Main Features of Arduino UNO:

The arduino UNO is a microcontroller based on the ATmega328P.
It has 14 digital IO pins 6 of which can be used as PWM outputs
A 16MHZ ceramic resonator
A USB connection Type B male
A power jack which takes in DC power of barrel type male connector
An ICP header and a reset button

It contains everything required to power the MCU and can be powered either from the 5V out of a USB connection from a computer or the 5V ac to dc adapter supply or even can run on batteries.

The best part about using an UNO is it's relative inexpensiveness while also being extremely safe and easy to tinker with.

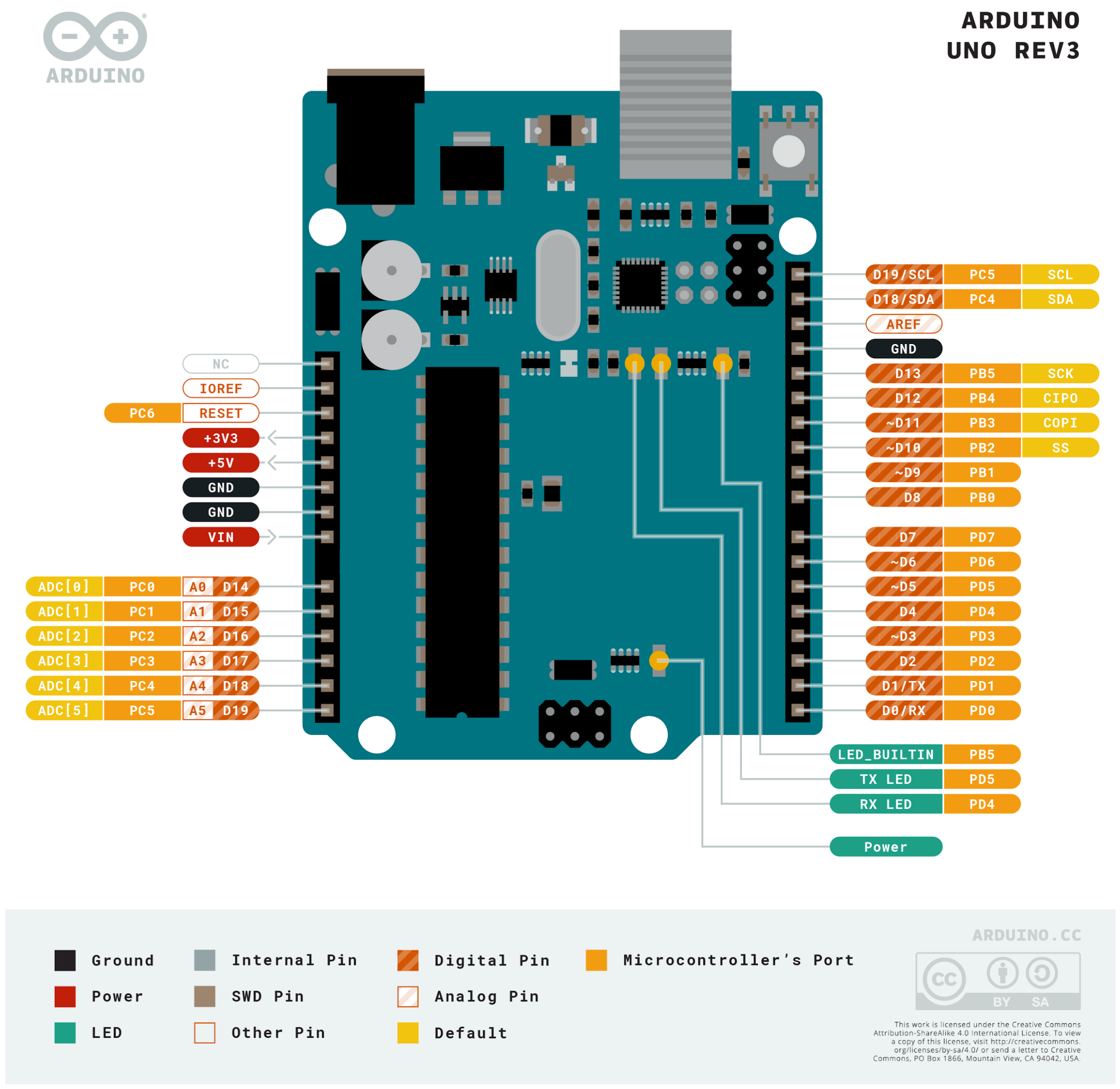
Tech Specs

Here you will find the technical specifications for the Arduino UNO R3.

