OLD DOMINION UNIVERSITY

Building Leaders for Advancing Science and Technology (BLAST)

Smart Technologies:
Introduction to Arduino and Hands-On Activities

Dr. Murat Kuzlu

Department of Engineering Technology

Outline

- What are Smart Technologies?
- What is a Microcontroller?
- What is Arduino?
- Overview of Arduino Development Board
- Setting Up the Arduino IDE
- Basic Arduino Program Structure
- Hands-on Activities
- Download Folders
  - Github Repository (Recommended)
  - **https://github.com/muratkuzlu/ODU_BLAST_2024**

About Me

- **Dr. Murat Kuzlu** (Senior Member, IEEE) is an Associate Professor at Old Dominion University (ODU).

- **Christopher Grogesky** graduated from Old Dominion University Majoring in EET with a concentration in Computer Engineering Technology

- **Madison Currie** is a current student at Old Dominion University Majoring in EET with a concentration in Computer Engineering Technology

What are Smart Technologies?

- Advanced systems and devices that are capable of learning, and performing tasks **intelligently**

- Utilize **sensors**, **controllers, embedded systems,** data analytics, and artificial intelligence (AI) to operate and improve performance over time
- Examples: smart home devices, smart cities, smart factories, smart vehicles, ….

- Key Components: **Sensors, Connectivity, Data Processing, User Interface**

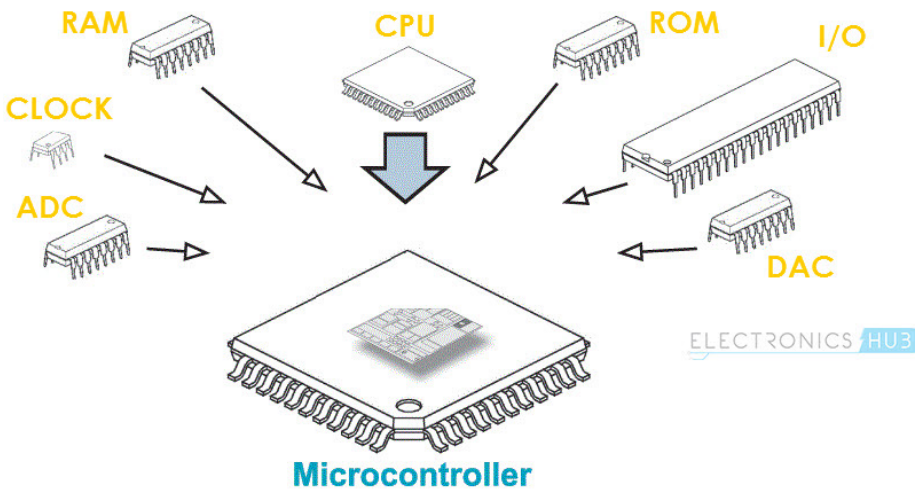
What is a Microcontroller

- A compact integrated circuit
- Called as a Computer-on-a-Chip or a Single-Chip-Computer
- Typically ***embedded*** inside some device

- **Key Components:**
  - Processor,
  - Memory
  - I/O peripherals
  - …

- **Applications:**
  - Consumer electronics,
  - Industrial automation,
  - Medical devices
  - ….

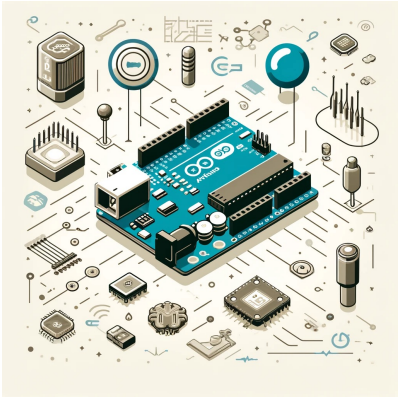
- **Examples:**
  - Arduino:
  - PIC Microcontrollers:
  - TI MSP430
  - ARM Cortex-M
  - …..



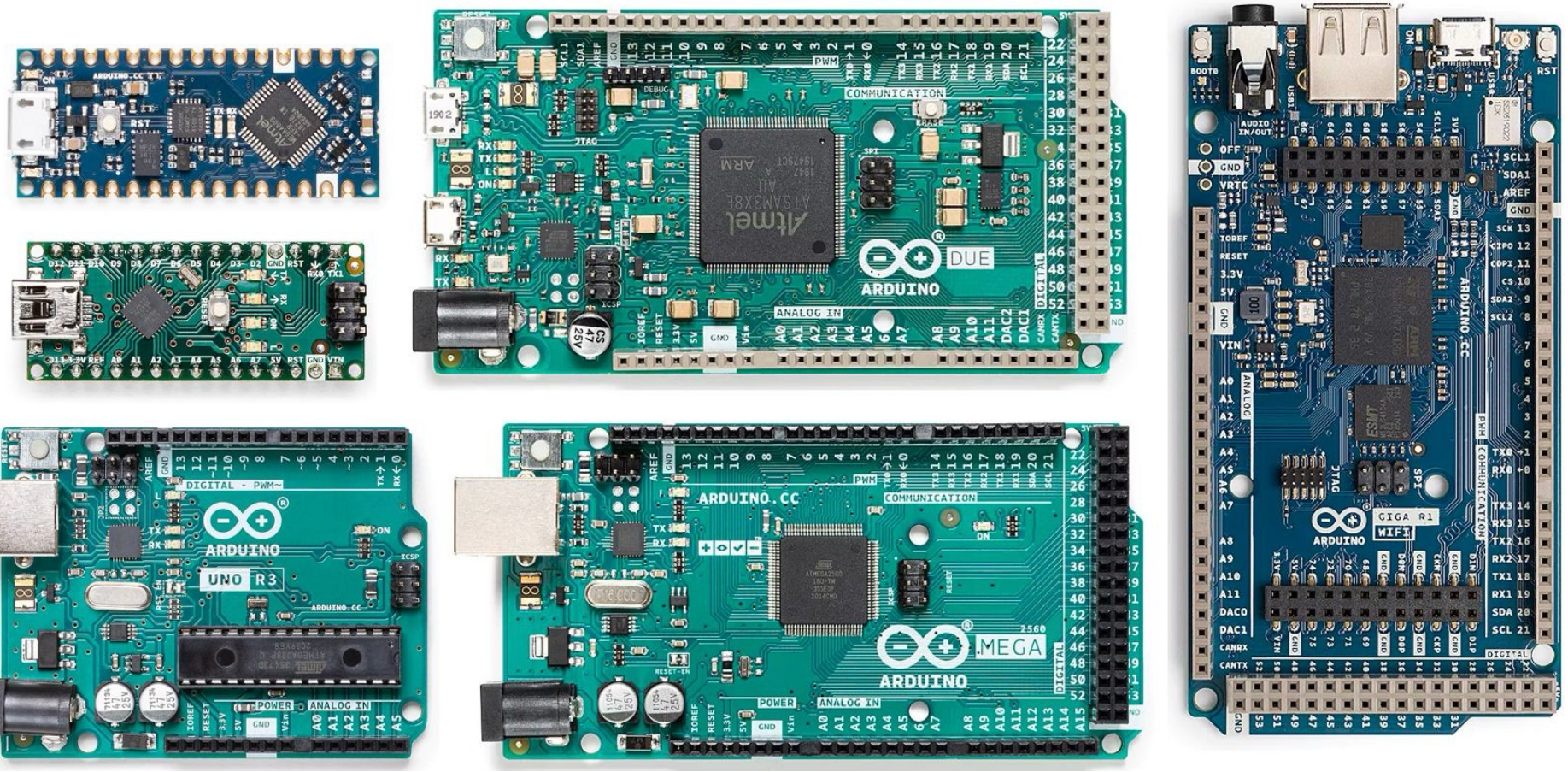
What is Arduino?

- An open-source electronics platform based on easy-to-use hardware and software
- **Easy to use**, connects to computer via USB
- Run in standalone mode and as an interface connected to PC/Macintosh computers
- **Inexpensive**, and comes with free authoring software

- **Key Components:**
  - Microcontroller
  - Development environment (IDE),
  - Libraries.…



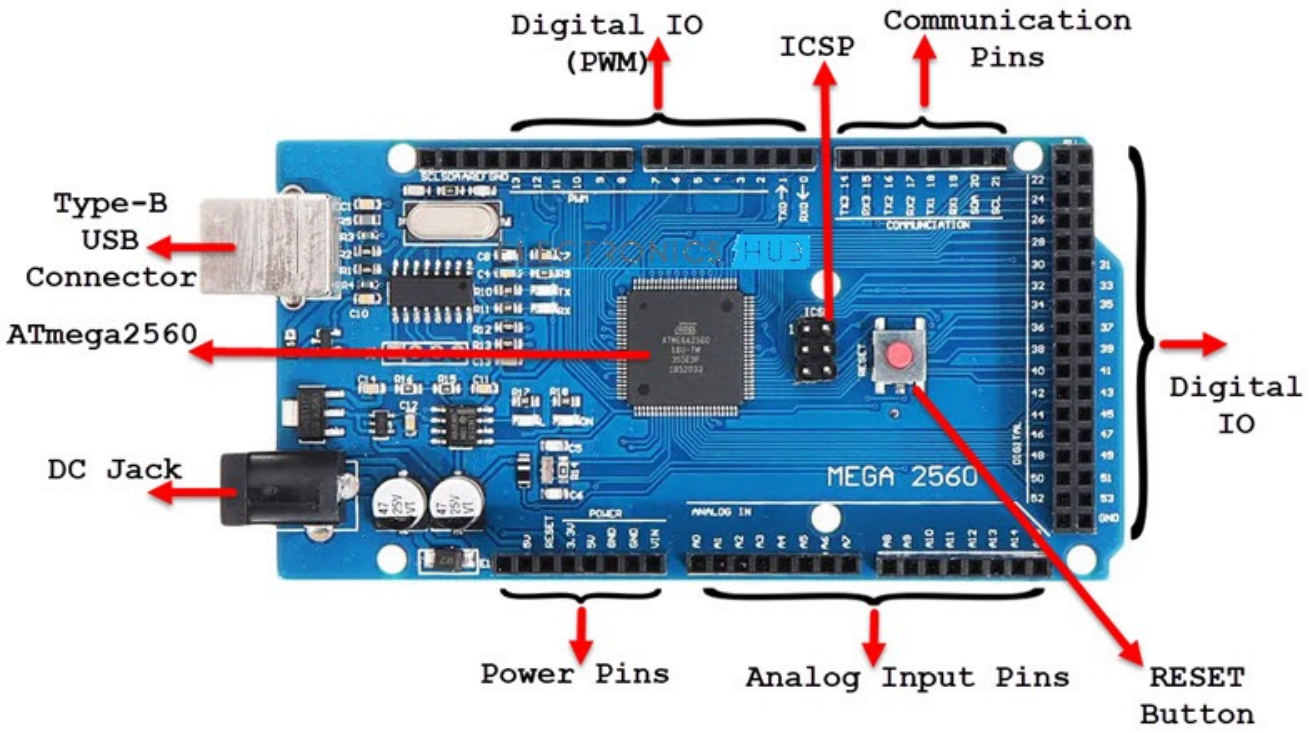
Overview of Arduino Development Board



Overview of Arduino Development Board: MEGA2560

- Based on ATmega2560 Microcontroller
- An 8-bit AVR Architecture based MCU from ATMEL.
- Available in a 100-pin Quad Flat Package

- Designed and developed to provide more number of IO lines (both Digital and Analog), more flash memory and more RAM when compared to UNO.



Overview of Arduino Development Kit

ELEGOO Mega 2560 Basic Starter Kit Tutorial

https://www.elegoo.com/blogs/arduino-projects/elegoo-mega-2560-basic-starter-kit-tutorial

Getting Started – Before Hands-on Activities -Optional

- Check out: https://github.com/muratkuzlu/ODU_BLAST_2024

Instructions:

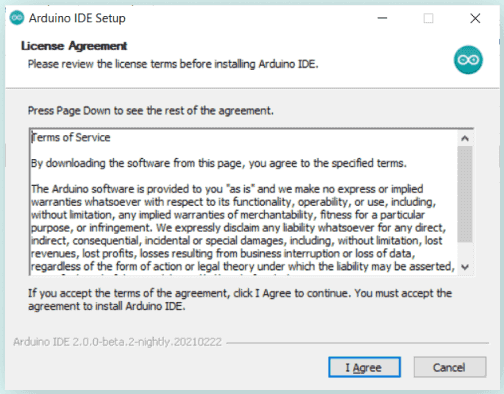
1. **Download & Install the Arduino environment (IDE)**
2. **Launch the Arduino IDE**
3. **Install Libraries**
4. **Open Serial Monitor**
5. **Connect the Board via USB Cable**
6. **Select your board**
7. **Select your serial port**
8. **Basic Arduino Program Structure**
9. **Verify a Sketch with the Arduino IDE**
10. **Upload a Sketch with the Arduino IDE**

