NORTHERN UNIVERSITY
OF BUSINESS & TECHNOLOGY KHULNA

Lab Report

Group: 05

Course Code: CSE 3102

Course Title: Computer Architecture Lab

Department of Computer Science and Engineering (CSE)

Submitted to:

Md. Mossadek Touhid

Lecturer, Department of CSE

Northern University of Business and technology Khulna

Date of Submission: 10/09/2024

Submitted by:

Masud Mehrab

ID: 11220120733

Riaz Ahmed

ID: 11220120740

Hridoy Hossain

ID: 200120530

Md. Sayeed Hossain

ID:11220120750

Md. Mahafujul Karim Sheikh

ID:11220120735

Sudarsan Sarker

ID:11220120752

Arghya Saha

ID: 11200120456

▪ Introduction

Project Name: Arduino Military Radar Using Ultrasonic Sensor.

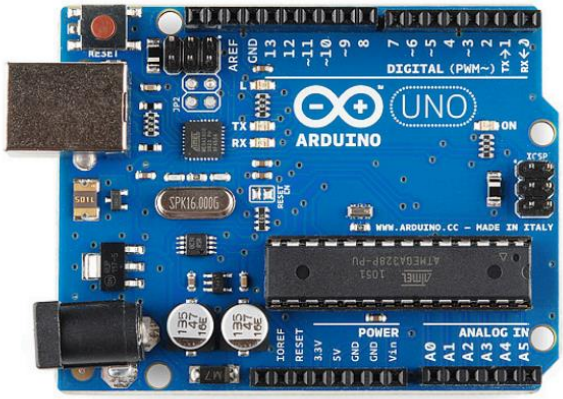
Objective: An Arduino military radar is a project that uses an Arduino to create a radar system capable of detecting objects and measuring their distance using ultrasonic sensor.

Radar systems play a pivotal role in both military and civilian applications, ranging from surveillance and navigation to object detection and tracking. By working on this project, we gain hands-on experience with a technology that has significant real-world implications. This practical exposure not only enhances our understanding of radar systems but also prepares us for future challenges in the field of technology and engineering.

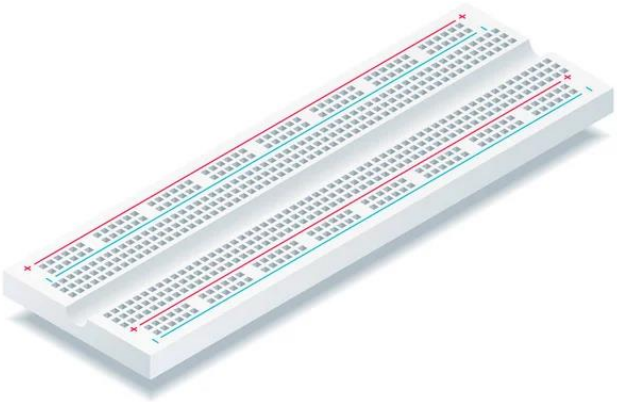
By focusing on a military radar system, we contribute to the advancement of defense technology. Our project, though on a smaller scale, mirrors the larger efforts in the defense sector to enhance surveillance and security. It instills a sense of purpose and responsibility, knowing that our work can potentially contribute to national security and safety.

▪ Required Components

1. Arduino Uno: This is a popular microcontroller board that acts as the brain of your project. It controls the other components and executes your program.



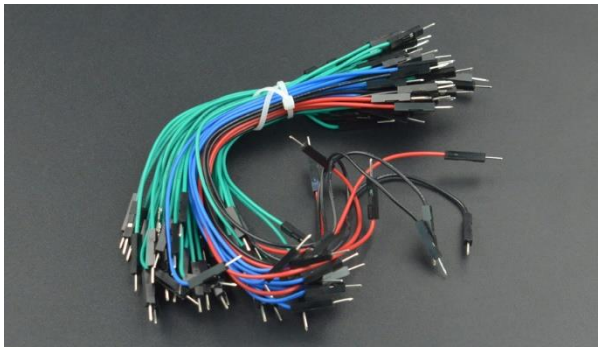
2. Breadboard: This is a prototyping tool used to connect the components together temporarily. It has a grid of holes where you can insert the wires and components.



3. Servo Motor: This is a mechanical actuator that can rotate to a specific angle. It's controlled by the Arduino using a pulse-width modulation (PWM) signal.

4. Ultrasonic Sensor: This sensor measures distance and detect objects by emitting sound waves and measuring the time it takes for the echo to return.