Board	**Name**	Arduino UNO R3
	SKU	A000066
Microcontroller	ATmega328P	
USB connector	USB-B	
Pins	**Built-in LED Pin**	13
	Digital I/O Pins	14
	Analog input pins	6
	PWM pins	6
Communication	**UART**	Yes
	I2C	Yes
	SPI	Yes

Power	**I/O Voltage**	5V
	Input voltage (nominal)	7-12V
	DC Current per I/O Pin	20 mA
	Power Supply Connector	Barrel Plug
Clock speed	**Main Processor**	ATmega328P 16 MHz
	USB-Serial Processor	ATmega16U2 16 MHz
Memory	**ATmega328P**	2KB SRAM, 32KB FLASH, 1KB EEPROM
Dimensions	**Weight**	25 g
	Width	53.4 mm
	Length	68.6 mm

Arduino UNO REV3 Pinout



Ground - negative terminal of power
Power - 5V positive terminal
LED - Indicator LEDs which are builtin for debugging or some form of indication
TX LED - blinks when it send transmissions like a new program or a message
RX LED - blinks when it receives a new program or a message

LED_BUILTIN - Builtin LED programmers can use to tinker with
SWD - Serial Wire Debugger used to debug code by setting breakpoints and inspecting variables mid execution
Digital Pin - IO pins that can only read/ write HIGH and LOW signals
Analog Pin - IO pins that can read and write analog values with 10bits of resolution or 1024 voltage values
Pins with ~ before are capable of giving out PWM output
Tx and rx pins - these pins are used in wired serial communication with the arduino controller, it can be used to control other arduino boards or interface with communication peripherals
SPI - SS/SCK/MISO/MOSI pins are the dedicated pins for SPI communication. They can be found on digital pins 10-13 of the Arduino Uno and on the ICSP headers.

- MISO (Master In Slave Out) - A line for sending data to the Master device
- MOSI (Master Out Slave In) - The Master line for sending data to peripheral devices
- SCK (Serial Clock) - A clock signal generated by the Master device to synchronize data transmission.

I2C - SCL/SDA pins are the dedicated pins for I2C communication. On the Arduino Uno they are found on Analog pins A4 and A5.

- SCL is the clock line which is designed to synchronize data transfers.
- SDA is the line used to transmit data.

Each device on the I2C bus has a unique address, up to 255 devices can be connected on the same bus.

Aref - Reference voltage for the analog inputs.

Interrupt - INT0 and INT1. Arduino Uno has two external interrupt pins.

External Interrupt - An external interrupt is a system interrupt that occurs when outside interference is present. Interference can come from the user or other hardware devices in the network. Common uses for these interrupts in Arduino are reading the frequency a square wave generated by encoders or waking up the processor upon an external event.

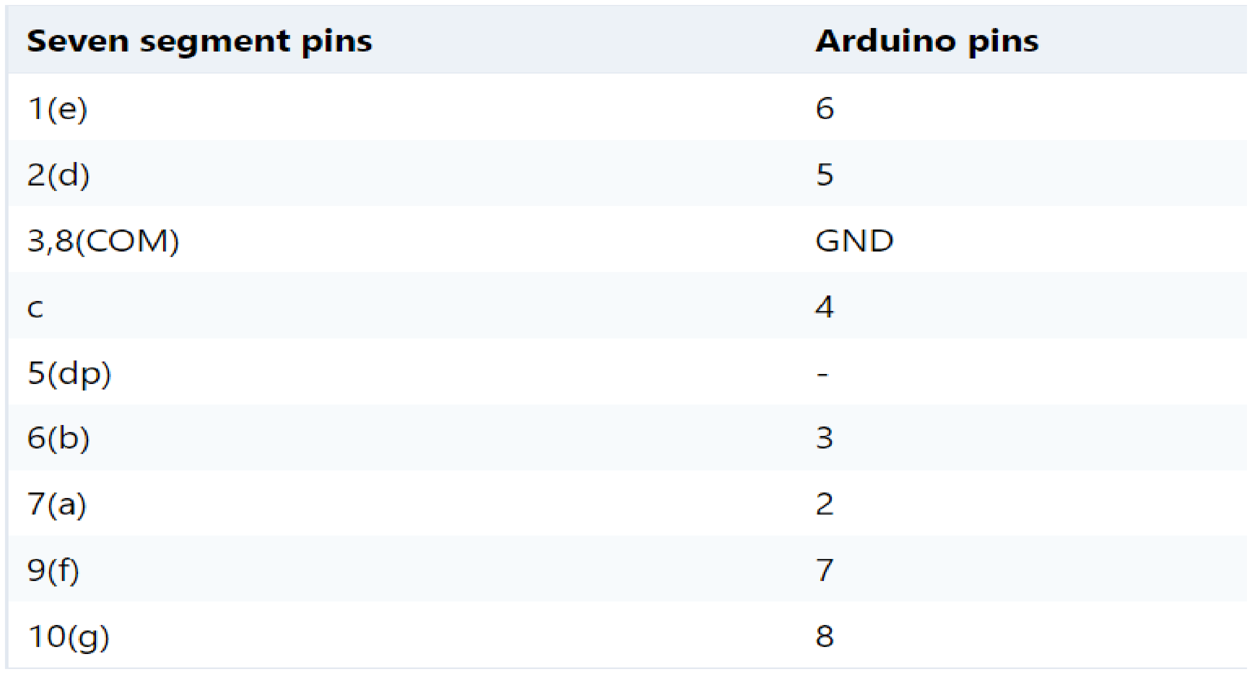
Arduino has two forms of interrupt:

- External
- Pin Change

There are two external interrupt pins on the ATmega168/328 called INT0 and INT1. both INT0 and INT1 are mapped to pins 2 and 3. In contrast, Pin Change interrupts can be activated on any of the pins.

