**A Practical activity Report submitted for Engineering Design**

**Project-II (UTA-014) by**

**Details of students**:

**-**

**Submitted to:**

**\_**

**GROUP NO.:**

****

**DEPARTMENT OF COMPUTER SCIENCE and ENGINEERING THAPAR**

**INSTITUTE OF ENGINEERING & TECHNOLOGY, (A DEEMED TO BE**

**UNIVERSITY), PATIALA, PUNJAB INDIA.**

# TABLE OF CONTENTS

|  |  |
| --- | --- |
| **ABSTRACT** | i |
| **DECLARATION** | ii |
| **INDEX** | iii |

**ABSTRACT**

The Engineering design Project is a one of its kind Project/Course. In a curriculum of theoretical courses, this course provides us with an opportunity to implement the theoretical knowledge into real life applications. The application of Programming and Circuits in real time applications is what this course all about

It gives us a clear view of Electronic and Computer based implementations.

This also describes the use of some of the major components of electronics, including capacitors, resistors, LEDs, microcontroller, operational amplifier and many more.

The computer part of the project includes all the coding part required to run the transmitter, receiver and the IR sensors circuit, with the help of Arduino UNO board, breadboard and connecting wires. The coding part is done on Arduino IDE, a software which is a text editor like a notepad with different features. It is used for writing code, compiling the code to check if any errors are there and uploading the code to the Arduino. It also includes supervisory control with the help of Zigbee which is done through software called XCTU.

A buggy was designed which uses IR sensor, receivers, transmitters and a

programmable Arduino UNO board to follow a line/path. The code was fed from the computers to the board. An ultrasonic sensor was attached to detect any obstacles and stop.

The Buggy also has the Receiver and IR circuit fitments on it. The Receiver circuit is responsible for receiving signals and pulses from the transmitter circuit that is attached on the gantry, and hence the buggy stops at the gantry.

# DECLARATION

We declare that this project report is based on our own work carried out during the course of our study in our Engineering-design II Computer Lab under the supervision of **Mr. Krishan Kumar.** We assert that the statements made and conclusions drawn are an outcome of our own research work. We further certify that the work contained in this report is original and has been done by us under the general supervision of our supervisor. We have followed the guidelines provided by the University in writing this report.

We also declare that this project is the outcome of our own effort, that it has not been submitted to any other university for the award of any degree.

# INDEX

|  |  |  |
| --- | --- | --- |
| **S. No.** | **Name of Experiments** | **Page No.** |
| 1. | Write a program to demonstrate control of DC Motor using forward, backward, left, right turn motion and  clock-wise/anti clock-wise rotation. | 1-3 |
| 2. | Write a program to read values of IR Sensor using analog and digital read and convert buggy into normal line  follower robocar. | 4-6 |
| 3. | To demonstrate the use of ultrasonic sensor by  integrating line following robocar with obstacle avoidance capability. | 7-10 |
| 4. | Write a program  a) to read the pulse width of gantry transmitter and trigger stop\_buggy function by detecting individual gantry, b) to demonstrate XBEE module  communication between 2 PC’s using X-CTU. | 11-14 |
| 5. | Bronze Challenge: Single buggy around track twice in clockwise direction, under full supervisory control.  Buggy can detect an obstacle, parks safely. Prints state of  the track and buggy at each gantry stop. | 15-16 |

**EXPERIMENT-1**

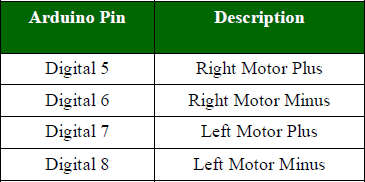
**OBJECTIVE:** Write a program to demonstrate control of DC Motor using forward, backward, left, right turn motion and clock-wise/anti clock-wise rotation.

