Internet of Things Lab Manual

Semester VI

Department of CSE (AIML/CSBS)
Institute of Engineering & Management, SaltLake

Institute of Engineering
& Management, SaltLake

Contents

INTRODUCTION TO INTERNET OF THINGS AND ARDUINO

Internet of Things (IOT)

IOT stands for “Internet of Things”. The IOT is a name for the vast collection of “things” that are being networked together in the home and workplace (up to 20 billion by 2020 according to Gardner, a technology consulting firm).

Characteristics of the IOT

Actuators

- IOT devices that do something. Lock doors, beep, turn lights on, or turn the TV on

Sensing

- IOT devices sense something about their environment

Networking

- These IOT devices talk to one another (M2M communication) or to servers located in the local network or on the Internet. Being on the network allows the device the common ability to consume and produce data.

EXPERIMENT 1: LED WITH PUSH BUTTON

DEPARTMENT OF CSE (AIML/CSBS)

NAME OF THE STUDENT	
ROLL NO.	
DATE OF EXPERIMENT	
DATE OF SUBMISSION	
GRADE/MARKS	
TEACHER'S SIGNATURE	

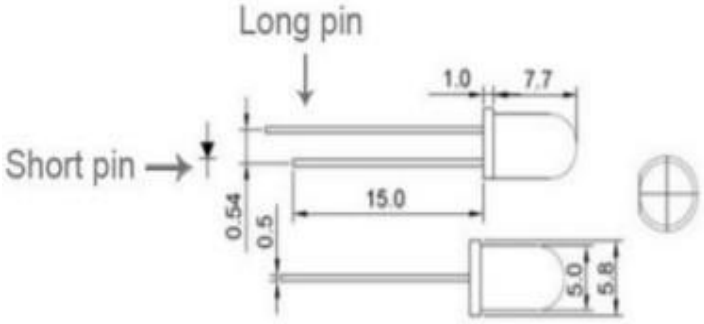
CONTROLLING THE LIGHT EMITTING DIODE (LED) WITH A PUSH BUTTON

Push-button is a very simple mechanism which is used to control electronic signal either by blocking it or allowing it to pass. This happens when mechanical pressure is applied to connect two points of the switch together. Push buttons or switches connect two points in a circuit when pressed. When the push-button is released, there is no connection between the two legs of the push-button. Here it turns on the built-in LED on pin 11 when the button is pressed. The LED stays ON as long as the button is being pressed

Hardware Required

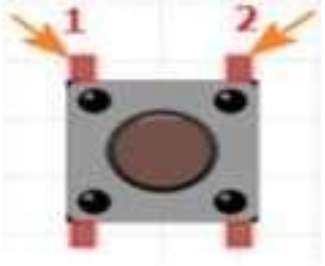
Arduino UNO,	Breadboard,	220Ω resistor,	Push Button,
USB Cable,	Jumper wires,	10KΩ resistor,	LED

LED Specifications



Long pin	+5V
Short pin	GND

Push Button



Size	6 x 6 x 5mm
Temperature	-30 ~ +70 °C

EXPERIMENT 1: LED WITH PUSH BUTTON

Steps of working

- Insert the push button into your breadboard and connect it to the digital pin 7(D7) which act as INPUT.
- Insert the LED into the breadboard. Attach the positive leg (the longer leg) to digital pin 11 of the Arduino Uno, and the negative leg via the 220-ohm resistor to GND. The pin D11 is taken as OUTPUT.
- The 10kΩ resistor used as PULL-UP resistor and 220 Ω resistors is used to limit the current through the LED.
- Upload the code
- Press the push-button to control the ON state of LED.

Algorithm

- This sketch works by setting pin D7 as for the push button as INPUT and pin 11 as an OUTPUT to power the LED.
- The initial state of the button is set to OFF.
- After that run a loop that continually reads the state from the pushbutton and sends that value as voltage to the LED. The LED will be ON accordingly.

Precautions:

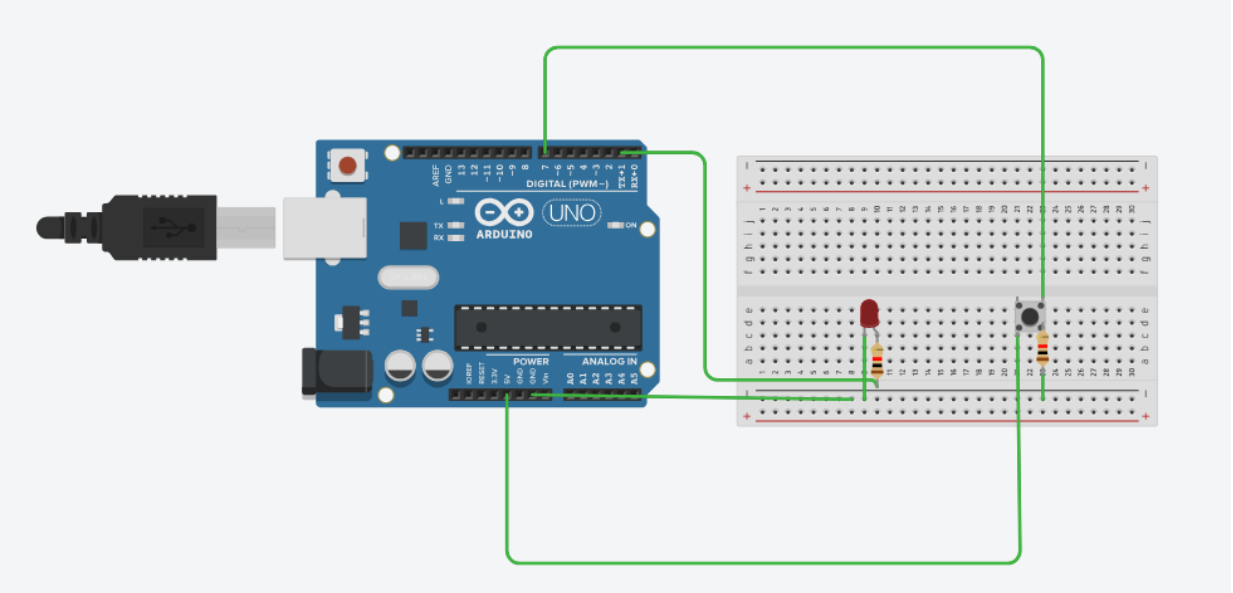
- The pushbutton is square so it is important to set it appropriately on breadboard.
- While making the connections make sure to use a pull-down resistor because directly connecting two points of a switch to the circuit will leave the input pin in floating condition and circuit may not work according to the program.
- It is very important to set pinMode() as OUTPUT first before using digitalWrite() function on that pin.
- If you do not set the pinMode() to OUTPUT, and connect an LED to a pin, when calling digitalWrite(HIGH), the LED may appear dim.