ICSP stands for In-Circuit Serial Programming. The name originated from In-System Programming headers (ISP). Manufacturers like Atmel who work with Arduino have developed their own in-circuit serial programming headers. These pins enable the user to program the Arduino boards' firmware. There are six ICSP pins available on the Arduino board that can be hooked to a programmer device via a programming cable.

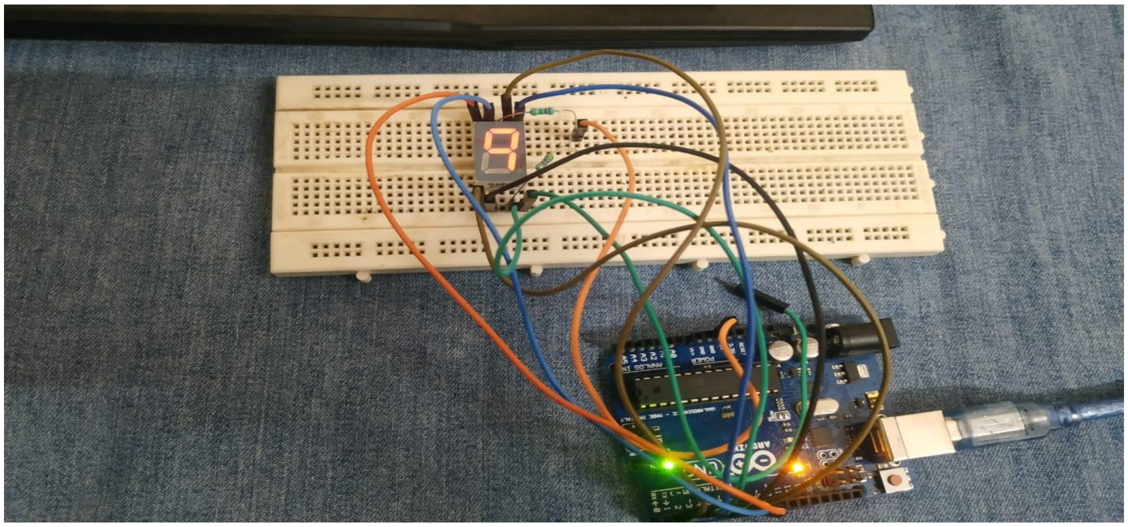
Seven Segment Display



Seven segment pins	Arduino pins
1(e)	6
2(d)	5
3,8(COM)	GND
c	4
5(dp)	-
6(b)	3
7(a)	2
9(f)	7
10(g)	8

Code :

```
int num_array[10][7] = {  { 1,1,1,1,1,1,0 },    // 0
                          { 0,1,1,0,0,0,0 },    // 1
                          { 1,1,0,1,1,0,1 },    // 2
                          { 1,1,1,1,0,0,1 },    // 3
                          { 0,1,1,0,0,1,1 },    // 4
                          { 1,0,1,1,0,1,1 },    // 5
                          { 1,0,1,1,1,1,1 },    // 6
                          { 1,1,1,0,0,0,0 },    // 7
                          { 1,1,1,1,1,1,1 },    // 8
                          { 1,1,1,0,0,1,1 }};   // 9

//function header
void Num_Write(int);

void setup()
{
  // set pin modes
  pinMode(2, OUTPUT);
  pinMode(3, OUTPUT);
  pinMode(4, OUTPUT);
  pinMode(5, OUTPUT);
  pinMode(6, OUTPUT);
  pinMode(7, OUTPUT);
  pinMode(8, OUTPUT);


}

void loop()
{

  //counter loop

  for (int counter = 10; counter > 0; --counter)
  {
   delay(1000);
   Num_Write(counter-1);
  }
  delay(3000);
}

// this functions writes values to the sev seg pins
void Num_Write(int number)
{
  int pin= 2;
  for (int j=0; j < 7; j++) {
   digitalWrite(pin, num_array[number][j]);
   pin++;
  }
}
```

