Minor Project – 2 :

RADAR PROJECT USING ARDUINO NANO AND ITS SENSORS

Done By: -

Name : Tushar Puranik

Guide : Muddesh H

Title: -

Radar Project using Arduino Nano and its sensors.

Aim: -

Create Radar system using Arduino and suitable sensors, whose output that is the main appearance is the visual narration in the suitable Application.

Problem Statement: -

Write a program and do suitable connections to make a radar system that detects any object that comes in the given radius. The visual representation of the radar system should be from 0 to 180` or 0 to 360`. Create an Alert System based on the size of the object that gets detected on the radar system and the output of Alert System should be as follows:

Range of detection (in °)	**Display**	**Message**	**LED’s**	**Buzzer (delay)**
5	Micro object	“*abc*”	Green	NA
10	Mini object	“*abc*”	Green blink	NA
15	Normal object	“*abc*”	Blue	NA
20	Lev – 1 object	“*abc*”	Blue blink	3000
25	Lev – 2 object	“*abc*”	Red blink	1500
30	Lev – 3 object	“*abc*”	Red	1000
35+	Mega object	“*abc*”	All LED’s	500

Survey(R/D) : -

- *Background and Objective:*

The objective of this project was to create an Radar System that detects any object if it appears in the range of 0 to 50 cm.

- *Materials and Procedure:*

For this project, 2 IDE's need to be used.

i. Arduino IDE:

The Arduino Integrated Development Environment is a Cross-Platform application that is written in functions from C and C++ It is used to write and upload programs to Arduino compatible boards, but also, with the help of third-party cores, other vendor development boards. Arduino IDE is a derivative of the Processing IDE,[10] however as of version 2.0, the Processing IDE will be replaced with the Visual Studio Code-based Eclipse Theia IDE framework.

For this project the programming of the sensors, OLED Monitor, LED's, Buzzer etc are done in the Arduino IDE. The distance calculated by the sensor and the angle is printed into the Serial Monitor.

ii. Processing IDE:

Processing is a free graphical library and integrated development environment (IDE) built for electronic arts, new media art, and visual design communities with the purpose of teaching non-programmers the fundamentals of computer programming in a visual context. Processing uses the Java language, with additional simplifications such as additional classes and aliased mathematical functions and operations. It also provides a graphical user interface for simplifying the compilation and execution stage. The Processing language and IDE have been the precursor to other projects including Arduino, Wiring and p5.js.

For this project the programming of the radar system is done in the Processing IDE. The Processing IDE takes the required information from the Serial Monitor of the Arduino IDE that are the distance and the angle variables and creates the Radar accordingly.

- *Results and Conclusions:*

Both the IDEs are unique in their own way but are dependent on the values achieved. The Processing IDE is the contemporary IDE assimilated in this project and the designing of Radar. The dimensions of the object were calculated in degree format by creating a variable whose values are the number of lines in the red region of the Radar that is created when any object is appeared.

- *Keywords and Methods:*

For creation of the Radar System in the Processing IDE, the keywords and functions that are used are as follows:

a) void setup()
b) void draw()
c) void serialEvent()
d) void DrawRadar()
e) void DrawRadarArcLine()
f) void DrawRadarAngledLine()
g) void DrawObject()
h) void DrawLine()
i) void DrawText()
j) int StringToInt()

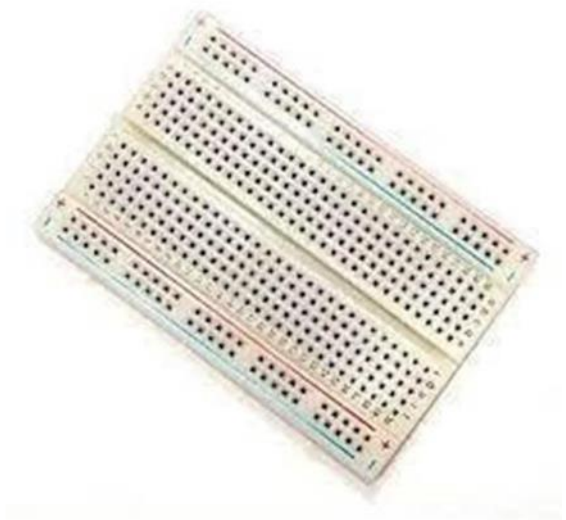
Requirements: -

- *Hardware:*

1. Arduino Nano : Arduino Nano is one type of microcontroller board. It can be built with a microcontroller like Atmega328.

