



Lab Manual

Course Name - Engineering Informatics

Course Code – 2207232


Course Credits – 04

Course Type - Program Core Course

Class & Branch – SY BTECH [All Branches]

MIT Academy of Engineering Alandi Pune

Revision 2023

 MIT Academy of Engineering AN AUTONOMOUS INSTITUTE	INDEX & CERTIFICATE	
	ACADEMIC YEAR	
Alandi (D), Pune – 412105	SEM/TRI	
DEPARTMENT OFENGG.	CLASS & BLOCK	

Module No	Title	Page no	Assessment points	Remark
01	Simulation of IoT Systems			
02	Data Acquisition , Storage and Retrieval Systems using Arduino / Raspberry Pi			
03	Data Presentation & Visualization			
04	Activity Details – For Continuous Assessment			

This is to certify that Mr/Ms -----
-----Roll No -----has successfully completed
the experiments for the course **Engineering Informatics** for
academic year 20_____.

Sign
Student

Sign
Course instructor

Module No. 1	Title: Simulation of IoT Systems	6 Hours
Design and implement an Arduino-Based 3D Scanner for Capturing and Digitizing Physical Objects. <ol style="list-style-type: none"> Identify the home appliances that require human interaction for its operations State the need of automation. Identify system component Design circuit diagram Assemble system components Program the interface System Testing 		

Problem statement –

Design and build a 3D scanner using Arduino that can capture physical objects and generate digital 3D models. The goal is to create an affordable and accessible solution for 3D scanning.

Objectives –

- Identification of components
- Circuit development
- Connections of various components
- Circuit simulation

Outcome –

1. Affordability: Creating a low-cost 3D scanner using Arduino components.
2. Customization: Adapting the design to different object sizes and shapes.
3. Digital Modeling: Transforming physical objects into digital 3D models.
4. Educational Value: Learn about laser scanning, motors, and programming.

Theory –

1. Laser Scanning Principle:
 - The heart of the 3D scanner lies in laser scanning. A laser beam is projected onto the object's surface, and the reflected light is captured by a sensor.
 - By measuring the time it takes for the laser to bounce back, we can determine the distance from the scanner to each point on the object.
 - These distance measurements create a point cloud, which forms the basis for the 3D model.
2. Stepper Motors:
 - Rotational Movement: One stepper motor rotates the model platform. It ensures a complete 360-degree scan of the physical object.
 - Vertical Adjustment: The second stepper motor controls the vertical movement of the laser scanner. It allows scanning from different angles.
3. Arduino Uno as the Brain:
 - The Arduino Uno coordinates the entire system.

- It receives commands to move the motors and trigger the laser sensor.
- Precise synchronization ensures accurate data collection.
- 4. Data Acquisition and Storage:
 - The laser scanner captures distance data point by point.
 - This data is saved to an SD card using an SD card module.
 - Later, on a PC, you can convert this raw scan data into usable 3D model files (e.g., STL or OBJ formats).
- 5. Calibration and Accuracy:
 - Proper calibration of motor steps, laser alignment, and sensor resolution is crucial.
 - Achieving consistent accuracy requires fine-tuning and testing.

Benefits –

- Cost-Effective: Arduino offers an affordable platform for DIY projects.
- Customizable: You can adapt the system to various scanning needs.
- Digital Modeling: Transform real-world objects into digital models for manipulation, analysis, or 3D printing.

Procedure –

- Assemble the frame using supporting rods and mounts.
- Attach stepper motors to the frame.
- Mount the laser sensor on a screw-based arrangement.
- Wire the Arduino Uno to the motors and laser sensor.
- Place an object on the platform and run the scanner to test.
- Save scan data to an SD card and convert it to 3D model files on a PC.

a. Enlist the components.