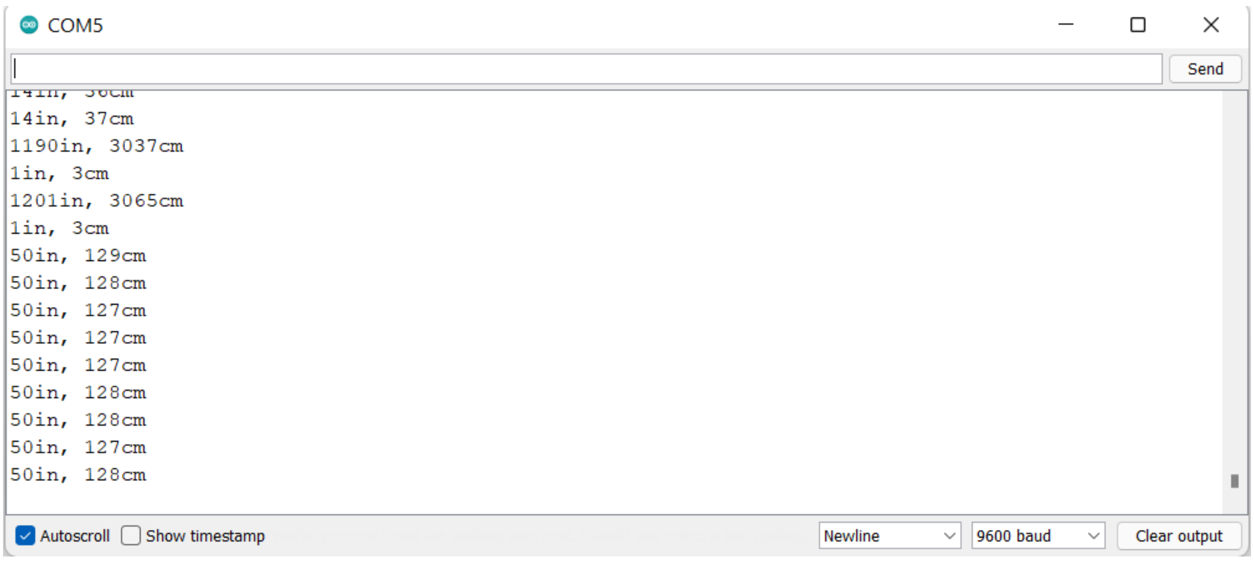


Ultrasonic Sensor

Code :

```
const int pingPin = 7; // Trigger Pin of Ultrasonic Sensor
const int echoPin = 6; // Echo Pin of Ultrasonic Sensor

void setup() {
  Serial.begin(9600); // Starting Serial Terminal
}

void loop() {
  long duration, inches, cm;
  pinMode(pingPin, OUTPUT);
  digitalWrite(pingPin, LOW);
  delayMicroseconds(2);
  digitalWrite(pingPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(pingPin, LOW);
  pinMode(echoPin, INPUT);
  duration = pulseIn(echoPin, HIGH);
  inches = microsecondsToInches(duration);
  cm = microsecondsToCentimeters(duration);
  Serial.print(inches);
  Serial.print("in, ");
  Serial.print(cm);
  Serial.print("cm");
  Serial.println();
  delay(100);
}

long microsecondsToInches(long microseconds) {
  return microseconds / 74 / 2;
}

long microsecondsToCentimeters(long microseconds) {
  return microseconds / 29 / 2;
}
```

Output:



```
14in, 37cm
1190in, 3037cm
1in, 3cm
1201in, 3065cm
1in, 3cm
50in, 129cm
50in, 128cm
50in, 127cm
50in, 127cm
50in, 127cm
50in, 128cm
50in, 128cm
50in, 127cm
50in, 128cm
```

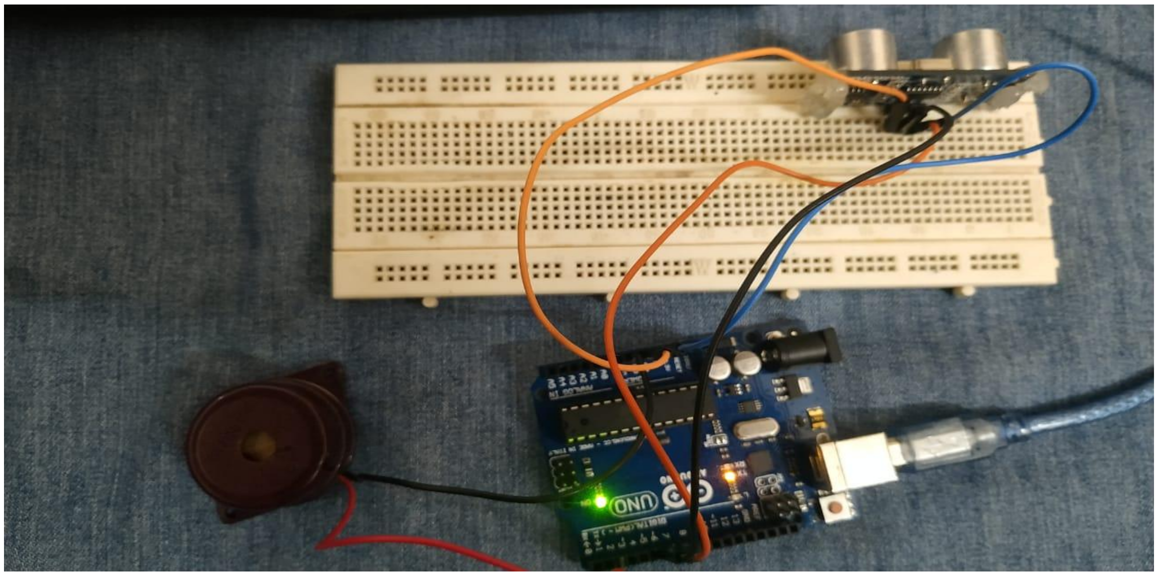
Detect Object Application :

