# A

**Practical Activity Report Submitted for**

**ENGINEERING DESIGN-III (UTA011)**

**By**

## Shivam Mittal 101603316

**Shobhit Gupta 101603319**

**Pranav Pratap Singh 101610068**

**Yash Bhardwaj 101603380**

**Submitted to**

**Dr. Raman Singh**



**THAPAR INSTITUTE OF ENGINEERING & TECHNOLOGY, PATIALA-147004, PUNJAB INDIA**

***Jan-June 2018***

# Index (List of Experiments):-

1. Introduction to Arduino Microcontroller.
2. Write a program in Arduino to blink a LED.
3. Write a program to design a pattern from sequence of multiple LED using for loop in Arduino.
4. Write a program to demonstrate sending data from the computer to the Arduino board and control brightness of LED
5. Write a program to demonstrate control of DC Motor using forward, backward, left, right, turn, motion and clock-wise/anti-clockwise rotation.
6. Write a program to read values of IR sensor using analog and digital read and convert buggy into normal line follower robo car.
7. To demonstrate the use of ultrasonic sensor by integrating line follower robo car with obstacle avoidance capability.

# EXPERIMENT-1

**Objective:**Introduction to Arduino Microcontroller

**Theory:**Arduino is an open-source electronics platform based on easy-to-use hardware and software. Arduino boards are able to read inputs - light on a sensor, a finger on a button, or a Twitter message - and turn it into an output - activating a motor, turning on an LED, publishing something online. You can tell your board what to do by sending a set of instructions to the microcontroller on the board. To do so you use the Arduino programming language (based on Wiring), and the Arduino Software (IDE), based on Processing.

## Logic/Circuit Diagram:



**Basic Arduino code definitions:**

* input: A pinmode that intakes some information.
* output: A pinmode that sends some information.
* setup( ): This is a function which is present in every Arduino sketch. Run once before the loop( ) function. Often used to set pinmode to input or output. The setup( ) function looks like : void setup( ) { //code goes here }
* loop( ): A function which is present in every single Arduino sketch. The loop( ) is where almost everything happens. The one exception to this is setup( ) as well as variable declaration. The loop( ) function looks like: void loop( ) { //code goes here

}

* HIGH: Electrical signal is present (5V for Uno) (ON or True in boolean logic)
* LOW: No presence of electrical signal (0V) (OFF or False in boolean logic)
* digitalRead: Get a HIGH or a LOW reading from a pin which is already declared as an input.
* digitalWrite: Assign a HIGH or a LOW value to a pin which is already declared as an output.

**Result analysis**: The basic features of Arduino UNO is cleared and now further experiments can be done with Arduino board.

.

# EXPERIMENT-2

**Objective:** Write a program in Arduino to blink a LED..

**Hardware Used:** Arduino UNO Board, multiple LEDs, bread board, jumping wires, data cable and a resistor.

**Software-**Arduino 1.6.5

**Theory-**A light-emitting diode (LED) is a semiconductor device which can turn electric energy into light energy via PN junctions. By wavelength, it can be categorized into laser diode, infrared light-emitting diode and visible light-emitting diode which is usually known as light-emitting diode (LED). LEDs are usually red, yellow, green, blue, or colour-changing. Colour-changing LEDs can change their colour with different voltages.

For running multiple LEDs we need to use for() loops and use arrays for making managing variables easier by grouping them together.

## CODE-

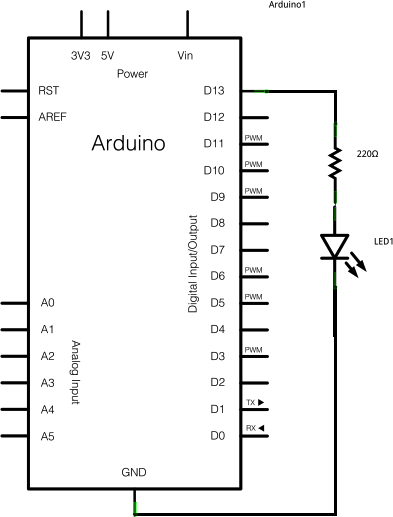
void setup() { pinMode(13, OUTPUT);

