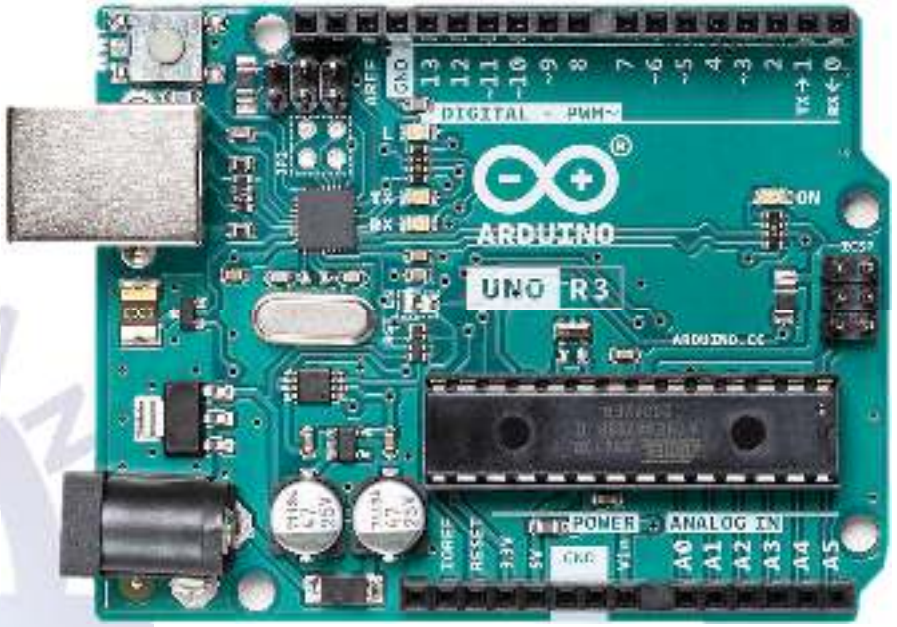
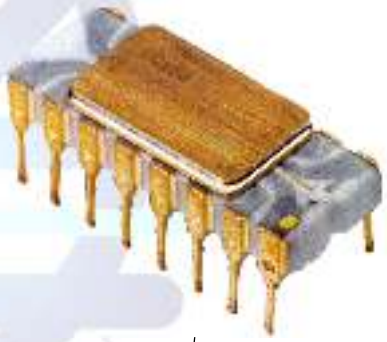
Arduino

A brief introduction to modern micro-controllers



Micro-Controllers

- History:
  - 1969: Four phase systems AL1.
  - 1971: Intel 4004
  - 2022: MARVELL 88MZ300 (SoC)

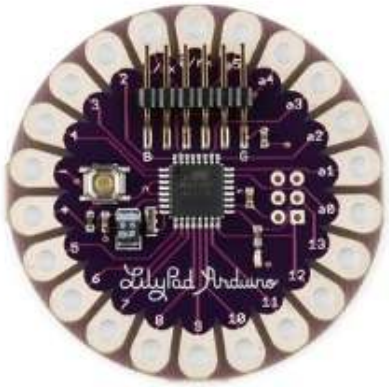


Why Arduino ?

- Microcontrollers are notorious for being difficult to program
- The goal of Arduino is to create an accessible way for software developers to enter the world of microcontroller programming
- Arduino is open source, both in its software and hardware
- Unlike most microcontroller interfaces, Arduino is cross-platform, so it can be run on Windows, Linux, and macOS
- Arduino can interact with other software on the computer like Flash or even web APIs

Types of Arduino Boards

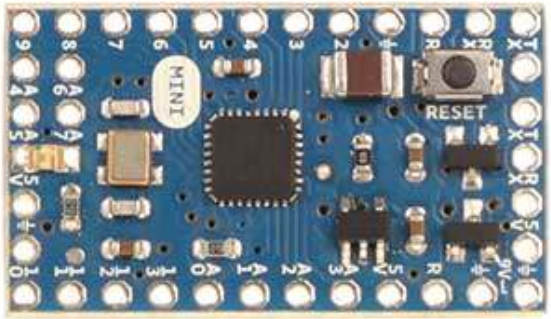
Arduino Nano



Arduino LilyPad

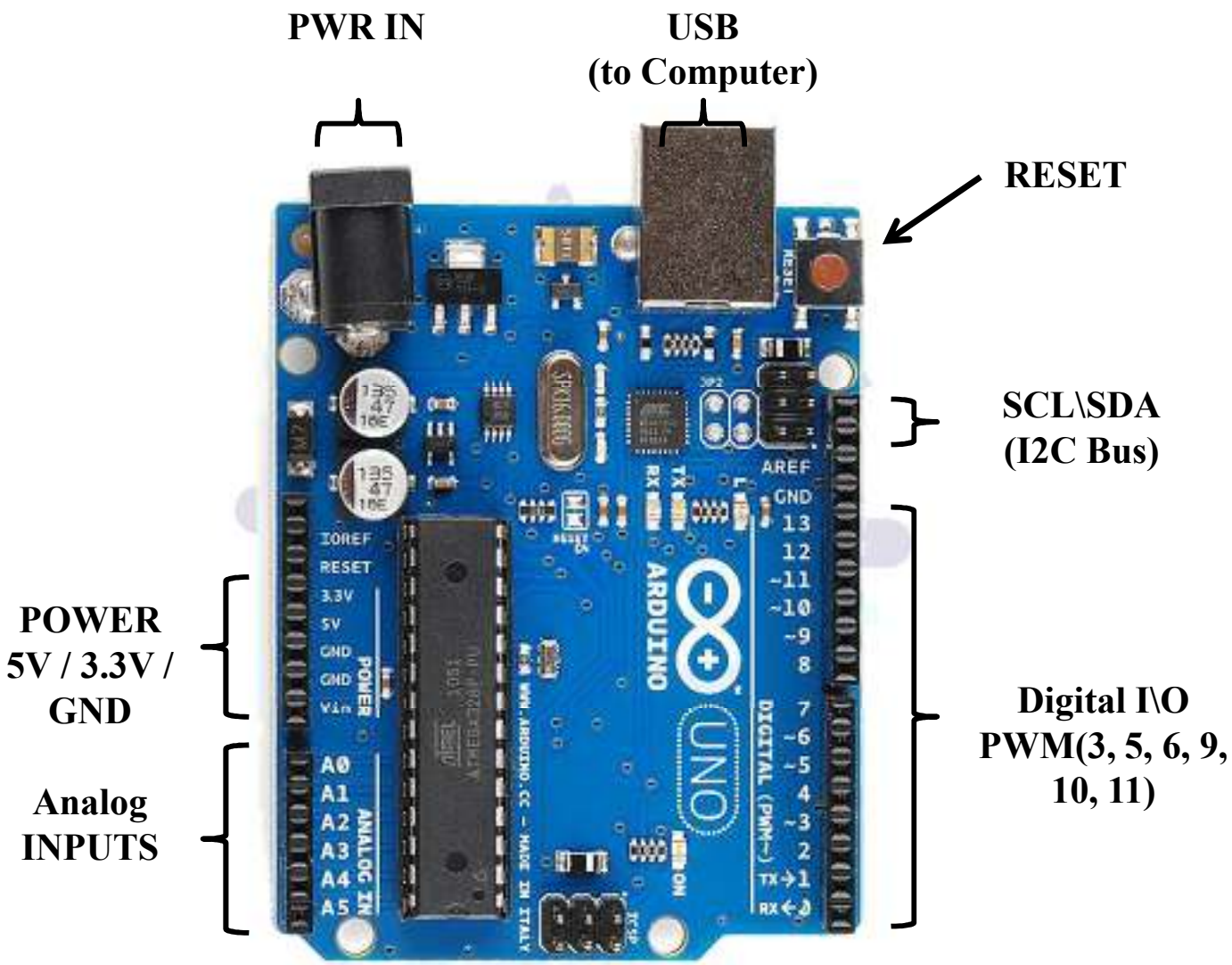
Arduino Mega

Arduino UNO



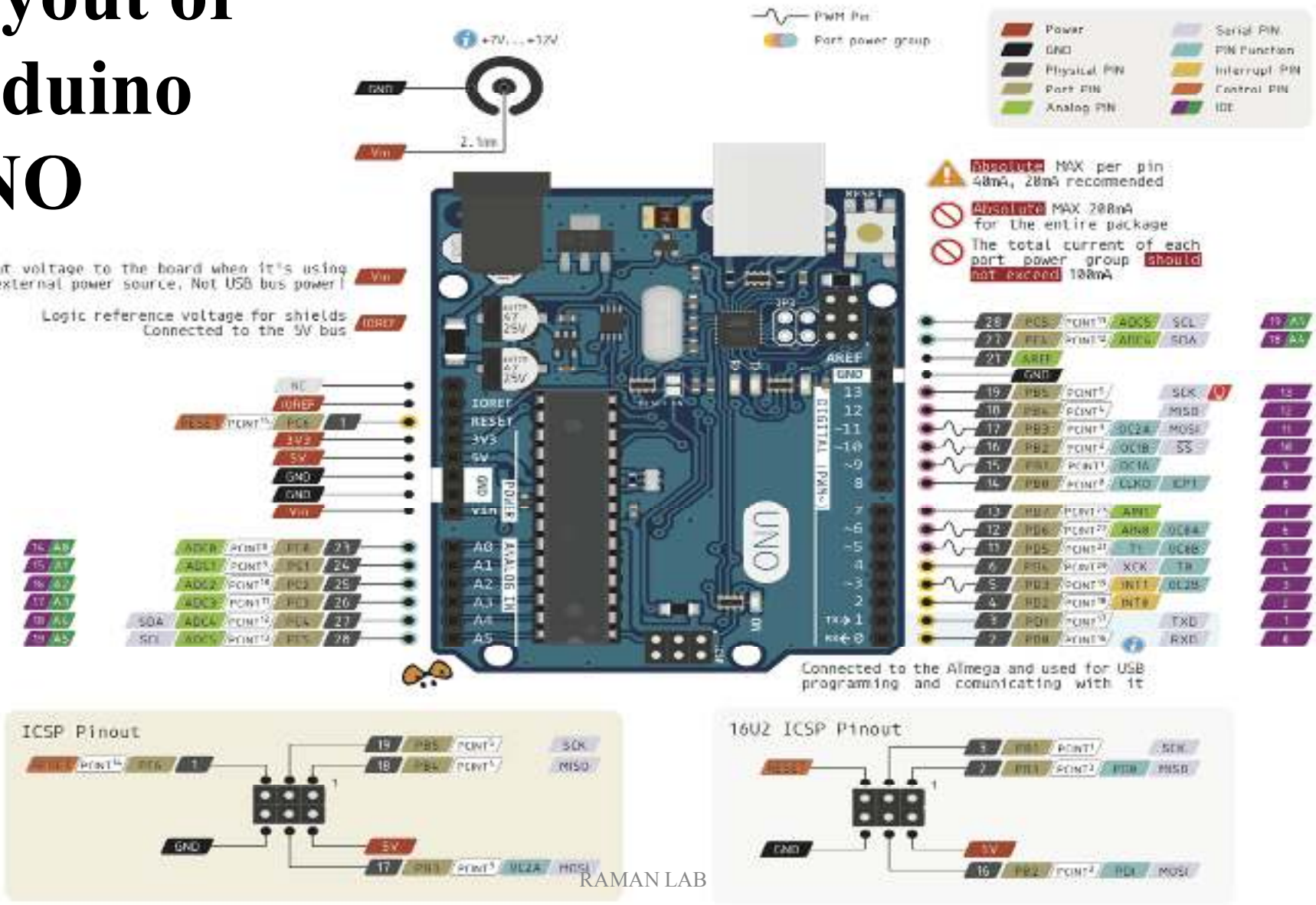
Arduino Mini

Arduino Leonardo


PWR IN
USB
(to Computer)
RESET
SCL\SDA
(I2C Bus)
POWER
5V / 3.3V /
GND
Analog
INPUTS
Digital I\O
PWM(3, 5, 6, 9,
10, 11)
IOREF
RESET
3.3V
5V
GND
GND
Vin
POWER
A0
A1
A2
A3
A4
A5
ANALOG IN
AREF
GND
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX→1
RX←0
DIGITAL (PWM~)
ARDUINO
UNO
WWW.ARDUINO.CC – MADE IN ITALY