Observations

Sr no.	Push Button State	LED State
1		
2		

EXPERIMENT 1: LED WITH PUSH BUTTON

Code & Circuit Diagram

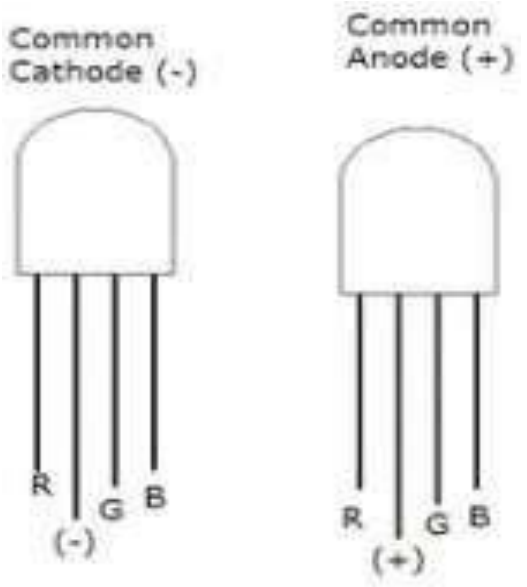


Code:

```
void setup() {
  pinMode(1, OUTPUT);
  pinMode(7, INPUT);
}
void loop() {
 if (digitalRead(7)==HIGH) {
  digitalWrite(1, HIGH);

}
else {
  digitalWrite(1, LOW);

  }
}
```

INTERFACING THE RGB LED WITH THE ARDUINO

There are actually two types of RGB LED's; the common cathode one and the common anode one. In the common cathode RGB led, the cathode of all the LED's is common and we give PWM signals to the anode of LED's while in the common anode RGB led, the anode of all the LED's is common and we give PWM signals to the cathode of LED's. Inside the RGB led, there are three more LED's. So, by changing the brightness of these LED's, we can obtain many other colors. To change brightness of RGB led, we can use the PWM pins of Arduino. The PWM pins will give signal different duty cycles to the RGB led to obtain different colors.



Hardware Required

Arduino UNO,	Breadboard,	3 X 220 Ω / 330 Ω resistor
USB Cable,	Jumper wires,	LED

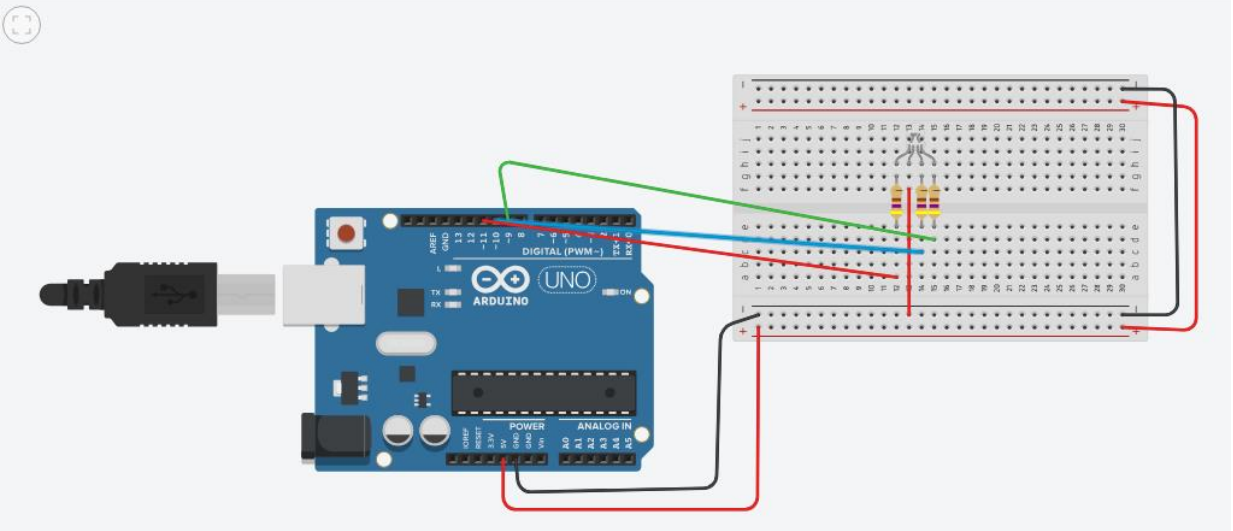
Steps of working

- Insert the RGB LED into your breadboard and connect its cathode pin to the GND of the Arduino.
- Insert the LED into the breadboard. Attach Red pin to pin 8, Green pin to pin 9 and Blue pin to pin 10 of the Arduino via the 220-ohm resistor, and the negative leg to GND.
- Upload the code
- Observe the changes in the color of the RGB LED.

Observations

Sr no.	Time (ms)	Color of LED
1		
2		
3		

Code & Circuit Diagram

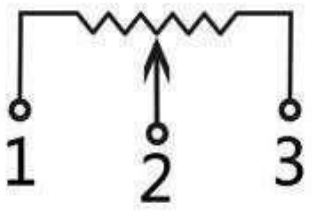


Code:

```
void setup()
{
 pinMode(11, OUTPUT);
 pinMode(10, OUTPUT);
 pinMode(9, OUTPUT);
}

void loop()
{
 analogWrite(11, 255);
 analogWrite(10, 0);
 analogWrite(9, 0);
 delay(1000); // Wait for 1000 millisecond(s)
 analogWrite(9, 51);
 analogWrite(11, 51);
 analogWrite(10, 255);
 delay(1000); // Wait for 1000 millisecond(s)
 analogWrite(10, 102);
 analogWrite(9, 255);
 analogWrite(11, 255);
 delay(1000); // Wait for 1000 millisecond(s)
}
```

CONTROLLING THE LED BLINK RATE WITH POTENTIOMETER INTERFACING WITH ARDUINO

A potentiometer is a variable resistor with a knob that allows altering the resistance of the potentiometer. The potentiometer manipulates a continuous analog signal, which represents physical measurements. The potentiometer is used with Arduino to control the blink rate of the LED. Potentiometer is adjustable resistor.