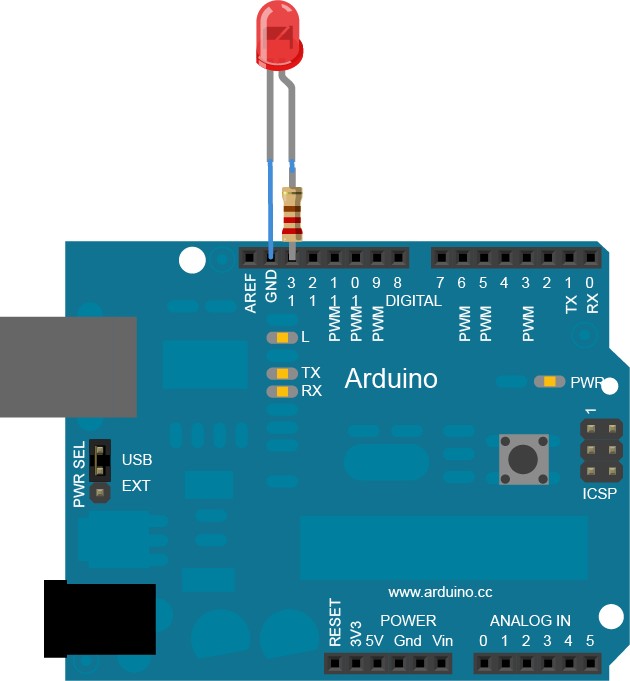
}

void loop() { digitalWrite(13, HIGH); delay(1000); digitalWrite(13,LOW); delay(1000);

}

## Logic/Circuit Diagram:





**RESULTS-**

We Successfully initiated the blinking of a LED using the Arduino board according to the code compiled.( When potential difference is created between two terminal of LEDs they start to glow.)

# EXPERIMENT-3

**Objective:**Write a program to design a pattern from sequence of multiple LED using for loop in Arduino.

**Hardware Used-** Arduino UNO Board and its connecting cable, 6 LED’s, 6 Resistors, Breadboard, Wires,

**Software-**Arduino 1.6.5

## Theory-

LED stands for Light Emitting Diode. A diode only allows electricity to flow through it one way, so if you hook it up backwards it won’t work. If you connect the LED directly to power and ground, too much current will go through the diode and destroy it. To keep that from happening we will use a resistor to limit the current.

Often you want to iterate over a series of pins and do something to each one. For instance, this example blinks 6 LEDs attached to the Arduino or Genuino by using a [**for()**](https://www.arduino.cc/en/Reference/For) loop to cycle back and forth through digital pins 2-7. The LEDS are turned on and off, in sequence, by using both the [digitalWrite()](https://www.arduino.cc/en/Reference/DigitalWrite) and [delay()](https://www.arduino.cc/en/Reference/Delay) functions .

## CODE-

int timer =100; voidsetup(){

for(int thisPin =2; thisPin <8; thisPin++){ pinMode(thisPin,OUTPUT);

}

}

voidloop(){

for(int thisPin =2; thisPin <8; thisPin++){

digitalWrite(thisPin,HIGH); delay(timer);

digitalWrite(thisPin,LOW); }

for(int thisPin =7; thisPin >=2; thisPin--){ digitalWrite(thisPin,HIGH);

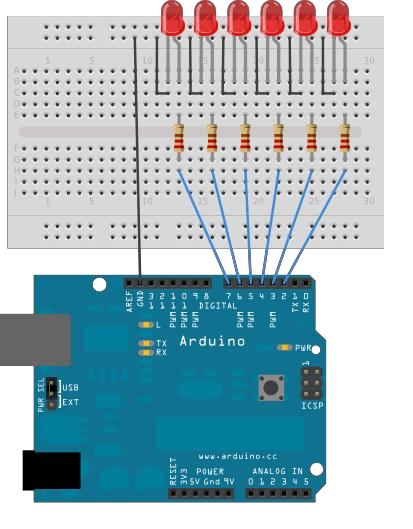
delay(timer);

digitalWrite(thisPin,LOW);

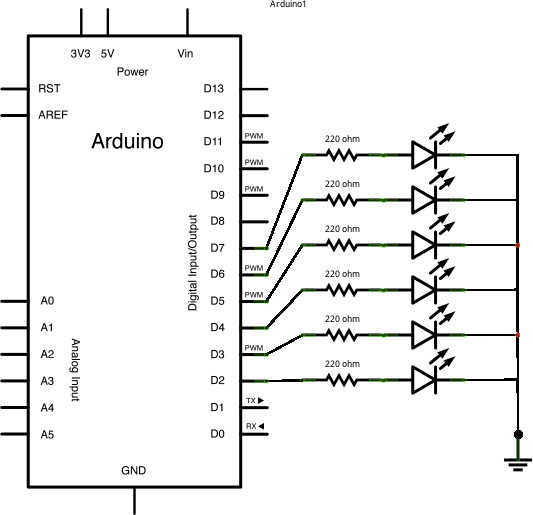
}

}

## Connection Diagram with Arduino Board:



**Logic/Circuit Diagram:**



**RESULTS-**

Successfully blinked 6 LED’s using Arduino board as per the code compiled. using

a [**for()**](https://www.arduino.cc/en/Reference/For) loop to cycle back and forth through digital pins 2-7. The LEDS are turned on and off, in sequence, by using both the digitalWrite() and delay() functions .