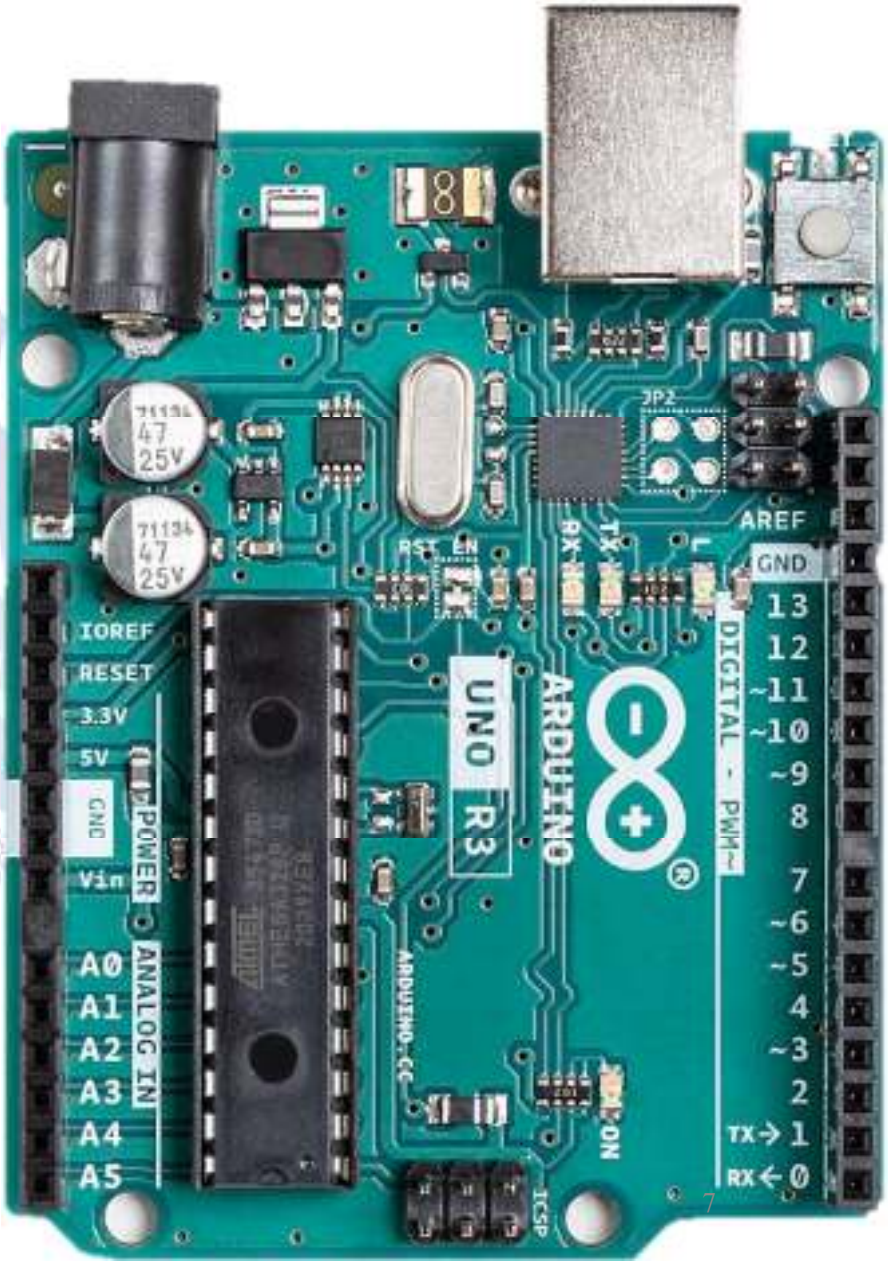
Arduino UNO

Layout of Arduino UNO



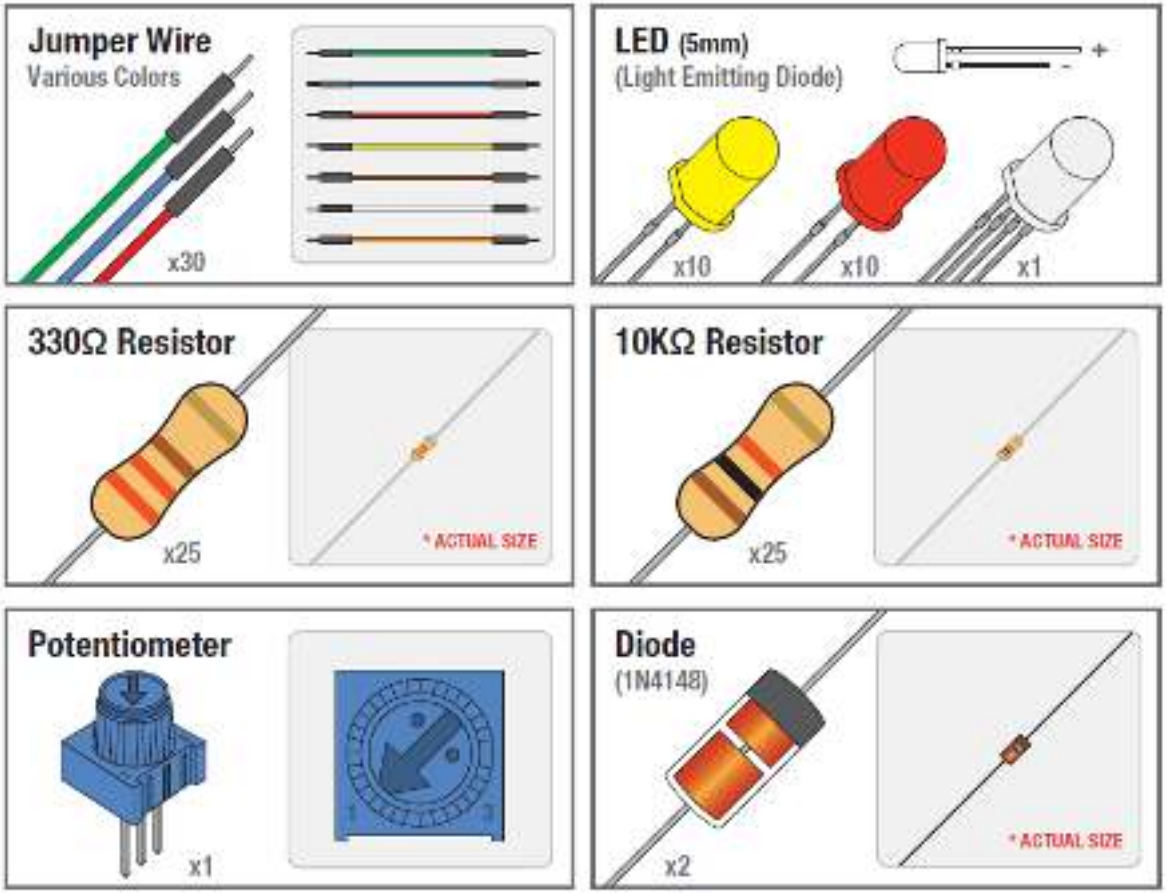
Specifications

- Atmega328 Microcontroller:
  - 32KB Flash Memory.
  - 8 Bit Resolution.
- Tech Specs:
  - USB Connector: USB B
  - DI/DO Pins: 14
  - AI Pins: 6
  - PWM Pins: 6
  - Communications:
    - UART
    - SPI
    - I2C
  - I/O Voltage: 5V
  - Rated Supply Voltage: 7-12V
  - Clock Speed:
    - Main Processor: ATmega328P 16 MHz
    - USB Serial Processor: ATmega16U2 16 MHz
  - Memory: Atmega328p
    - 2KB SRAM, 32KB FLASH, 1KB EEPROM



Components

Name	Image	Type	Function
Push Button		Digital Input	Switch - Closes or opens circuit
Trim potentiometer		Analog Input	Variable resistor
Photoresistor		Analog Input	Light Dependent Resistor (LDR)
Relay		Digital Output	Switch driven by a small signal
Temp Sensor		Analog Input	Temp Dependent Resistor
Flex Sensor		Analog Input	Variable resistor
Soft Trimpot		Analog Input	Variable resistor
RGB LED		Dig & Analog Output	16,777,216 different colors


Jumper Wire
Various Colors
x30
LED (5mm)
(Light Emitting Diode)
+
-
x10
x10
x1
330Ω Resistor
x25
* ACTUAL SIZE
10KΩ Resistor
x25
* ACTUAL SIZE
Potentiometer
x1
Diode
(1N4148)
x2
* ACTUAL SIZE

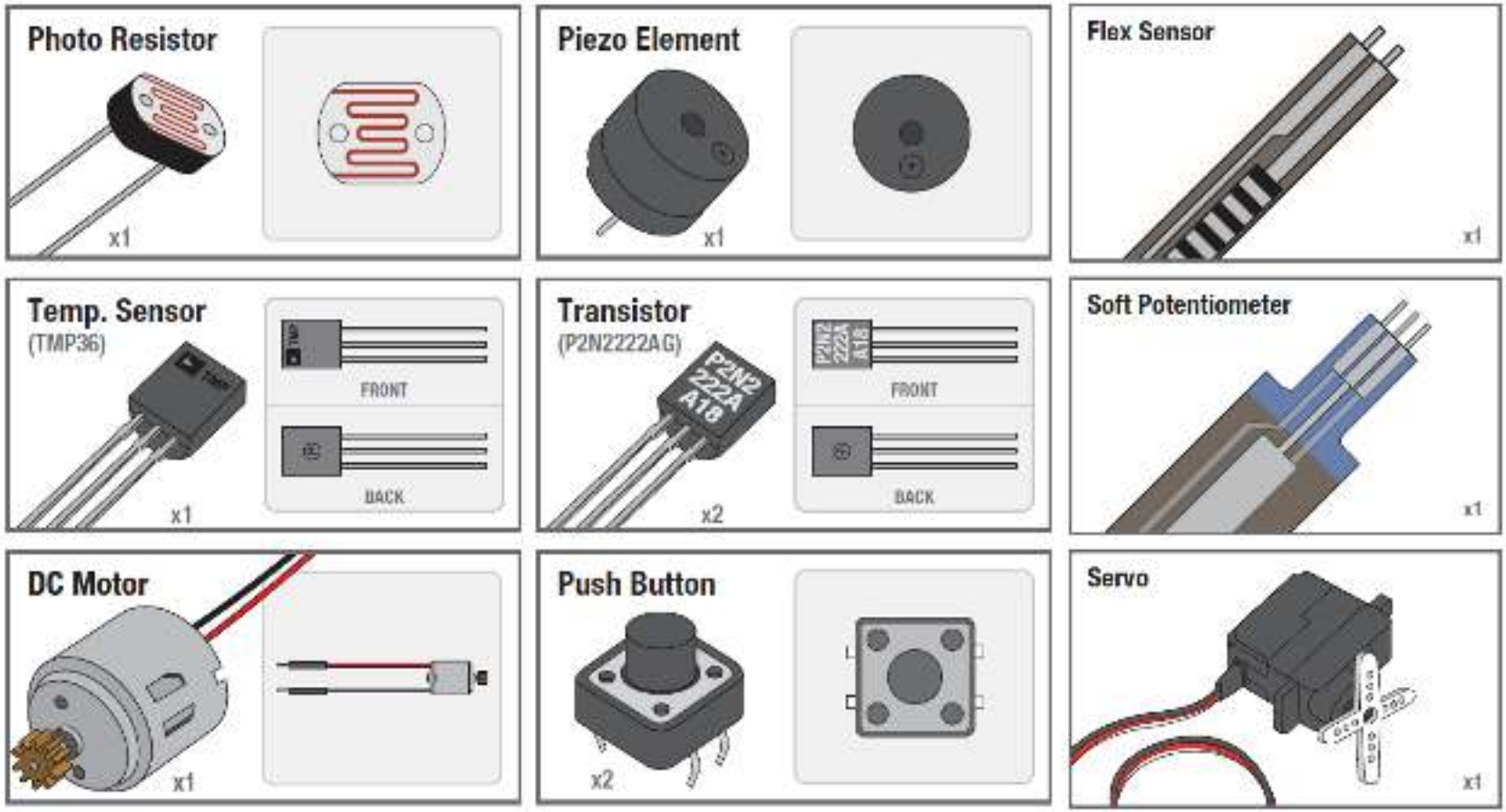
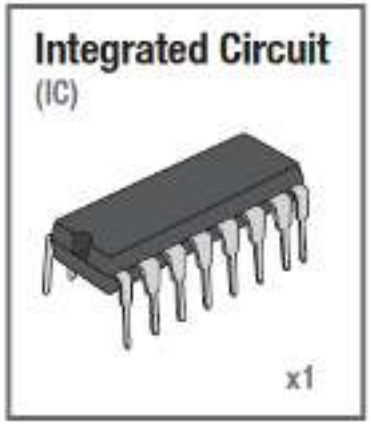
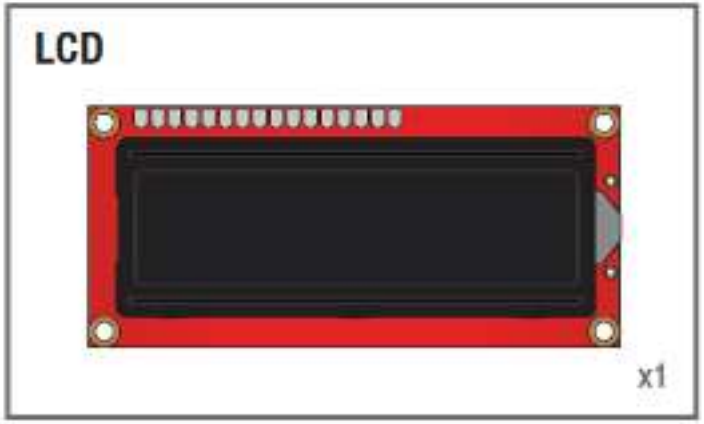
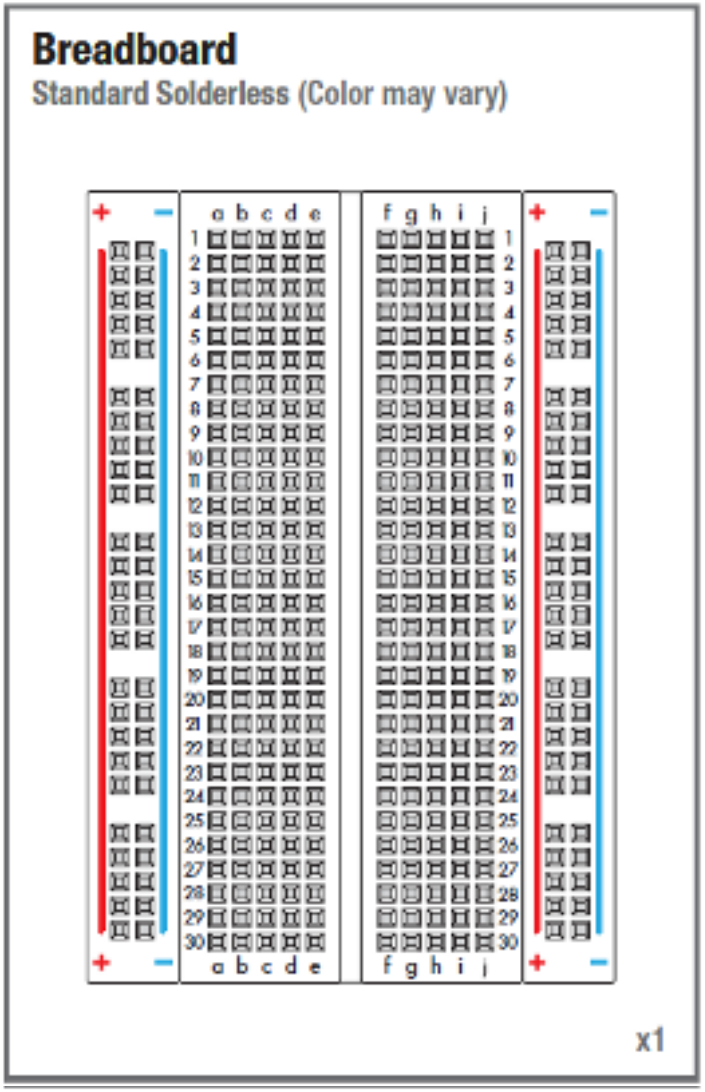