Code :

```
#define pirPin 2
int calibrationTime = 30;
long unsigned int lowIn;
long unsigned int pause = 5000;
boolean lockLow = true;
boolean takeLowTime;
int pirStat = 0;
```

```
const int buzzer = 9; //buzzer to arduino pin 9


void setup(){
 Serial.begin(9600); // Starting Serial Terminal
 pinMode(buzzer, OUTPUT); // Set buzzer - pin 9 as an output
 pinMode(pirPin, INPUT);

}



const int pingPin = 7; // Trigger Pin of Ultrasonic Sensor
const int echoPin = 6; // Echo Pin of Ultrasonic Sensor



long ultrasonic() {
  long duration, inches, cm;
  pinMode(pingPin, OUTPUT);
  digitalWrite(pingPin, LOW);
  delayMicroseconds(2);
  digitalWrite(pingPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(pingPin, LOW);
  pinMode(echoPin, INPUT);
  duration = pulseIn(echoPin, HIGH);
  cm = microsecondsToCentimeters(duration);
  Serial.print(cm);
  Serial.print("cm");
  Serial.println();
  delay(100);
  return cm;
}

long microsecondsToCentimeters(long microseconds) {
  return microseconds / 29 / 2;
}
```

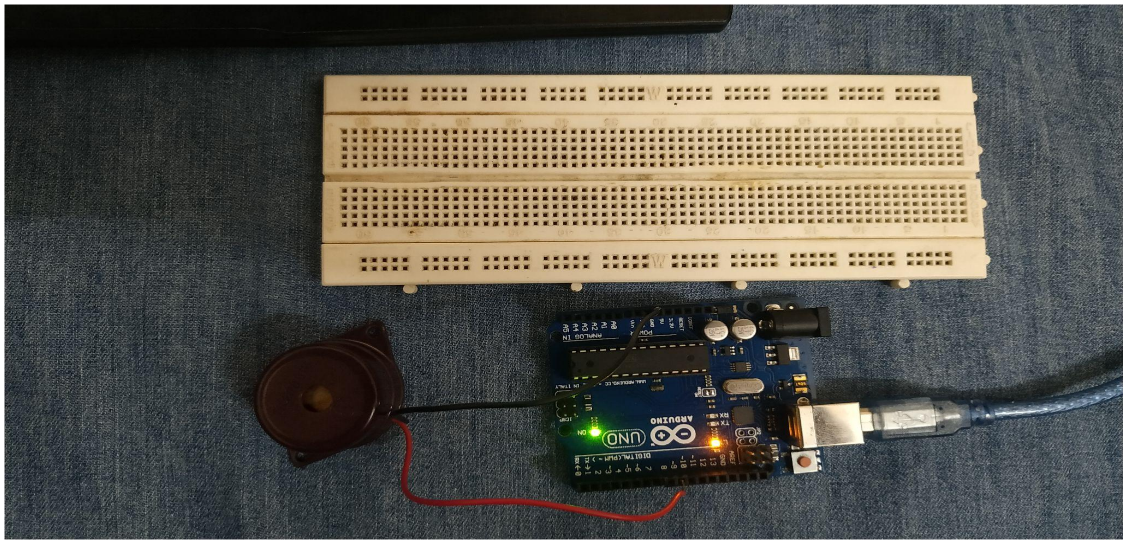
```
void loop() {

   if(ultrasonic()<30){
     tone(buzzer, 1000); // Send 1KHz sound signal...
      digitalWrite(pirPin,HIGH);
   }
   else{
    noTone(buzzer);
     digitalWrite(pirPin, LOW);
   }
}
```



Buzzer :

Code :