1. 1 x Arduino NANO/UNO
2. 1 x Sharp 0A51SK
3. 2 x A4988 driver
4. 2 x NEMA17 motors
5. 1 x SD module
6. 1 x 12V-2A power supply
7. 1 x screw PCB connector
8. Female pins
9. 1 x micro-SD card
10. 1 x drilled PCB
11. 2 x 47uF cap
12. 1 x limit switch
13. Wires
14. 2 x smooth rods (200mm/8mm)
15. 4 x LM8UU bearing
16. 1 x lead screw (200mm/8mm)
17. M3 screws

b. State the need.

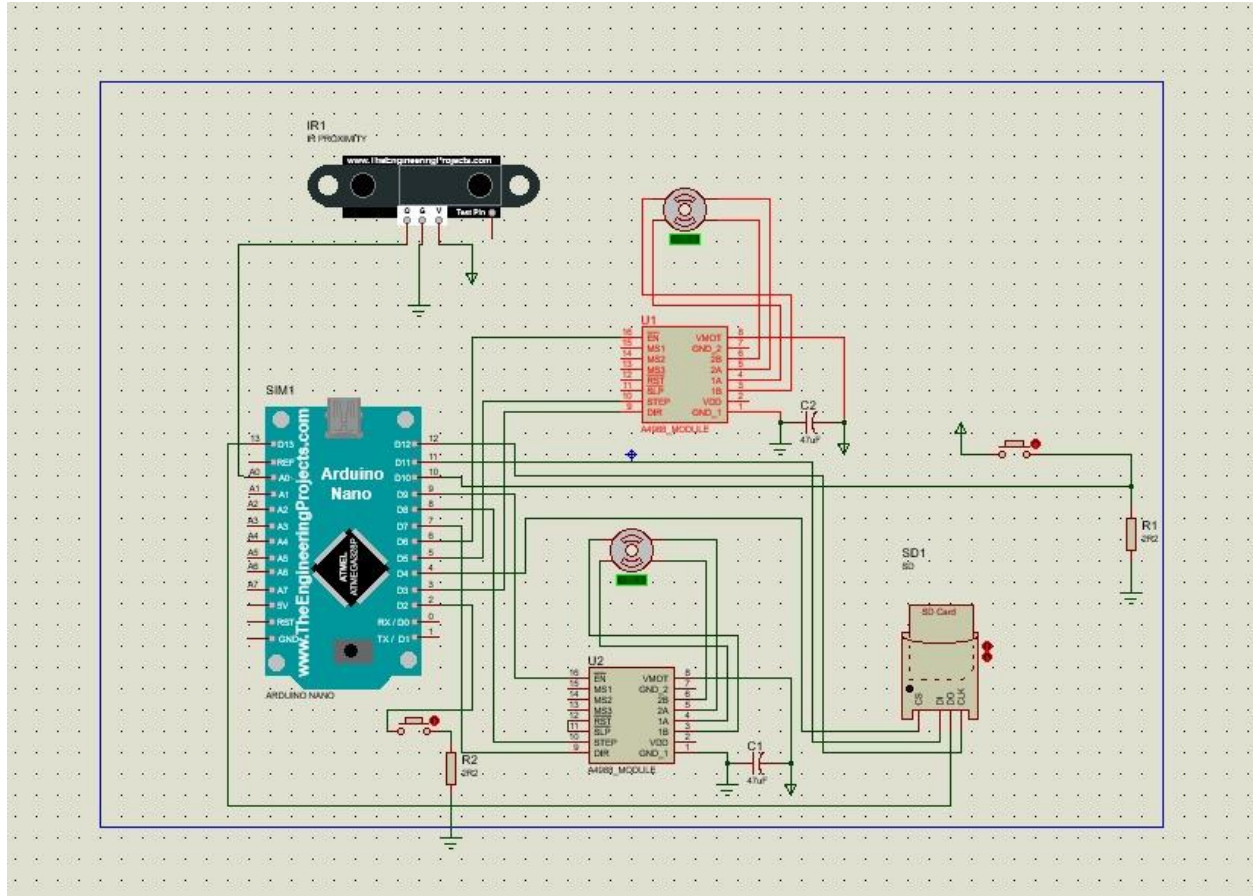
The need for this Arduino 3D scanner project stems from the desire to create a cost-effective, DIY solution for 3D scanning of small objects. Traditional 3D scanners can be expensive, so this project offers an affordable alternative using readily available components like an Arduino Uno, stepper motors, and a distance sensor. It enables hobbyists, makers, and educators to explore 3D scanning technology, develop their technical skills, and create digital models of physical objects for various applications such as 3D printing and digital archiving.