Photo Resistor
x1
Piezo Element
x1
Flex Sensor
x1
Temp. Sensor
(TMP36)
FRONT
BACK
x1
Transistor
(P2N2222AG)
P2N2 222A A18
FRONT
BACK
x2
Soft Potentiometer
x1
DC Motor
x1
Push Button
x2
Servo
x1


Relay
x1


Integrated Circuit
(IC)
x1

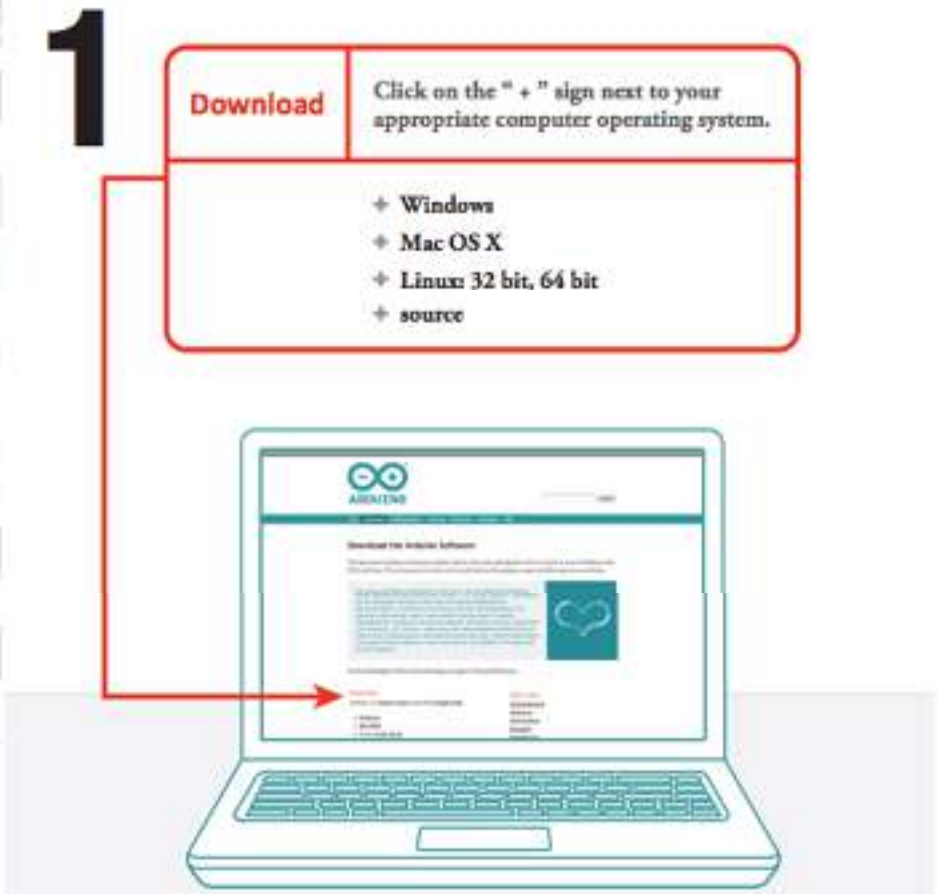

LCD
x1


Breadboard
Standard Solderless (Color may vary)
x1

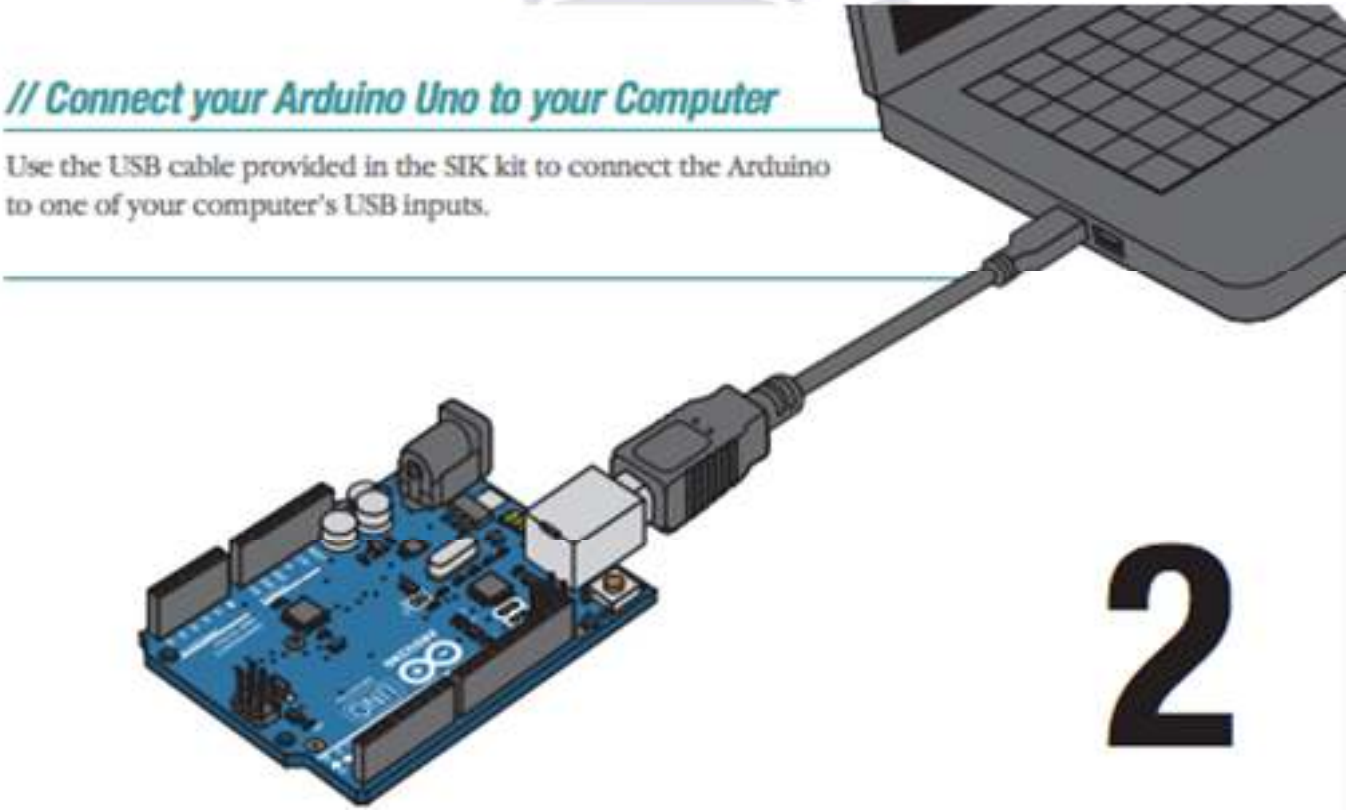
DownLoad Arduino IDE

(Integrated Development Environment)

arduino.cc/en/main/software



Connect Arduino to your Computer



Install Arduino Drivers

3

// Install Drivers

Depending on your computer's operating system, you will need to follow specific instructions. Please consult the URLs below for specific instructions on how to install the drivers onto your Arduino Uno.

* You will need to scroll to the section labeled "Install the drivers".

Windows Installation Process

Go to the web address below to access the instructions for installations on a Windows-based computer.

