GUJARAT TECHNOLOGICAL UNIVERSITY

Vishwakarma Government Engineering College

Chandkheda, Ahmedabad

Project Report On

Micro-Controller Based Voting Machine

Under subject of
Design Engineering 2B
Semester – 6th (EC)

Submitted by:
Group Id: 676025

Sr. No.	Student Name	Enrollment Number
1	Sumit Vaghela	220170111141
2	Mohit Topiya	220170111139
3	Varu Anand	220170111142
4	Darshan Dodiya	220170111151

PROF. ARUN NANDURBARKAR
(FACULTY GUIDE/MENTOR)

Acknowledgement

We would like to express our heartfelt gratitude to our project guide, Prof. Arun Nandurbarkar for his unwavering support, invaluable insights, and constructive feedback throughout the course of this project. His guidance has been instrumental in shaping the quality and direction of our work.

We extend our sincere thanks to the faculty of Design Engineering, whose knowledge and expertise provided us with a strong foundation for completing our domain work. Their guidance in creating various models and project documents was essential to our success.

We are deeply appreciative of the encouragement and assistance from our friends and colleagues, whose inspirational support played a significant role in helping us navigate the challenges we faced during this project.

Thank you all for contributing to the successful completion of our project.

Abstract

This project focuses on designing and developing an “Electronic Voting Machine (EVM)” using “Arduino” to provide a low-cost, efficient, and secure voting system. The EVM is built using an Arduino microcontroller, which acts as the core processing unit, alongside peripherals such as push buttons, LEDs, and an LCD display. The system is designed to allow voters to cast their votes electronically by pressing a button corresponding to their preferred candidate, with the Arduino securely recording and displaying the votes.

The primary goal of this project is to demonstrate the feasibility of using Arduino in real-world voting applications, ensuring simplicity, accuracy, and user-friendliness. The system prevents multiple voting through unique identification, stores the voting data, and displays the results after all votes are cast. Additionally, the machine is designed to be tamper-resistant and ensures that every vote is counted correctly.

By leveraging the flexibility of Arduino, this project offers an accessible approach to electronic voting, making it suitable for small-scale elections, educational purposes, and prototype development.

Table of content

1. Introduction

1.1 Overview of Electronic Voting Machines (EVM)

Electronic Voting Machines (EVMs) are electronic devices designed to cast and count votes during elections. These machines eliminate the need for traditional paper ballots, offering a more efficient and tamper-resistant system for managing elections. EVMs have been widely adopted in many countries due to their ease of use, reliability, and ability to quickly produce results.

1.2 Importance of EVM in Modern Elections

In modern elections, accuracy, speed, and security are paramount. EVMs ensure that each vote is counted exactly once, eliminating the risks of over-voting or manual errors associated with paper ballots. Additionally, EVMs allow for faster counting, reducing delays in election results and ensuring a more transparent and trustworthy voting process.

1.3 Project Objective

The objective of this project is to design a low-cost, Arduino-based EVM that demonstrates how technology can improve voting systems. By utilizing Arduino, the project aims to create a simple, reliable, and secure prototype for small-scale voting environments such as schools, colleges, or local elections

2. Literature Review

2.1 Traditional Voting Systems

Traditional voting systems rely on paper ballots, where voters mark their choices manually. While this method has been used for centuries, it is prone to human error, fraud, and delays in counting. Furthermore, large-scale elections often require substantial resources for managing logistics and security.

2.2 Existing Electronic Voting Systems

Several types of electronic voting systems have been developed, ranging from Direct Recording Electronic (DRE) machines to optical scan machines. These systems have made elections more secure and efficient. However, existing systems are often expensive and complex, making them inaccessible for smaller institutions.

2.3 Role of Microcontrollers in EVMs

Microcontrollers like Arduino provide a low-cost solution for designing EVMs. These compact, programmable devices can manage input (such as buttons) and output (like LCD screens) efficiently, allowing for the development of reliable and secure voting systems.

3. System Design and Architecture

3.1 Overview of the Arduino Platform

Arduino is an open-source microcontroller platform that offers flexibility for creating a wide variety of electronics projects. It is widely used due to its simplicity, accessibility, and the large community supporting its development. In this project, Arduino is the core processing unit that handles input from voters and displays the results.