**SOFTWARE USED:** Arduino IDE

## HARDWARE USED:

|  |  |  |
| --- | --- | --- |
| **Sr. No.** | **Name of Components** | **Value** |
| 1. | Arduino UNO | Microcontroller |
| 2. | USB Cable | 1 |
| 3. | Nvis 3302ARD RoboCar | 1 |
| 4. | IR Sensors | 2 |
| 5. | Wires |  |

**THEORY:** The RoboCar is an assembled structure of Nvis 3302ARD. The machine is driven by DC motors which are powered by rechargeable batteries. A motor is a machine that converts electrical energy into mechanical energy(rotation).Its features includes – wireless control of robocar using XBee, detection of objects using ultrasonic sensor, detection of path using IR sensors and many more. The code which is made according to problem statement is uploaded on to the robocar using USB cable.



1

## CODING:

void stop1(){ digitalWrite(6,LOW); digitalWrite(8,LOW); digitalWrite(7,LOW); digitalWrite(5,LOW); delay(1000);

}

void stop2(){ digitalWrite(6,LOW); digitalWrite(8,LOW); digitalWrite(7,LOW); digitalWrite(5,LOW); delay(2000);

}

void left(){ digitalWrite(6,LOW); digitalWrite(8,HIGH); digitalWrite(7,LOW); digitalWrite(5,LOW); delay(1500);

}

void right(){ digitalWrite(6,LOW); digitalWrite(8,LOW); digitalWrite(5,HIGH); digitalWrite(7,LOW); delay(1500);

}

void forw(){ digitalWrite(5,HIGH); digitalWrite(8,HIGH); digitalWrite(7,LOW); digitalWrite(6,LOW); delay(2000);

}

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

delay(2000);

}

void clockw(){ digitalWrite(7,HIGH); digitalWrite(6,LOW); digitalWrite(5,HIGH); digitalWrite(8,LOW); delay(3100);

}

void anticlock(){ digitalWrite(7,LOW); digitalWrite(6,HIGH); digitalWrite(5,LOW); digitalWrite(8,HIGH); delay(3000);

}

void setup() {

// put your setup code here, to run once: pinMode(5,OUTPUT); pinMode(6,OUTPUT); pinMode(7,OUTPUT); pinMode(8,OUTPUT);

}

void loop() { forw();

stop1();

back();

stop1();

back();

stop1();

forw();

stop1();

right();

stop1();

forw();

stop1();

back();

stop1();

right();

stop1();

forw();

stop1();

back();

stop1();

right();

stop1();

forw();

stop1();

back();

stop1();

left();

stop2();

clockw();

stop1(); anticlock(); stop1();

while(1){

}

}

## MODEL:



**RESULT ANALYSIS:**

In this experiment, we have learnt how to use a DC Motor for the forward, backward, left, right turn motion and clock-wise/anti clock-wise rotation in a RoboCar.

**Signature of Faculty Member**

# EXPERIMENT-2

**OBJECTIVE:** Write a program to read values of IR Sensor using analog and digital read and convert buggy into normal line follower robocar.

**SOFTWARE USED:** Arduino IDE

## HARDWARE USED:

|  |  |  |
| --- | --- | --- |
| **Sr. No.** | **Name of Components** | **Value** |
| 1. | Arduino UNO | Microcontroller |
| 2. | USB Cable | 1 |
| 3. | Nvis 3302ARD RoboCar | 1 |
| 4. | IR Sensors | 2 |
| 5. | Wires |  |

**THEORY:** IR sensor basically works on intensity of light RGB code for white is (255,255,255) and for Black is (0,0,0)

An IR sensor is a device that emits signals in order to sense some aspects of the surroundings which detects IR radiation falling on it. The emitter is an IR LED (Light Emitting Diode) and the detector is an IR photodiode which is sensitive to IR light. In our buggy, the IR sensor helps it to move only on the black lines of our path defined. We use the pre defined functions from the previous experiment to control the movement of the buggy on the path in the Arduino IDE which in turn gives the instructions to the IR sensor according to values read by the IR sensor on the analog pins..