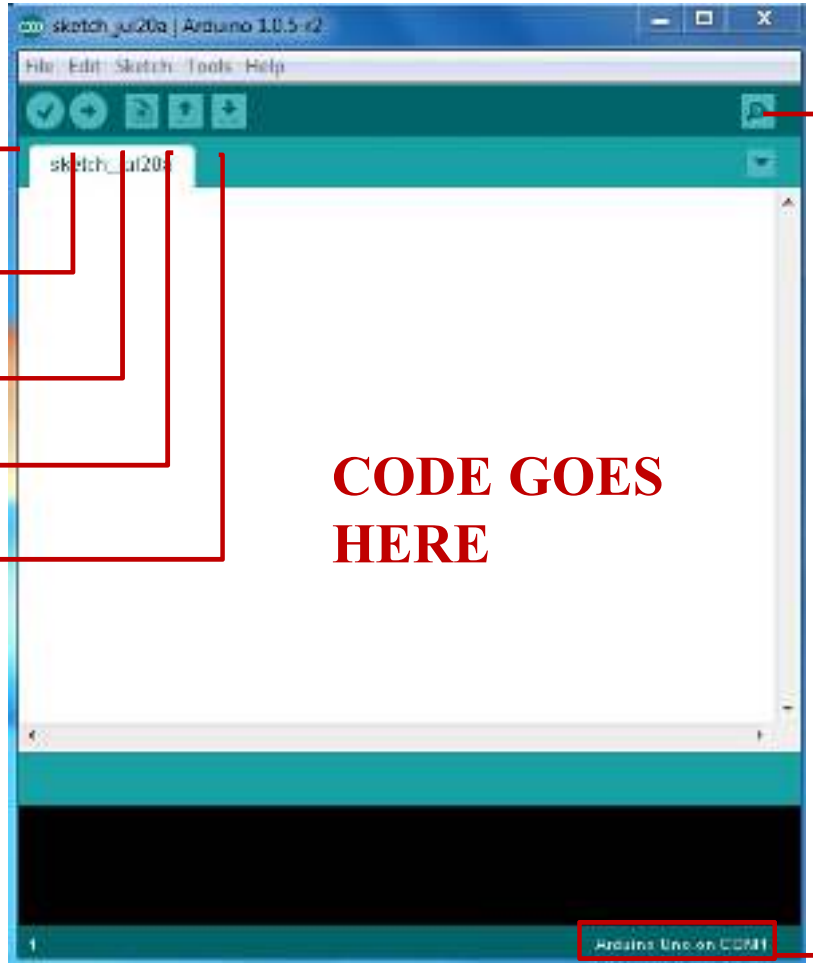
http://arduino.cc/en/Guide/Windows

Macintosh OS X Installation Process

Macs do not require you to install drivers. Enter the following URL if you have questions. Otherwise proceed to next page.

http://arduino.cc/en/Guide/MacOSX

Open Arduino IDE



VERIFY

UPLOAD

NEW TAB

OPEN

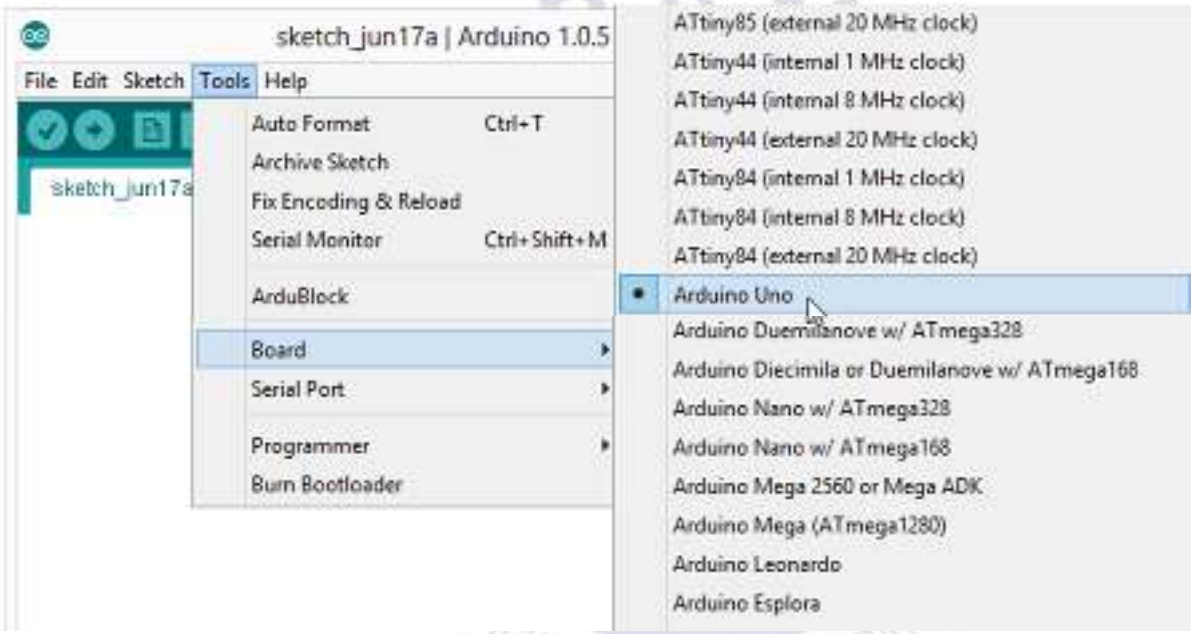
SAVE

CODE GOES HERE

SERIAL MONITOR

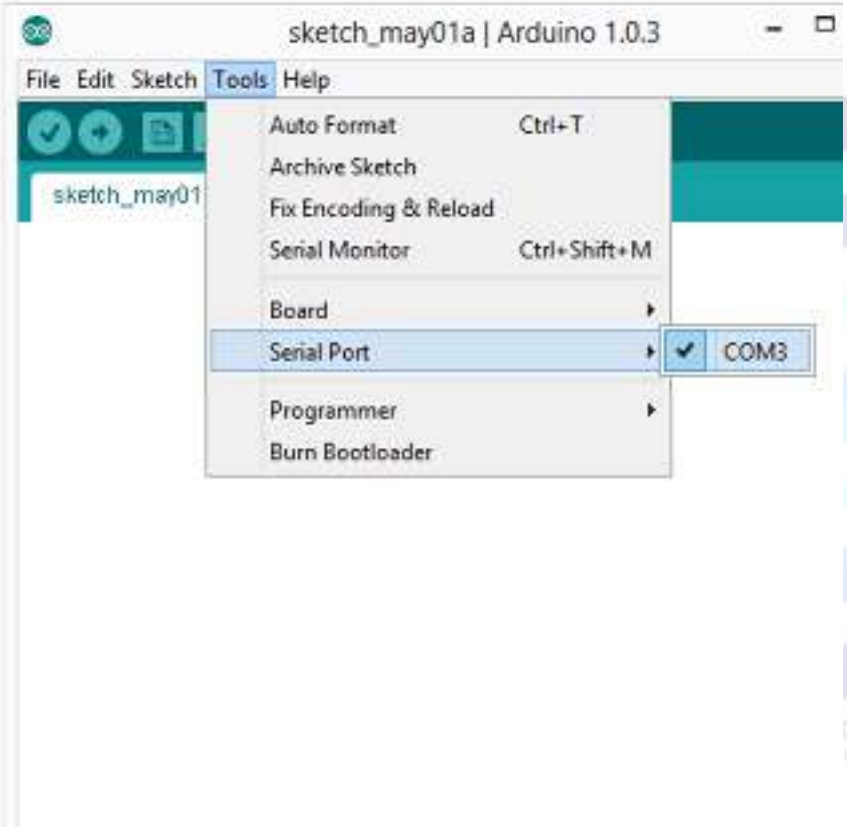
BOARD & SERIAL PORT SELECTIONS

Settings: Tools → Board



•Next, double-check that the proper board is selected under the Tools→Board menu.

Settings: Tools → Serial Port



- Your computer communicates to the Arduino microcontroller via a serial port → through a USB-Serial adapter.

- Check to make sure that the drivers are properly installed.

Basic Coding Notebook

Data Types:

Type	Sign	Bytes	Bits	Range		Other Info.
				Min	Max	
char	signed	1	8	-128	127	ASCII
char	unsigned	1	8	0	255	ASCII
byte		1	8	0	255	
int(Uno+)	signed	2	16	-32768	32767	Uno model +others
short		2	16	-32768	32767	
int(Uno+)	unsigned	2	16	0	65535	Uno model +others
word		2	16	0	65535	Same as unsigned int
long	signed	4	32	-2147483648	2147483647	Append with ‘L’
Long	unsigned	4	32	0	4294967295	
float		4	32	-3.4028235E+38	3.4028235E+38	6-7 dec digits of precision
double (Uno+)		4	32	-3.4028235E+38	3.4028235E+38	Same as float

Commonly used operators:

- Sum: +
- Product: *
- Division: /
- Subtraction: -
- Modulo: %
- Exponent: ^

- AND: &&
- OR: ||
- NOT: !
- Equal to: = =
- Greater than: >
- Less than: <
- Greater than equal to: >=
- Less than equal to: <=
- Not Equal to: !=

Basic Coding

- ***setup()* function**
  - Called when a sketch starts.
  - To initialize variables, pin modes, start using libraries, etc.
  - Will only run once, after each power-up or reset of the Arduino board.
- ***loop()* function**
  - Loops consecutively.
  - Code in the loop() section of the sketch is used to actively control the Arduino board.
- ***Commenting***
  - Any line that starts with two slashes (//) will not be read by the compiler, so you can write anything you want after it.

- ***pinMode()***
  - Instruction used to set the mode (INPUT or OUTPUT) in which we are going to use a pin.
  - E.g.: pinMode (13, OUTPUT);
  - i.e. setting pin13 as output.

- ***digitalWrite()***
  - Write a HIGH or a LOW value to a digital pin.
  - E.g.: digitalWrite (11, HIGH);
  - i.e. setting pin 11 to high.

- ***digitalRead()***
  - Reads the value from a specified digital pin, either HIGH or LOW
  - E.g.: int inPin=7;

    val = digitalRead(inPin);
  - ie. reads the value from inPin and assigns it to val.

- ***delay()***
  - Pauses the program for the amount of time (in milliseconds) specified as parameter.
  - E.g.: delay(1000);
  - ie. waits for a second (1000 ms = 1 s)

Setup

The setup section is used for assigning input and outputs (Examples: motors, LED's, sensors etc) to ports on the Arduino
To do this we use the command "pinMode"

```
void setup() {

    pinMode(9, OUTPUT);

}
```

port #

Input or Output

Loop

```
void loop() {

    digitalWrite(9, HIGH);
    delay(1000);
    digitalWrite(9, LOW);
    delay(1000);

}
```

Port # from setup

Turn the LED on or off

Wait for 1 second or 1000 milliseconds

Blink §

```
void setup() {
  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 t
  delay(1000);                       // wait for a second
  digitalWrite(LED_BUILTIN, LOW);    // turn the LED off by making
  delay(1000);                       // wait for a second
}
```

TASK 1

- Using 3 LED's (red, yellow and green) build a traffic light that
  - Illuminates the green LED for 5 seconds
  - Illuminates the yellow LED for 2 seconds
  - Illuminates the red LED for 5 seconds
  - repeats the sequence
- Note that after each illumination period the LED is turned off!

TASK 2

- Modify Task 1 to have an advanced green (blinking green LED) for 3 seconds before illuminating the green LED for 5 seconds

Control Structure & Loop

If conditioning:

```
if(condition)
{
Statement-1
….
Statement-N

}
else if(condition)
{
Statement
}
else{Statement}
```

Switch case:

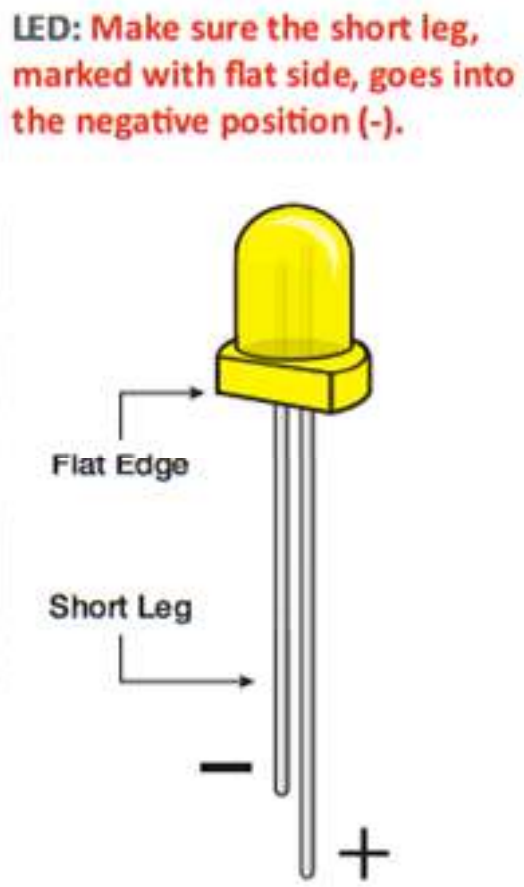
```
switch(var)
{
Case 1:
// do something when var equal to
1
break;

Case 2:
// do something when var equal to
2
break;

default:
//if nothing else matches, do the
default
//default is optional
}
```

```
while(expression)
 { Block of statements; }


do { Block of statements; }
while (expression);


for ( initialize; control; increment or decrement)
{ // statement block }
```