Hardware Required

Arduino UNO,	Breadboard,	220Ω resistor,	5mm LED,
USB Cable,	Jumper wires,	10KΩ potentiometer	

Steps of working

- Insert the potentiometer into your breadboard and connect its center pin to the analog pin A2 and the remaining pin to GND on the breadboard.
- Insert the LED into the breadboard. Attach the positive leg (the longer leg) to pin 13 of the Arduino via 220-ohm resistor, and the negative leg to GND.
- Upload the code as given below.
- Turn the potentiometer to control the brightness of the LED and move the position of pin 2 by rotating the knob, changing the resistance value from pin 2 to both ends.
- Observe the changes in the blinking rate of the LED.

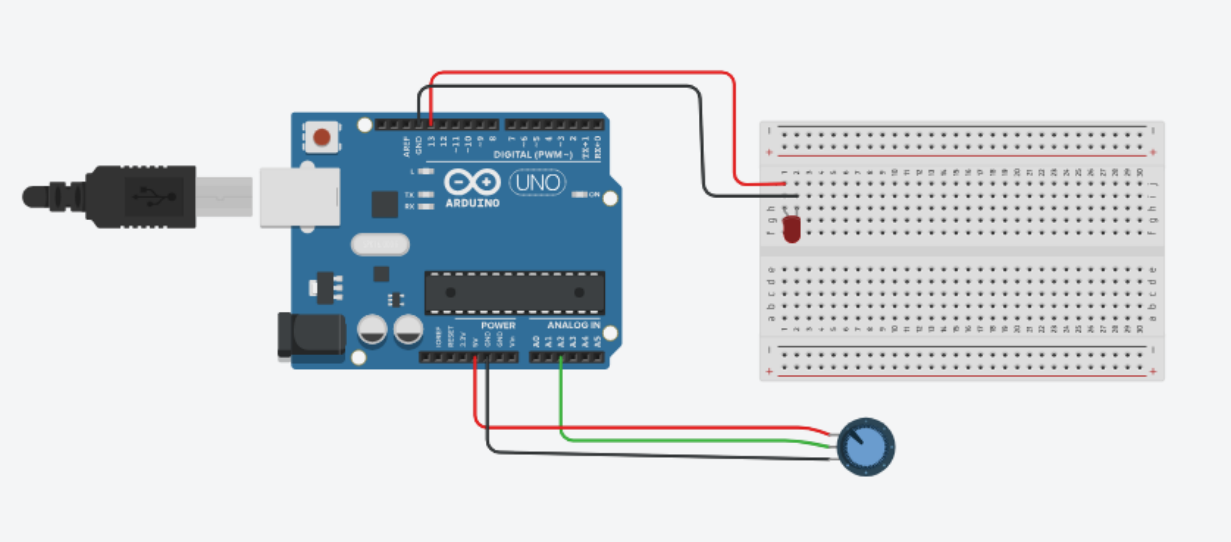
Algorithm

- This sketch works by setting pin A2 as for the potentiometer and pin 9 as an OUTPUT to power the LED
- After that run a loop that continually reads the value from the potentiometer and sends that value as voltage to the LED
- The voltage value is between 0–5 volts, and the brightness of the LED will vary accordingly

Observations

Sr no.	Voltage	Light Intensity
1		
2		
3		
4		
5		

Code & Circuit Diagram



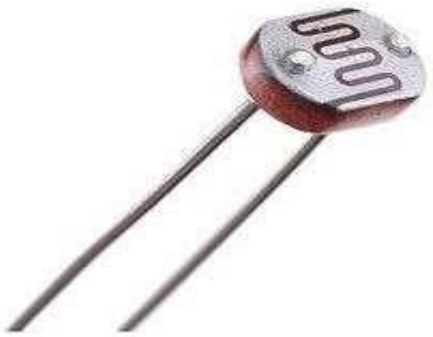
Code:

```
int potpin = A2;
int ledpin = 13;
int val;

void setup()
{
pinMode(ledpin, OUTPUT);
Serial.begin(9600);
}

void loop()
{
 val = analogRead(potpin);
```

```
Serial.println(val);
  digitalWrite(ledpin, HIGH);
  delay(val);
  digitalWrite(ledpin, LOW);
  delay(val);
}
```

DETECTION OF THE LIGHT USING PHOTO RESISTOR

A photo resistor or photocell is a light-controlled variable resistor made of a high resistance semiconductor. The resistance of a photo resistor decreases with increasing incident light intensity. A photo resistor can be applied in light-sensitive detector circuits, and light- and dark-activated switching circuits. It's also called light-dependent resistor (LDR).



Hardware Required

Arduino UNO,	Breadboard,	220Ω resistor,	Photo Resistor,
USB Cable,	Jumper wires,	10KΩ resistor,	LED

Steps of working

- Insert the photo resistor into your breadboard and connect its pin to the analog pin A0 and the remaining pin to supply on the breadboard.
- Insert the LED into the breadboard. Attach the positive leg (the longer leg) to pin 9 of the Arduino via the 220-ohm resistor, and the negative leg to GND.
- Insert the 10K-ohm resistor
- Upload the code
- Turn the photo resistor to ON the LED
- Observe the changes in the state of the LED.