## CODING:

void setup()

{

pinMode(A0,INPUT); pinMode(A1,INPUT); pinMode(5,OUTPUT);

pinMode(6,OUTPUT); pinMode(7,OUTPUT); pinMode(8,OUTPUT);

}

void loop()

{

int p,q; p=digitalRead(A0); q=digitalRead(A1);

if ((p==0 && q==0) || (p==1 && q==1))

{

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

}

else if(p==0 && q==1)

{

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

}

else

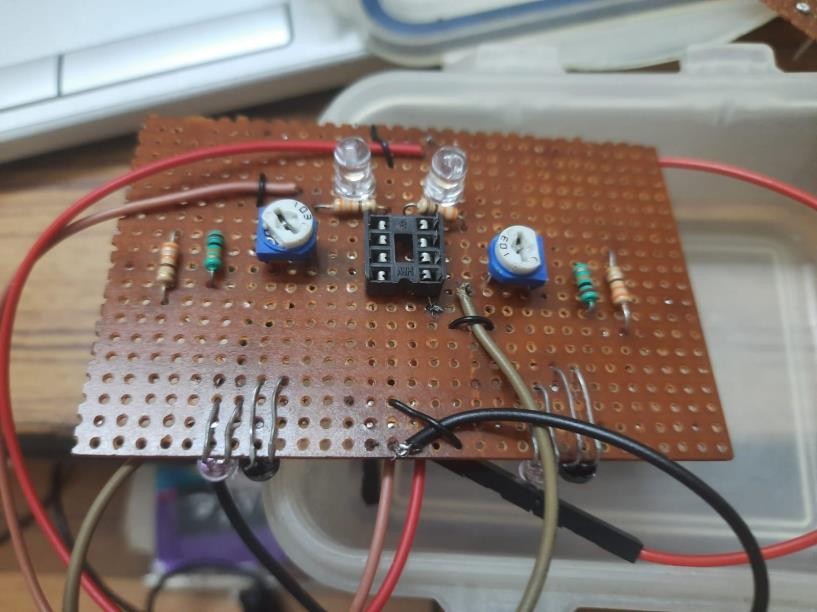
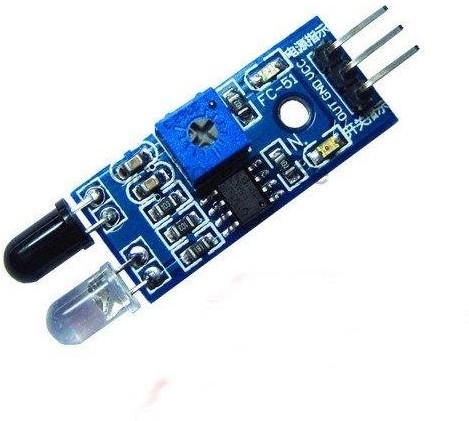
{

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

}

}

## MODEL:



**RESULT ANALYSIS:**

In this Experiment we learnt how to take input from IR Sensors using analog pins. And also that how can we make the Buggy follow the black path and execute the proper lpath follow of the buggy y adjusting the sensitivity of the IR Sensor..

**Signature of Faculty Member**

# EXPERIMENT-3

**OBJECTIVE:** To demonstrate the use of ultrasonic sensor by integrating line following robocar with obstacle avoidance capability.

**SOFTWARE USED:** Arduino IDE

## HARDWARE USED:

|  |  |  |
| --- | --- | --- |
| **Sr. No.** | **Name of Components** | **Value** |
| 1. | Arduino UNO | Microcontroller |
| 2. | USB Cable | 1 |
| 3. | Nvis 3302ARD RoboCar | 1 |
| 4. | IR Sensors | 2 |
| 5. | Ultrasonic Sensor | 1 |
| 6. | Jumper Wires |  |

**THEORY:** An ultrasonic sensor is an instrument that measures the distance to an object and detects the obstacle using ultrasonic sound waves. An ultrasonic sensor uses a transducer to send and receive ultrasonic pulses that relay back information about an object’s proximity. Thus, we use the ultrasonic sensor on the buggy to measure the distance between the buggy and obstacle. We stop the buggy as soon as the distance between the buggy and the obstacle is less than 15cm. Hence, we design the code to control the working of the ultrasonic sensor on the robocar.

