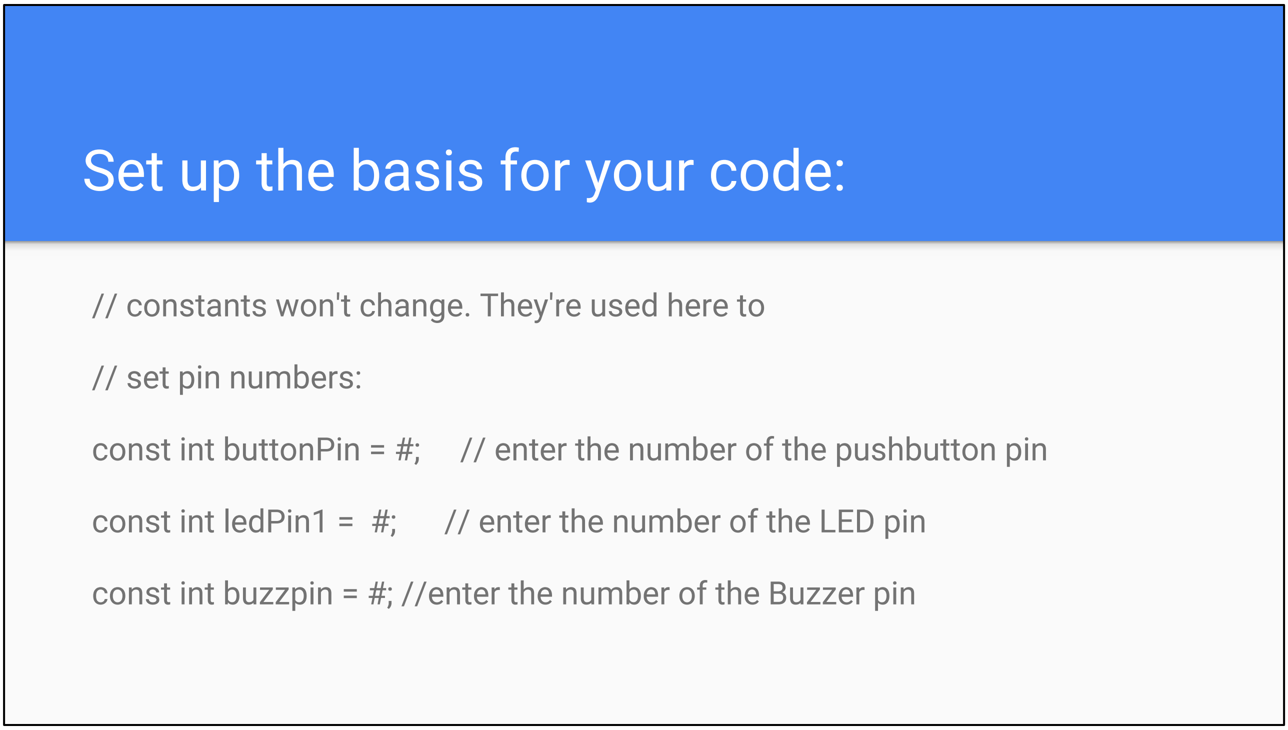
Example setup



I am using Pin 0-2 for my LEDs and pin 4 for my buzzer, my button while be on on pin 3: can you remember how to wire a button