2. Ultrasonic Sensor: An ultrasonic sensor is an electronic device that measures the distance of a target object by emitting ultrasonic sound waves, and converts the reflected sound into an electrical signal.

3. Bread Board : A breadboard is a solderless device for temporary prototype with electronics and test circuit designs.



4. Arduino Nano Cable:

The Arduino Nano is a small, complete, and breadboardfriendly board based on the ATmega328 . It lacks only a DC power jack, and works with a Mini-B USB cable instead of a standard one.

5. Led bulbs 5mm : These bulbs are used for low power small projects.

6. Piezoelectric Buzzer: A buzzer or beeper is an audio signalling device, which may be mechanical, electromechanical, or piezoelectric (piezo for short).

7. Resistors : A resistor is a passive two-terminal electrical component that implements electrical resistance as a circuit element.

8. OLED Monitor :

The display connects to Arduino using only four wires – two for power and two for data, making the wiring very simple. The data connection

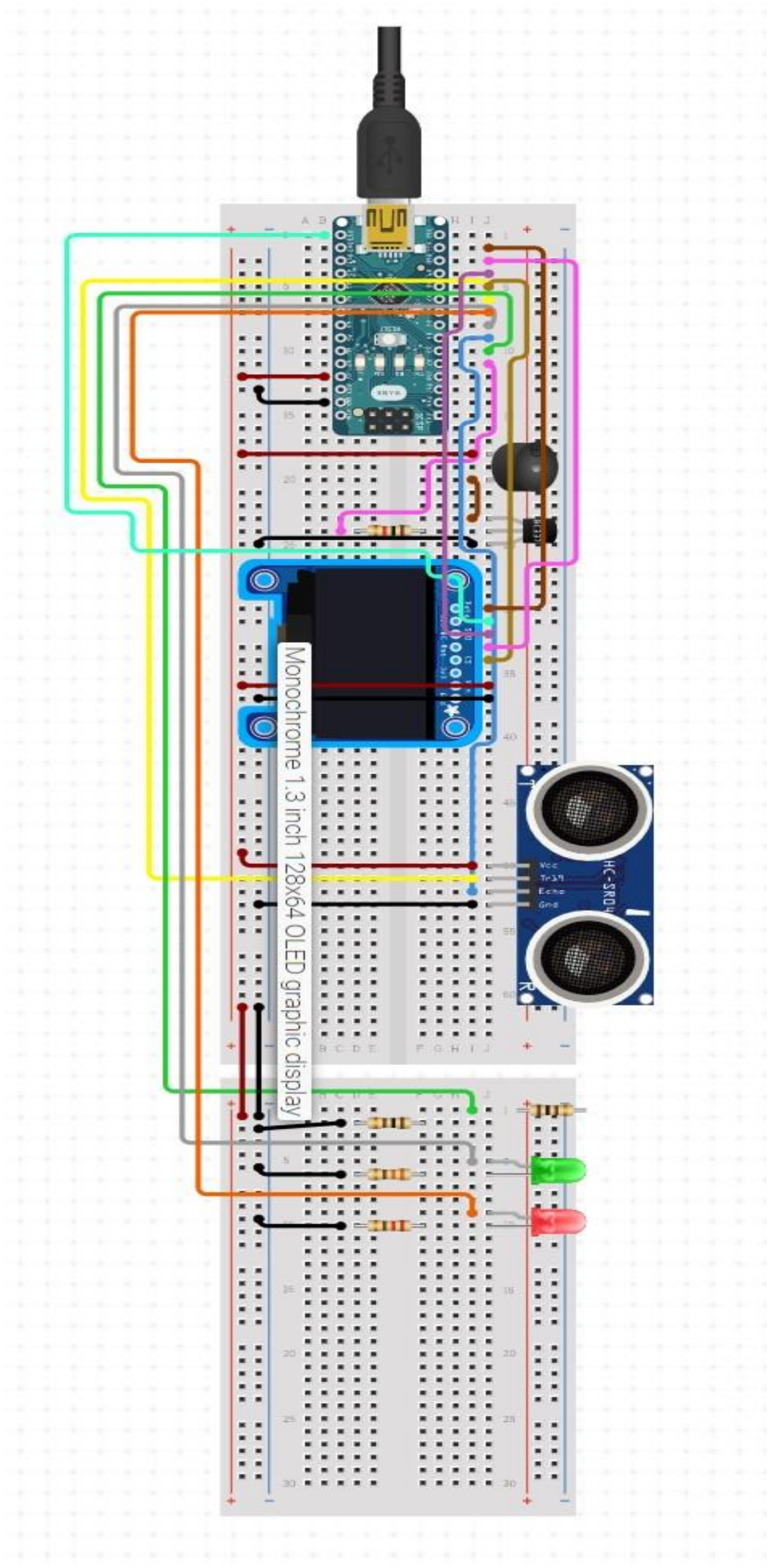
is I2C (I^{2}C, IIC or InterIntegrated Circuit). This interface is sometimes called TWI (Two Wire Interface).