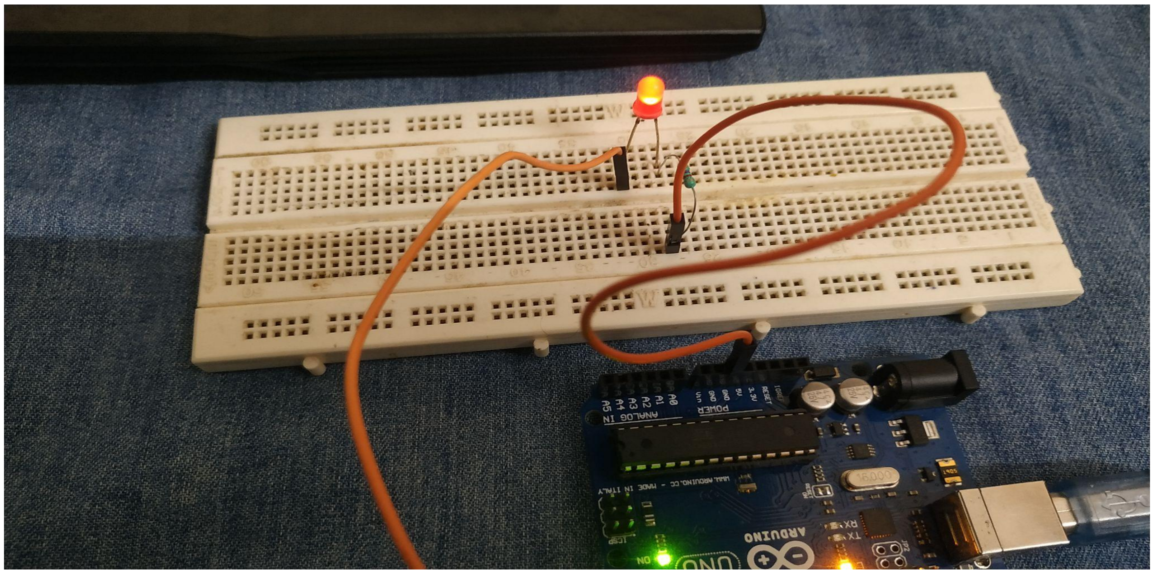
```
const int buzzer = 9; //buzzer to arduino pin 9


void setup(){

  pinMode(buzzer, OUTPUT); // Set buzzer - pin 9 as an output

}

void loop(){
  tone(buzzer, 1000); // Send 1KHz sound signal...
  delay(1000);        // ...for 1 sec
  noTone(buzzer);     // Stop sound...
  delay(1000);        // ...for 1sec

}
```



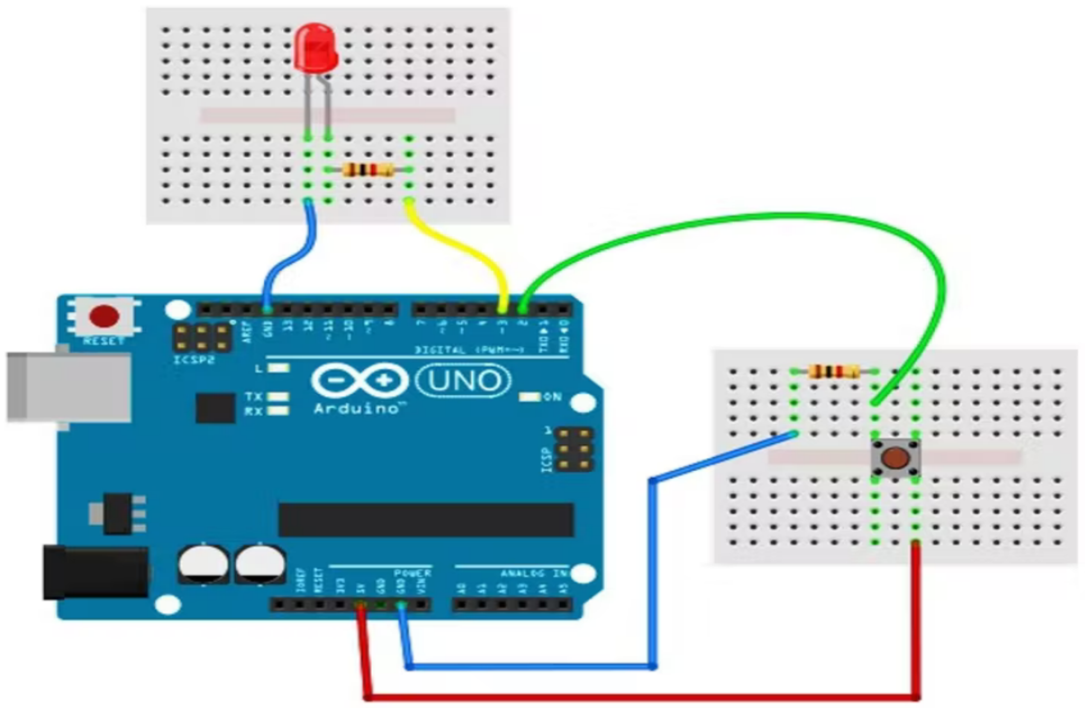
Blinking LED

Code :

```
/*
  Blink
  Turns on an LED on for one second, then off for one second, repeatedly.
*/

// the setup function runs once when you press reset or power the board

void setup() {  // initialize digital pin 13 as an output.
  pinMode(2, OUTPUT);
}

// the loop function runs over and over again forever

void loop() {
  digitalWrite(2, HIGH); // turn the LED on (HIGH is the voltage level)
  delay(1000); // wait for a second
  digitalWrite(2, LOW); // turn the LED off by making the voltage LOW
  delay(1000); // wait for a second
}
```