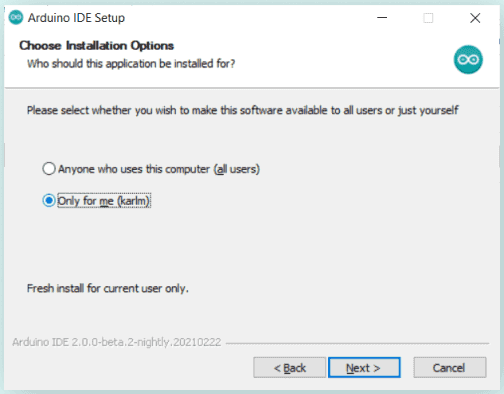
Setting Up the Arduino IDE - Optional

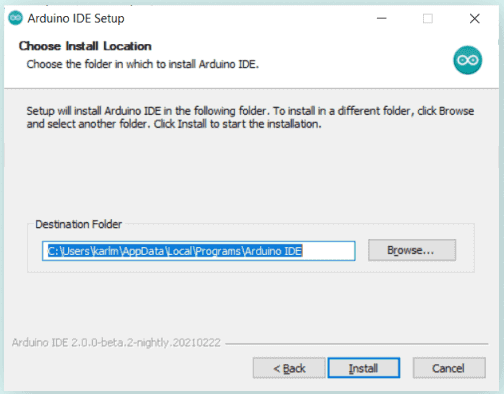
Download and Installing IDE

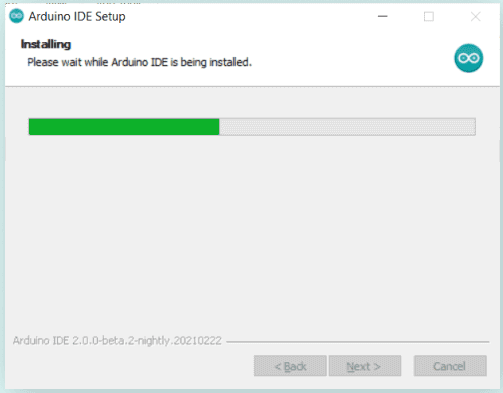
- The Arduino Integrated Development Environment (IDE) is the software side of the Arduino platform.
- **Go to https://www.arduino.cc/en/software and find below page.**

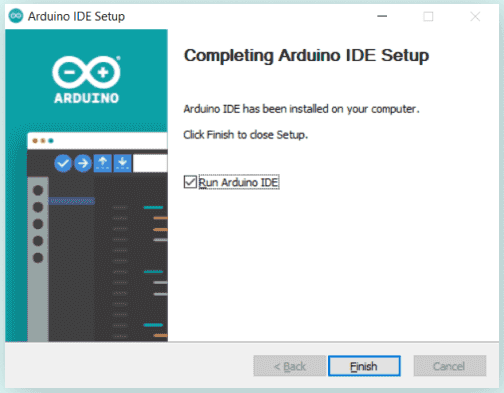
Installing Arduino (Windows)











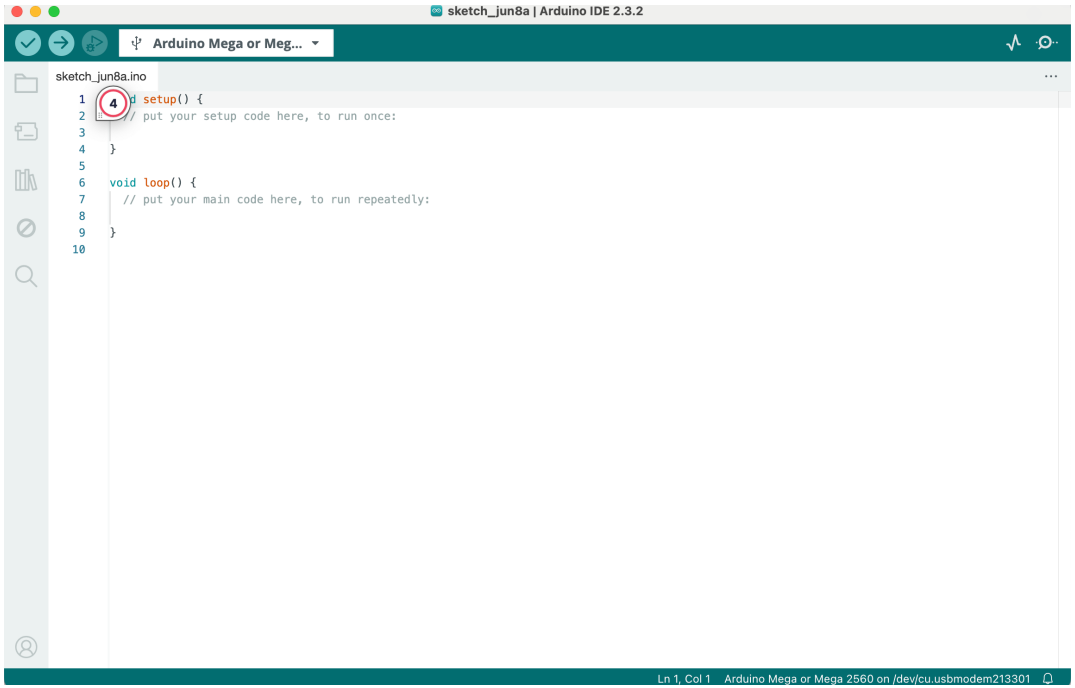
[13] https://docs.arduino.cc/software/ide-v2/tutorials/getting-started/ide-v2-downloading-and-installing/

Installing Arduino (macOS)

[14] https://docs.arduino.cc/software/ide-v2/tutorials/getting-started/ide-v2-downloading-and-installing/

Launch the Arduino IDE

Double-click to enter the desired development environment



Install Libraries and Open Serial Monitor

What are Libraries?

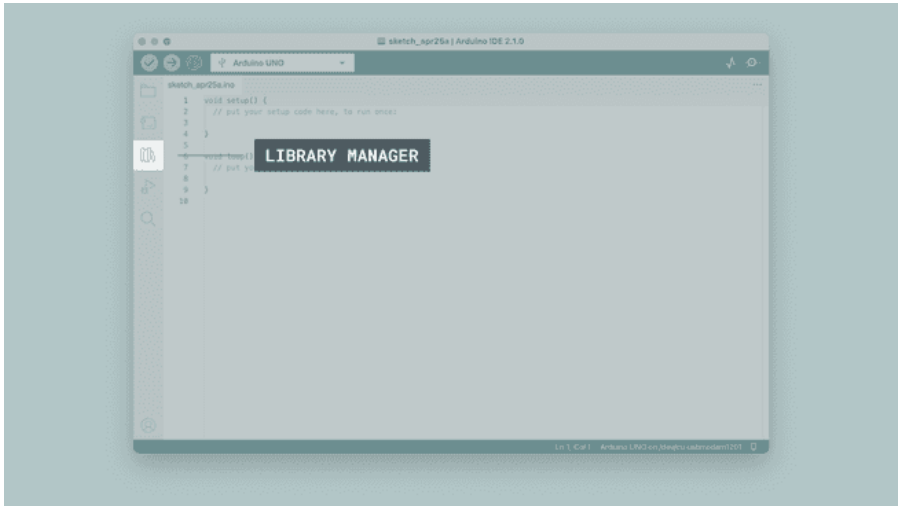
Libraries are a collection of code that makes it easy for you to connect to a sensor, display, module, etc. For example, the built-in LiquidCrystal library makes it easy to talk to character LCD displays. There are hundreds of additional libraries available on the Internet for download. The built-in libraries and some of these additional libraries are listed in the reference. To use the additional libraries, you will need to install them.

Arduino Serial Monitor (Windows, Mac, Linux)

The Arduino Integrated Development Environment (IDE) is the software side of the Arduino platform. And, because using a terminal is such a big part of working with Arduinos and other microcontrollers, they decided to include a serial terminal with the software. Within the Arduino environment, this is called the Serial Monitor.

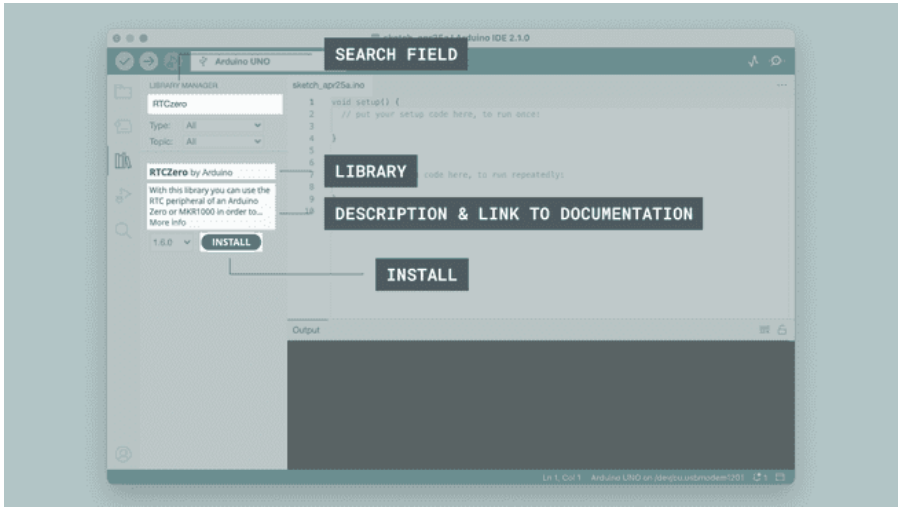
Install Libraries

1. Open the Arduino IDE 2.
2. With the editor open, let's take a look at the left column. Here, we can see a couple of icons. Let's click the on the "library" icon.



3. A list will now appear of all available libraries, where we can also search for the library we want to use. In this example, we are going to install the RTCZero library. Click on the "INSTALL" button to install the library.

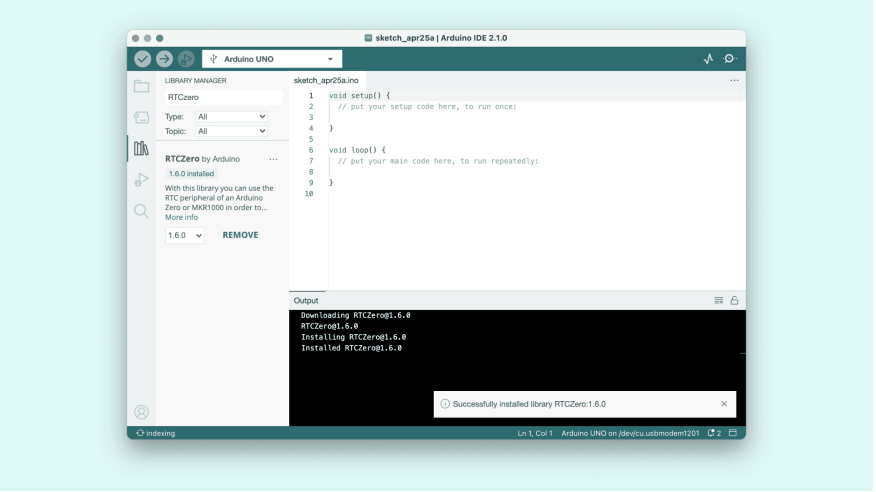
Please also try “DHT11” for the temperature and humidity sensor



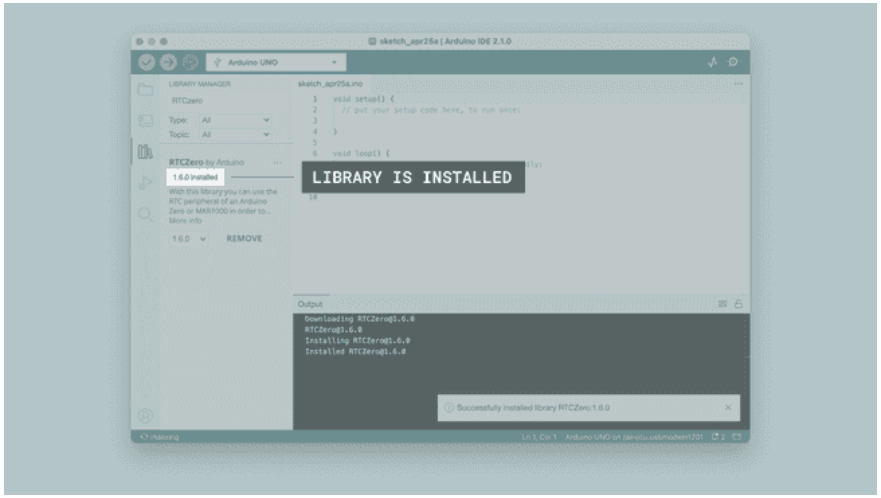
[17] https://docs.arduino.cc/software/ide-v2/tutorials/getting-started/ide-v2-downloading-and-installing/

Install Libraries

4. This process should not take too long, but allow up to a minute to install it.



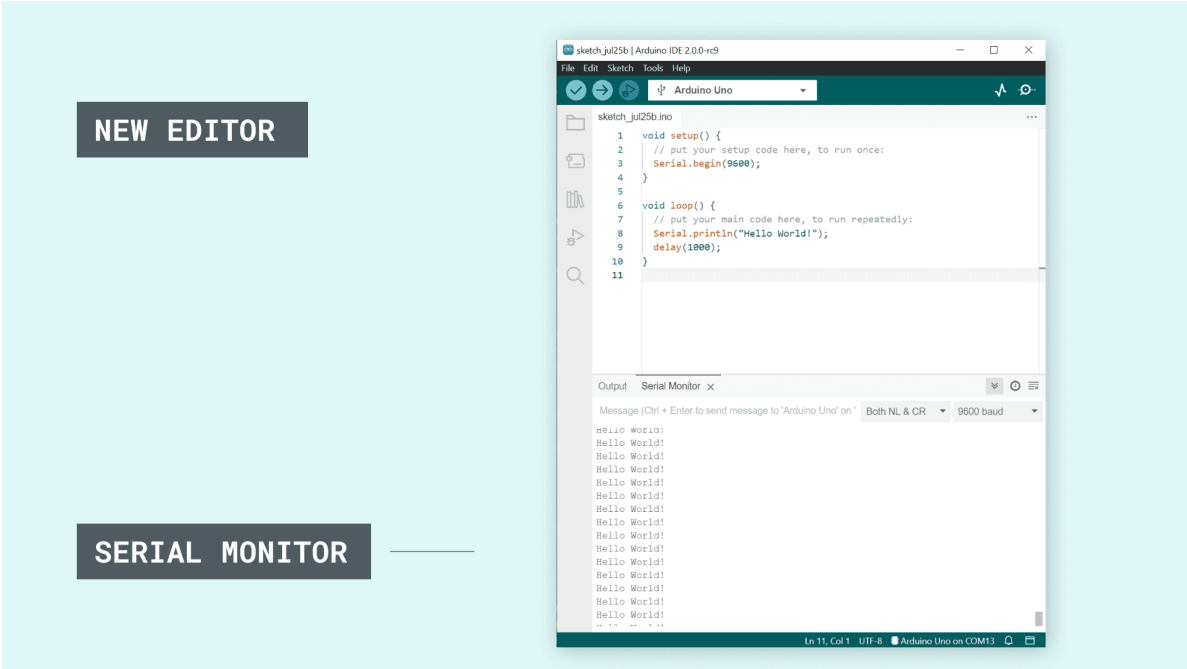
5. When it is finished, we can take a look at the library in the library manager column, where it should say **"INSTALLED"**.