The HC-SR04 module hosts the ultrasonic **transmitter**, the **receiver** and **control** circuit. The HC-SR04 has four pins namely Vcc, Trigger, Echo, GND and they are explained in detail below:

1. **VCC**: 5V DC supply voltage is connected to this pin.
2. **Trigger**: The **trigger signal** for starting the transmission is given to this pin. The trigger signal must be a pulse with 10uS high time.

When the module receives a valid trigger signal, it issues 8 pulses of 40KHz ultrasonic sound from the transmitter.

The echo of this sound is picked by the echo pin.

1. **Echo**: At this pin, the module outputs will be received as a waveform with high time proportional to the distance.
2. **GND**: Ground is connected to this pin.

## CODING:

#include<NewPing.h> #define TRIGGER\_PIN 13

#define ECHO\_PIN 12

#define MAX\_DISTANCE 200

NewPing sonar(TRIGGER\_PIN,ECHO\_PIN,MAX\_DISTANCE);

void setup()

{

pinMode(A0,INPUT); pinMode(A2,INPUT); Serial.begin(9600);

pinMode(5,OUTPUT); pinMode(6,OUTPUT); pinMode(7,OUTPUT); pinMode(8,OUTPUT); pinMode(TRIGGER\_PIN, OUTPUT); pinMode(ECHO\_PIN, INPUT);

Serial.begin(9600);

Serial.println("Ultrasonic Sensor HC-SR04 Test"); Serial.println("with Arduino UNO R3");

}

void stop1(){ digitalWrite(5,LOW); digitalWrite(8,LOW); digitalWrite(6,LOW); digitalWrite(7,LOW);

}

void loop()

{

int p,q; p=digitalRead(A0); q=digitalRead(A2); long duration;

int distance; digitalWrite(TRIGGER\_PIN, LOW); delayMicroseconds(2);

digitalWrite(TRIGGER\_PIN, HIGH); delayMicroseconds(10); digitalWrite(TRIGGER\_PIN, LOW);

duration = pulseIn(ECHO\_PIN, HIGH);

distance = duration \* 0.034 / 2; // Speed of sound wave divided by 2 (go and back) Serial.print("Distance: ");

Serial.print(distance); Serial.println(" cm");

if(distance<15 && distance>0)

{

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

}

else if ((p==0 && q==0) || (p==1 && q==1))

{

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

}

else if(p==0 && q==1)

{

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

}

else

{

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

}

}

## COMPONENTS:



**RESULT ANALYSIS:**

In this experiment, we learnt how to use the Ultrasonic sensor on the buggy for distance measurement. We also learned to design the code for controlling the motion of the buggy and detecting the obstacles in its path.

**Signature of Faculty Member**

**EXPERIMENT-4**

**OBJECTIVE:** Write a program a) to read the pulse width of gantry transmitter and trigger stop\_buggy function by detecting individual gantry, b) to demonstrate XBEE module communication between 2 PC’s using X-CTU.

**SOFTWARE USED:** Arduino IDE

## HARDWARE USED:

|  |  |  |
| --- | --- | --- |
| **Sr. No.** | **Name of Components** | **Value** |
| 1. | Arduino UNO | Microcontroller |
| 2. | USB Cable | 1 |
| 3. | Nvis 3302ARD RoboCar | 1 |
| 4. | IR Sensors | 2 |
| 5. | XBee Module | 1 |
| 6. | Jumper Wires |  |

**THEORY:** XBee module is used for the communication between 2 PC’s using the X- CTU software. It is configured first and then attached to the buggy so that it can be controlled from the X-CTU software on its own.

As soon as the receiver on the top of buggy receives signals from the transmitter that is attached on the gantry, our buggy stops.