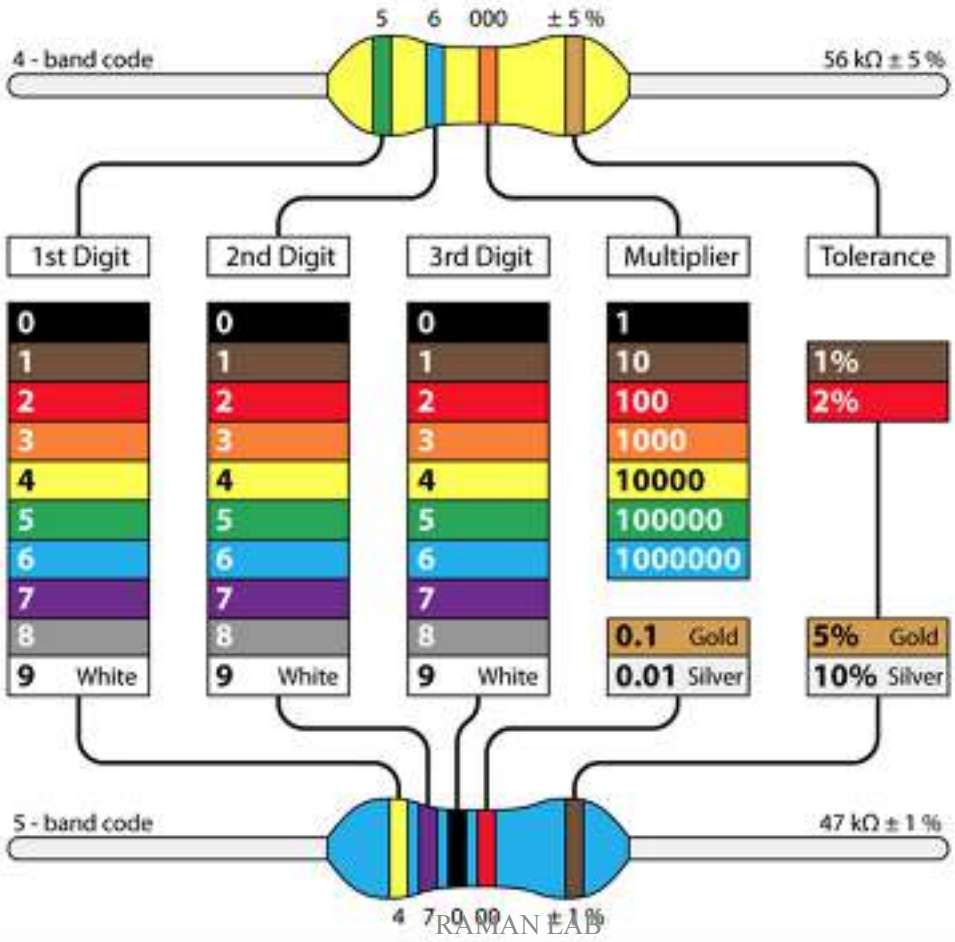
Practical Hands On

Circuit #1: Blinking an LED

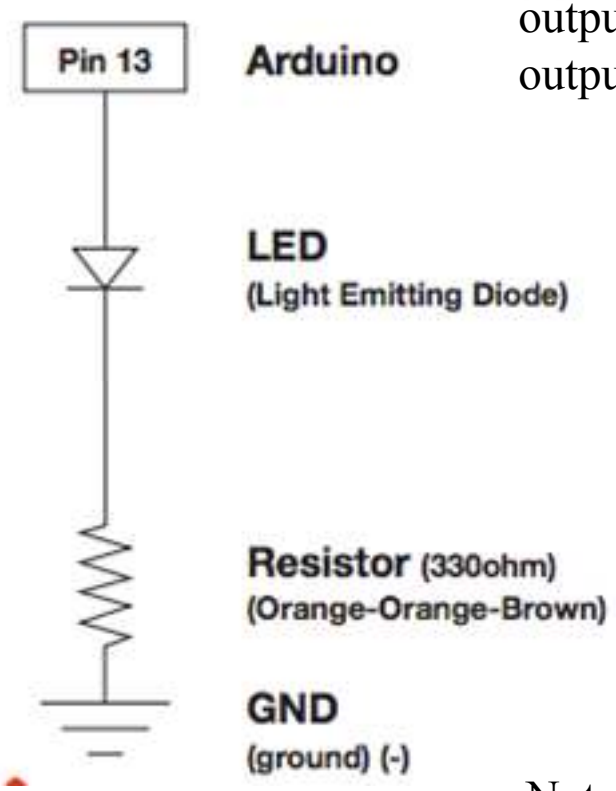
LEDs (light emitting diodes) are small, powerful lights used in many different applications.



Resistor Color Code



Circuit Schematic



Pin 13 is a digital pin and can be used as an output or an input. Here we will use it to output power to the LED.

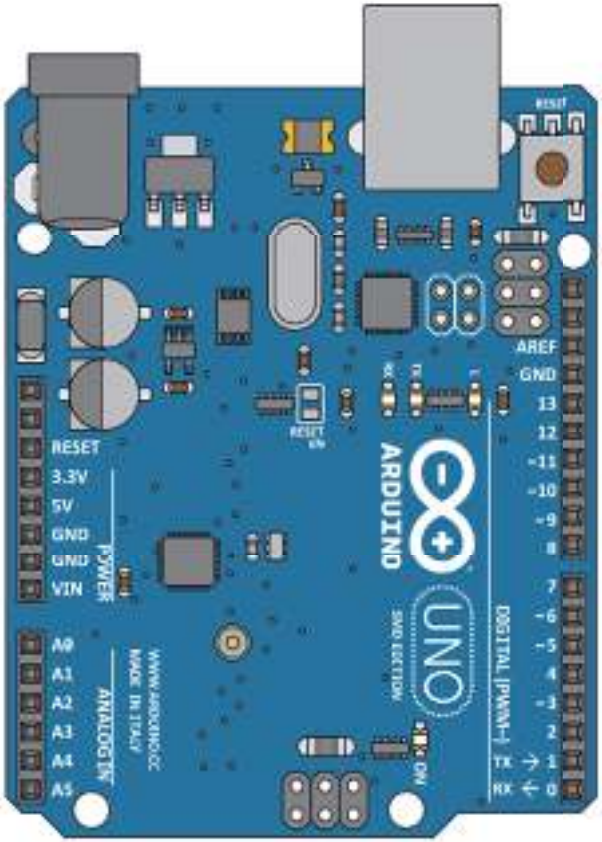
Pin 13 will be connected to the positive lead on the LED.

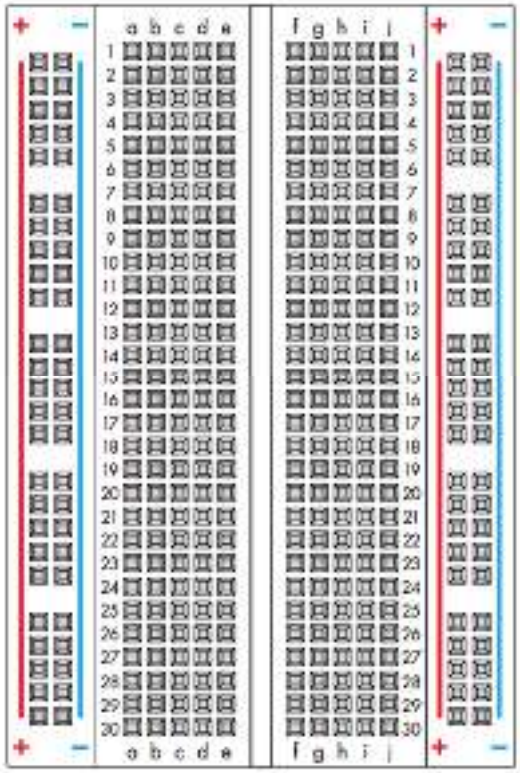
The negative lead on the LED will be connected to one leg of the resistor.

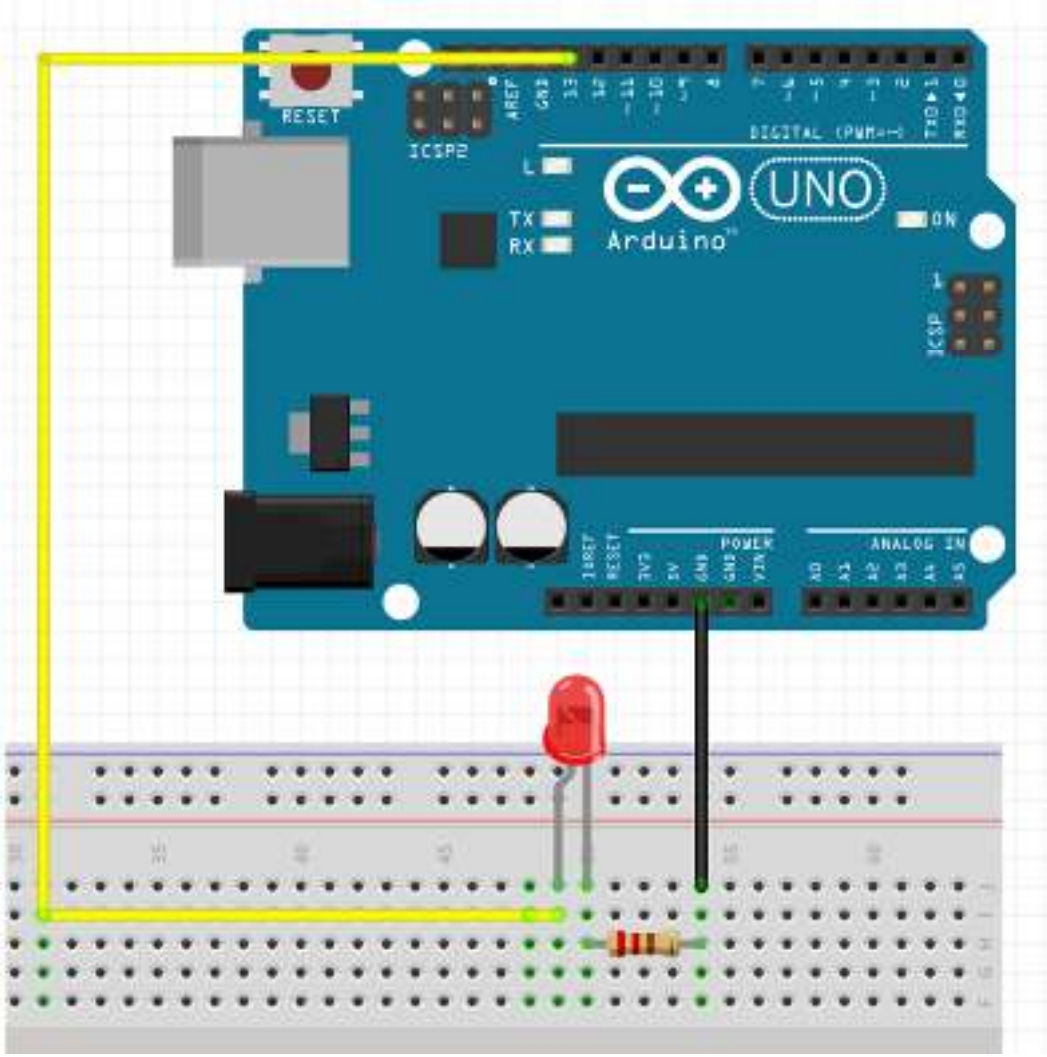
The other leg of the resistor will be connected to ground to complete the circuit.

Note: These components are all in series (one after the other, like beads on a string).

Design it!