 https://docs.arduino.cc/software/ide-v2/tutorials/getting-started/ide-v2-downloading-and-installing/

Open Serial Monitor

1. Open the Arduino IDE 2.
2. Notice how the Serial Monitor is located at the bottom of the editor

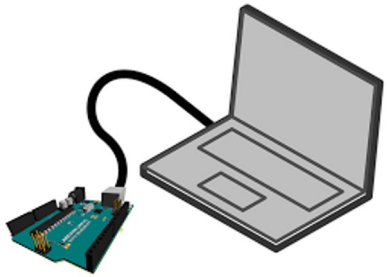


[19] https://docs.arduino.cc/software/ide-v2/tutorials/getting-started/ide-v2-downloading-and-installing/

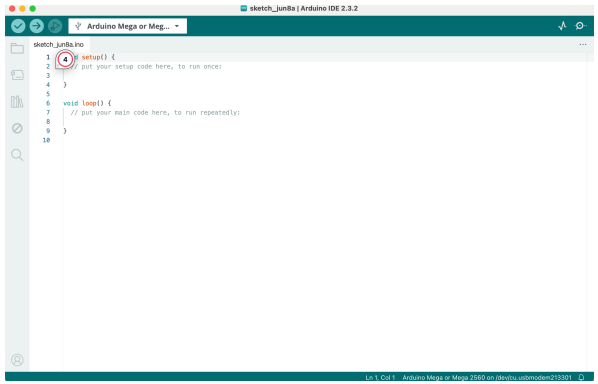
Connect the Board via USB Cable

- Items:
- Arduino Mega2560 board’
- USB cable (Type A to Type B)
- Computer with USB port(s)

- Connect the Arduino Mega2560 to Your Computer

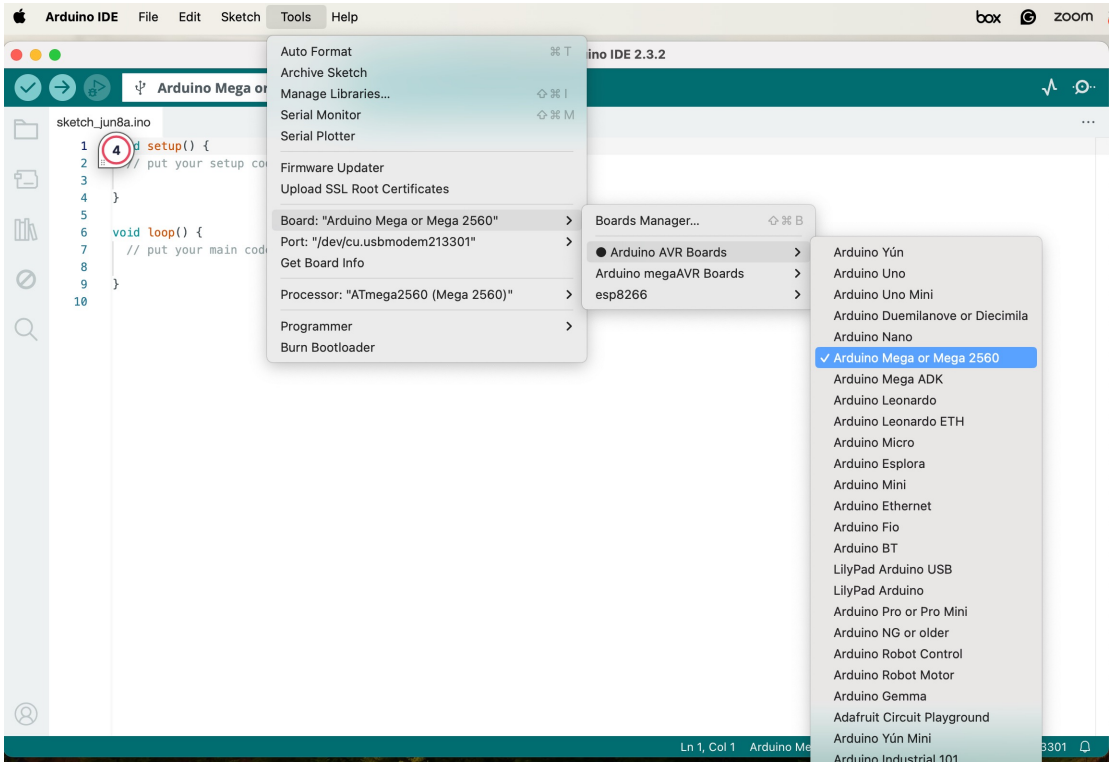


- Launch Arduino IDE



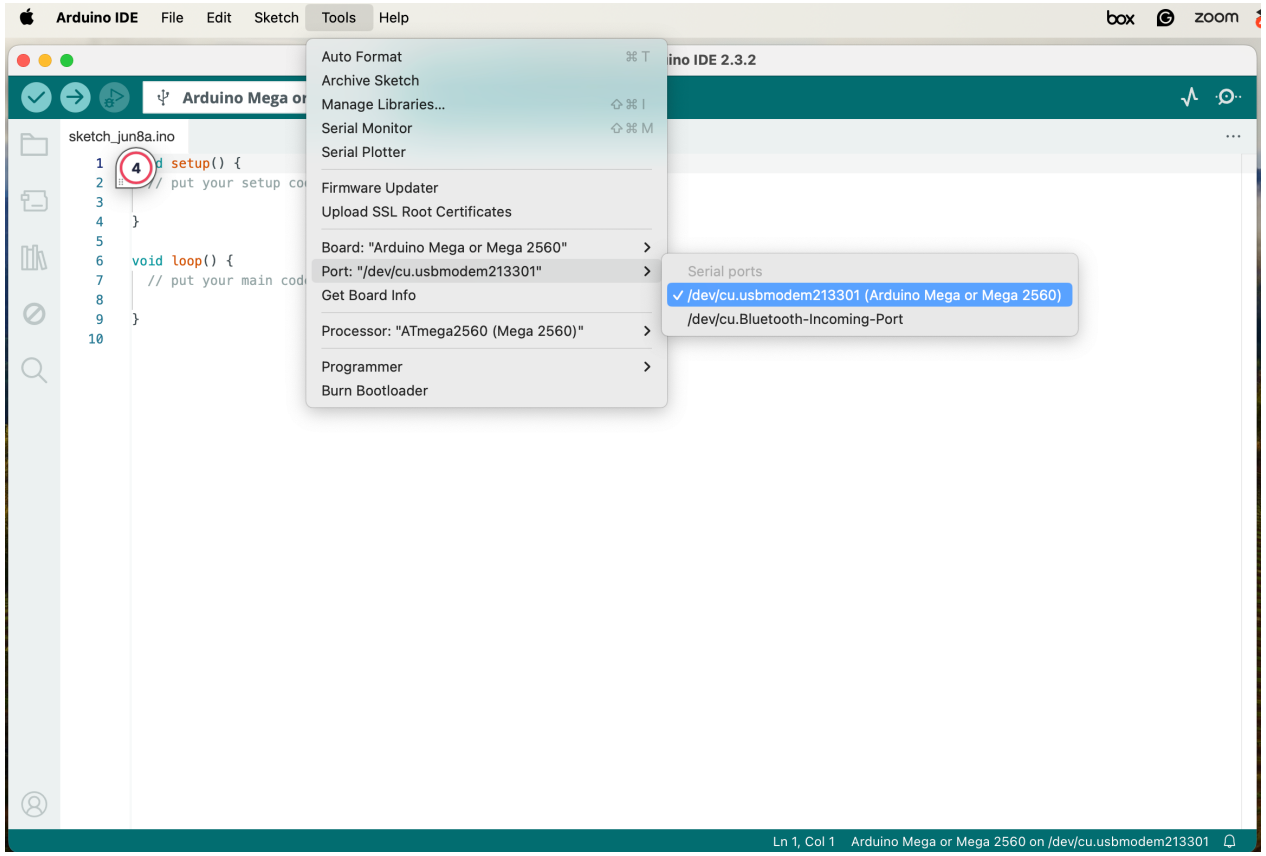
Select the Arduino Board

- In the Arduino IDE, go to the Tools menu.
- Select Board, and Arduino AVR Boards, then Arduino Mega or Mega 2560

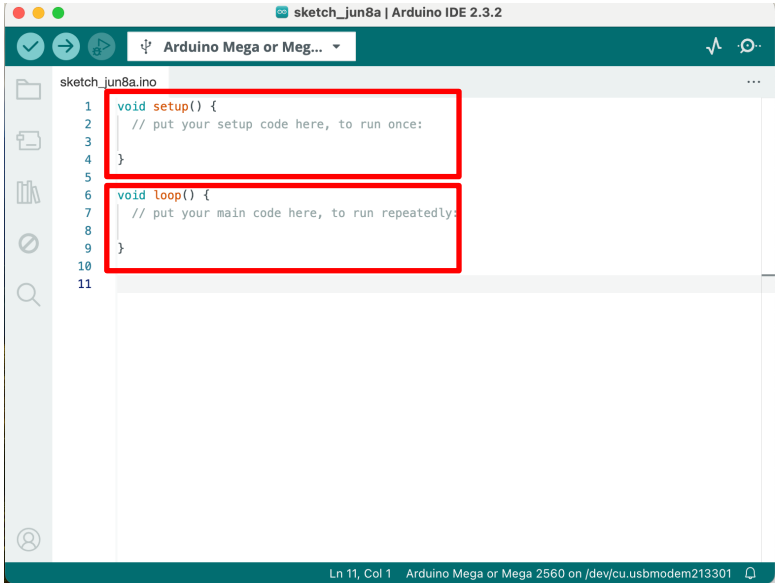


Select the Port

- Still in the Tools menu, select Port.Choose the port that corresponds to your Arduino Mega2560.
- It should be labeled something like COM3 on Windows, /dev/tty.usbmodem on macOS, or /dev/ttyUSB on Linux.



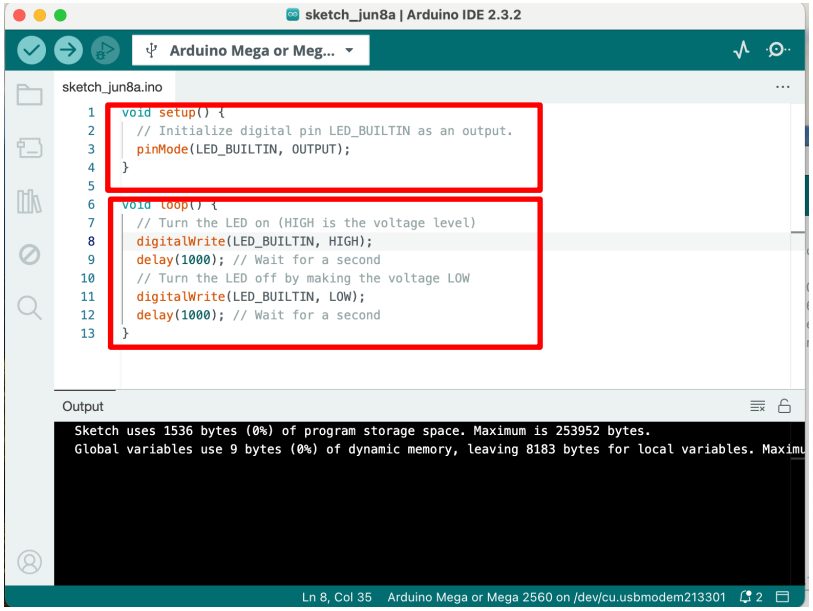
Basic Arduino Program Structure



- **Programming Language: C/C++**

- **void setup()**: Runs once when the program starts, used for initialization.
- **void loop()**: Runs continuously after setup().

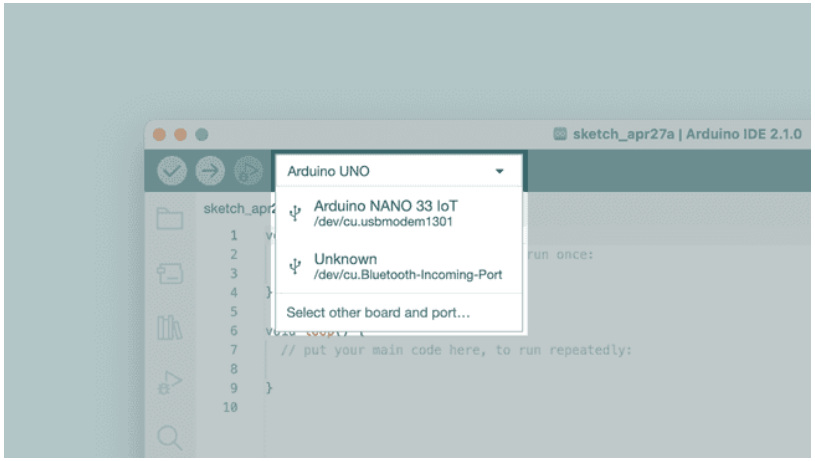
- Please note that
- Sketch: A program you write to run on an Arduino board



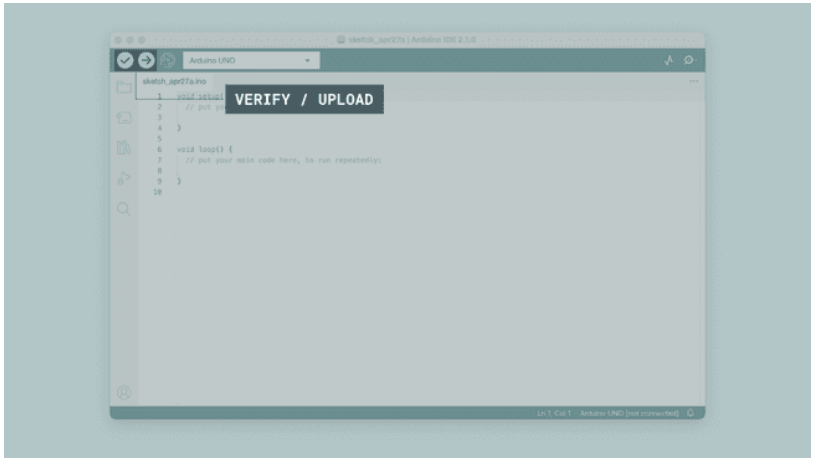
Verify/Upload a Sketch with the Arduino IDE

1. Open the Arduino IDE 2.

2. Make sure your board connects to the computer and select it from the drop down menu. You'll know that there is a connection to the board when the board name appears in **bold**.