Algorithm

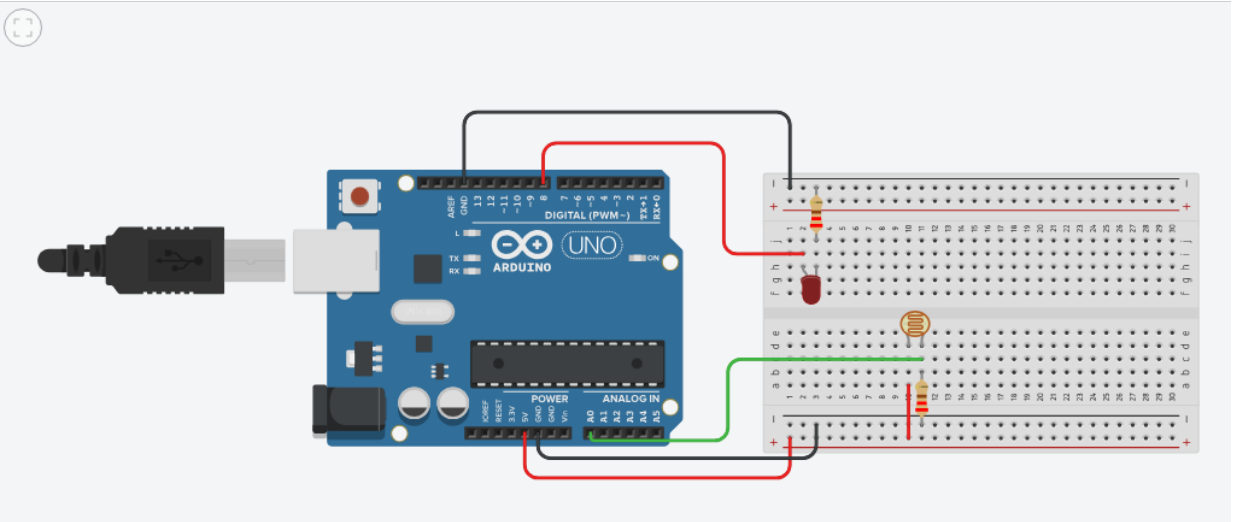
- This sketch works by setting pin A0 as for the photo sensor and pin 9 as an OUTPUT to power the LED.
- After that run a loop that continually reads value from the photo resistor and sends that value as voltage to the LED. The LED will vary accordingly.

EXPERIMENT 4: PHOTO RESISTOR

Observations

Sr no.	Light detected	LED state
1		
2		

Code & Circuit Diagram

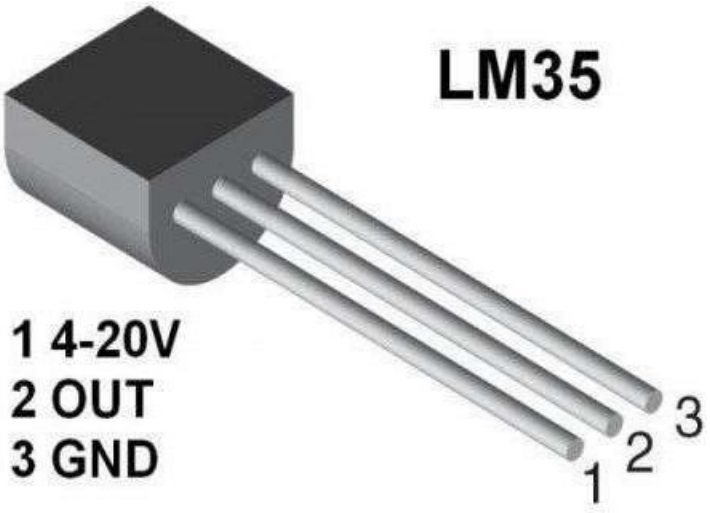


Code:

```
int sensorPin = A0;
int ledPin = 8;
int lightVal;
void setup()
{
pinMode(ledPin, OUTPUT);
Serial.begin(9600);
}
void loop() {
lightVal = analogRead(sensorPin);
Serial.println(lightVal);
if(lightVal < 100) {
digitalWrite(ledPin,HIGH);
}
else {
  digitalWrite(ledPin, LOW);
}
}
```

INTERFACING OF TEMPERATURE SENSOR WITH ARDUINO

The LM35 series are precision integrated-circuit temperature devices with an output voltage linearly proportional to the Centigrade temperature. LM35 is three terminal linear temperature sensors from National semiconductors. It can measure temperature from -55 degree Celsius to +150 degree Celsius. The voltage output of the LM35 increases 10mV per degree Celsius rise in temperature. LM35 can be operated from a 5V supply and the stand by current is less than 60uA.



Hardware Required

Arduino UNO, Breadboard, LM35, USB Cable, Jumper wires

Steps of working

- Insert the temperature sensor into your breadboard and connect its pin1 to the supply.
- Connect its center pin to the analog pin A0 and the remaining pin3 to GND on the breadboard.
- Upload the code as given below.
- Vary the temperature and read the voltage changes.
- Open the Arduino IDE's serial monitor to see the results.

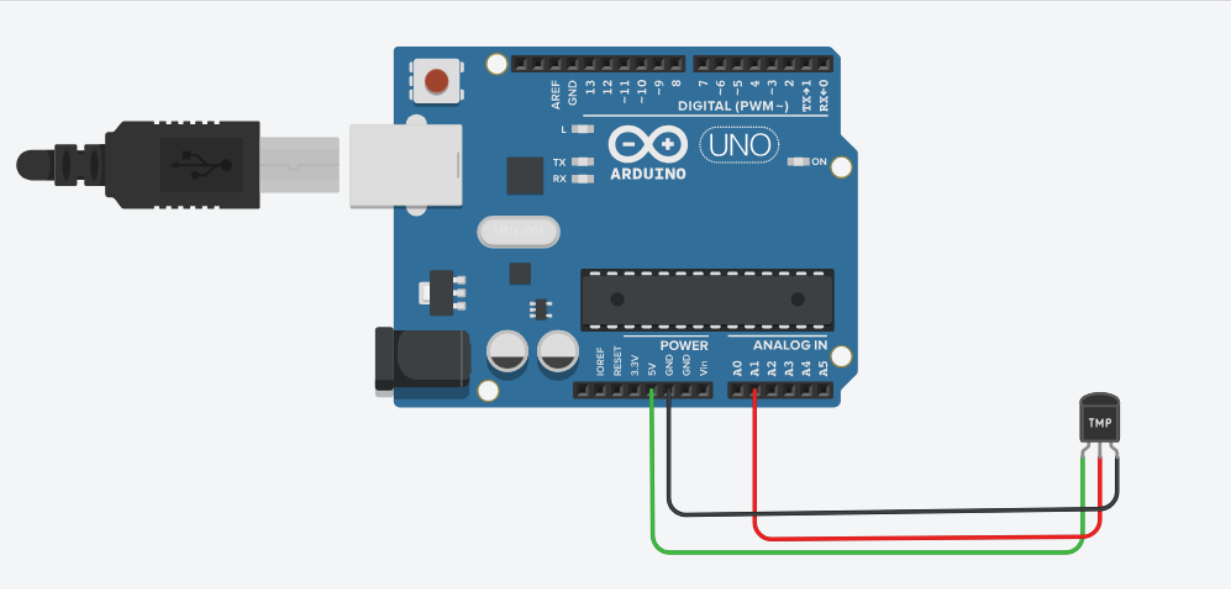
Algorithm

- This sketch works by setting pin A0 as for the temperature sensor
- After that run a loop that continually reads the value from the sensor and sends that value as voltage
- The voltage value is between 0–5 V, when temperature varies accordingly