5. Wires: These wires to connect the ultrasonic sensor, servo motor, and Arduino Uno together on the breadboard.

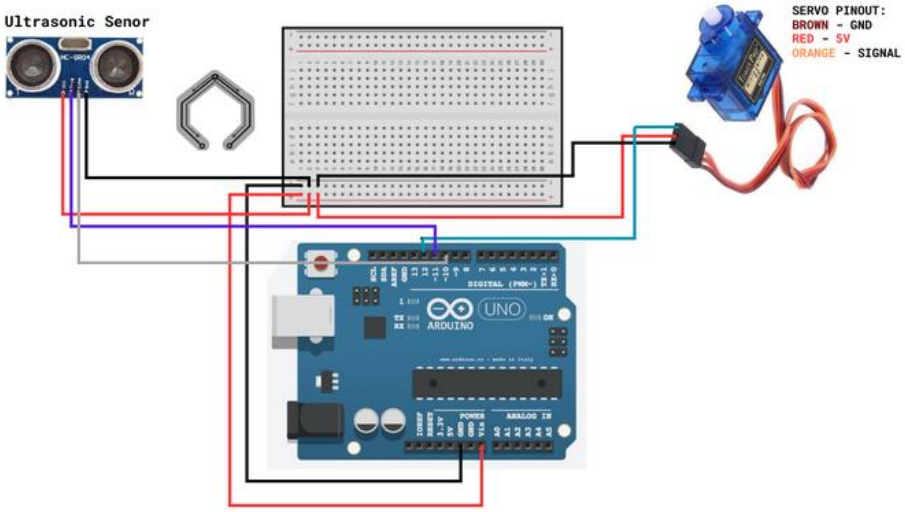


6. Power Supply or USB Cable: Provide power to the Arduino board. USB Type A cables can be standard (USB 2.0) for high-speed.

7. Arduino Software (IDE): This is the software you use to write and upload your program to the Arduino Uno.

8. Processing: A graphical library to write code and 3D visualization our Radar.

▪ Hardware Block Diagram



Pic: Diagram for Radar with Arduino using ultrasonic sensor

The provided diagram shows a circuit diagram for connecting an ultrasonic sensor and a servo motor to an Arduino Uno. Here are the steps and connections involved in applying this diagram:

1. Setup and initialization:

Connect the Arduino to the computer and set up the hardware components, including the ultrasonic sensor and the servo motor, breadboard as shown in the circuit diagram.

2: Connect the components:
Connect the VCC pin of the ultrasonic sensor to the 5V pin on the Arduino.
Connect the Trig pin of the ultrasonic sensor to digital pin 2 on the

Connect the Echo pin of the ultrasonic sensor to digital pin 3 on the Arduino.
Connect the GND pin of the ultrasonic sensor to the GND pin on the Arduino.
Connect the GND pin of the servo motor to the GND pin on the Arduino.
Connect the 5V pin of the servo motor to the 5V pin on the Arduino.
Connect the Signal pin of the servo motor to digital pin 9 on the Arduino.

3: Write and Upload Arduino code:

The code will read the distance measured by the ultrasonic sensor and control the servo motor to rotate based on the distance. Connect the Arduino to your computer using a USB cable. Open the Arduino IDE and upload the code to the board.

4. Display using processing:

Use the Processing IDE (a programming environment) to create a visual representation of the radar system. The data received from the Arduino is processed and displayed on the screen in real-time, showing the position of any detected objects within the range of the ultrasonic sensor. The process repeats continuously, with the radar system scanning the area, detecting objects, and updating the display in real-time

5. Object Detection:

The servo motor rotates the ultrasonic sensor, scanning the area. The ultrasonic sensor sends out sound waves and waits for any echoes to return. If an object is detected within the sensor's range, the Arduino records the distance and angle of the object.

▪ Source Code

▫Arduino Code:

The shared code is written in C++ that designed to control a servo motor and measure distances using an ultrasonic sensor. Here's a breakdown of this full program:

```
#include <Servo.h>.
const int trigPin = 10;
const int echoPin = 11;
long duration;
int distance;
Servo myServo;
void setup() {
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  Serial.begin(9600);
  myServo.attach(12);
}
void loop() {
  for(int i=15;i<=165;i++){
  myServo.write(i);
  delay(30);
  distance = calculateDistance();
  Serial.print(i);
  Serial.print(",");
```

```
  Serial.print(distance); // Sends the distance value into the Serial Port
  Serial.print(".");
  }
  for(int i=165;i>15;i--){
  myServo.write(i);
  delay(30);
  distance = calculateDistance();
  Serial.print(i);
  Serial.print(",");
  Serial.print(distance);
  Serial.print(".");
  }
}
// Function for calculating the distance measured by the Ultrasonic sensor
int calculateDistance(){

  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  // Sets the trigPin on HIGH state for 10 micro seconds
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  duration = pulseIn(echoPin, HIGH);
  distance= duration*0.034/2;
  return distance;
}
```

▫Processing Code:

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

String angle="";
String distance="";
String data="";
String noObject;
float pixsDistance;
int iAngle, iDistance;
int index1=0;
int index2=0;
PFont orcFont;
void setup() {

 size (1200, 700);
 smooth();
 myPort = new Serial(this,"COM5", 9600);
 myPort.bufferUntil('.');
'.'. So actually it reads this: angle,distance.
}
void draw() {
```