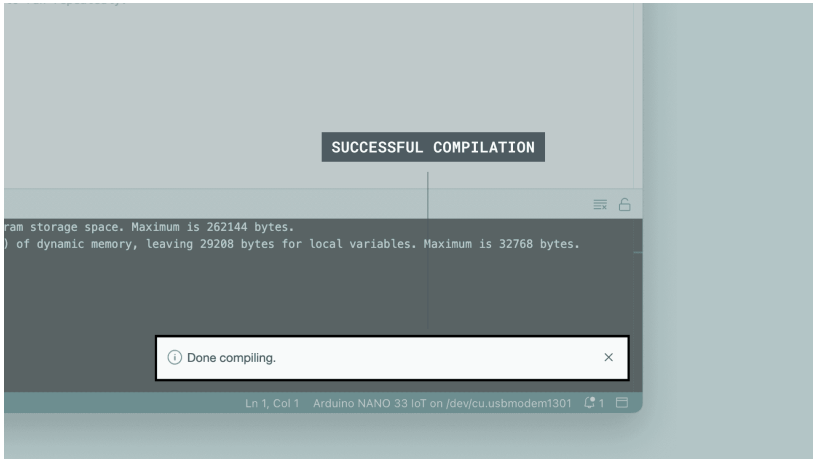
3. With the editor open, let's take a look at the toolbar at the top. At the very left, there is a **checkmark** and an **arrow pointing right**. The checkmark is used to **verify**, and the arrow is used to **upload.**



[24] https://docs.arduino.cc/software/ide-v2/tutorials/getting-started/ide-v2-downloading-and-installing/

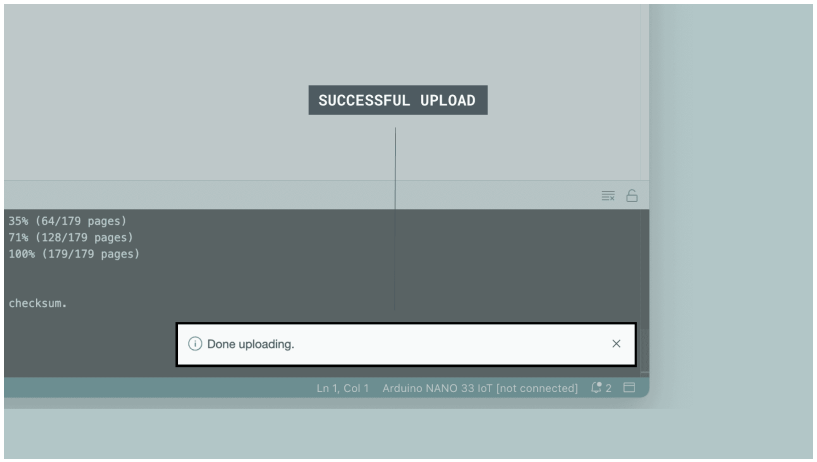
Verify/Upload a Sketch with the Arduino IDE

4. Click on the verify tool (checkmark). Since we are verifying an empty sketch, we can be sure it is going to compile. After a few seconds, we can see the result of the action in the console (black box in the bottom).



5. With the board selected, we are good to go! Click on the **upload** button, and it will start uploading the sketch to the board.

6. When it is finished, a notification pops up in the bottom right of your IDE window. Of course, sometimes there are some complications when uploading, and these errors will be listed here as well.



Introduction to Arduino and Hands-On Activities

Topics of this workshop

- **Getting started**
  (setup and programming of hardware)

- **Reading and Writing**
  (physical computing: sensors and actuators)

- **Monitoring and controlling**
  (sensors, actuator)

Getting started

The **IDE** (**I**ntegrated **D**evelopment **E**nvironment) allows you to **program** your board, i.e., “make it do something new”

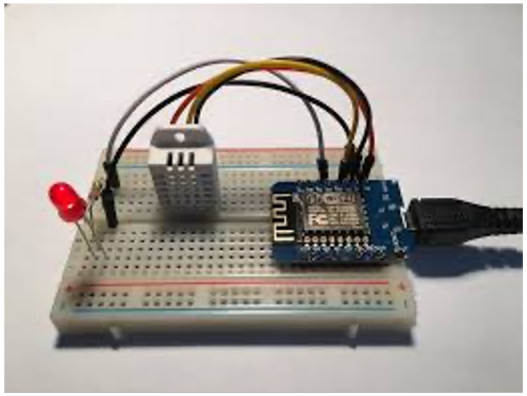
You **edit** a program on your computer, then **upload** it to your board where it’s stored in the program memory (flash) and **executed** in RAM

Please Note: Once it has been programmed, your board can run on its own, without another computer

Reading and Writing

Hardware has an **interface to the real world**

GPIO (**G**eneral **P**urpose **I**nput/**O**utput) pin (an **input or output** connected to something)



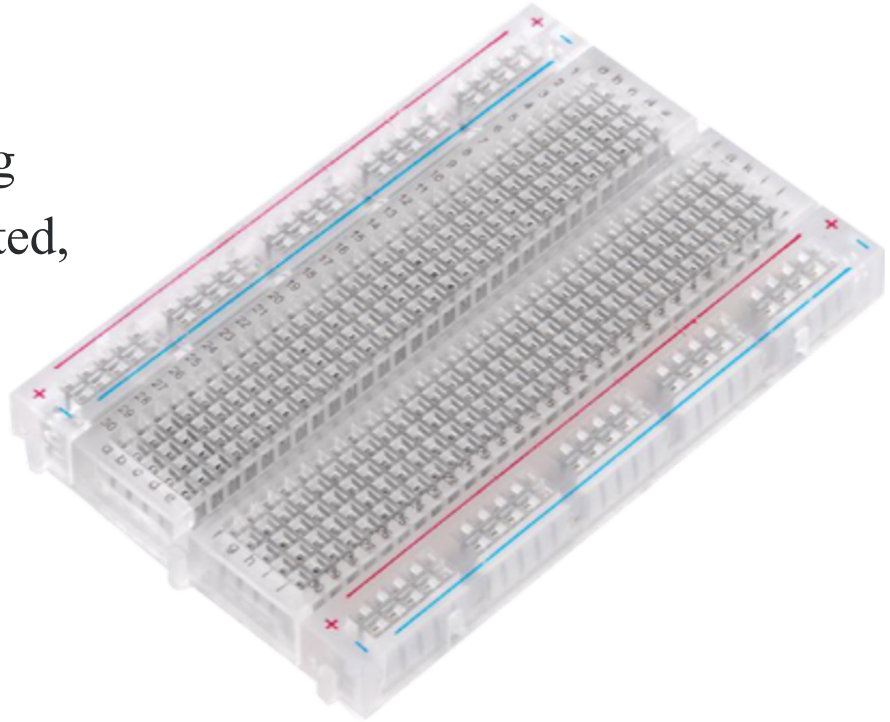
Measure: **read** sensor value from **input** pin
Manipulate: **write** actuator value to **output** pin

Inputs and outputs can be
Digital: Vale is either HIGH or LOW (e.g., LED ON/OFF)
Analog: Values ranges, usually 0-255 (e.g., LED brightness)

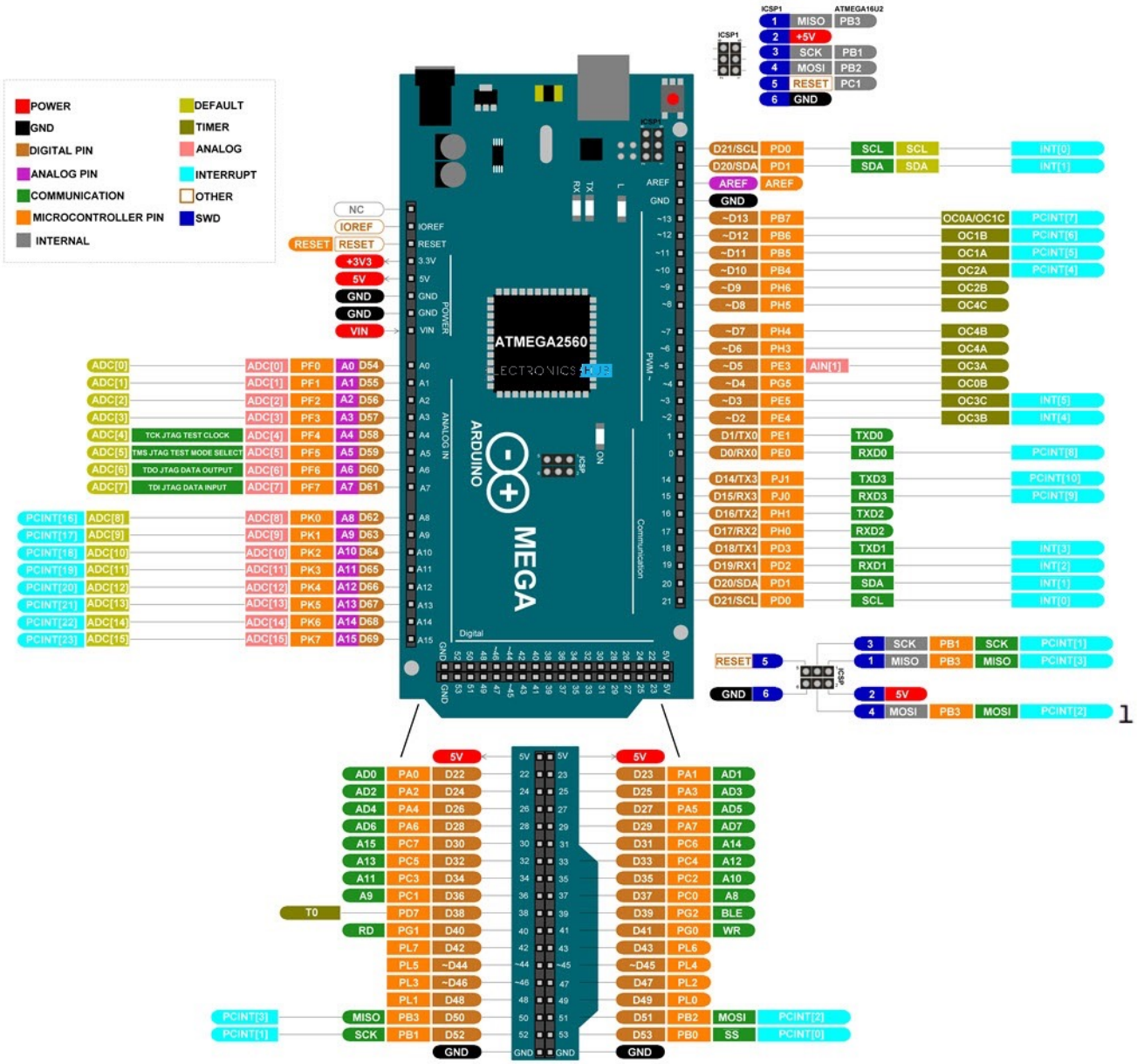
Prototyping Circuits Solderless Breadboard

- One of the most useful tools in an engineer or Maker's toolkit.