c. Identify system component available in Simulation Software –

1. Arduino NANO/UNO
2. 1 x Sharp 0A51SK
3. 2 x A4988 driver
4. 2 x NEMA17 motors
5. 1 x SD module
6. 1 x 12V-2A power supply
7. 1 x screw PCB connector
8. 1 x micro-SD card
9. 1 x drilled PCB
10. 2 x 47uF cap
11. 1 x limit switch
12. Wires

d. Design circuit diagram [Draw the diagram here]

e. Assemble system components - Simulate the circuit in Simulation Software



f. Program the interface – Write the program if any.

```
#include <Stepper.h>
#include <SD.h>

// Editable variables
int scan_amount = 40; // Amount of scans for each point
String file = "scan_001.txt"; // Name of the saved file on the SD card
int z_axis_height = 12; // in cm
int step_delay = 3000; // in us
float z_layer_height = 0.5; // in mm
int lead_screw_rotations_per_cm = 8;
int steps_per_rotation_for_motor = 200;
```

```

int distance_to_center = 8; // in cm

// Stepper motor setup
const int steps_per_revolution = steps_per_rotation_for_motor * lead_screw_rotations_per_cm;
// calculate total steps per cm
Stepper myStepper(steps_per_revolution, 8, 9, 10, 11); // pins for the stepper motor

void setup() {
  // Initialize SD card
  Serial.begin(9600);
  while (!SD.begin()) {
    Serial.println("SD card initialization failed!");
    return;
  }
  Serial.println("SD card initialized.");

  // Set up the stepper motor
  myStepper.setSpeed(60); // set the speed of the stepper motor (rpm)
}

void loop() {
  // Perform scanning
  for (int height = 0; height < z_axis_height; height++) {
    // Move to the next layer
    myStepper.step(steps_per_revolution / (10 / z_layer_height)); // move the stepper motor
    delayMicroseconds(step_delay);

    // Scan the current layer
    for (int scan = 0; scan < scan_amount; scan++) {
      // Read sensor data
      int sensorValue = analogRead(A0); // replace A0 with the actual sensor pin

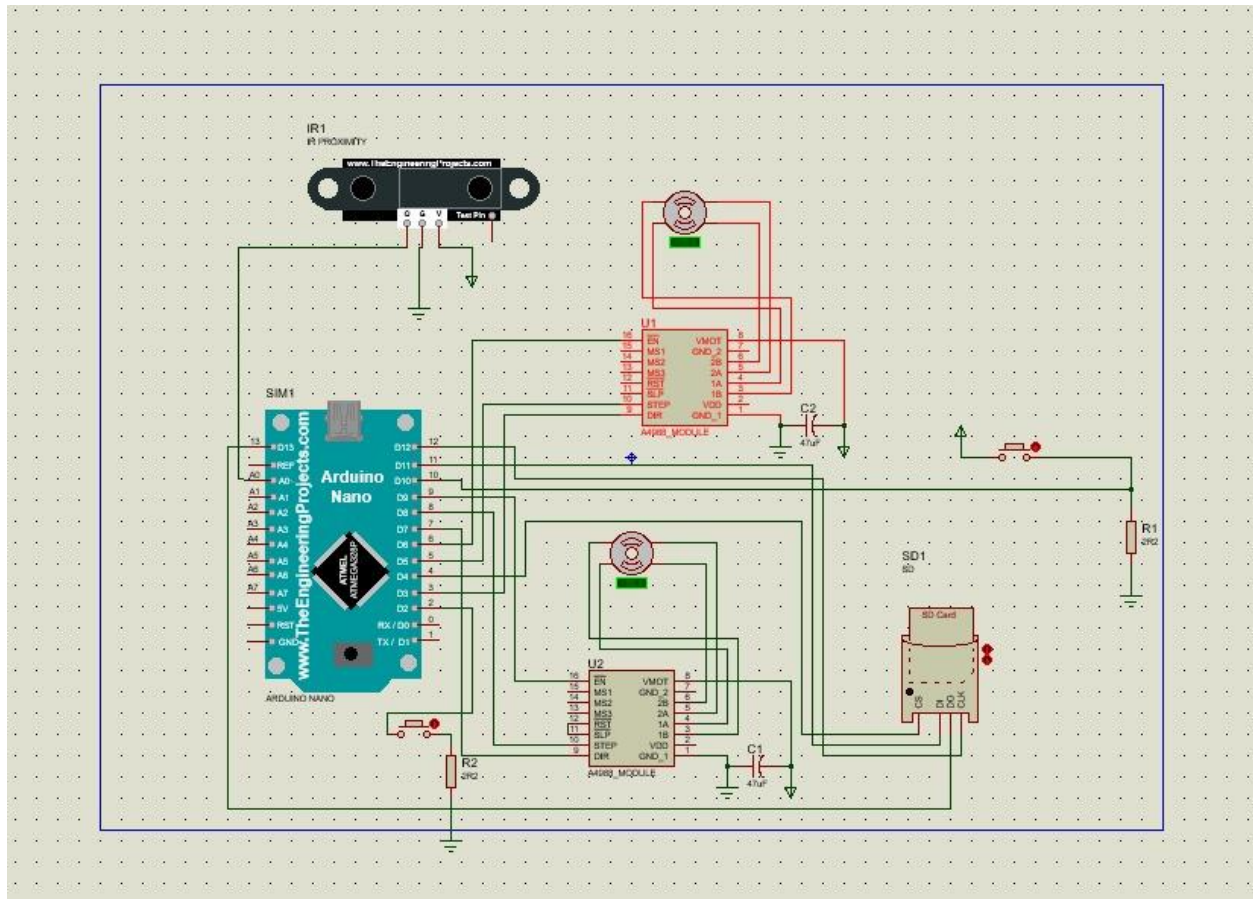
      // Save data to SD card
      File dataFile = SD.open(file, FILE_WRITE);
      if (dataFile) {
        dataFile.println(String(height * 10) + "," + String(sensorValue));
        dataFile.close();
      } else {
        Serial.println("Error opening file for writing");
      }
    }
  }

  // Stop scanning
  Serial.println("Scanning complete");
  while (true); // halt the program
}