```
  fill(98,245,31);

  noStroke();
  fill(0,4);
  rect(0, 0, width, height-height*0.065);

  fill(98,245,31);

  drawRadar();
  drawLine();
  drawObject();
  drawText();
}
void serialEvent (Serial myPort) {
  data = myPort.readStringUntil('.');
  data = data.substring(0,data.length()-1);

  index1 = data.indexOf(",");
  angle= data.substring(0, index1);
  distance= data.substring(index1+1, data.length());
  iAngle = int(angle);
  iDistance = int(distance);
}
void drawRadar() {
  pushMatrix();
  translate(width/2,height-height*0.074);
```

```
  noFill();
  strokeWeight(2);
  stroke(98,245,31);
  // draws the arc lines
  arc(0,0,(width-width*0.0625),(width-width*0.0625),PI,TWO_PI);
  arc(0,0,(width-width*0.27),(width-width*0.27),PI,TWO_PI);
  arc(0,0,(width-width*0.479),(width-width*0.479),PI,TWO_PI);
  arc(0,0,(width-width*0.687),(width-width*0.687),PI,TWO_PI);
  // draws the angle lines
  line(-width/2,0,width/2,0);
  line(0,0,(-width/2)*cos(radians(30)),(-width/2)*sin(radians(30)));
  line(0,0,(-width/2)*cos(radians(60)),(-width/2)*sin(radians(60)));
  line(0,0,(-width/2)*cos(radians(90)),(-width/2)*sin(radians(90)));
  line(0,0,(-width/2)*cos(radians(120)),(-width/2)*sin(radians(120)));
  line(0,0,(-width/2)*cos(radians(150)),(-width/2)*sin(radians(150)));
  line((-width/2)*cos(radians(30)),0,width/2,0);
  popMatrix();
}
void drawObject() {
  pushMatrix();
  translate(width/2,height-height*0.074);
  strokeWeight(9);
  stroke(255,10,10); // red color
  pixsDistance = iDistance*((height-height*0.1666)*0.025);

  if(iDistance<40){
```

```
  line(pixsDistance*cos(radians(iAngle)),-
pixsDistance*sin(radians(iAngle)),(width-width*0.505)*cos(radians(iAngle)),-
(width-width*0.505)*sin(radians(iAngle)));
  }
  popMatrix();
}
void drawLine() {
  pushMatrix();
  strokeWeight(9);
  stroke(30,250,60);
  translate(width/2,height-height*0.074);
  line(0,0,(height-height*0.12)*cos(radians(iAngle)),-(height-
height*0.12)*sin(radians(iAngle)));
  popMatrix();
}
void drawText() {

  pushMatrix();
  if(iDistance>40) {
  noObject = "Out of Range";
  }
  else {
  noObject = "In Range";
  }
  fill(0,0,0);
  noStroke();
  rect(0, height-height*0.0648, width, height);
```

```
  fill(98,245,31);
  textSize(25);


  text("10cm",width-width*0.3854,height-height*0.0833);
  text("20cm",width-width*0.281,height-height*0.0833);
  text("30cm",width-width*0.177,height-height*0.0833);
  text("40cm",width-width*0.0729,height-height*0.0833);
  textSize(40);
  text("SciCraft ", width-width*0.875, height-height*0.0277);
  text("Angle: " + iAngle +"  ", width-width*0.48, height-height*0.0277);
  text("Distance: ", width-width*0.26, height-height*0.0277);
  if(iDistance<40) {
  text("        " + iDistance +" cm", width-width*0.225, height-height*0.0277);
  }
  textSize(25);
  fill(98,245,60);
  translate((width-width*0.4994)+width/2*cos(radians(30)),(height-
height*0.0907)-width/2*sin(radians(30)));
  rotate(-radians(-60));
  text("30 ",0,0);
  resetMatrix();
  translate((width-width*0.503)+width/2*cos(radians(60)),(height-
height*0.0888)-width/2*sin(radians(60)));
  rotate(-radians(-30));
  text("60 ",0,0);
  resetMatrix();
```

```
  translate((width-width*0.507)+width/2*cos(radians(90)),(height-
height*0.0833)-width/2*sin(radians(90)));
  rotate(radians(0));
  text("90 ",0,0);
  resetMatrix();
  translate(width-width*0.513+width/2*cos(radians(120)),(height-
height*0.07129)-width/2*sin(radians(120)));
  rotate(radians(-30));
  text("120 ",0,0);
  resetMatrix();
  translate((width-width*0.5104)+width/2*cos(radians(150)),(height-
height*0.0574)-width/2*sin(radians(150)));
  rotate(radians(-60));
  text("150 ",0,0);
  popMatrix();
}
```

▪ Conclusion

Our choice to develop an Arduino-based military radar system using an ultrasonic sensor is driven by its real-world relevance, the opportunity for skill development, interdisciplinary learning, and the potential for innovation.

By focusing on a military radar system, we contribute to the advancement of defense technology. Our project, though on a smaller scale, mirrors the larger efforts in the defense sector to enhance surveillance and security.

Radar systems are widely used in military and civilian applications for surveillance, navigation, and object detection. This project gives you hands-on experience with technology that has significant real-world relevance.

Working on this project helps us develop valuable skills in electronics, programming, and data visualization, which are essential for a career in technology and engineering.