- The three most important things:
  - A breadboard is easier than soldering
  - A lot of those little holes are connected, or not connected, which ones?
  - Sometimes breadboards break



MEGA2560 Pinout

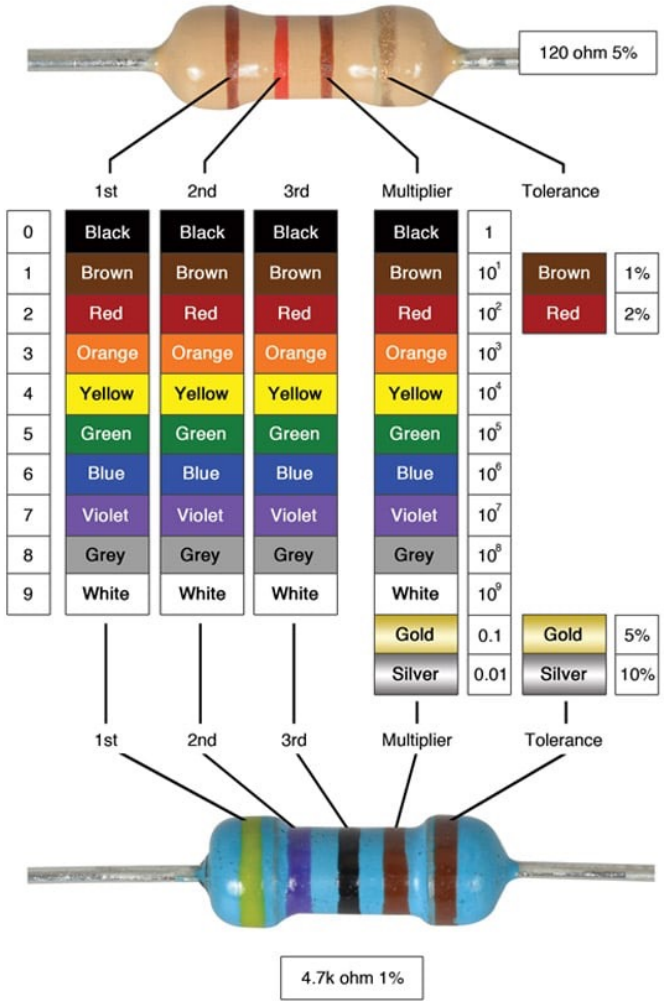


The Resistor

Resistors are the **workhorse of electronics**

Resistance is **measured in Ω** (Ohm) and adds up in series; a resistors orientation doesn't matter

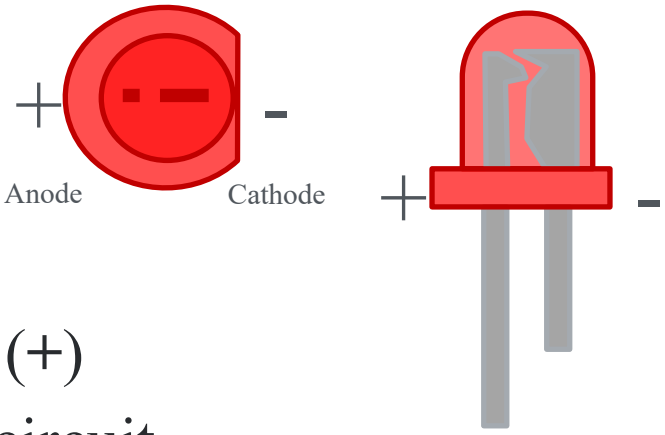
A resistors Ω value is **color-coded** right on it



https://www.jameco.com/Jameco/workshop/CircuitNotes/circuit-notes-resistors.html

The LED

The **LED** (**L**ight **E**mitting **D**iode)
is a simple, digital **actuator**



LEDs have a **short leg** (**-**) and a **long leg** (+)
and it matters how they are oriented in a circuit

To prevent damage, LEDs are used together with a 1KΩ **resistor** (or anything from 300Ω to 2KΩ)

The Buzzer

The piezo buzzer is an electronic devices commonly used t
sound.

There are two main types of buzzers that can be used with
buzzers and passive buzzers.

- An active buzzer has a built-in oscillating source, which means it can generate sound by simply applying a DC voltage.
- A passive buzzer requires an external oscillating signal to produce sound.

Connect the positive leg (longer leg) of the buzzer to the Arduino pin and the negative leg to the ground.

Hands-On Activities

Hands on Activities and Source Code

Copy&Paste or Download examples, from GitHub:
https://https://github.com/muratkuzlu/ODU_BLAST_2024

Focus on **end-to-end results, not details**

Hands-on Activity - I

Arduino MEGA2560

Blinking the Arduino built-in LED

We will introduce how to blink the built-in LED.

Hardware

- MEGA2560 x 1
- USB cable x 1
- PC x 1

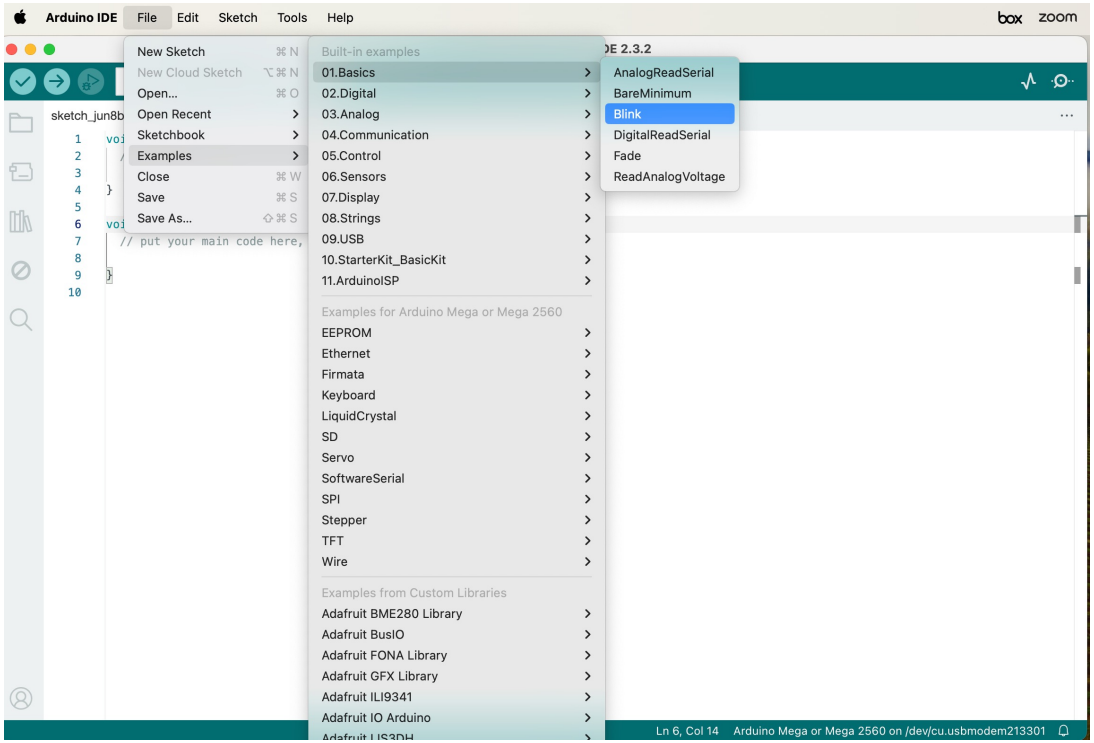
Software

- Arduino IDE

Blinking the Arduino built-in LED

Set up

- Connect the board to your computer via USB port
- Launch Arduino IDE
- **Go to File > Examples > Basic > Blink**



Blinking the Arduino built-in LED

Code

HIGH = digital 1 (5V) means LED is **on**,
LOW = digital 0 (0V) means LED is **off**

➔ Copy the following code to the IDE

```
// the setup function runs once when you press reset or power the board
void setup() {
// initialize digital pin LED_BUILTIN as an output.
pinMode(LED_BUILTIN, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
digitalWrite(LED_BUILTIN, HIGH); // turn the LED on (HIGH is the voltage level)
delay(1000); // wait for a second
digitalWrite(LED_BUILTIN, LOW); // turn the LED off by making the voltage LOW
delay(1000); // wait for a second
}
```

Initialize variables. Runs once

Used to actively control the Arduino board. Run repeatedly

➔ Upload

Digital input/output with Arduino – Remember!

pinMode(pin, mode)->
pinMode(LED_BUILTIN, OUTPUT);

Sets pin to either INPUT or OUTPUT

digitalWrite(pin, value)->
digitalWrite(LED_BUILTIN, LOW); // turn the LED off by making

Writes HIGH or LOW to a pin

---

pinMode(pin, mode)->
pinMode(LED_BUILTIN, INPUT);

Sets pin to either INPUT or OUTPUT

digitalRead(pin)->
digitalRead(LED_BUILTIN)->

Reads a pin

Hands-on Activity - II

Arduino MEGA2560

Blinking an LED with Arduino

We will introduce how to blink an external LED.

Hardware

- MEGA2560 x 1
- USB cable x 1
- PC x 1
- LED x 1
- 220 ohm ohm resistor x 1
- Wiring

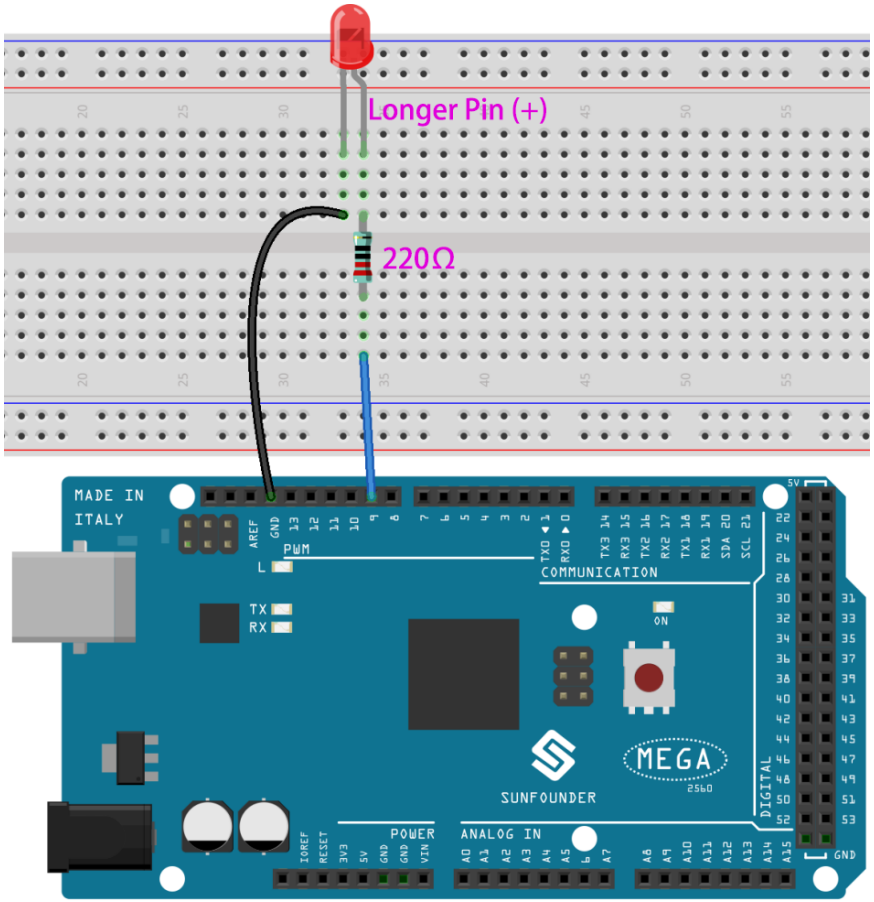
Software

- Arduino IDE

Blinking an LED with Arduino

Set up

- Connect one end of the 220ohm resistor to pin 9 of the Mega 2560 and the other end to the anode (the long pin) of the LED, and the cathode (the short pin) of the LED to GND.
- Connect the long leg of the LED (the positive leg, called the anode) to the other end of the resistor (220 ohm).
- Connect the short leg of the LED (the negative leg, called the cathode) to the GND of the Mega 2560 .



Blinking an LED with Arduino

Code

➔ Copy the following code to the IDE

```
/*********************************************/
const int ledPin = 9;//the number of the LED pin
/*********************************************/
void setup()
{
  pinMode(ledPin,OUTPUT);//initialize the digital pin as an output
}
/*********************************************/
//the loop routine runs over and over again forever
void loop()
{
  digitalWrite(ledPin,HIGH);//turn the LED on
  delay(500);              //wait for half a second
  digitalWrite(ledPin,LOW); //turn the LED off
  delay(500);              //wait for half a second
}
/*********************************************/
```

Specifying what pin is going to be used

Initialize variables. Runs once

Used to actively control the Arduino board. Run repeatedly

➔ Upload

Hands-on Activity - III

Arduino MEGA2560

RGB LED with Arduino

We will introduce how to control an RGB LED

Hardware

- MEGA2560 x 1
- USB cable x 1
- PC x 1
- RGB LED x 1
- 220 ohm ohm resistor x 3
- Wiring

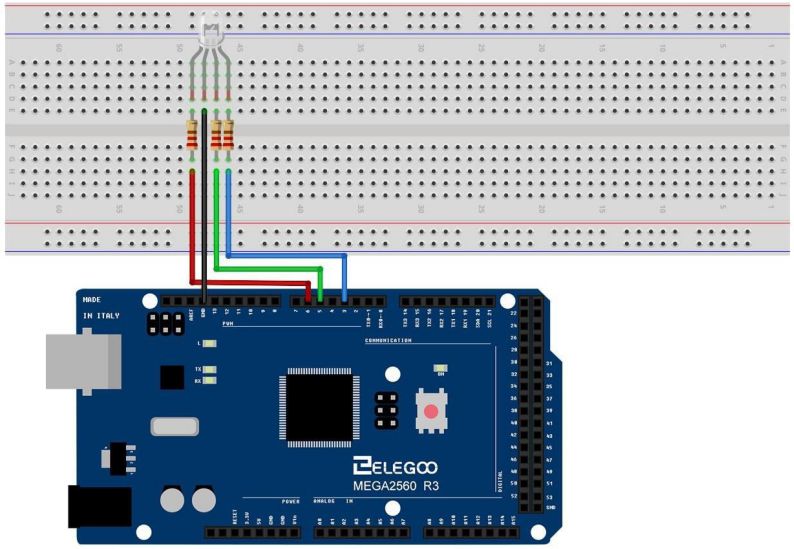
Software

- Arduino IDE

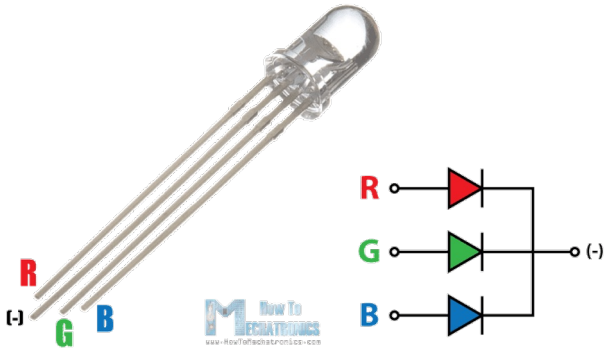
RGB LED with Arduino

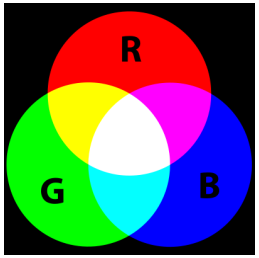
Set up

- Connect the longest lead(common cathode) of the RGB to ground.
- Connect each of the other three leads to a 220Ω resistor
- These three positive leads of the LEDs (one red, one green and one blue) are connected to MEGA 2560 output pins using these resistors.