9. Jumper Cables : Jumper wires are used to connect two points in a circuit. All Electronics stocks jumper wire in a variety of lengths and assortments. Frequently used with breadboards and other prototyping tools in order to make it easy to change a circuit as needed.

- *Software*

1. Any device with Arduino IDE on it.
2. Any Device with Processing IDE on it.
3. User must have basic knowledge required for programming of the Arduino Nano and Processing IDE.

Circuit Diagram: -



Source Code: -

- *Arduino Code:*

```
#include<Adafruit_GFX.h>
#include<Adafruit_SSD1306.h>
#include<wire.h>
#include<SPI.h>
Adafruit_SSD1306 display(-1);
const int led1=2;//green
const int led2=3;//red
const int led3=4;//blue
const int buzzer=5;
const int soundTriggerPin = 11;
const int soundEchoPin = 12;
const int startingAngle = 0;
const int minimumAngle = 0;
const int maximumAngle = 180;
const int rotationSpeed = 1;
int firstAngle;
int lastAngle;
int sizeOfObject;
int timer;

void setup(void)
{
  Serial.begin(9600);
  pinMode(led3,OUTPUT);
  pinMode(led1,OUTPUT);
  pinMode(led2,OUTPUT);
  pinMode(buzzer,OUTPUT);
  pinMode(soundTriggerPin, OUTPUT);
  pinMode(soundEchoPin, INPUT);
  display.begin(SSD1306_SWITCHCAPVCC ,
0x3C);
  display.clearDisplay();
  display.setTextColor(WHITE);

}
void loop(void)
{
    static int motorAngle = startingAngle;
    static int motorRotateAmount = rotationSpeed;
    int distance=CalculateDistance();
    delay(10);
    SerialOutput(motorAngle, CalculateDistance());
    motorAngle += motorRotateAmount;
    if(motorAngle <= minimumAngle || motorAngle
>= maximumAngle)
    {
        motorRotateAmount = -motorRotateAmount;
    }
    if(distance<=50)
        {
          timer++;
          sizeOfObject=timer;
        }
        else
        {
          timer=0;
          sizeOfObject=0;
        }
```

```
   oledMonitor(sizeOfObject);
}
void cd()
{
  display.clearDisplay();
}

void oledMonitor(int sizeOfObject)
{
   digitalWrite(led1,LOW);
    digitalWrite(led2,LOW);
    digitalWrite(led3,LOW);
 if(sizeOfObject<=5 && sizeOfObject>0)
  {
    cd();
    printing("Micro Object","No Problem");
    digitalWrite(led1,HIGH);
  }
  else if(sizeOfObject<=10 && sizeOfObject>5)
  {
```

```
  cd();
  printing("Mini Object","Good To Go");
  digitalWrite(led1,HIGH);
  delay(100);
  digitalWrite(led1,LOW);
}
else if(sizeOfObject>10 && sizeOfObject<=15)
{
  cd();
  printing("Normal Object","Observe");
  digitalWrite(led3,HIGH);
}
else if(sizeOfObject>15 && sizeOfObject<=20)
{
  cd();
  printing("Lev-1 Object","Be Careful");
  digitalWrite(led3,HIGH);
  delay(100);
  digitalWrite(led3,LOW);
   digitalWrite(buzzer,HIGH);
   delay(100);
  digitalWrite(buzzer,LOW);
}
else if(sizeOfObject>20 && sizeOfObject<=25)
{
  cd();
  printing("Lev-2 Object","Risk");
  digitalWrite(led2,HIGH);
  digitalWrite(buzzer,HIGH);
  delay(100);
  digitalWrite(buzzer,LOW);
  digitalWrite(led2,LOW);
}
else if(sizeOfObject>25 && sizeOfObject<=30)
{
  cd();
  printing("Lev-3 Object","High Risk");
  digitalWrite(led2,HIGH);
  digitalWrite(buzzer,HIGH);
  delay(100);
  digitalWrite(buzzer,LOW);
}
else if(sizeOfObject>30 && sizeOfObject<=35)
{
  cd();
  printing("Lev-3 Object","Very High Risk");
  digitalWrite(led1,HIGH);
  digitalWrite(led2,HIGH);
  digitalWrite(led3,HIGH);
  digitalWrite(buzzer,HIGH);
  delay(100);
  digitalWrite(buzzer,LOW);
   digitalWrite(led1,LOW);
  digitalWrite(led2,LOW);
  digitalWrite(led3,LOW);
}
else if(sizeOfObject>35)
{
  cd();
  printing("Mega Object","Escape");
```

```
    digitalWrite(led1,HIGH);
    digitalWrite(led2,HIGH);
    digitalWrite(led3,HIGH);
    digitalWrite(buzzer,HIGH);
    delay(500);
    digitalWrite(buzzer,LOW);
  }
}
void printing(String x,String y)
{
  display.setTextSize(1);
  display.setCursor(25,5);
  display.println(x);
  display.setCursor(25,20);
  display.println(y);
  display.display();

}
int CalculateDistance(void)
{
    //digitalWrite(soundTriggerPin, LOW);
    //delayMicroseconds(2);
    digitalWrite(soundTriggerPin, HIGH);
    delayMicroseconds(10);
    digitalWrite(soundTriggerPin, LOW);
      long duration = pulseIn(soundEchoPin, HIGH);
    float distance = duration * 0.017F;
    return int(distance);
}
void SerialOutput(const int angle, const int distance)
{
    String angleString = String(angle);
    String distanceString = String(distance);
    Serial.println(angleString + "," + distanceString);
}
```