.

# EXPERIMENT-4

**Objective -**Write a program to demonstrate sending data from the computer to the arduino board and control the brightness of LED.

**Hardware Used-**Arduino board, LED, 220 Ohm resistor, Connecting wires

**Software-**Arduino 1.6.5

## Theory-

LED stands for Light Emitting Diode. A diode only allows electricity to flow through it one way, so if you hook it up backwards it won’t work. If you connect the LED directly to power and ground, too much current will go through the diode and destroy it. To keep that from happening we will use a resistor to limit the current. The higher the value of the resistor lesser is the electricity “flow” through it. Resistors have color bands on them that let you know what value they are. Resistance is measured in ohms. The two leads (sometimes called “legs”) of an LED are called an anode and a cathode. The anode is the longer lead. A potentiometer is a simple knob that provides a variable resistance, which we can read into the Arduino board as an analog value. In this example, that value controls the rate at which an LED blinks.We connect three wires to the Arduino board. The first goes to ground from one of the outer pins of the potentiometer. The second goes from 5 volts to the other outer pin of the potentiometer. The third goes from analog input 2 to the middle pin of the potentiometer. By turning the shaft of the potentiometer, we change the amount of resistance on either side of the wiper, which is connected to the centre pin of the potentiometer. This changes the relative "closeness" of that pin to 5 volts and ground, giving us a different analog input. When the shaft is turned all the way in one direction, there are 0 volts going to the pin, and we read 0. When the shaft is turned all the way in the other direction, there are 5 volts going to the pin and we read 1023. In between, analogRead() returns a number between 0 and 1023 that is proportional to the amount of voltage being applied to the pin.

This experiment shows how to send data from a personal computer to an Arduino or Genuino board to control the brightness of an LED. The data is sent in individual bytes, each of which ranges in value from 0 to 255. The sketch reads these bytes and uses them to set the brightness of the LED.

## Code:

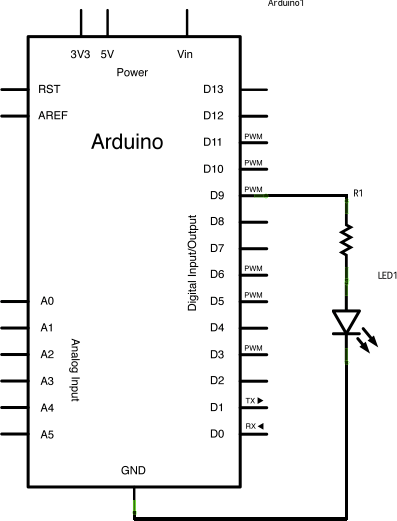
void setup() { Serial.begin(9600); pinMode (9,OUTPUT);

}

void loop() { x=serial.parseInt(); delay(3000); analogWrite(3,x); delay(1000);

}

## Logic /Circuit diagram:



**RESULTS-**

The brightness of the LED can be controlled by using this program . The data is sent in individual bytes, each of which ranges in value from 0 to 255. It reads these bytes and uses them to set the brightness of the LED.

# EXPERIMENT-5

**Objective-** A program to demonstrate control of DC Motor using forward, backward, left, right turn motion and clock-wise/anti clock- wise rotation.

**Hardware Used-**Arduino UNO Board and its connecting cable, IC L293D, DC motor, Breadboard, Wires, Arduino IDE

**Software-**Arduino 1.6.5

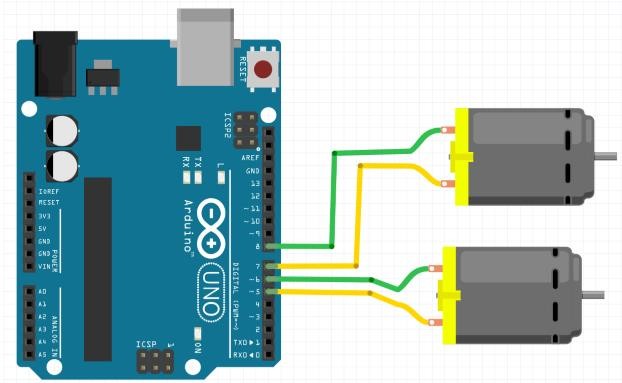
## Theory-

A DC motor is any of a class of rotary electrical machines that converts direct current electrical power into mechanical power. A DC motor's speed can be controlled over a wide range, using either a variable supply voltage or by changing the strength of current in its field windings. L293D is a typical Motor driver or Motor Driver IC which allows DC motor to drive on either direction. L293D is a 16-pin IC which can control a set of two DC motors simultaneously in any direction. It means that you can control two DC motor with a single L293D IC.