RESET
3.3V
5V
GND
GND
VIN
POWER
A0
A1
A2
A3
A4
A5
ANALOG IN
WWW.ARDUINO.CC
MADE IN ITALY
RESET
ON
ARDUINO
UNO
SMD EDITION
DIGITAL (PWM~)
TX
RX
AREF
GND
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
1
0




RESET
ICSP2
L
TX
RX
UNO
Arduino
ON
ICSP
DIGITAL (PWM~)
POWER
ANALOG IN

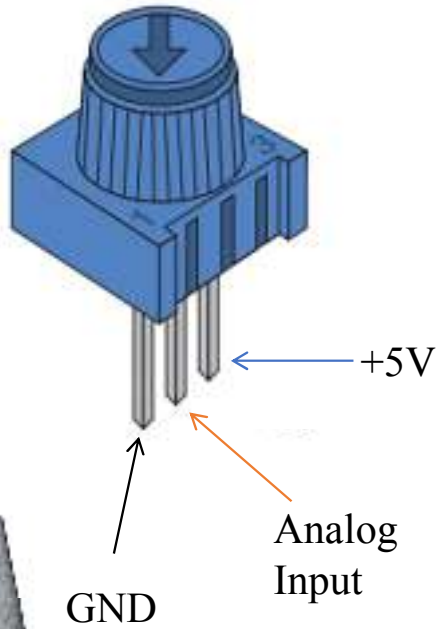
Create the Sketch

```
/*
 Blink
Turns on an LED for one second then off for one second, repeatedly.
*/
void setup() {
pinMode(13, OUTPUT);
}
void loop() {
digitalWrite(13, HIGH);
delay(1000);
digitalWrite(13, LOW);
delay(1000);
}
```

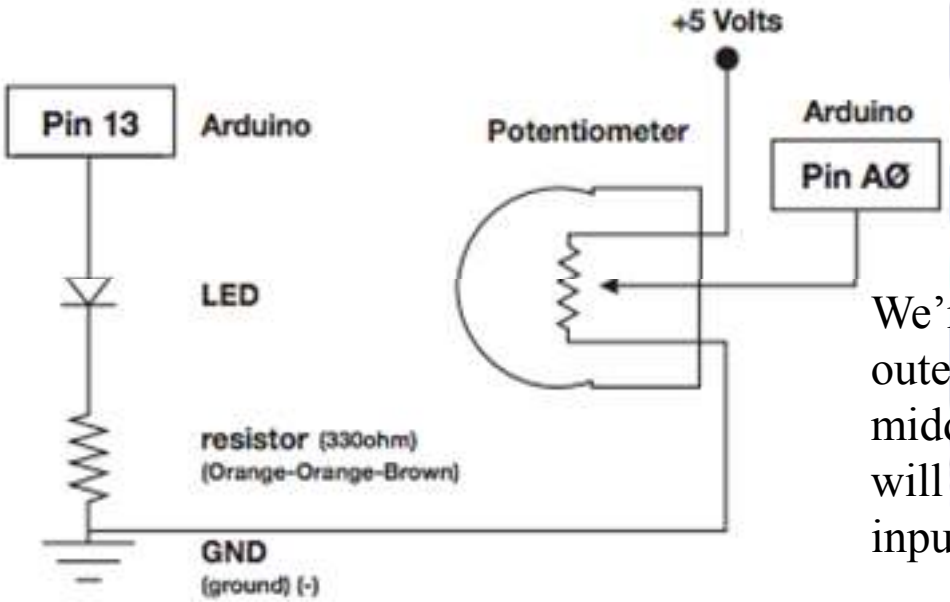
Circuit #2: Potentiometer

How to read analog input from the physical world using a potentiometer ("pot" for short) and control the blink rate of an LED. We'll also learn how to use the serial monitor to watch how the voltage changes.

When it's connected with 5V across its two outer pins, the middle pin outputs a voltage between 0 and 5V, depending on the position of the knob. In this way, it can be used as a "voltage divider".



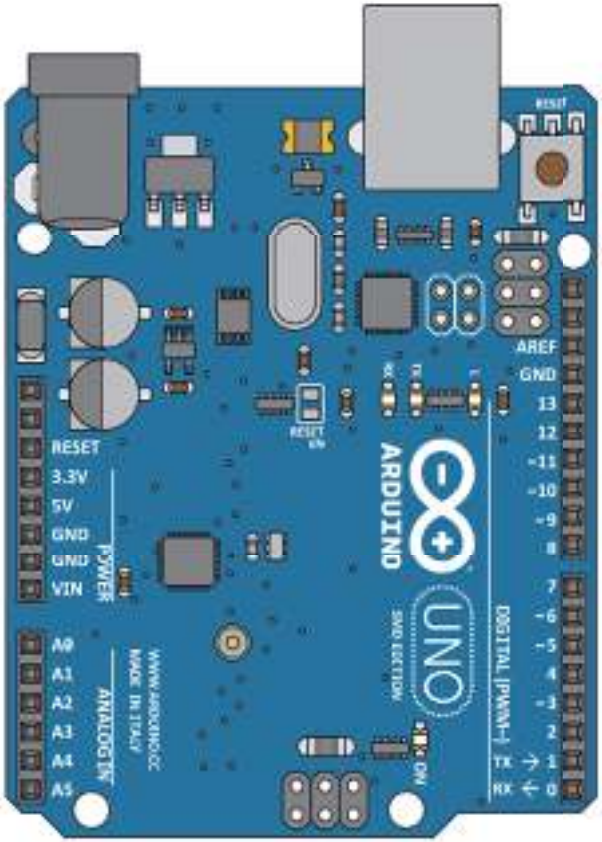
Circuit Schematic

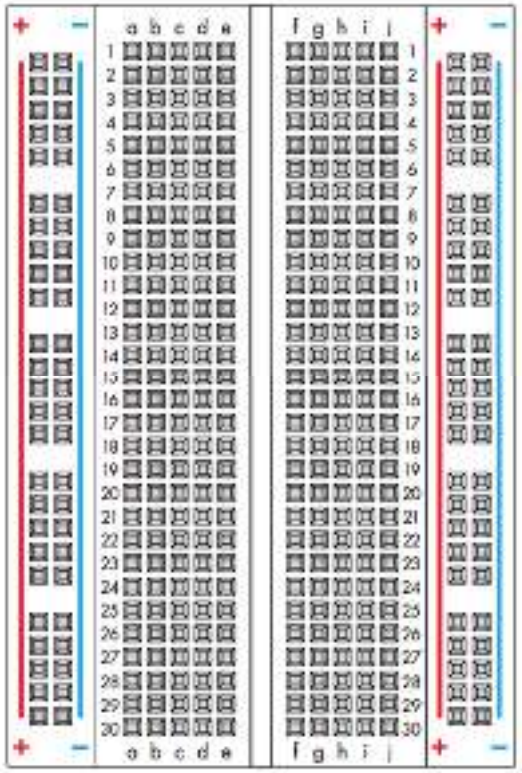


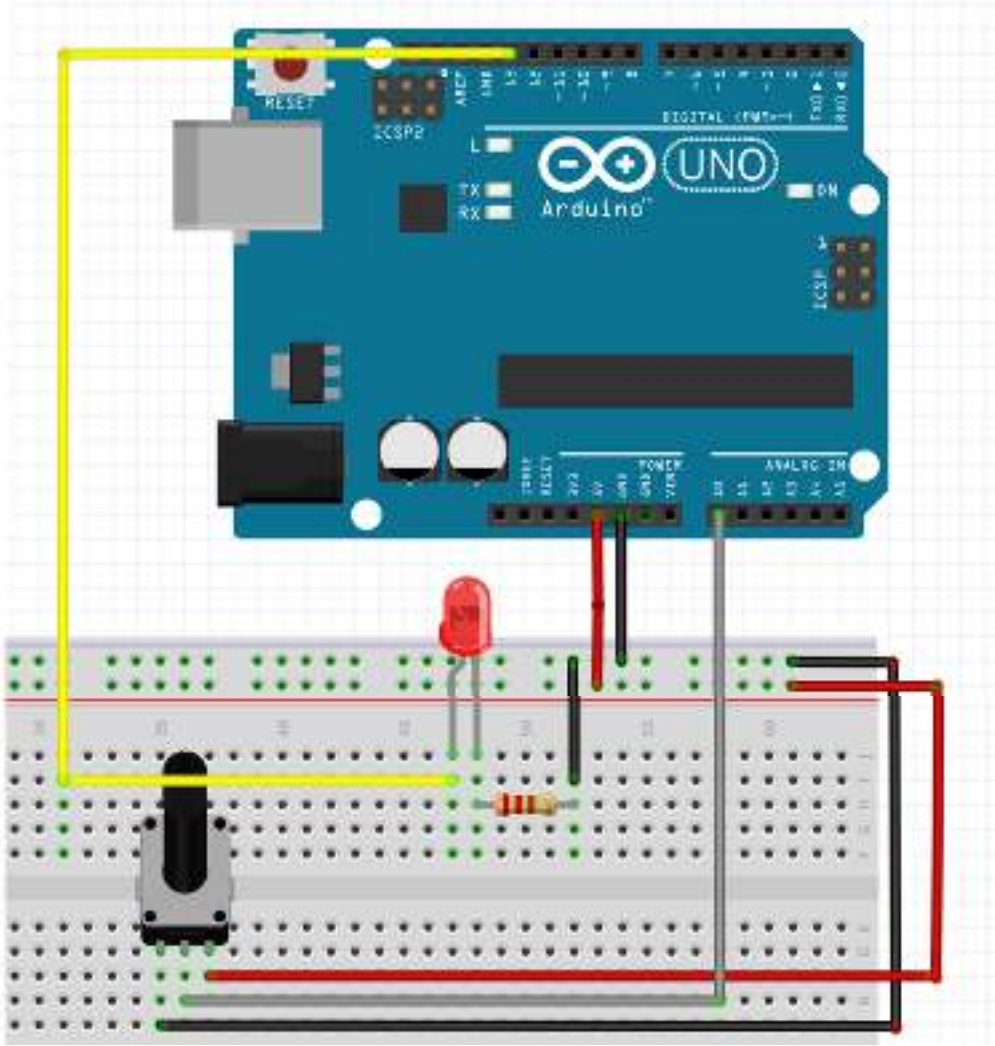
We're adding a potentiometer to control the blink rate of the LED.

We're running 5V across the outer pins of the pot. The middle pin of the potentiometer will be connected to analog input pin 0.

Design it!


RESET
3.3V
5V
GND
GND
VIN
POWER
A0
A1
A2
A3
A4
A5
ANALOG IN
WWW.ARDUINO.CC
MADE IN ITALY
RESET
ON
ARDUINO
UNO
SMD EDITION
DIGITAL (PWM~)
TX
RX
AREF
GND
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
1
0




RESET
ICSP2
L
TX
RX
UNO
Arduino
ON
ICSP
POWER
ANALOG IN

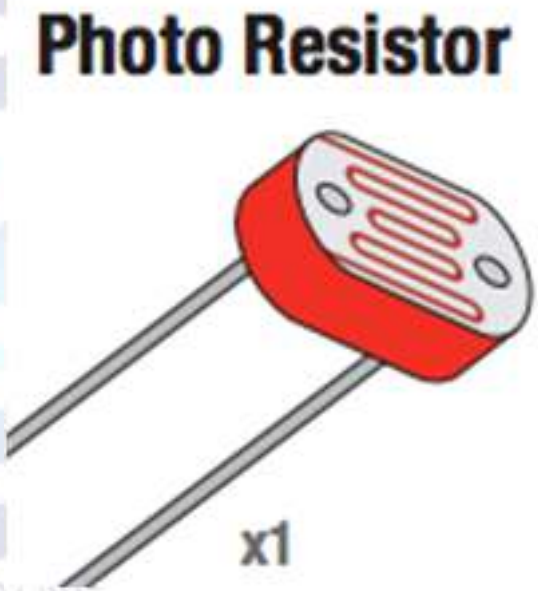
Create the Sketch

```
int sensorPin =0;
int ledPin =13;
void setup() {
Serial.begin(9600);
pinMode(ledPin, OUTPUT);
}
void loop() {
int sensorValue;
sensorValue = analogRead(sensorPin);
digitalWrite(ledPin, HIGH);
delay(sensorValue);
digitalWrite(ledPin, LOW);
delay(sensorValue);
Serial.println(sensorValue);
}
```

Circuit #3
Photo Resistor (Light Sensor)

- Photoresistors change resistance based on how much light the sensor receives.
- Use our photo resistor in a “voltage divider” configuration. Output:
- High voltage = lot of light
- Low voltage = little light

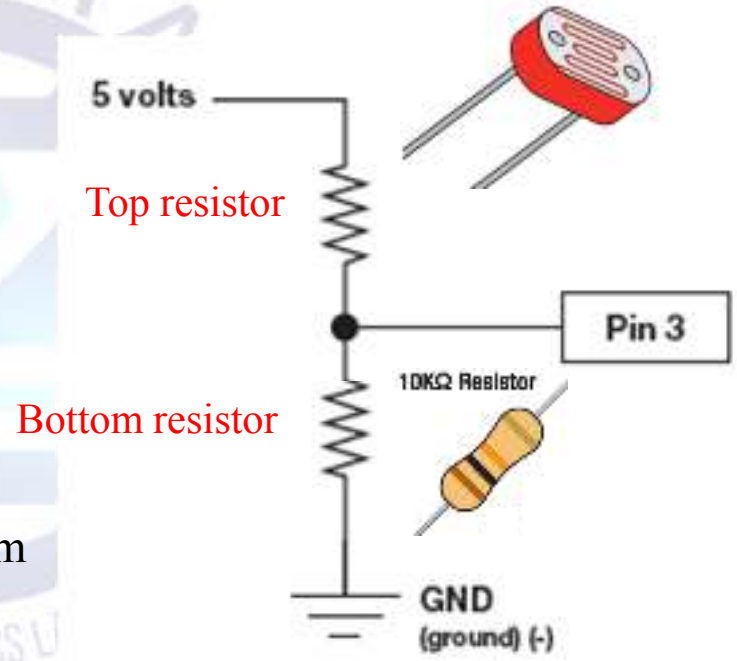
- Brighten and dim an LED based on the light level picked up by the photo resistor.