- *Processing Code:*

```
import processing.serial.*;
import java.awt.event.KeyEvent;
import java.io.IOException;

Serial myPort;
PFont orcFont;
int iAngle;
int iDistance;
void setup() {
size(1350, 760);
   smooth();

   myPort = new Serial(this, "COM3", 9600);
   myPort.clear();
   myPort.bufferUntil('\n');
     orcFont = loadFont("OCRAExtended-30.vlw");
}
void draw()
```

```
{
    fill(98, 245, 31);
    textFont(orcFont);
    noStroke();
    fill(0, 4);
    rect(0, 0, width, 0.935 * height);
    fill(98, 245, 31);

    DrawRadar();
    DrawLine();
    DrawObject();
    DrawText();
}
void serialEvent (Serial myPort)
{
    try {
        String data = myPort.readStringUntil('\n');
            if (data == null) {
            return;
        }
```

```
         int commaIndex = data.indexOf(",");
        String angle = data.substring(0, 
commaIndex);
        String distance = 
data.substring(commaIndex+1, data.length()-1);
   iAngle = StringToInt(angle);
        iDistance = StringToInt(distance);
    } catch(RuntimeException e) {}
}
void DrawRadar()
{
    pushMatrix();
    translate(width/2, 0.926 * height);
    noFill();
    strokeWeight(2);
    stroke(98, 245, 31);

    // draws the arc lines
    DrawRadarArcLine(0.9375);
    DrawRadarArcLine(0.7300);
    DrawRadarArcLine(0.5210);
    DrawRadarArcLine(0.3130);

    // draws the angle lines
    final int halfWidth = width/2;
    line(-halfWidth, 0, halfWidth, 0);
    for(int angle = 30; angle <= 150; angle+=30)
{
        DrawRadarAngledLine(angle);
    }
    line(-halfWidth * cos(radians(30)), 0, halfWidth,
0);
        popMatrix();
}
void DrawRadarArcLine(final float coefficient)
{
    arc(0, 0, coefficient * width, coefficient * width,
PI, TWO_PI);
}
void DrawRadarAngledLine(final int angle){
    line(0, 0, (-width/2) * cos(radians(angle)), (-
width/2) * sin(radians(angle)));
```

```
}
void DrawObject()
{
    pushMatrix();
    translate(width/2, 0.926 * height);
    strokeWeight(9);
    stroke(255,10,10);
    int pixsDistance = int(iDistance * 0.020835 *
height);
    if(iDistance < 40 && iDistance != 0) {
        float cos = cos(radians(iAngle));
        float sin = sin(radians(iAngle));
        int x1 = +int(pixsDistance * cos);
        int y1 = -int(pixsDistance * sin);
        int x2 = +int(0.495 * width * cos);
        int y2 = -int(0.495 * width * sin);

        line(x1, y1, x2, y2);
    }
      popMatrix();
}
```

```
void DrawLine()
{
    pushMatrix();
    strokeWeight(9);
    stroke(30, 250, 60);
    translate(width/2, 0.926 * height);

    float angle = radians(iAngle);