```

g. System Testing – Test the implemented system and observe the output. Attach printout of simulated circuit. Write your observations while testing the circuit. Enlist errors occurred etc.

Simulated Circuit:



Observations:

Component Integration: All components were correctly placed and connected as per the circuit diagram.

Functionality Test: The Arduino successfully controlled the stepper motors and processed distance measurements from the sensor.

Data Storage: The data was correctly saved to the SD card.

Errors and Corrections:

Wiring Issues: Initial wiring errors caused incorrect motor movements, corrected by verifying connections.

Sensor Calibration: The distance sensor needed calibration for accurate measurements, resolved by adjusting the sensor position and parameters.

Power Supply: Inconsistent power supply caused interruptions, fixed by using a stable power source.

Overall, the system performed well after troubleshooting and adjustments, successfully capturing and storing 3D scan data.

Result & Conclusion - Elaborate your learning and findings after completion of Module 1.

Results:

The Arduino 3D scanner was successfully assembled and operated as intended.

It scanned objects and saved the data to an SD card.

Conclusion:

This project provided valuable hands-on experience with electronics, motor control, and data processing. It demonstrated the feasibility of creating a low-cost, DIY 3D scanner using Arduino, making 3D scanning technology more accessible and educational. The project underscored the importance of component integration and calibration for achieving accurate and reliable scans.