Every separate pin for Green or Blue or Red color is called Anode. You will always connect "+" to it.





RGB LED with Arduino - I

Code

➔ Copy the following code to the IDE

```
// Define Pins
#define BLUE 3
#define GREEN 5
#define RED 6

void setup()
{
pinMode(RED, OUTPUT);
pinMode(GREEN, OUTPUT);
pinMode(BLUE, OUTPUT);
digitalWrite(RED, HIGH);
digitalWrite(GREEN, LOW);
digitalWrite(BLUE, LOW);
}

// define variables
int redValue;
int greenValue;
int blueValue;
```

Specifying what pin is going to be used

Initialize variables. Runs once

Define Variables

RGB LED with Arduino - II

Code

➔ Copy the following code to the IDE

```
// main loop
void loop()
{
#define delayTime 10 // fading time between colors

redValue = 255; // choose a value between 1 and 255 to change the color.
greenValue = 0;
blueValue = 0;


for(int i = 0; i < 255; i += 1) // fades out red bring green full when i=255
{
redValue -= 1;
greenValue += 1;
analogWrite(RED, redValue);
analogWrite(GREEN, greenValue);
delay(delayTime);
}

redValue = 0;
greenValue = 255;
blueValue = 0;
```

Used to actively control the Arduino board. Run repeatedly

➔ Upload

RGB LED with Arduino - III

Code

➔ Copy the following code to the IDE

```
for(int i = 0; i < 255; i += 1) // fades out green bring blue full when i=255
{
greenValue -= 1;
blueValue += 1;
analogWrite(GREEN, greenValue);
analogWrite(BLUE, blueValue);
delay(delayTime);
}

redValue = 0;
greenValue = 0;
blueValue = 255;

for(int i = 0; i < 255; i += 1) // fades out blue bring red full when i=255
{
// The following code has been rearranged to match the other two similar sections
blueValue -= 1;
redValue += 1;
analogWrite(BLUE, blueValue);
analogWrite(RED, redValue);
delay(delayTime);
}
}
```

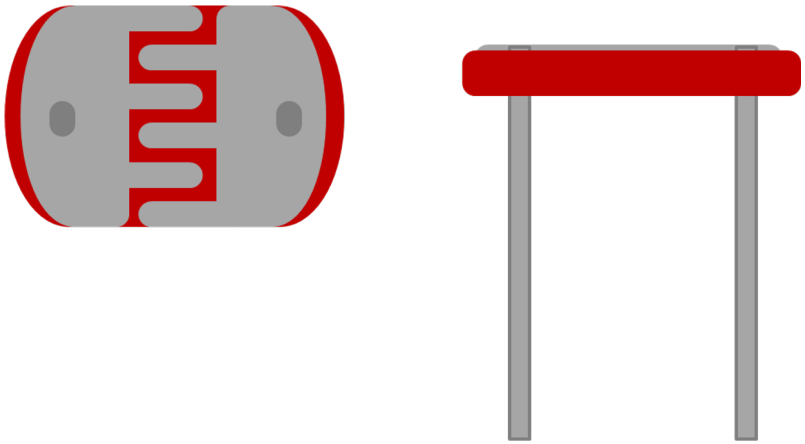
➔ Upload

Hands-on Activity - IV

Arduino MEGA2560

Photoresistor (LDR)

A photoresistor or **LDR** (**l**ight **d**ependent **r**esistor) is a resistor whose resistance depends on light intensity. That is, as more light **shines** on the LDR, its resistance **decreases**. Conversely less light or more darkness on the LDR surface causes its resistance to increase.



An LDR can be used as a simple, **analog sensor**

The orientation of an LDR does not matter

LDR with Arduino

We will introduce how to how to measure light intensity using a photo resistor.

Hardware

- MEGA2560 x 1
- LDR / photoresistor
- 10k ohm resistor
- Breadboard
- USB cable
- Connecting Wires

Software

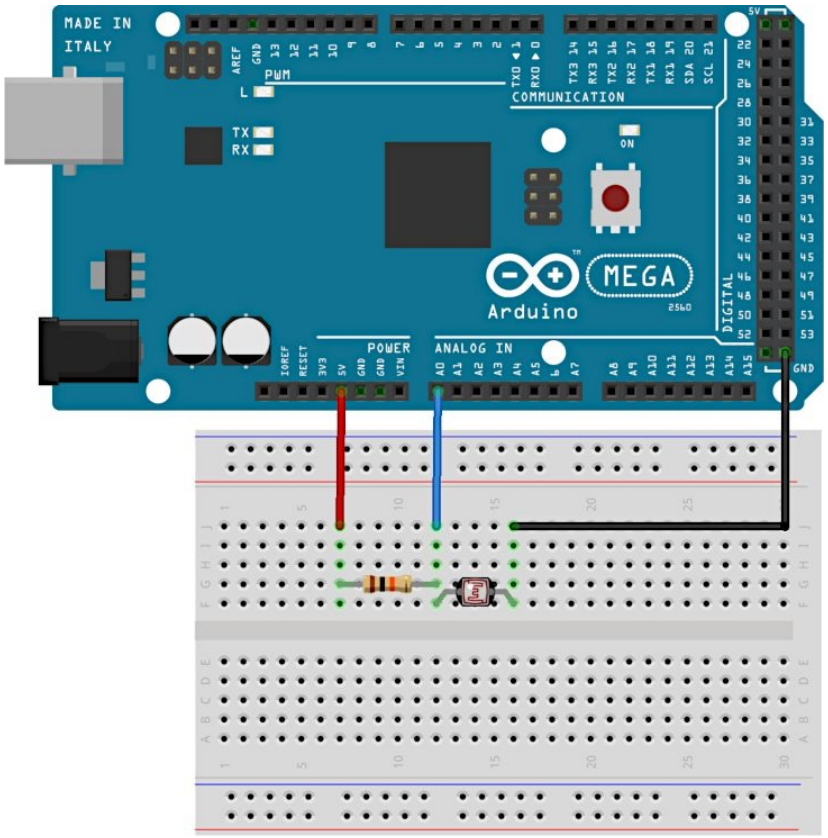
- Arduino IDE

LDR with Arduino

Set up

The LDR output is actually analog in nature, so it gets connected to the A0 pin of the Mega2560.

Connect the LDR between the A0 pin and GND. Additionally, connect the 10k resistor between A0 and 5V.



Note: this setup is a *voltage-divider*, *as* the total voltage is divided between LDR and resistor to keep 0V < **A0** < 5V

LDR with Arduino

Code

➔ Copy the following code to the IDE

```
void setup()
{
Serial.begin(9600);
delay(1000);
}

void loop() {

int sensorValue = analogRead(A0); // read the input on analog pin 0

float voltage = sensorValue * (5.0 / 1023.0); // Convert the analog reading (which goes from 0 - 1023) to a voltage
(0 - 5V)
delay(1000);

Serial.println(voltage); // print out the value you read
}
```

Serial Port Setting

➔ Upload

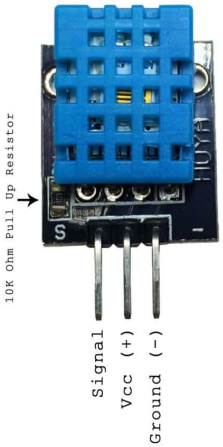
Hands-on Activity - V

Arduino MEGA2560

Temperature & Humidity Sensor DHT11

The DHT11 sensor can detect temperature (C and F) & humidity.

The DHT11 detects water vapor by measuring the electrical resistance between two electrodes. The humidity sensing component is a moisture holding substrate with electrodes applied to the surface.
It has everything it requires built into it, so it will work very well with the Arduino boards. This sensor is used in conjunction with the DHT11 Library.



DHT11 with Arduino

We will learn how to set up the DHT11 Humidity and Temperature sensor on your NodeMCU. And learn about how the Humidity sensor works, and how to check output readings from the Serial monitor.

Hardware

- MEGA2560
- DHT11 Humidity and Temperature sensor
- Breadboard
- Jumper Wires (Optional)
- USB Cable

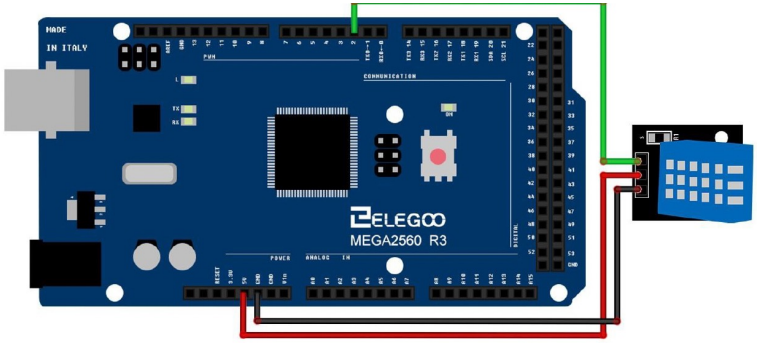
Software

- Arduino IDE

DHT11 with Arduino

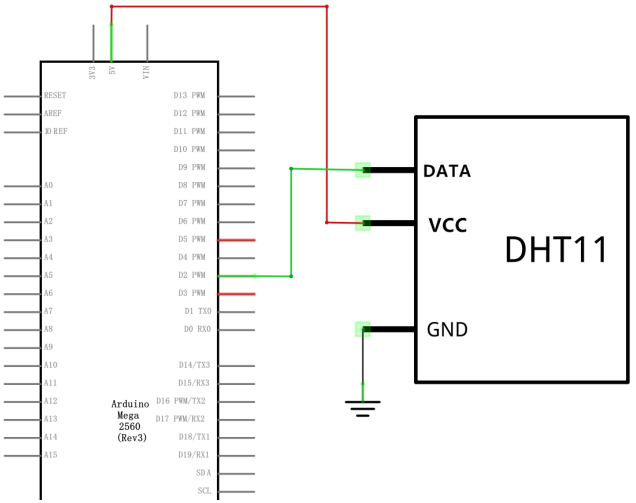
Set up

Wiring the **DHT11** to the Arduino board is really easy, but the connections are different depending on which type you have either 3-pins or 4-pins



The **wiring connections** are made as follows:

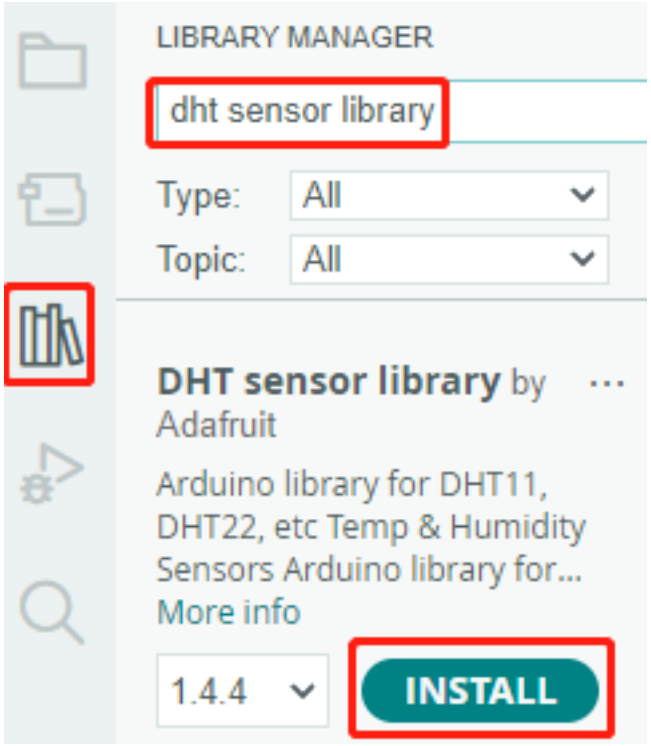
- **Pin 1** of the DHT11 (S, data) goes into Digital Pin (2) of the Arduino board.
- **Pin 2** of the DHT11 (Vcc) goes into **+5V** of the Arduino board.
- **Pin 3** of the DHT11 goes into Ground Pin (**GND**) of the Arduino board.



DHT11 with Arduino

Set up

Add DHT11 sensor library!



Digital input with Arduino

Code

➔ Copy the following code to the IDE

Add the DHT.h library

Define Variables

Initialize variables. Runs once

Used to actively control the Arduino board. Run repeatedly

```
#include "DHT.h"

#define DHTPIN 2 // Set the pin connected to the DHT11 data pin
#define DHTTYPE DHT11 // DHT 11
DHT dht(DHTPIN, DHTTYPE);

void setup() {
Serial.begin(9600);
Serial.println("DHT11 test!");
dht.begin();
}

void loop() {
// Wait a few seconds between measurements.
delay(2000);

float humidity = dht.readHumidity();
// Read temperature as Celsius (the default)
float temperature = dht.readTemperature();

// Print the humidity and temperature
Serial.print("Humidity: ");
Serial.print(humidity);
Serial.print(" %\t");
Serial.print("Temperature: ");
Serial.print(temperature);
Serial.println(" *C");
}
```

➔ Upload

Hands-on Activity - VI

Arduino MEGA2560

Buzzer with Arduino

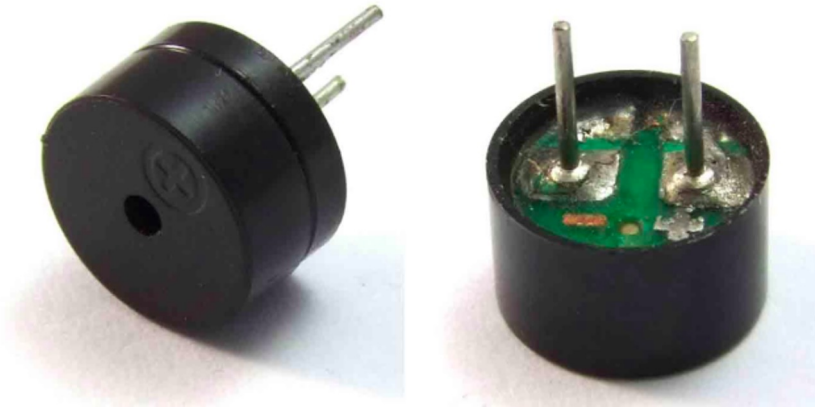
We will introduce how to use an Active Buzzer with the MEGA2560. You will generate sounds.