    int x = int(+0.88 * height * cos(angle));
    int y = int(-0.88 * height * sin(angle));
    line(0, 0, x, y);
    popMatrix();
}
void DrawText()
{
    pushMatrix();
    fill(0, 0, 0);
    noStroke();
    rect(0, 0.9352 * height, width, height);
    fill(98, 245, 31);
    textSize(25);
    text("10cm", 0.6146 * width, 0.9167 * height);
    text("20cm", 0.7190 * width, 0.9167 * height);
    text("30cm", 0.8230 * width, 0.9167 * height);
    text("40cm", 0.9271 * width, 0.9167 * height);


    textSize(40);
    text("Object: " + (iDistance > 40 ? "Out of
Range" : "In Range"), 0.125 * width, 0.9723 *
height);
    text("Angle: " + iAngle + " °", 0.52 * width,
0.9723 * height);
    text("Distance: ", 0.74 * width, 0.9723 * height);
    if(iDistance < 40) {
        text("        " + iDistance +" cm", 0.775 *
width, 0.9723 * height);
    }
     textSize(25);
    fill(98, 245, 60);
    translate(0.5006 * width + width/2 *
cos(radians(30)), 0.9093 * height - width/2 *
sin(radians(30)));
```

```
    rotate(-radians(-60));
    text("30°",0,0);

    resetMatrix();

    translate(0.497 * width + width/2 *
cos(radians(60)), 0.9112 * height - width/2 *
sin(radians(60)));
    rotate(-radians(-30));
    text("60°",0,0);
     resetMatrix();
     translate(0.493 * width + width/2 *
cos(radians(90)), 0.9167 * height - width/2 *
sin(radians(90)));
    rotate(radians(0));
    text("90°",0,0);
    resetMatrix();

    translate(0.487 * width + width/2 *
cos(radians(120)), 0.92871 * height - width/2 *
sin(radians(120)));
    rotate(radians(-30));
    text("120°",0,0);
    resetMatrix();


    translate(0.4896 * width + width/2 *
cos(radians(150)), 0.9426 * height - width/2 *
sin(radians(150)));
    rotate(radians(-60));
    text("150°",0,0);
    popMatrix();
}
int StringToInt(String string)
{
    int value = 0;
    for(int i = 0; i < string.length(); ++i) {
        if(string.charAt(i) >= '0' && string.charAt(i)
<= '9') {
            value *= 10;
            value += (string.charAt(i) - '0');
        }}
     return value;
}
```

Applications: -

- Air Traffic Control
- Military Control
- Defence Systems
- Obstacle avoiding robots
- Vital sign Monitoring
- Meteorologists use radar to monitor precipitation and wind.
- Marine Radars are used to measure the bearing and distance of ships.

Outcomes: -

We have successfully created a Radar System using the suitable IDEs. The results satisfy the instructions of the given table. The connections are done properly to get correct results.

THANK YOU

Done by: -

Name : Tushar Puranik

Guide : Muddesh H