And the button

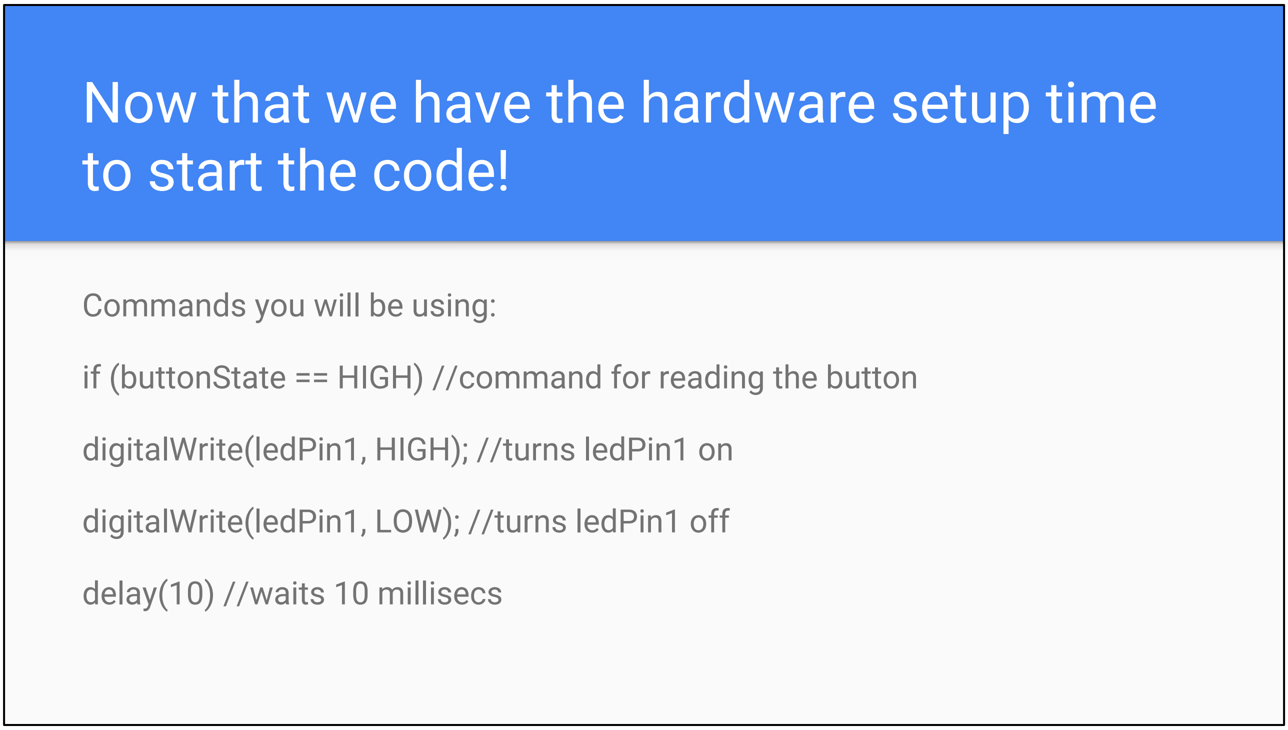




Set up the basis for your code:

```
// constants won't change. They're used here to

// set pin numbers:

const int buttonPin = #;     // enter the number of the pushbutton pin

const int ledPin1 =  #;      // enter the number of the LED pin

const int buzzpin = #; //enter the number of the Buzzer pin
```

This is in the folder you downloaded earlier as it goes off screen



Now that we have the hardware setup time to start the code!

Commands you will be using:

```
if (buttonState == HIGH) //command for reading the button

digitalWrite(ledPin1, HIGH); //turns ledPin1 on

digitalWrite(ledPin1, LOW); //turns ledPin1 off

delay(10) //waits 10 millisecs
```

These are the buildings blocks for many basic arduino projects, by changing a few minor details you can create 1000s of diffrent projects,

Anyone want to share their alarm?

Did you make the LEDs Flash?

How long does it ring?

How many LEDs are there?

Did you build a status indicator with the RGB Leds?

Any other details you would like to demonstrate?

FREE ARDUINO PROJECT TIME

If we have any time left then you are free to use the materials, work with each other, and ask me questions to create your own arduino project :)

HAVE FUN!

THANK YOU!

Thanks for coming, if you have any further questions then feel free to email me at **tobycowles1015@gmail.com** visit TGCPROJECTS.TK if I ever get it working again for poorly edited tutorials for electronics projects or if you forget me email you can also reach me through the contact page at TGCTECH.TK