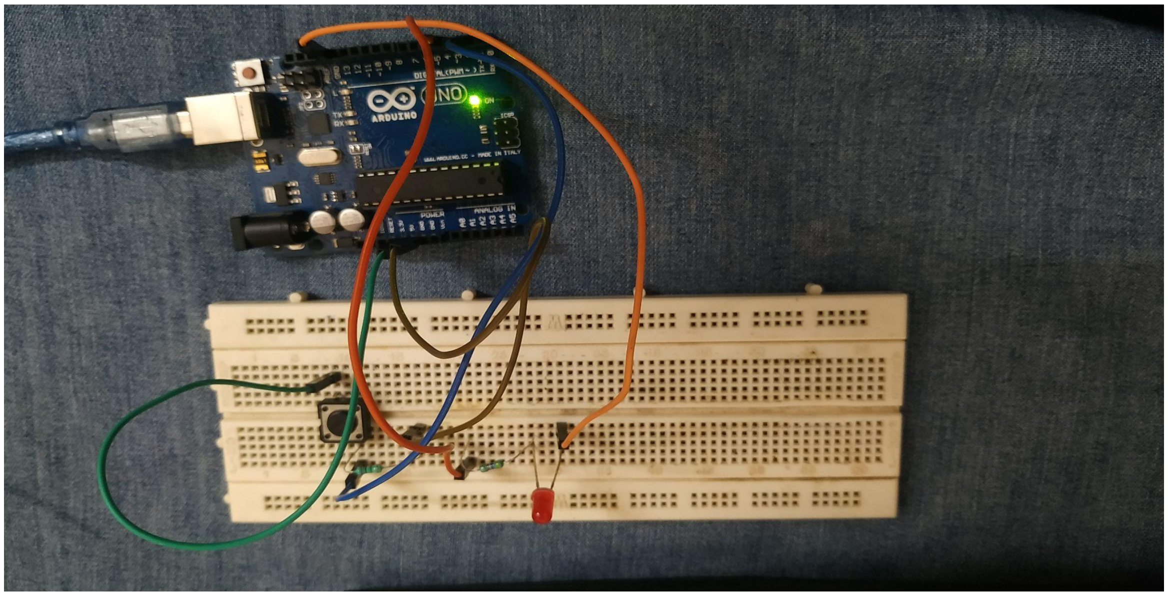


Push Button LED :



Code :

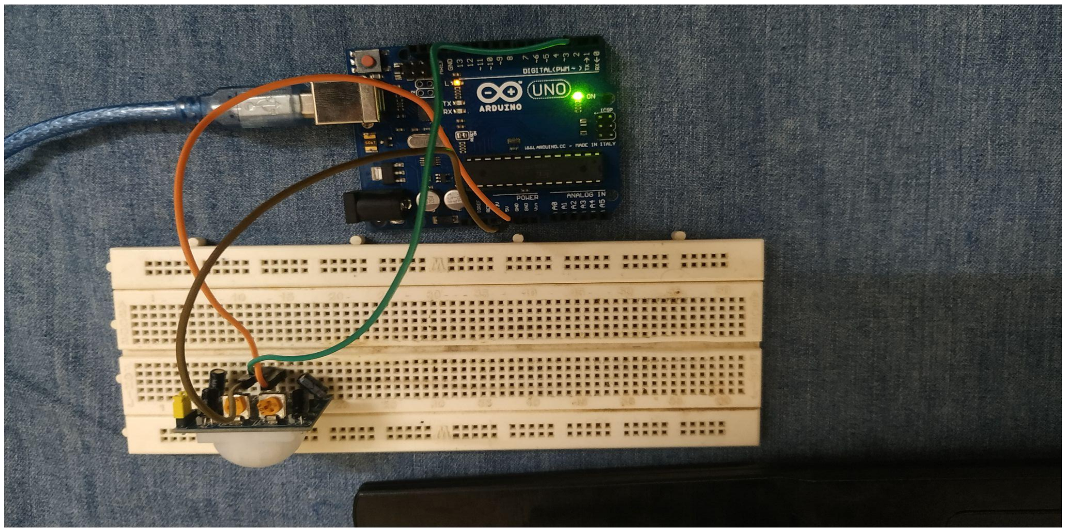
```
const int BUTTON = 2;
const int LED = 3;
int BUTTONstate = 0;

void setup()
{
  pinMode(BUTTON, INPUT);
  pinMode(LED, OUTPUT);
}

void loop()
{
  BUTTONstate = digitalRead(BUTTON);
  if (BUTTONstate == HIGH)
  {
    digitalWrite(LED, HIGH);
  }
  else{
    digitalWrite(LED, LOW);
  }
}
```