* 1. Each network has 1 coordinator
  2. Coordinator selects channel and PAN ID
  3. Other devices then join the PAN
  4. 16-bit address is always 0
  5. Assigns 16-bit address for the router and end devices

## CODING:

int pin5=5; int pin6=6; int pin7=7; int pin8=8; int irPin=4;

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 right()

{

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

}

void left()

{

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

}

void Stop()

{

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

}

void setup()

{

pinMode(pin5,OUTPUT); pinMode(pin6,OUTPUT); pinMode(pin7,OUTPUT); pinMode(pin8,OUTPUT); Serial.begin(9600);

}

void loop()

{

if (Serial.available() > 0)

{

char s = Serial.read(); switch (s)

{

case 'F':

{

forward(); Serial.println("Forward"); break;

}

case 'S':

{

Stop(); Serial.println("Stop"); break;

}

case 'L':

{

left(); Serial.println("Left"); break; }

case 'R':

{

right(); Serial.println("Right"); break;

}

}

}

if (digitalRead(irPin)==HIGH)

{

int d = pulseIn(irPin,HIGH); if (d>500 and d<1500)

{

Serial.println(d); Serial.println("Gantry: Crossed"); Stop();

delay(1000); forward();

}

}

}

## XBee 1mW Wire Antenna - Series 1 (802.15.4) - WRL-08665 - SparkFun ElectronicsCOMPONENT:

**RESULT ANALYSIS:**

In this experiment we learnt about XBee Module and the X-CTU application used to configure the XBee. And also to stop the buggy at the gantry and display the distance respectively.

**Signature of Faculty Member**

**EXPERIMENT-5**

**OBJECTIVE:** Bronze Challenge: Single buggy around track twice in clockwise direction, under full supervisory control. Buggy can detect an obstacle, parks safely. Prints state of the track and buggy at each gantry stop.

**SOFTWARE USED:** Arduino IDE

## HARDWARE USED:

|  |  |  |
| --- | --- | --- |
| **Sr. No.** | **Name of Components** | **Value** |
| 1. | Arduino UNO | Microcontroller |
| 2. | USB Cable | 1 |
| 3. | Nvis 3302ARD RoboCar | 1 |
| 4. | IR Sensors | 2 |
| 5. | Ultrasonic Sensor | 1 |
| 6. | XBee Module | 1 |
| 7. | Jumper Wires |  |

**THEORY:** The bronze challenge i.e. the buggy is required to traverse the whole path twice in clockwise direction while following the path, detect obstacles, park at the right position, stop at each of the three gantry and work with XBee configuration to give initial start to the buggy. This is done with the help of all the circuits prepared earlier and all the experiments performed. We are using the transmitter, receiver and the IR circuits soldered on our buggy to achieve this challenge.

## CODING:

int t1=A2; int t2=A0; int pin5=5; int pin6=6; int pin7=8; int pin8=7; int irPin=4; int flag=0;

unsigned long d=0;

static int gantryCounter=0; static long StartTime=0; static long CurrentTime = 0;

unsigned long ElapsedTime = 0; static long StartTimeG=millis();; static long CurrentTimeG = 0; unsigned long ElapsedTimeG = 0; long previousMillisU = millis(); long intervalU = 500;

#include <NewPing.h> #define TRIGGER\_PIN 13

#define ECHO\_PIN 12

#define MAX\_DISTANCE 200

NewPing sonar(TRIGGER\_PIN, ECHO\_PIN, MAX\_DISTANCE);

void setup() { pinMode(pin5,OUTPUT); pinMode(pin6,OUTPUT); pinMode(pin7,OUTPUT); pinMode(pin8,OUTPUT); pinMode(t1,INPUT); pinMode(t2,INPUT); Serial.begin(9600);

Serial.print("+++"); // Enter xbee AT command mode, NB no carriage return here delay(1500); // Guard time

Serial.println("ATID 3333, CH C, CN");

}

void loop() { if(flag==0)

{

if (Serial.available() > 0)

{

char s = Serial.read(); switch (s) {

case 'Y':

{

flag=1;

}

}

}

}

unsigned long currentMillisU = millis(); if(currentMillisU - previousMillisU > intervalU)

{

previousMillisU = currentMillisU; detectObstacle();

}

if (flag==1)

{

gantry();

}

if (flag==3)