Hardware

- MEGA2560
- USB cable
- PC
- Active buzzer
- Jumper Wires (Optional)

Software

- Arduino IDE

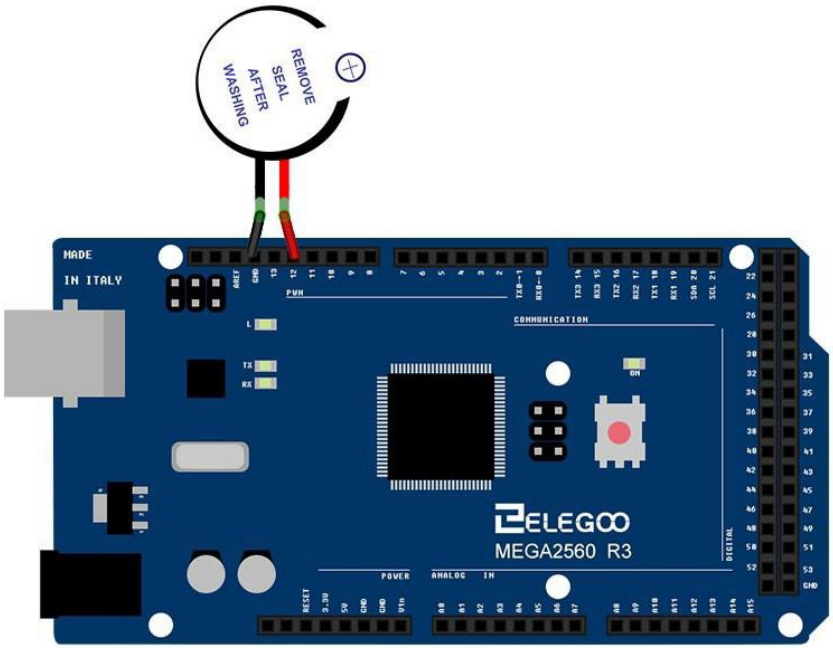


Buzzer with Arduino

Set up

- Connect the positive lead to the pin 12, and the negative lead to the GND.

That's it!



Buzzer with Arduino

Code

➔ Copy the following code to the IDE

```
int buzzer = 12;//the pin of the active buzzer
void setup()
{
pinMode(buzzer, OUTPUT); //initialize the buzzer pin as an output
}
void loop()
{
int sound_duration = 500;
for (int i = 0; i < 20; i++)
{
//use the if function to gradually shorten the interval of the sound
if (i < 5)
{
sound_duration = 500;
} else if (i < 10)
{
sound_duration = 300;
} else if (i < 20)
{
sound_duration = 100;
}
//activate the active buzzer
digitalWrite(buzzer, HIGH);
delay(sound_duration);//wait for sound_duration ms
digitalWrite(buzzer, LOW); //deactivate the active buzzer
delay(sound_duration);//wait for sound_duration ms
}
//activate the active buzzer
digitalWrite(buzzer, HIGH);
delay(5000);//keep playing sound for 5 seconds.
}
```

➔ Upload

Hands-on Activity - VII

Arduino MEGA2560

4x4 Keypad with Arduino

We will introduce how to use a 4x4 matrix keypad with the Arduino. We will then monitor the inputs of the keypad.

Hardware

- Mega2560 x 1
- Breadboard x 1
- Micro USB cable x 1
- PC x 1
- Membrane Switch Module Keypad
- Male to Male Wires x 8

Software

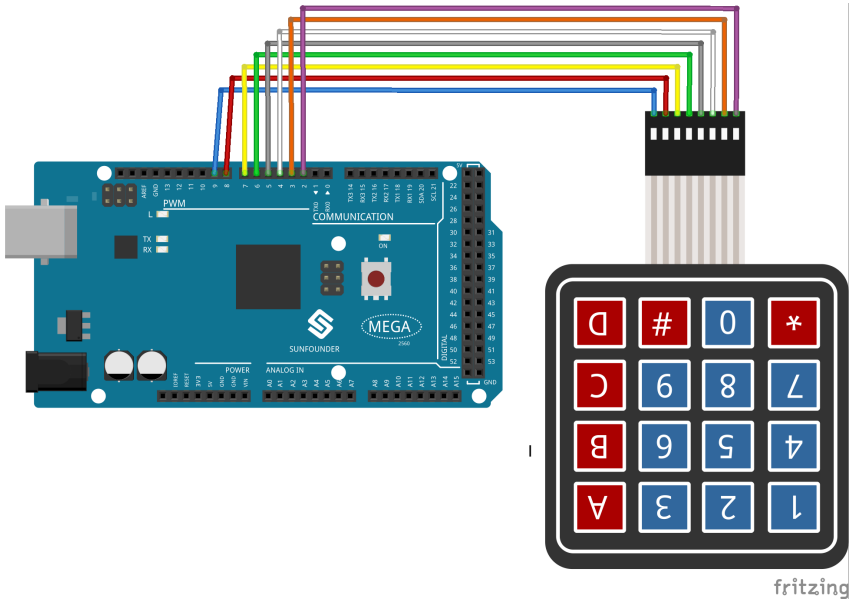
- Arduino IDE

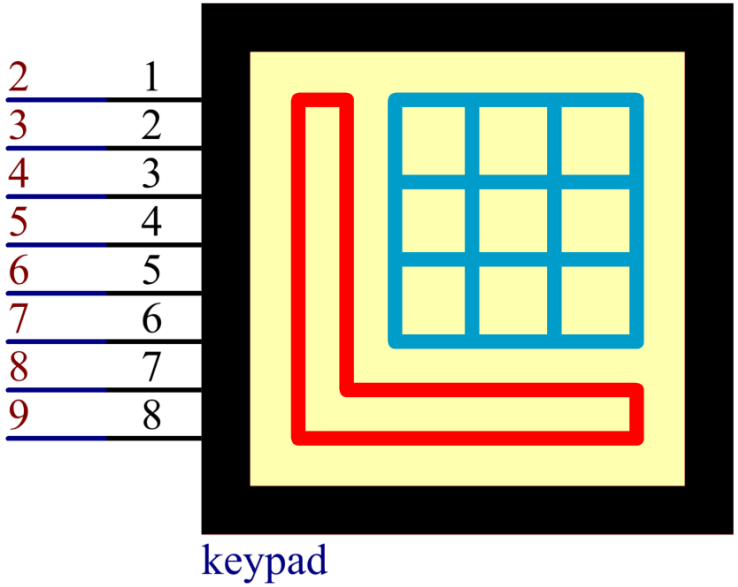
4x4 Keypad with Arduino

Set up

- Connect the pins 1~8 of Keypad to connect to the digital pins 2~9.

That's it!

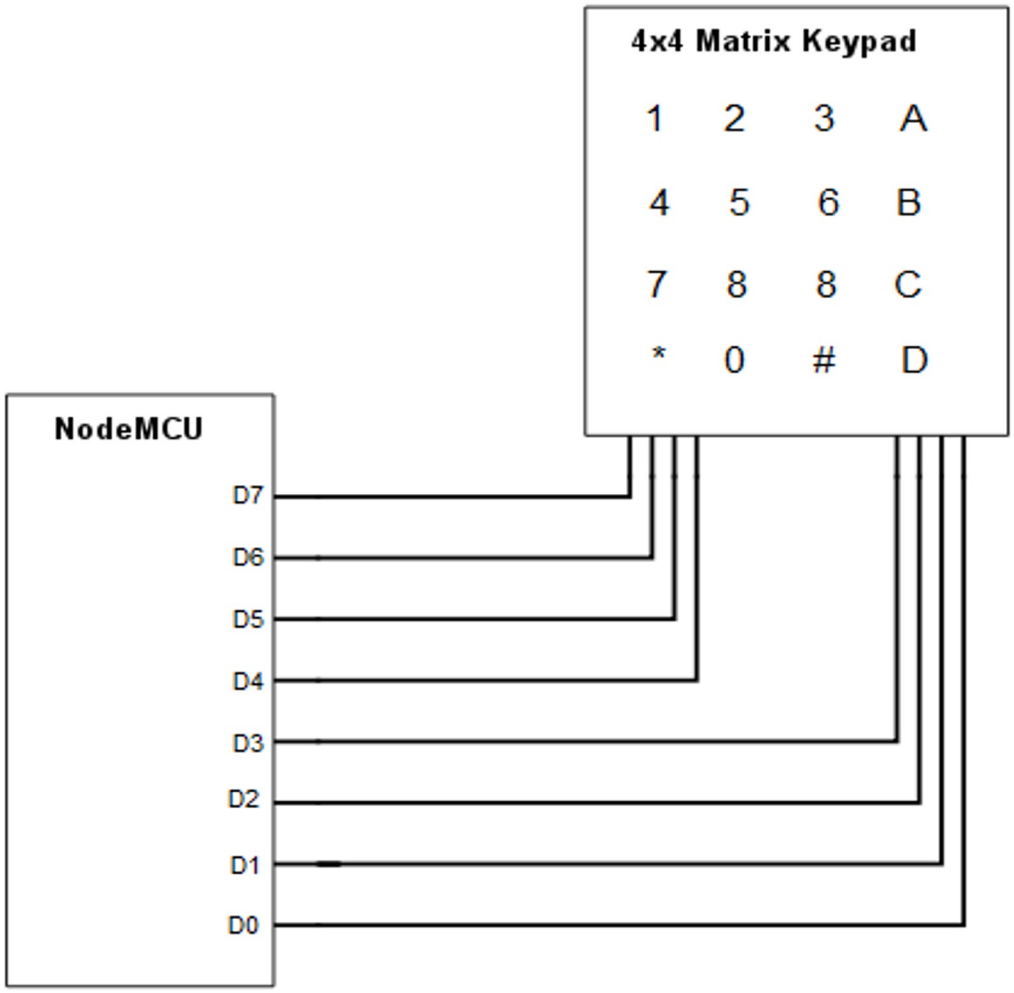




4x4 Keypad with Arduino

Set up

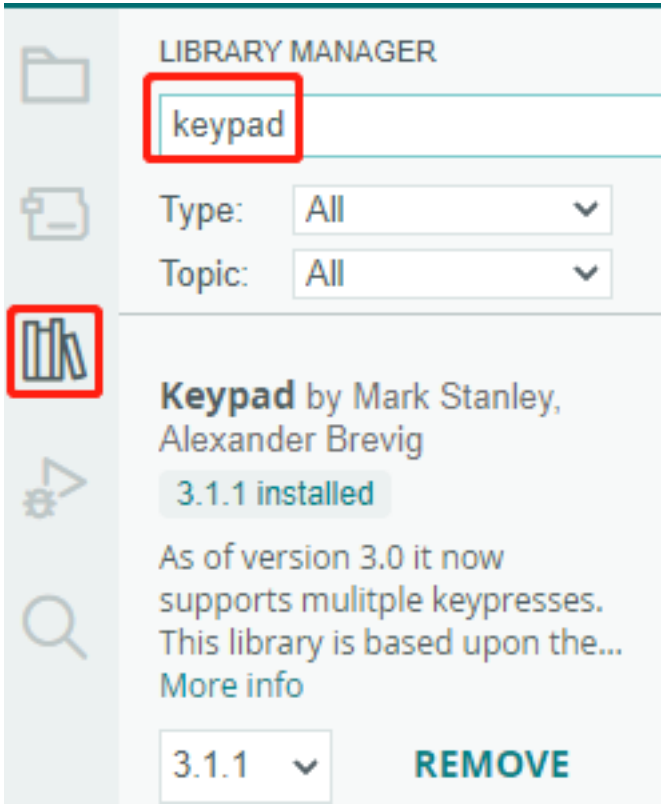
- With the screen of Membrane Switch Module facing you, start from left to right when wiring each of the eight male to male wires
- The first wire connects to pin (D7)
- The second wire connects to pin (D6)
- The third wire connects to pin (D5)
- The fourth wire connects to pin (D4)
- The fifth wire connects to pin (D3)
- The sixth wire connects to pin (D2)
- The seventh wire connects to pin (D1)
- The eighth wire connects to pin (D0)



4x4 Keypad with Arduino

Set up

Add keypad sensor library!



4x4 Keypad with Arduino

```
#include <Keypad.h>

const byte ROWS = 4; //four rows
const byte COLS = 4; //four columns
//define the cymbols on the buttons of the keypads
char hexaKeys[ROWS][COLS] =
{
{ '1','2','3','A' },
{ '4','5','6','B' },
{ '7','8','9','C' },
{ '*','0','#','D' }
};
byte rowPins[ROWS] = {2, 3, 4, 5}; //connect to the row pinouts of the keypad
byte colPins[COLS] = {6, 7, 8, 9}; //connect to the column pinouts of the keypad

//initialize an instance of class NewKeypad
Keypad customKeypad = Keypad( makeKeymap(hexaKeys), rowPins, colPins, ROWS, COLS);

void setup(){
Serial.begin(9600);
}
void loop(){
char customKey = customKeypad.getKey();
if (customKey){
Serial.println(customKey);
}
}
```

Passing the matrix of keys to a macro that will cast it to a char array

Initialize the Serial Connection

Obtain the key that is being pressed

Hands-on Activity - VIII

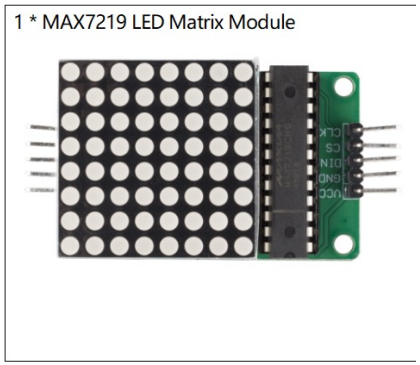
Arduino MEGA2560

LED Matrix Module with Arduino

We will introduce how to use a LED Matrix Module with the Arduino.