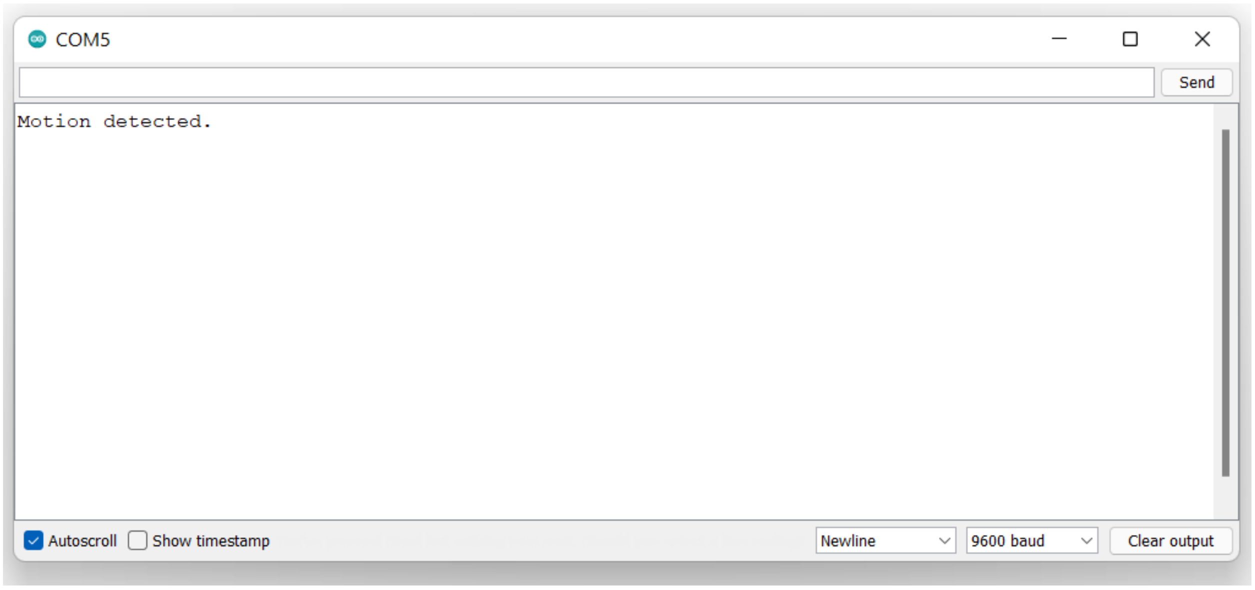


PIR Sensor :

Code :

```
#define pirPin 2
int calibrationTime = 30;
long unsigned int lowIn;
long unsigned int pause = 5000;
boolean lockLow = true;
boolean takeLowTime;
int PIRValue = 0;

void setup() {
  Serial.begin(9600);
  pinMode(pirPin, INPUT);
}

void loop() {
  PIRSensor();
}

void PIRSensor() {
    if(digitalRead(pirPin) == HIGH) {
      if(lockLow) {
        PIRValue = 1;
        lockLow = false;
        Serial.println("Motion detected.");
        delay(50);
      }
      takeLowTime = true;
    }
    if(digitalRead(pirPin) == LOW) {
      if(takeLowTime){
        lowIn = millis();takeLowTime = false;
      }
      if(!lockLow && millis() - lowIn > pause) {
        PIRValue = 0;
        lockLow = true;
        Serial.println("Motion ended.");
        delay(50);
      }
    }
  }
```



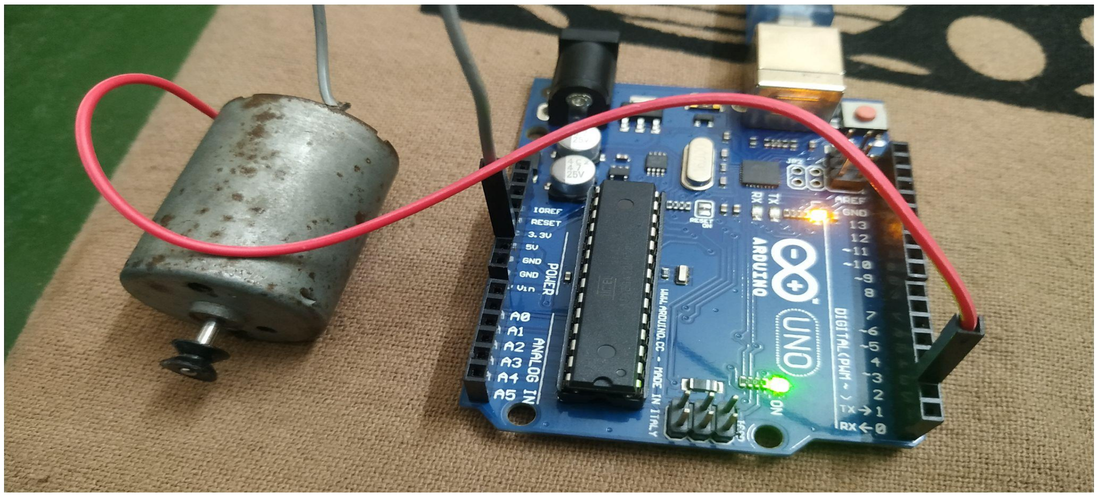


```
Motion detected.
```

DC Motor :

Code :

```
/*
  Blink
  Turns on an LED on for one second, then off for one second, repeatedly.
*/

// the setup function runs once when you press reset or power the board

void setup() {  // initialize digital pin 13 as an output.
  pinMode(2, OUTPUT);
}

// the loop function runs over and over again forever

void loop() {
  digitalWrite(2, HIGH); // turn the LED on (HIGH is the voltage level)
  delay(5000); // wait for a second
  digitalWrite(2, LOW); // turn the LED off by making the voltage LOW
  delay(5000); // wait for a second
}
```


ARDUINO
UNO
POWER
ANALOG IN
DIGITAL (PWM ~)
ON
WWW.ARDUINO.CC - MADE IN ITALY