{

//gantryParking(); CurrentTimeG=millis();

ElapsedTimeG = CurrentTimeG-StartTimeG;

if(ElapsedTimeG<2000)

{

flag=3; leftBlind();

}

if(ElapsedTimeG>2000 && ElapsedTimeG<7500)

{

flag=3; normalLineFollow();

}

if(ElapsedTimeG>7500)

{

stopBuggy(); Serial.print("Buggy:1 Parked"); Serial.println(ElapsedTimeG); delay(2000);

flag=-1;

}

}

}

void gantry()

{

int r1=digitalRead(t1); int r2=digitalRead(t2);

if(r1==LOW&&r2==LOW)

{

digitalWrite(pin5,HIGH); digitalWrite(pin6,LOW); digitalWrite(pin7,HIGH); digitalWrite(pin8,LOW);

}

if(r1==LOW&&r2==HIGH)

{

digitalWrite(pin5,HIGH); digitalWrite(pin6,LOW); digitalWrite(pin7,LOW); digitalWrite(pin8,LOW);

} if(r1==HIGH&&r2==LOW)

{

digitalWrite(pin5,LOW); digitalWrite(pin6,LOW); digitalWrite(pin7,HIGH); digitalWrite(pin8,LOW);

} if(r1==HIGH&&r2==HIGH)

{

digitalWrite(pin5,HIGH); digitalWrite(pin6,LOW); digitalWrite(pin7,HIGH); digitalWrite(pin8,LOW);

}

if (digitalRead(irPin)==HIGH)

{

StartTime = millis();

d = pulseIn(irPin,HIGH);

if (d>500 and d<1500) //(d > 500 and d < 1500)

{

Serial.println(d); Serial.println("Gantry: 1"); stopBuggy();

delay(1000);

}

else if (d>2000 and d< 3000) //(d> 1500 and d < 2500)

{

Serial.println(d); Serial.println("Gantry: 2"); stopBuggy(); gantryCounter=gantryCounter+1; Serial.print("The gantry Counter is: "); Serial.println(gantryCounter); stopBuggy();

delay(1000);

}

else if (d>3000 and d<4000) //(d > 2500 and d < 3500)

{

Serial.println(d); Serial.println("Gantry: 3");

delay(1000);

}

else {

//Serial.println(d); Serial.println("Gantry: Unknown");

}

if (gantryCounter>=2)

{

StartTimeG = millis(); flag=3;

}

else

{

gantry();

}

}

}

void stopBuggy()

{

digitalWrite(pin5,LOW); digitalWrite(pin6,LOW); digitalWrite(pin7,LOW); digitalWrite(pin8,LOW);

}

void normalLineFollow()

{

int r1=digitalRead(t1); int r2=digitalRead(t2);

if(r1==LOW&&r2==LOW)

{

digitalWrite(pin5,HIGH); digitalWrite(pin6,LOW); digitalWrite(pin7,HIGH); digitalWrite(pin8,LOW);

}

if(r1==HIGH&&r2==LOW)

{

digitalWrite(pin5,LOW); digitalWrite(pin6,LOW); digitalWrite(pin7,HIGH); digitalWrite(pin8,LOW);

} if(r1==LOW&&r2==HIGH)

{

digitalWrite(pin5,HIGH); digitalWrite(pin6,LOW);

digitalWrite(pin7,LOW); digitalWrite(pin8,LOW);

}

if(r1==HIGH&&r2==HIGH)

{

digitalWrite(pin5,HIGH); digitalWrite(pin6,LOW); digitalWrite(pin7,HIGH); digitalWrite(pin8,LOW);

}

}

void leftBlind()

{

int r2=digitalRead(t2); if(r2==LOW)

{

digitalWrite(pin5,LOW); digitalWrite(pin6,LOW); digitalWrite(pin7,HIGH); digitalWrite(pin8,LOW);

}

if(r2==HIGH)

{

digitalWrite(pin5,HIGH); digitalWrite(pin6,LOW); digitalWrite(pin7,HIGH); digitalWrite(pin8,LOW);