Hardware

- Mega2560 x 1
- Breadboard x 1
- Micro USB cable x 1
- PC x 1
- LED Matrix Module x 1
- Connecting Wires



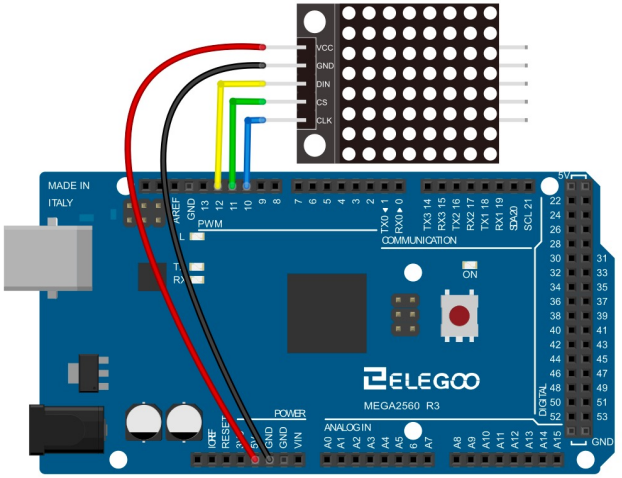
Software

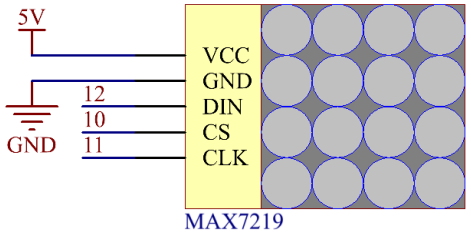
- Arduino IDE

LED Matrix Module with Arduino

Set up

- Connect the VCC pin of MAX7219 connected to 5V, GND to ground, DIN to digital pin 12, CS to digital pin 11, CLK to digital pin10.

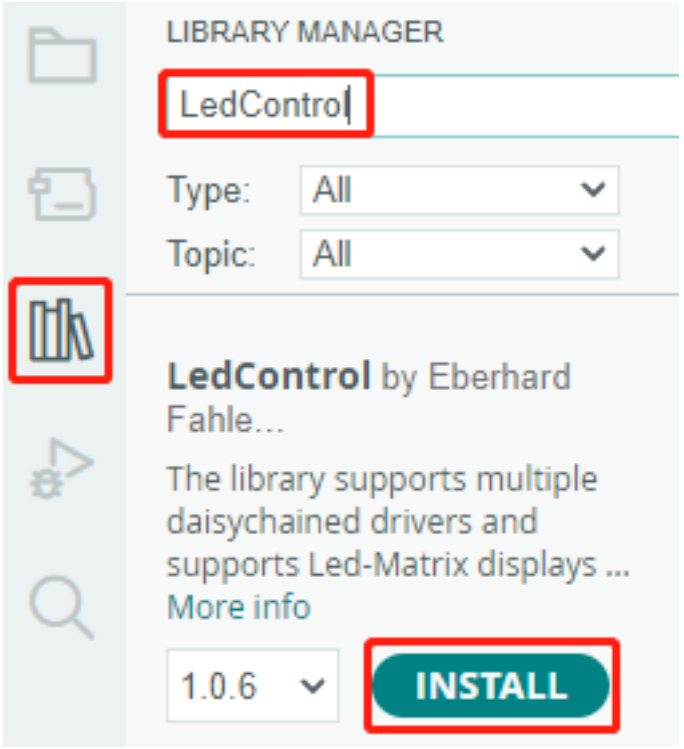




LED Matrix Module with Arduino

Set up

Add LED Matrix Module library!



LED Matrix Module with Arduino I

```
#include "LedControl.h"

LedControl lc=LedControl(12,10,11,1);

/* we always wait a bit between updates of the display */
unsigned long delaytime1=500;
unsigned long delaytime2=50;
void setup() {
/*
The MAX72XX is in power-saving mode on startup,
we have to do a wakeup call
*/
lc.shutdown(0,false);
/* Set the brightness to a medium values */
lc.setIntensity(0,8);
/* and clear the display */
lc.clearDisplay(0);
}
```

LED Matrix Module with Arduino II

```
void writeArduinoOnMatrix() {
/* here is the data for the characters */
byte a[5]={B01111110,B10001000,B10001000,B10001000,B01111110};
byte r[5]={B00010000,B00100000,B00100000,B00010000,B00111110};
byte d[5]={B11111110,B00010010,B00100010,B00100010,B00011100};
byte u[5]={B00111110,B00000100,B00000010,B00000010,B00111100};
byte i[5]={B00000000,B00000010,B10111110,B00100010,B00000000};
byte n[5]={B00011110,B00100000,B00100000,B00010000,B00111110};
byte o[5]={B00011100,B00100010,B00100010,B00100010,B00011100};

/* now display them one by one with a small delay */
lc.setRow(0,0,a[0]);
lc.setRow(0,1,a[1]);
lc.setRow(0,2,a[2]);
lc.setRow(0,3,a[3]);
lc.setRow(0,4,a[4]);
delay(delaytime1);
lc.setRow(0,0,r[0]);
lc.setRow(0,1,r[1]);
lc.setRow(0,2,r[2]);
lc.setRow(0,3,r[3]);
lc.setRow(0,4,r[4]);
delay(delaytime1);
lc.setRow(0,0,d[0]);
lc.setRow(0,1,d[1]);
lc.setRow(0,2,d[2]);
lc.setRow(0,3,d[3]);
lc.setRow(0,4,d[4]);
delay(delaytime1);
lc.setRow(0,0,u[0]);
lc.setRow(0,1,u[1]);
lc.setRow(0,2,u[2]);
lc.setRow(0,3,u[3]);
lc.setRow(0,4,u[4]);
delay(delaytime1);
lc.setRow(0,0,i[0]);
lc.setRow(0,1,i[1]);
lc.setRow(0,2,i[2]);
lc.setRow(0,3,i[3]);
lc.setRow(0,4,i[4]);
delay(delaytime1);
lc.setRow(0,0,n[0]);
lc.setRow(0,1,n[1]);
lc.setRow(0,2,n[2]);
lc.setRow(0,3,n[3]);
lc.setRow(0,4,n[4]);
delay(delaytime1);
lc.setRow(0,0,o[0]);
lc.setRow(0,1,o[1]);
lc.setRow(0,2,o[2]);
lc.setRow(0,3,o[3]);
lc.setRow(0,4,o[4]);
delay(delaytime1);
lc.setRow(0,0,0);
lc.setRow(0,1,0);
lc.setRow(0,2,0);
lc.setRow(0,3,0);
lc.setRow(0,4,0);
delay(delaytime1);
}
```

LED Matrix Module with Arduino III

```
void rows() {
for(int row=0;row<8;row++) {
delay(delaytime2);
lc.setRow(0,row,B10100000);
delay(delaytime2);
lc.setRow(0,row,(byte)0);
for(int i=0;i<row;i++) {
delay(delaytime2);
lc.setRow(0,row,B10100000);
delay(delaytime2);
lc.setRow(0,row,(byte)0);
}
}
}

void columns() {
for(int col=0;col<8;col++) {
delay(delaytime2);
lc.setColumn(0,col,B00100000);
delay(delaytime2);
lc.setColumn(0,col,(byte)0);
for(int i=0;i<col;i++) {
delay(delaytime2);
lc.setColumn(0,col,B00100000);
delay(delaytime2);
lc.setColumn(0,col,(byte)0);
}
}
}
```

LED Matrix Module with Arduino IV

```
/*
This function will light up every Led on the matrix.
The led will blink along with the row-number.
row number 4 (index==3) will blink 4 times etc.
*/
void single() {
for(int row=0;row<8;row++) {
for(int col=0;col<8;col++) {
delay(delaytime2);
lc.setLed(0,row,col,true);
delay(delaytime2);
for(int i=0;i<col;i++) {
lc.setLed(0,row,col,false);
delay(delaytime2);
lc.setLed(0,row,col,true);
delay(delaytime2);
}
}
}
}

void loop() {
writeArduinoOnMatrix();
rows();
columns();
single();
}
```

Hands-on Activity - IX

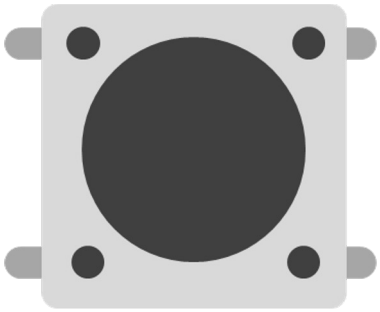
Arduino MEGA2560

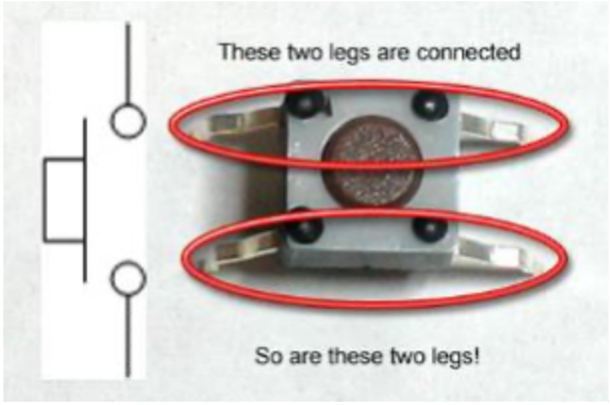
The switch

A switch is a simple, digital **sensor**

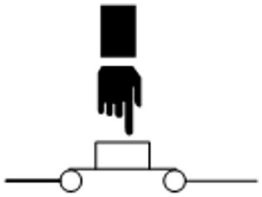
Switches come in different forms, but all of them in some way **open** or **close** a gap in a wire

The **pushbutton** switch has four legs for easier mounting, but only two of them are needed





Normally Open
When not pressed



Closed
When pressed

Switch with Arduino

We will introduce how to control the switch.

Hardware

- Mega2560 x 1
- Push Button x1
- LED x1
- 10 K ohm Resistor x1
- 220-ohm Resistor x1
- Bread Board x1
- PC x1

Software

- Arduino IDE

Switch with Arduino

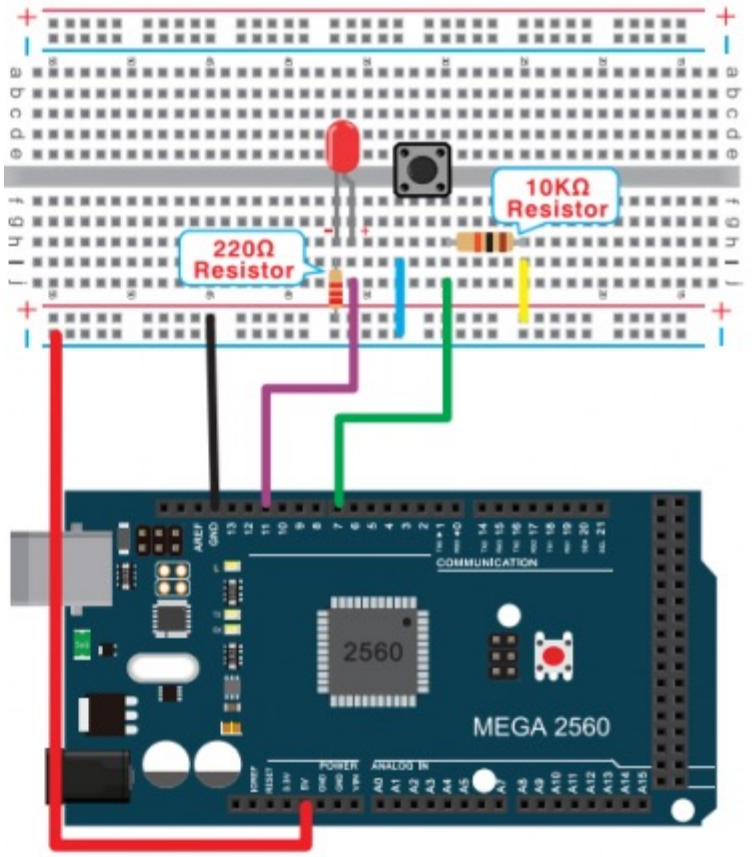
Set up

Push Button connections :

- The first pin goes from one leg of the pushbutton to Ground (**GND**).
- The **second pin** goes from the corresponding leg of the pushbutton through 10K Ohm resistor to the 5v supply pin.
- The **third pin** connects to a Digital I/O pin (here pin **7**) which reads the button's state.

LED connections:

- LED **Anode** is connected to Digital I/O pin (here pin **11**) and Cathode 220 Ohm resistor to ground (**GND)** pin.



https://wiki.keyestudio.com/052043_Super_Learning_Kit_for_Arduino#Project_6:_Button-controlled_LED

Switch with Arduino

Code

➔ Copy the following code to the IDE

```
int ledpin = 11; // initialize pin 11
int inpin = 7; // initialize pin 7
int val; // define val

void setup() {
  pinMode(ledpin, OUTPUT); // set LED pin as “output”
  pinMode(inpin, INPUT); // set button pin as “input”
}

void loop() {
  val = digitalRead(inpin); // read the level value of pin 7 and assign if to val
  if (val == LOW) // check if the button is pressed, if yes, turn on the LED
  {
    digitalWrite(ledpin, LOW);
  } else {
    digitalWrite(ledpin, HIGH);
  }
}
```

➔ Upload

More

• Google & Youtube - search for projects, solutions to occurring problems and data sheets for components

• https://www.adafruit.com/ - Learning materials, guides, example projects, forums, store

• https://github.com/ - Largest code host, lots of projects and sample code

https://docs.sunfounder.com/projects/ - Arduino examples

Dr. Murat Kuzlu

mkuzlu@odu.edu