## Operation:

There are 4 input pins for l293d, pin 7,8 on the left and pin 5,6 on the right as shown on the pin diagram. Left input pins will regulate the rotation of motor connected across left side and right input for motor on the right hand side. The motors are rotated on the basis of the inputs provided across the input pins as LOGIC 0 or LOGIC 1.

In simple you need to provide Logic 0 or 1 across the input pins for rotating the motor.



## Code:

void setup() { pinMode(5,OUTPUT); pinMode(6,OUTPUT); pinMode(7,OUTPUT); pinMode(8,OUTPUT);

}

void forward()

{

digitalWrite(5,HIGH); digitalWrite(6,LOW); digitalWrite(7,LOW); digitalWrite(8,HIGH);

}

void backward()

{

digitalWrite(5,LOW); digitalWrite(6,HIGH); digitalWrite(7,HIGH); digitalWrite(8,LOW);

}

void left()

{

digitalWrite(5,HIGH); digitalWrite(6,LOW); digitalWrite(7,LOW); digitalWrite(8,LOW);}

void right()

{

digitalWrite(5,LOW); digitalWrite(6,LOW); digitalWrite(7,LOW); digitalWrite(8,HIGH);

}

void clockwise()

{

digitalWrite(5,LOW); digitalWrite(6,HIGH); digitalWrite(7,LOW); digitalWrite(8,HIGH);

}

void anticlockwise()

{

digitalWrite(5,HIGH); digitalWrite(6,LOW); digitalWrite(7,HIGH); digitalWrite(8,LOW);

}

void loop() { forward(); delay(5000); anticlockwise(); delay(5000); forward(); delay(5000); clockwise(); delay(5000); backward(); delay(5000); forward(); delay(5000); left(); delay(5000); right(); delay(5000);

}

## RESULTS-

In this experiment we successfully managed to run a DC motor using an Arduino board.

# EXPERIMENT-6

**Objective:** Write a program to read values of IR sensor using analog and digital read and convert buggy into normal line follower robo car.

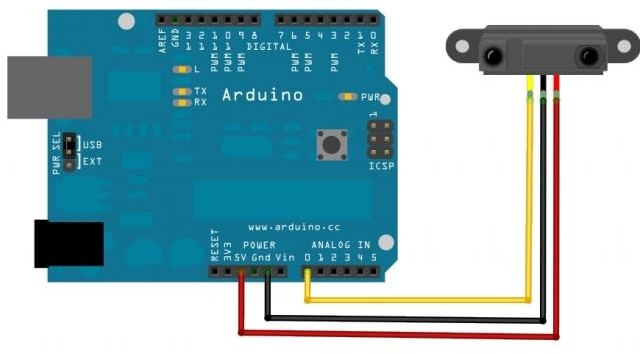
**Hardware Used-** Arduino UNO Board and its connecting cable, IR sensor, Breadboard, Wires

**Software-**Arduino 1.6.5

**Theory:-**

IR Sensors work by using a specific light sensor to detect a select light wavelength in the Infra-Red (IR) spectrum. By using an LED which produces light at the same wavelength as what the sensor is looking for, you can look at the intensity of the received light. When an object is close to the sensor, the light from the LED bounces off the object and into the light sensor. This results in a large jump in the intensity, which we already know can be detected using a threshold. The PING sensor emits short bursts of sound and listens for this sound to echo off of nearby objects. The frequency of the sound is too high for humans to hear (it is ultrasonic). The PING sensor measures the time of flight of the sound burst. A user then computes the distance to an object using this time of flight and the speed of sound (1,126 ft/s).

## Connection Diagram using Arduino Board:



**Operation:**

The Arduino triggers the PING by sending a 5ms pulse to the sensor through pin 13, which is initially configured as an Arduino OUTPUT.

When the PING receives the 5ms pulse from the Arduino, it sends a 40kHz (ultrasonic) burst of sound out its “speaker” and sets pin 13 to HIGH.

The PING then waits for the sound burst to reflect off of something and return to the “microphone” where it is detected; the PING then sets pin 12 to LOW.

The Arduino uses the pulseIn command to measure the time of flight of the sound wave in microseconds (the time that pin 12, when configured as an input, is HIGH).

The “time of flight” of the sound wave in ms is stored in the variable “duration.”