}

}

void detectObstacle()

{

delay(50);

unsigned int distanceCm; distanceCm = sonar.ping\_cm(); pinMode(ECHO\_PIN,OUTPUT);

digitalWrite(ECHO\_PIN,LOW); pinMode(ECHO\_PIN,INPUT);

Serial.print("Ping: "); Serial.println(distanceCm); Serial.println("cm");

if((distanceCm<15) && (distanceCm>0))

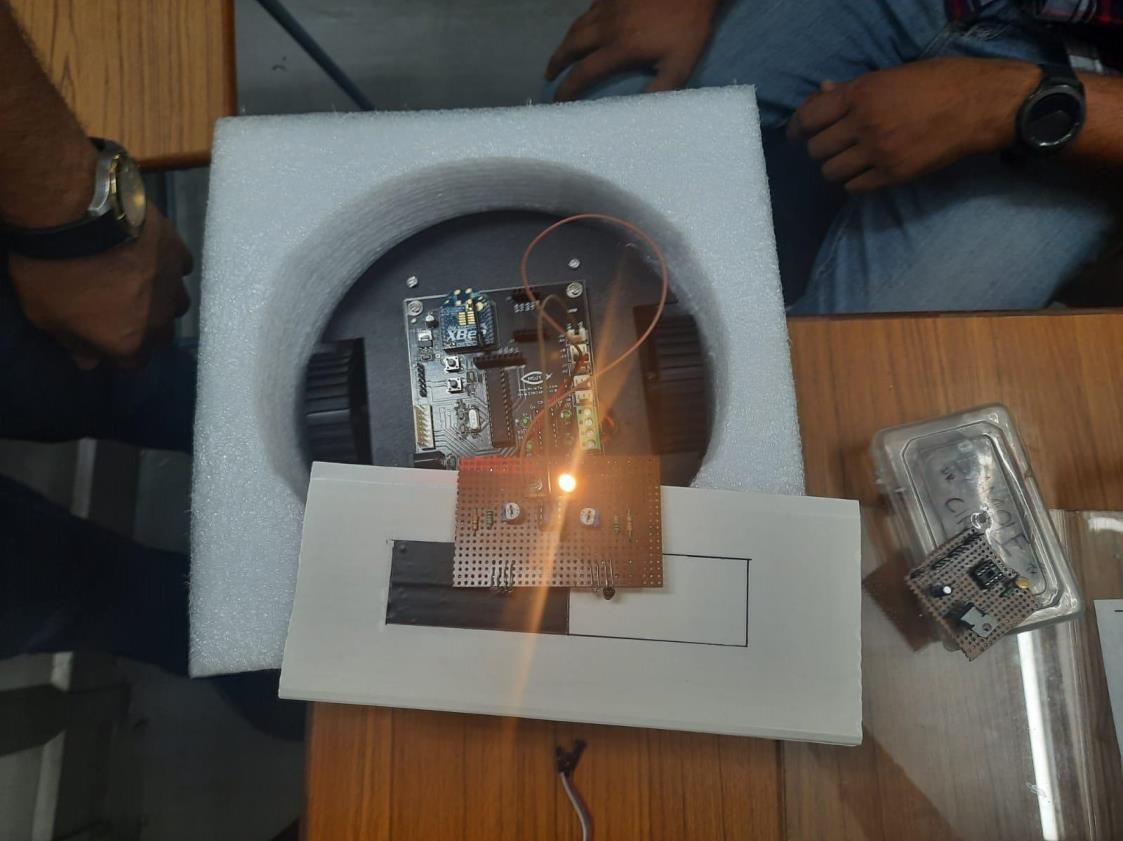
{

stopBuggy(); delay(1000);

}

}

## MODEL:



**RESULT ANALYSIS:**

In this experiment we are performing the bronze challenge i.e. the buggy is required to traverse the whole path twice, detect obstacles, park at the right position, stop at each gantry and work with XBee configuration. We understood the working of the working of the buggy along with the working of various sensors and zigbee for wireless communication.

**Signature of Faculty Member**