Observations

Sr no.	Voltage	Temperature
1		
2		
3		
4		
5		

Code & Circuit Diagram



Code:

```
float temp;
int tempPin =A1;
void setup()
{
  Serial.begin(9600);
}
void loop()
{
 temp=analogRead(tempPin); //read analog voltage to temperature
  temp=temp*0.48828125;
  Serial.print("Temperature= ");
  Serial.print(temp);  //Display temp value
```

```
  Serial.print("*C");
  Serial.println();
  delay(1000);
}
```

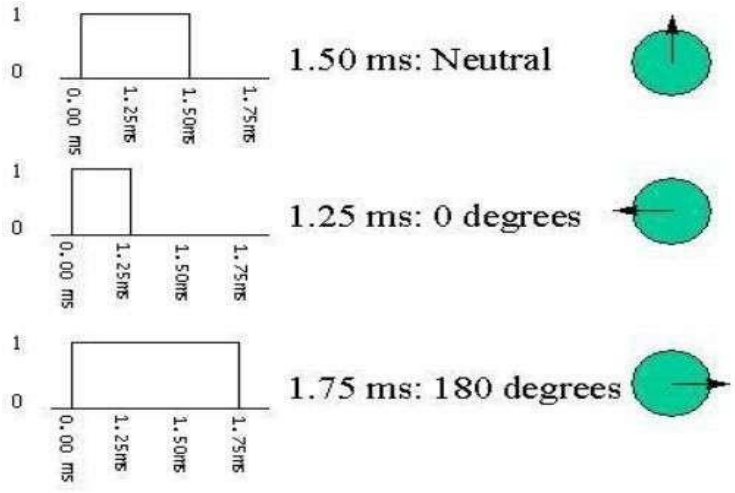
INTERFACING SERVO MOTOR WITH THE ARDUINO

A Servo Motor is a small device that has an output shaft. This shaft can be positioned to specific angular positions by sending the servo a coded signal. As long as the coded signal exists on the input line, the servo will maintain the angular position of the shaft. If the coded signal changes, the angular position of the shaft changes. Servo motors have three terminals – power, ground, and signal. The power wire is typically red, and should be connected to the 5V pin on the Arduino. The ground wire is typically black or brown as shown in figure

Specifications

GND	common ground for both the motor and logic
5V	positive voltage that powers the servo
Control	Input for the control system

The control wire is used to communicate the angle. The angle is determined by the duration of a pulse that is applied to the control wire. This is called Pulse Coded Modulation. The servo expects to see a pulse every 20 milliseconds (.02 seconds). The length of the pulse will determine how far the motor turns. A 1.5 millisecond pulse, for example, will make the motor turn to the 90-degree position (often called as the neutral position). If the pulse is shorter than 1.5 milliseconds, then the motor will turn the shaft closer to 0 degrees. If the pulse is longer than 1.5 milliseconds, the shaft turns closer to 180 degrees.



EXPERIMENT 6: SERVO MOTOR

Hardware Required

Arduino UNO, Breadboard, Servo Motor, USB Cable, Jumper wires

Steps of working

- The servo motor has a female connector with three pins. The darkest or even black one is usually the ground. Connect this to the Arduino GND.
- Connect the power cable that in all standards should be red to 5V on the Arduino.
- Connect the remaining line on the servo connector to a digital pin on the Arduino.
- Upload the code
- Observe the position of the shaft.

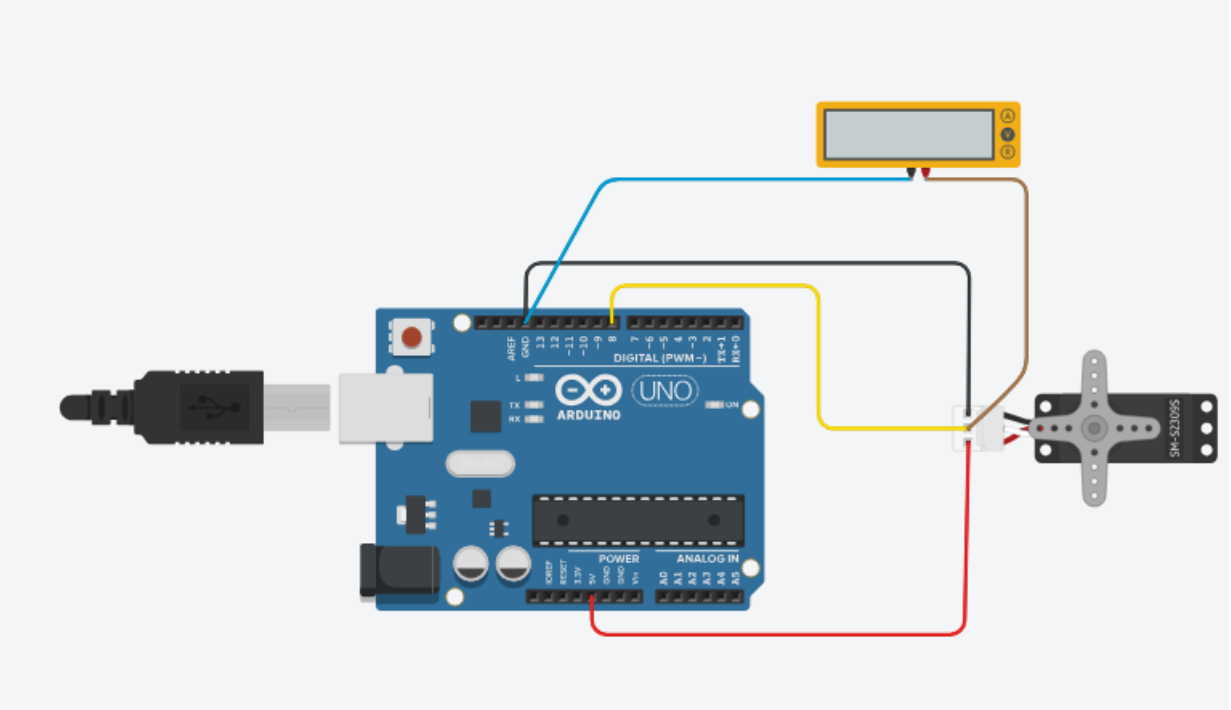
Algorithm

- This sketch works by setting pin D9 as for the control of servo motor
- After that run a loop that continually increment the value of the index of rotation angle and sends that value as voltage to the D9
- The voltage value is between 0–5 volts, and the rotation angle of the servo motor will vary accordingly.