Resistive Sensors & Voltage Dividers

Resistor in disguise!

- Arduino measures voltage, not resistance. Need a voltage divider:
- Consists of two resistors.
- The “top” resistor is the sensor.
- The “bottom” resistor is a normal, fixed resistor (usually 10 KΩ).



When top resistor is connected to 5V and the bottom resistor to ground, the middle will output a voltage proportional to the values of the two resistors.

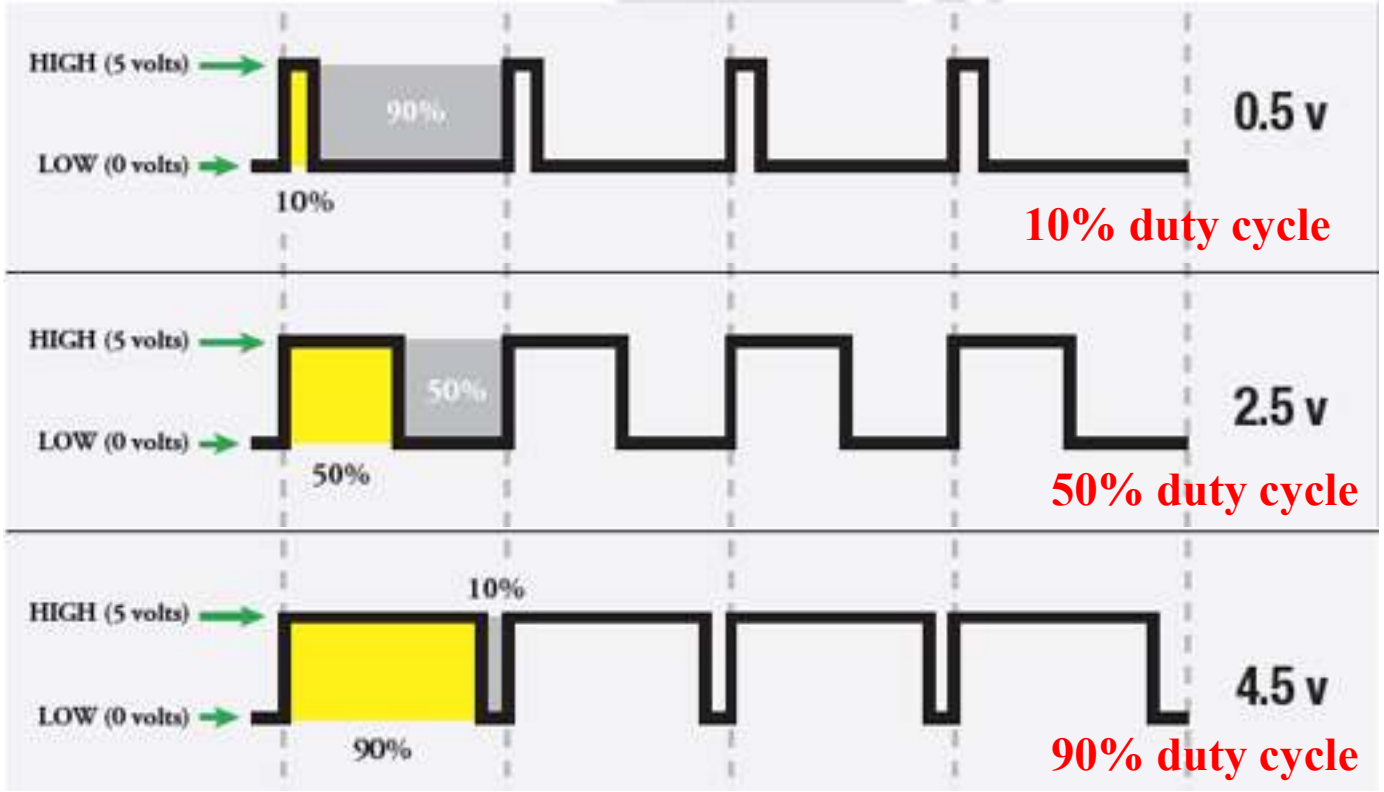
Pulse Width Modulation (PWM)

- Arduino is so fast it can blink a pin on and off 1,000 times per second.
- PWM pins also vary amount of time blinking pin spends on HIGH vs. LOW.

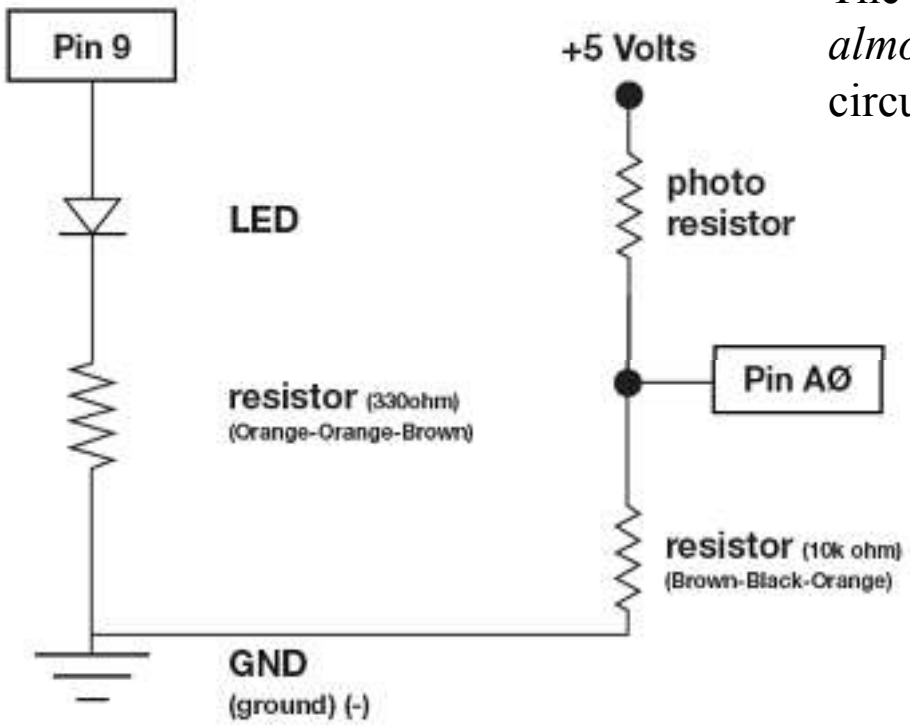
- Use function: analogWrite(pin, value)
- Choose a pin marked by a ~
- Value is the duty cycle
  - 0 = always OFF
  - 255 = always ON
  - 127 = on HALF the time

    (50% duty cycle)

Pulse Width Modulation (PWM)

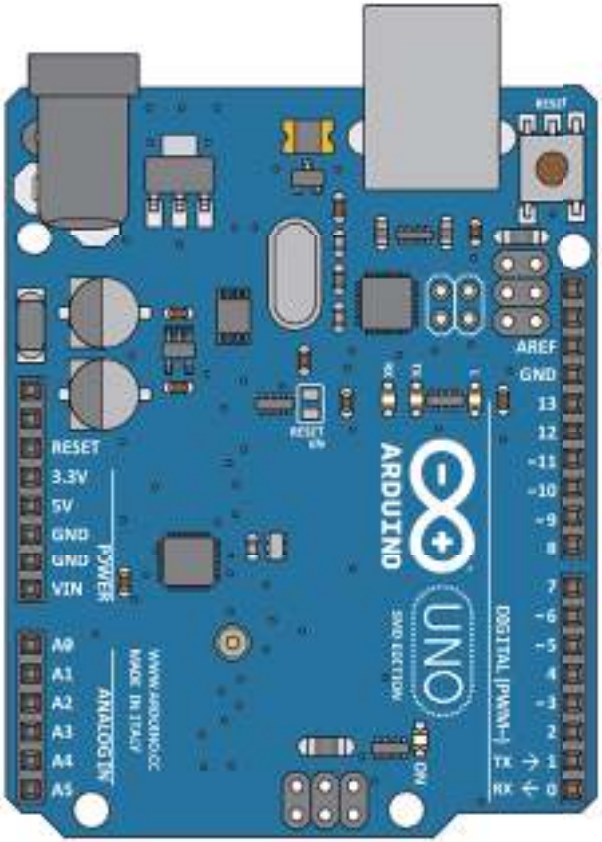


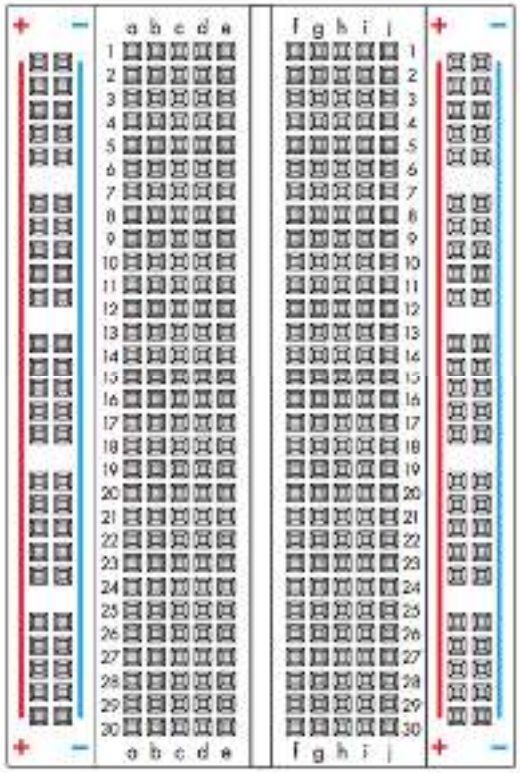
Circuit Schematic

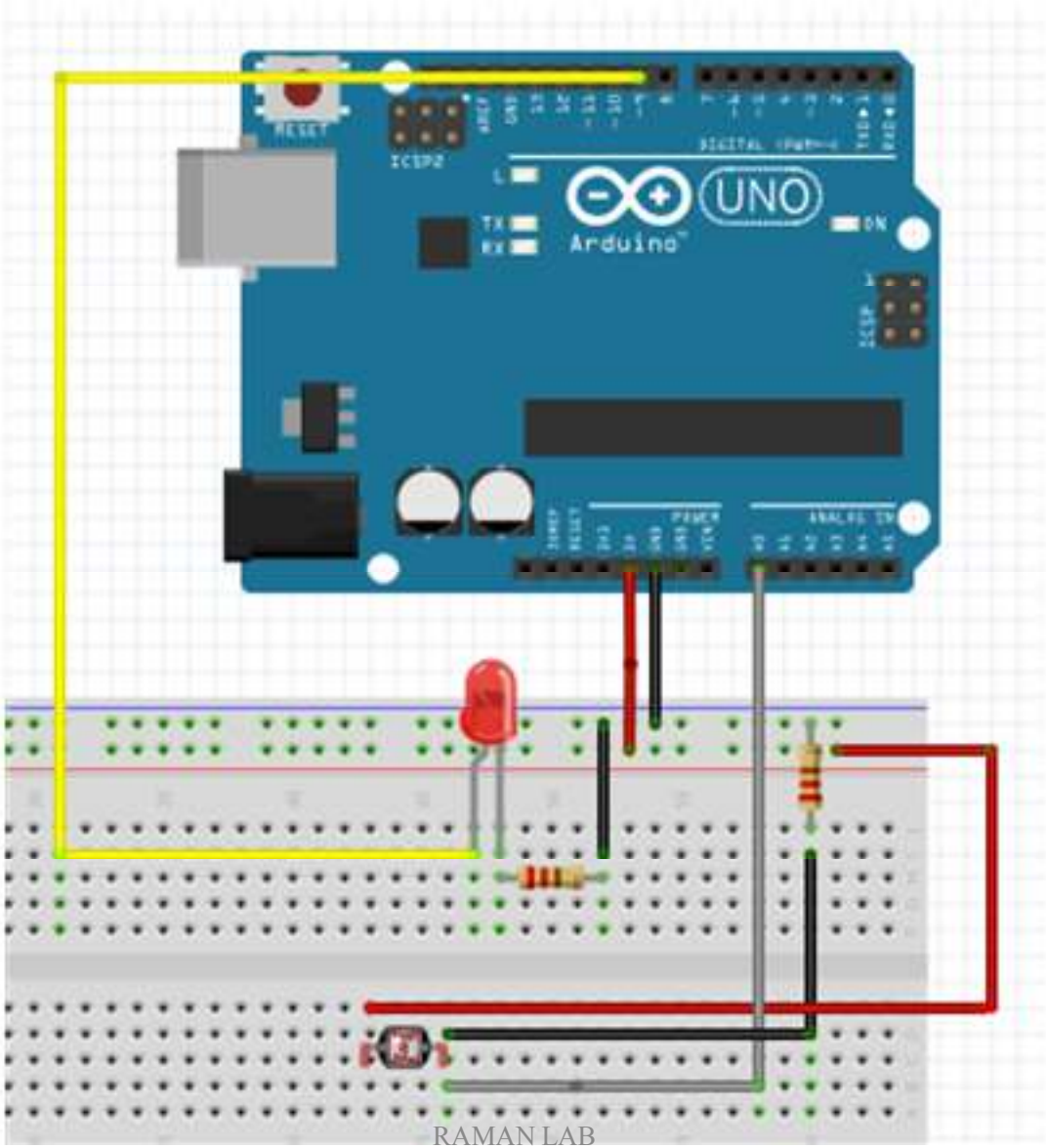


The left side of the schematic is *almost* the same as the previous circuits. What's changed?