## Code:

int r1,r2;

void setup() {

pinMode (5,OUTPUT); pinMode (6,OUTPUT); pinMode (7,OUTPUT); pinMode (8,OUTPUT); pinMode (A0,INPUT); pinMode (A2,INPUT); Serial.begin(9600);

}

void forward(void){ digitalWrite(5,HIGH); digitalWrite(6,LOW); digitalWrite(7,LOW); digitalWrite(8,HIGH);

}

void backward (void){ digitalWrite(5,LOW); digitalWrite(6,HIGH); digitalWrite(7,HIGH); digitalWrite(8,LOW);

}

void right(void){ digitalWrite(5,LOW); digitalWrite(6,LOW); digitalWrite(7,LOW); digitalWrite(8,HIGH);

}

void left(void){ digitalWrite(5,HIGH); digitalWrite(6,LOW); digitalWrite(7,LOW); digitalWrite(8,LOW);

}

void loop(){ r1=digitalRead(A0); r2=digitalRead(A2); if(r1==1 && r2==1); forward();

if(r1==0 && r2==0); forward();

if(r1==1 && r2==0); right();

if(r1==0 && r2==1); left();

}

## RESULTS-

Buggy senses black line by using IR sensors and then send signals to Arduino, If both side sensor senses white colour or black colour the buggy moves forward. If left sensor comes on black line then robot turn left side. If right sensor sense black line then robot turn right side until both sensor comes at white surface. When white surface comes robot starts moving on forward again.

## EXPERIMENT- 7

**Objective:** To demonstrate the use of ultrasonic sensor by integrating line follower robo car with obstacle avoidance capability.

**Hardware used-** Ultrasonic sensor, robotic buggy.

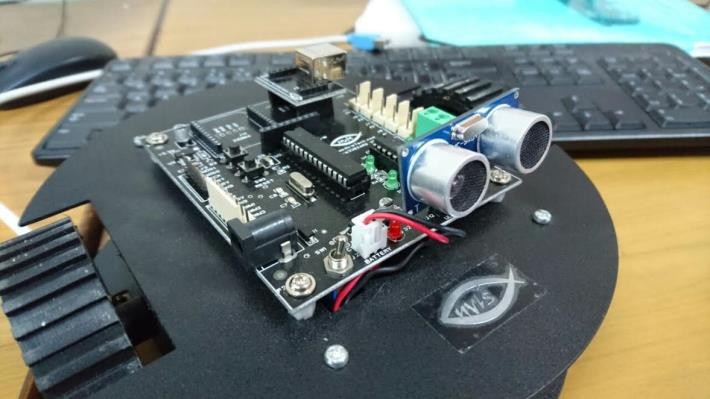
**Software used-** Arduino IDE.

**Theory:** Ultrasonic sensors generate high-frequency sound waves and evaluate the echo which is received back by the sensor, measuring the time interval between sending the signal and receiving the echo to determine the distance to an object.

**ULTRASONIC SENSOR HC-SR04**

***Working****:* The Ultrasonic Sensor sends out a high-frequency sound pulse and then times how long it takes for the echo of the sound to reflect back. The sensor has 2 openings on its front. One opening transmits ultrasonic waves, (like a tiny speaker), the other receives them, (like a tiny microphone). The speed of sound is approximately 341 meters (1100 feet) per second in air. The ultrasonic sensor uses this information along with the time difference between sending and receiving the sound pulse to determine the distance to an object. It uses the following mathematical equation: Distance = Time x Speed of Sound divided by 2 Time

= the time between when an ultrasonic wave is transmitted and when it is received. You divide this number by 2 because the sound wave has to travel to the object and back.



## Code:

#include <Newping.h> int trigger=10;

int echo=12;

int Maxdist=200;

NewPing Sonar(trigger,echo,Maxdist);

void setup()

{

pinMode(10,OUTPUT); pinMode(12,INPUT); pinMode(8,OUTPUT); pinMode(6,OUTPUT); pinMode(5,OUTPUT); pinMode(7,OUTPUT); Serial.begin(9600);

}

void forward(){ digitalWrite(5,HIGH); digitalWrite(6,LOW); digitalWrite(7,LOW); digitalWrite(8,HIGH);

}

void stop(){ digitalWrite(5,LOW); digitalWrite(6,LOW); digitalWrite(7,LOW); digitalWrite(8,LOW);

}

void loop()

{

int x=Sonar.ping\_cm();

if(x<=10)

{

stop();

}

Else

{

forward();

} }

**Result :**The Experiment to make obstacle avoiding Robo Car was done.

**Signature of Faculty member**