Observations

Sr no.	Voltage	Position of Shaft
1		
2		
3		
4		
5		

Code & Circuit Diagram



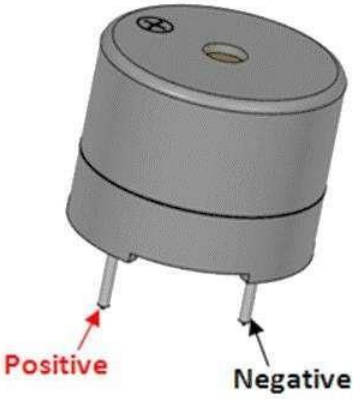
Code:

```
#include <Servo.h>;
Servo myservo;
int pos=0;
void setup()
{
myservo.attach(8);
}
void loop() {

for(pos=0;pos<=180;pos++)
{
myservo.write(pos);
delay (15);
}
delay (1000);
for (pos=180; pos>=0;pos--)
{
myservo.write (pos);

delay (15);
}
```

```
delay(1000);
}
```

INTERFACING OF THE ACTIVE BUZZER WITH THE ARDUINO

A piezo buzzer is a type of electronic device that's used to produce beeps and tones. The working principle of the device is piezoelectric effect. The main component of this device is a piezo crystal, which is a special material that changes shape when a voltage applied to it. The active buzzer will only generate sound when it will be electrified. It generates sound at only one frequency. This buzzer operates at an audible frequency of about 2 KHz



Specifications

Specification	Range
Voltage	3.3-5V
Frequency	2KHz

Pin Name	Description
Positive	Identified by (+) symbol or longer terminal lead. Can be powered by 6V DC
Negative	Identified by short terminal lead. Typically connected to ground of circuit

Hardware Required

Arduino UNO, Breadboard, Buzzer / piezo speaker,
USB Cable, Jumper wires, 220Ω resistor

Steps of working

- Connect the Supply wire (RED) of the buzzer to the Digital Pin 9 of the Arduino through a 100-ohm resistor.
- Connect the Ground wire (BLACK) of the buzzer to any Ground Pin on the Arduino.
- Upload the code
- Observe the changes in the pitch and volume of the buzzer.

EXPERIMENT 7: ACTIVE BUZZER

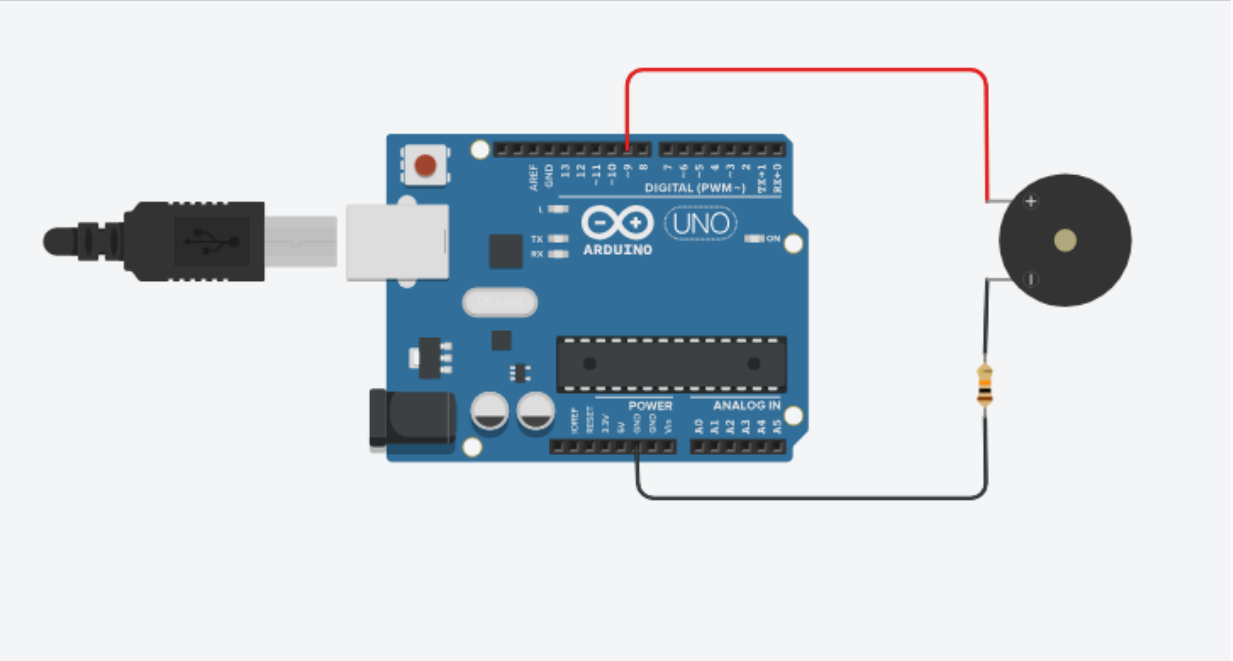
Algorithm

- This sketch works by setting pin D9 as for the control the buzzer
- After that run a loop that continually sends that value as voltage high or low to the D9 using the function digitalWrite( )
- The voltage value and the tone generated from the buzzer will vary accordingly.