Design it!


RESET
3.3V
5V
GND
GND
VIN
POWER
A0
A1
A2
A3
A4
A5
ANALOG IN
WWW.ARDUINO.CC
MADE IN ITALY
RESET
ON
ARDUINO
UNO
SMD EDITION
DIGITAL (PWM~)
TX
RX
AREF
GND
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
1
0




RESET
ICSP2
UNO
Arduino
ON
ICSP
POWER
ANALOG IN

Create the Sketch

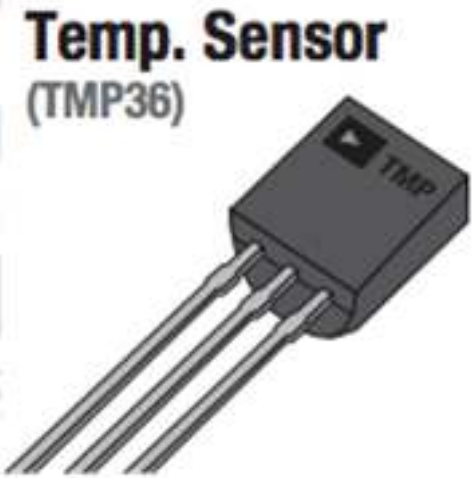
```
const int sensorPin=0;
const int ledPin=9;
int lightLevel, high=0, low=1023;
void setup() {
pinMode(ledPin, OUTPUT);
Serial.begin(9600);
}
void loop() {
lightLevel = analogRead(sensorPin);
autoTune();
analogWrite(ledPin, lightLevel);
Serial.println(lightLevel);
}
```

Create the Sketch

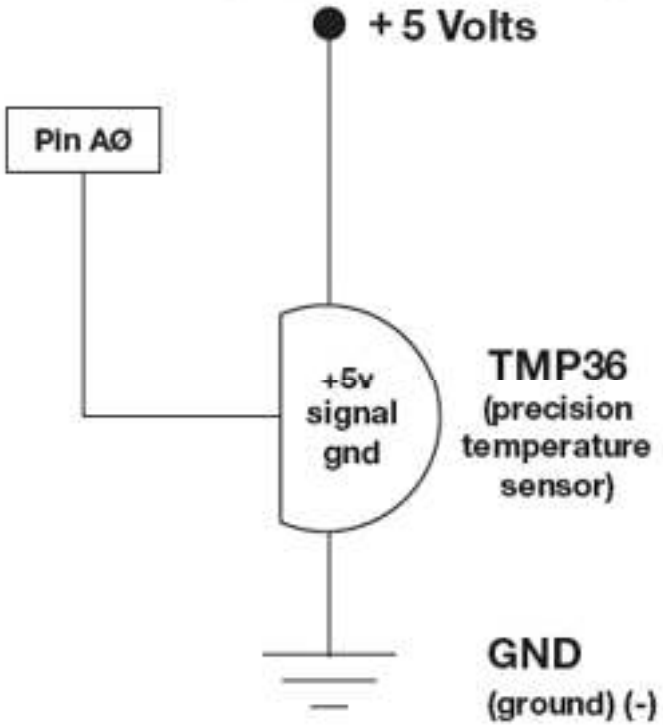
```
void autoTune()
{
if(lightLevel<low)
{
  low=lightLevel;
}
if(lightLevel>high)
{
  high=lightLevel;
}
lightLevel=map(lightLevel, low+30, high-30, 0, 255);
lightLevel=constrain(lightLevel, 0, 255);
}
```

Circuit #4: Temperature Sensor

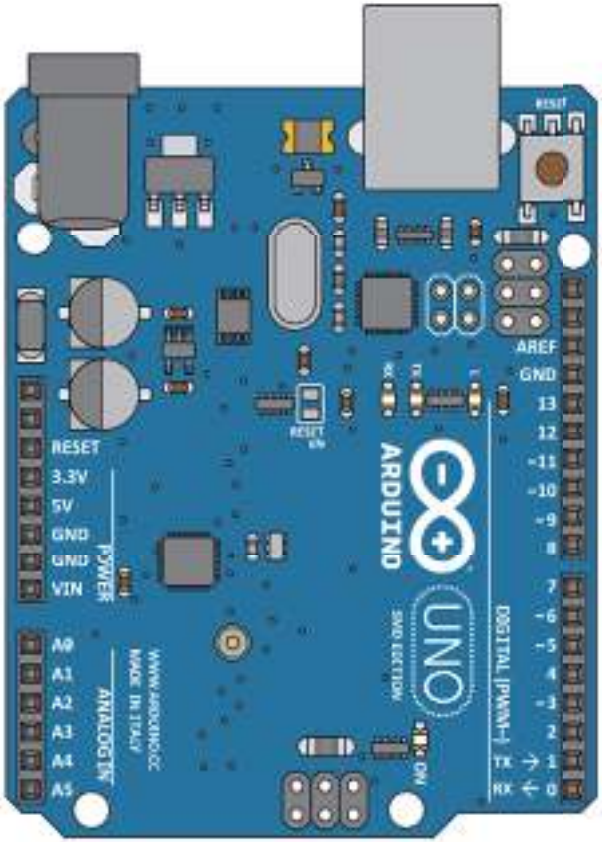
- Temperature sensors are used to measure ambient temperature.
- Sensor we're using has three pins – positive, ground, and a signal. For every centigrade degree it reads, it outputs 10 millivolts.
- We'll integrate the temperature sensor with Arduino and use the serial monitor to display the temperature.

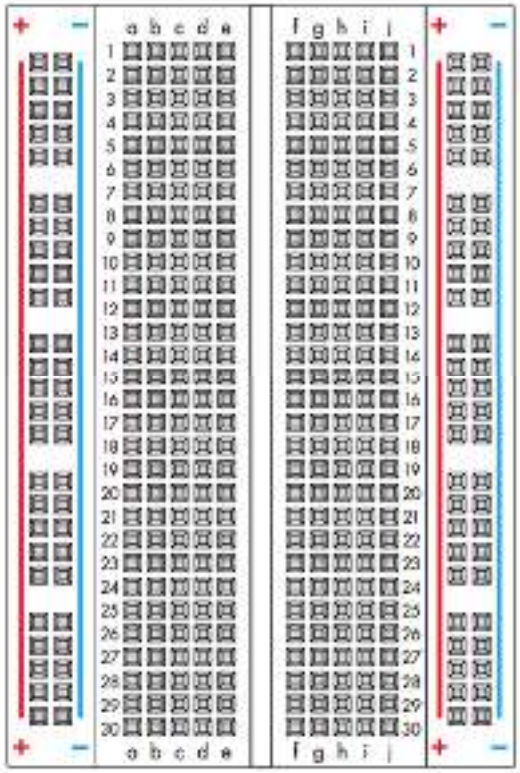


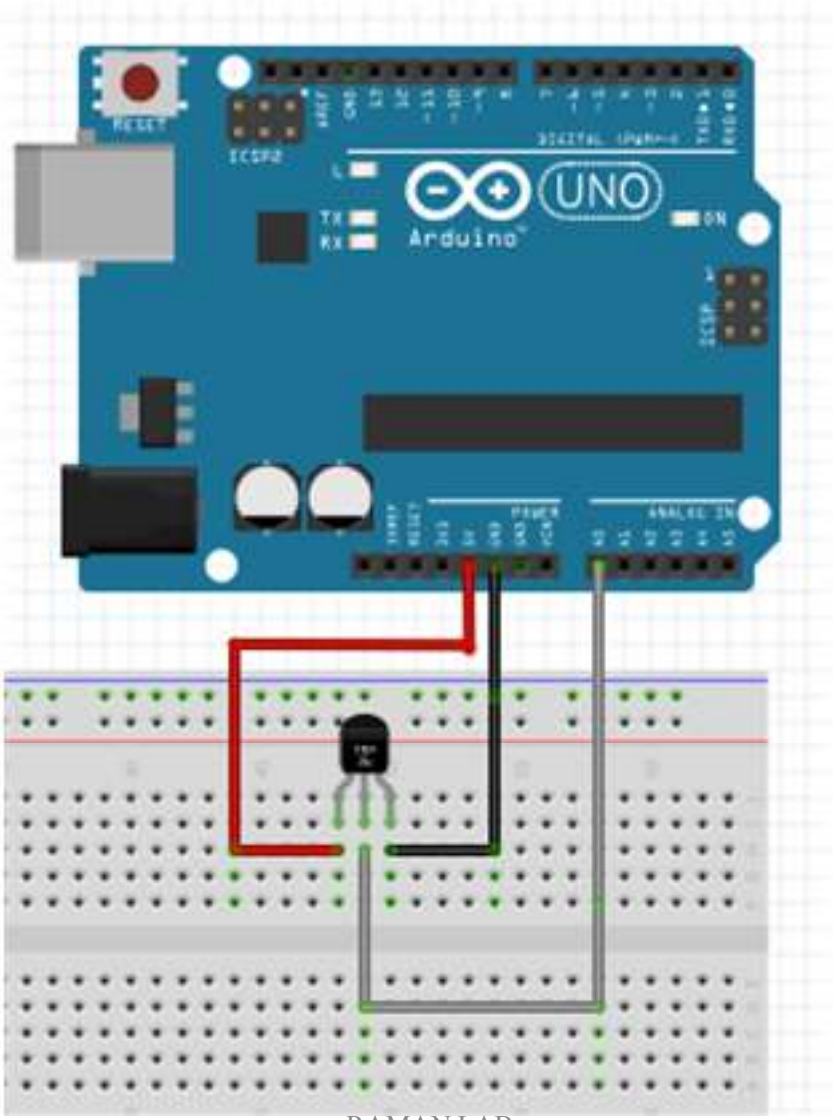
Circuit Schematic



Design it!


RESET
3.3V
5V
GND
GND
VIN
POWER
A0
A1
A2
A3
A4
A5
ANALOG IN
WWW.ARDUINO.CC
MADE IN ITALY
RESET
ON
ARDUINO
UNO
SMD EDITION
DIGITAL (PWM~)
TX
RX
AREF
GND
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
1
0


a b c d e
f g h i j


RESET
ICSP2
UNO
Arduino
ON
ICSP
POWER
ANALOG IN

Create the Sketch

```
const int temperaturePin = 0;
void setup() {
Serial.begin(9600);
}void loop() {
float voltage, degreesC, degreesF;
voltage = getVoltage(temperaturePin);
degreesC = (voltage - 0.5) * 100.0;
degreesF = degreesC * (9.0/5.0) +
32.0;
Serial.print("voltage: ");
Serial.print(voltage);
Serial.print("  deg C: ");
Serial.print(degreesC);
Serial.print("  deg F: ");
Serial.println(degreesF);
delay(1000);}
```

```
float getVoltage(int pin)
{
  return(analogRead(pin) *
0.004882814);
}
```


THANK YOU!