Observations

Sr no.	Change the Value	Frequency of Tone
1		
2		
3		

Circuit Diagram



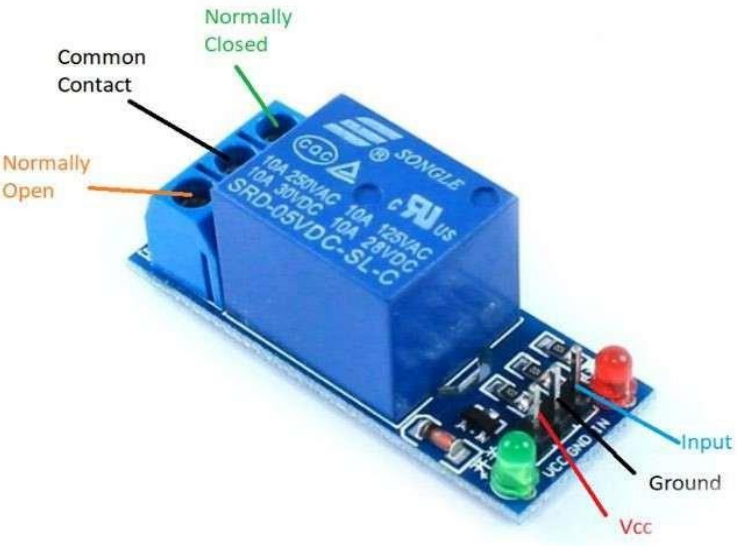
EXPERIMENT 7: ACTIVE BUZZER

Code

```
int buzzer = 9;
int i;
void setup()
{
pinMode (buzzer,OUTPUT);
}
void loop(){

for(i=0;i<80;i++)
{
digitalWrite(buzzer,HIGH);
delay(10);
digitalWrite(buzzer,LOW);
delay(10);
}

for(i=0;i<100;i++){
digitalWrite(buzzer,HIGH);
delay(20);
digitalWrite(buzzer,LOW);
delay(20);

}
}
```

INTERFACING OF THE RELAY WITH THE ARDUINO

Relay is an electromagnetic switch, which is controlled by small current, and used to switch ON and OFF relatively much larger current. Means by applying small current we can switch ON the relay which allows much larger current to flow



Hardware Required

Arduino UNO, Breadboard, 5V Relay, USB Cable, Jumper wires

Steps of working

- The relay module connected with three pins. We will connect the relay module with Arduino in the normally open state. The black one of relay is usually the ground. Connect this to the Arduino GND.
- Connect the red wire of relay module to 5V of the Arduino.
- Connect the signal pin of relay module to a digital pin 6 of the Arduino.
- Upload the code
- Observe the clicking sound of the relay that states the ON and OFF constantly.

Algorithm

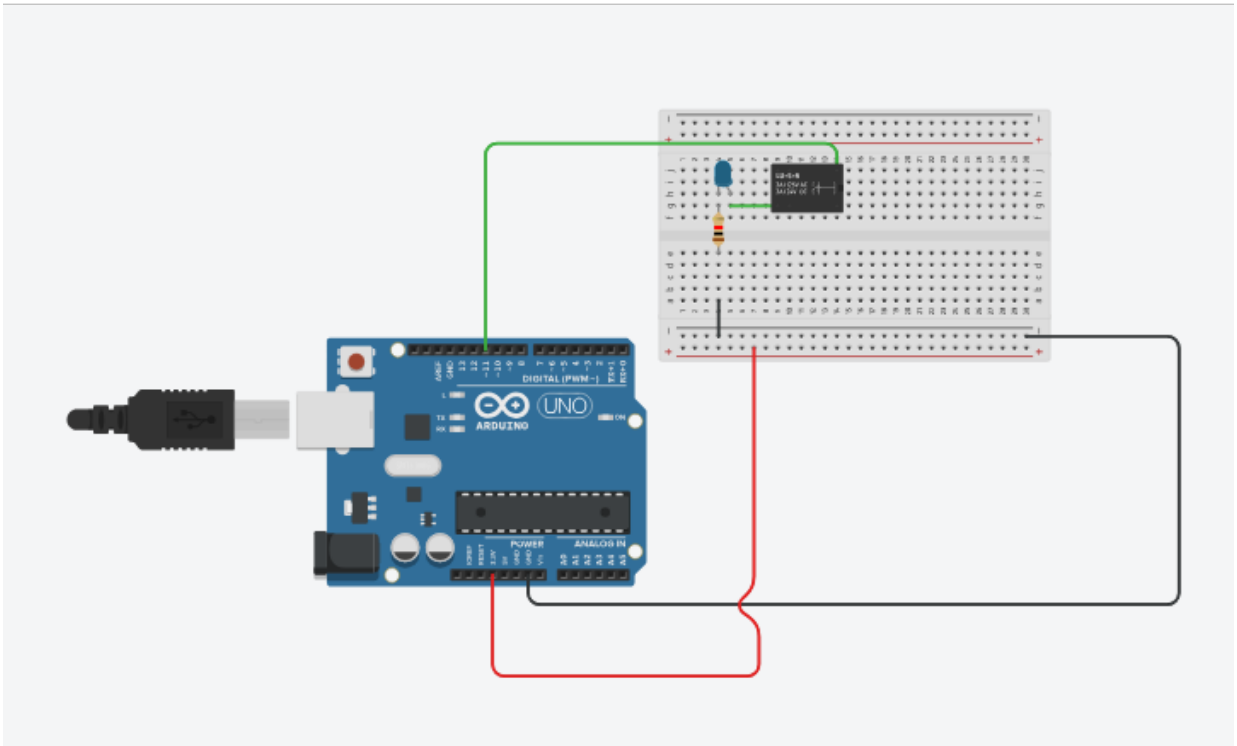
- This sketch works by setting 5V supply pin of Arduino as for the control of relay module
- After that run a loop that continually sends that value as voltage to the D6 with the delay given.

EXPERIMENT 8: RELAY

Observations

Sr no.	Delay	Relay Status
1		
2		

Code & Circuit Diagram



Code:

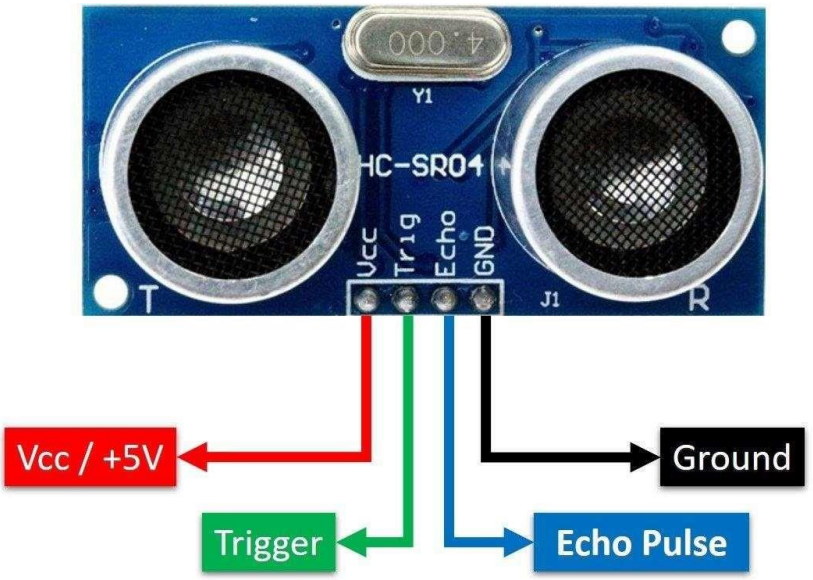
```
void setup()
{
  pinMode(11, OUTPUT);
}

void loop()
{
  digitalWrite(11, HIGH);
  delay(1000); // Wait for 1000 millisecond(s)
  digitalWrite(11, LOW);
  delay(1000); // Wait for 1000 millisecond(s)
}
```

BUILDING INTRUSION DETECTION SYSTEM WITH ARDUINO AND ULTRASONIC SENSOR

An intrusion detection system (IDS) is a device or software application that monitors a network or systems for malicious activity

Ultrasonic Sensors

The HC-SR04 ultrasonic sensor uses SONAR to determine the distance of an object just like the bats do. It offers excellent non-contact range detection with high accuracy and stable readings in an easy- to-use package from 2 cm to 400 cm or 1" to 13 feet. It comes complete with ultrasonic transmitter and receiver module. The ultrasonic sensor uses the reflection of sound in obtaining the time between the wave sent and the wave received. It usually sent a wave at the transmission terminal and receives the reflected waves. The time taken is used together with the normal speed of sound in air (340ms-1) to determine the distance between the sensor and the obstacle. The Ultrasonic sensor is used here for the intruder detection. The sound via a buzzer occurs when an object comes near to the sensor. The distance to which the sensor will respond can be easily adjusted in the program.



Hardware Required

Arduino UNO,	Breadboard,	Red LED,	HC-SR04 Ultrasonic Sensor,
USB Cable,	Jumper wires,	Green LED,	Buzzer

Steps of working

- Insert the Ultrasonic sensor into your breadboard and connect its Echo pin to the digital pin 2 and the Trigger pin to digital pin 3 of the Arduino.
- Insert the RED and Green LED into the breadboard. Attach the positive leg (the longer leg) of red LED to signal pin of the Buzzer via the 220- ohm resistor, and the negative leg to GND. The green LED is connected to digital pin 8 of the Arduino.
- Upload the code.
- Observe the LEDs and take some object in front of ultrasonic sensor.
- Observe the changes in the LED and buzzer sound.

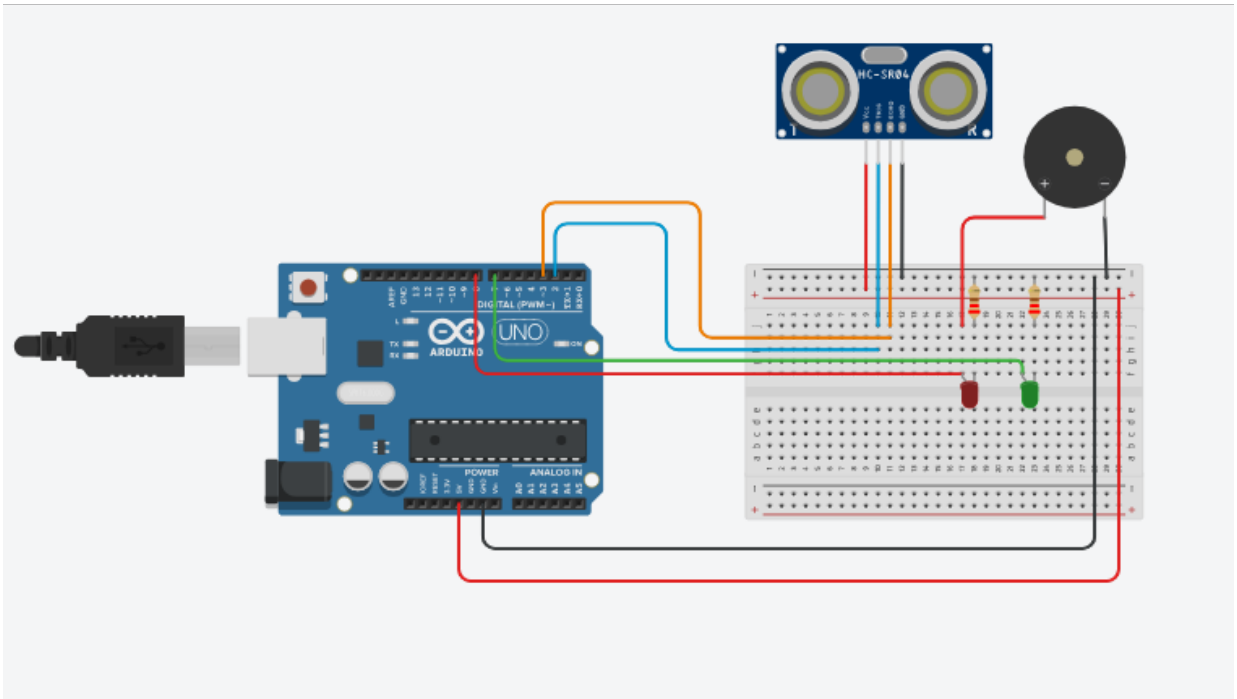
Algorithm

- This sketch works by setting pin 2 as for the ultrasonic sensors and pin 8, pin9 & pin 10 as an OUTPUT to power the LEDs and buzzer
- After that run a loop that continually reads the value from the echo pin and sends that value as voltage to the LEDs
- The color of the LED which glows will vary accordingly to the detection of object in the given range

Observations

Sr no.	Object Detected	LED	Buzzer
1			
2			

Circuit Diagram



Code

```
int echo=3;
#define trig 2
#define outA 8
#define outB 7
float duration;
float distance;
const int intruderDistance = 100;
void setup() {
pinMode(trig, OUTPUT);
pinMode(echo, INPUT);
pinMode(outA, OUTPUT);
pinMode(outB, OUTPUT);
Serial.begin(9600);

}
void loop() {
time_Measurement();
distance = (float)duration * (0.0343) / 2;
Serial.println(distance);
alarm_condition();
}
void time_Measurement()
{
delay(2);
digitalWrite(trig, HIGH);
delay(10);
digitalWrite(trig, LOW);
```

```
duration = pulseIn(echo, HIGH);
}
void alarm_condition()
{
if(distance<=intruderDistance)
{
digitalWrite(outA,HIGH);
digitalWrite(outB,LOW);
}
else
{
digitalWrite(outA,LOW);
digitalWrite (outB, HIGH);
}
}
```