3.2 Hardware Components

3.2.1 Arduino Microcontroller: The Arduino board processes the votes and controls all other components in the system.

3.2.2 Push Buttons: Each candidate in the election is assigned a button. Voters cast their vote by pressing the corresponding button.

3.2.3 LEDs: LEDs provide visual confirmation that a vote has been successfully cast.

3.2.4 LCD Display: The display shows the voting options and presents the results at the end of the election.

3.2.5 Power Supply: A stable power source is essential for the Arduino and the connected peripherals to function reliably.

3.3 Circuit Design

The circuit design connects the push buttons, LEDs, and LCD display to the Arduino board. Each component is assigned specific pins on the Arduino, allowing the microcontroller to read inputs and send output commands. Proper connections are made to ensure secure voting.

3.4 Working Principle

The system works by allowing a voter to press one of the buttons corresponding to a candidate. The Arduino records the vote and provides visual feedback through the LEDs. Once voting is complete, the system displays the final vote count on the LCD screen.

4. Software Design

4.1 Arduino IDE Overview

The Arduino Integrated Development Environment (IDE) is used to write and upload code to the Arduino board. It supports the C++ programming language and provides libraries to simplify the process of reading inputs and controlling outputs.

4.2 Program Flow and Logic

4.2.1 Vote Casting Process: When a voter presses a button, the Arduino records the input and increments the vote count for the selected candidate.

4.2.2 Vote Counting Process: The Arduino stores the vote counts in its memory, ensuring each vote is only counted once.

4.2.3 Displaying Results: After all votes are cast, the total votes for each candidate are displayed on the LCD screen, providing a transparent outcome.

4.3 Code Explanation

The code starts by initializing the Arduino, configuring input pins for the buttons and output pins for the LEDs and LCD. A loop function continuously monitors button presses, records the votes, and controls the display to provide feedback to the user.

5. Security Features

5.1 Preventing Multiple Voting

To prevent multiple voting, the system is designed to allow only one vote per voter. This is achieved by deactivating the buttons after a vote is cast, ensuring that each person can only vote once.

5.2 Data Integrity and Tamper Protection

The Arduino's internal memory securely stores votes, reducing the risk of tampering. Additionally, physical measures such as secure housing for the machine can prevent unauthorized access to the system.

5.3 Reliability of Results

The system ensures that each vote is accurately recorded and the final count is free from errors. The Arduino is tested to handle multiple inputs efficiently without data corruption, ensuring reliable results.

6. Testing and Validation

6.1 Simulation and Testing Environment

Testing is conducted in a controlled environment, where different scenarios are simulated to ensure the system works as expected. The machine is tested with multiple votes to confirm the accuracy of the results.

6.2 Test Cases and Results

Various test cases, such as single votes, multiple votes, and invalid inputs, are executed to evaluate the system's performance. The results are recorded to verify that the machine functions as designed, ensuring accurate vote counts and reliable operation.

7. Conclusion

7.1 Summary of Findings

This project successfully demonstrates the use of Arduino to design a low-cost, reliable, and secure EVM. The system is capable of recording votes, preventing multiple voting, and displaying results efficiently. It can be used for small-scale elections or educational purposes.

7.2 Future Enhancements

Future improvements could include adding biometric authentication for enhanced security, increasing storage capacity for large-scale elections, and integrating wireless data transfer for remote monitoring of results.

8.Code

```
#include <LiquidCrystal.h>
LiquidCrystal lcd(13, 12, 11, 10, 9, 8);

#define S1 7  // MOHIT
#define S2 6  // VARU
#define S3 5  // KAI
#define S4 4  // DARSH
#define S5 3  // Result / Reset / Power

int vote1 = 0, vote2 = 0, vote3 = 0, vote4 = 0;
bool isOn = true;
bool voteInProgress = false;

void showWelcomeAnimation() {
  lcd.clear();
  for (int i = 0; i < 16; i++) {
    lcd.setCursor(i, 0);
    lcd.print(">");
    delay(50);
  }
  lcd.setCursor(2, 0); lcd.print("Electronic");
  lcd.setCursor(1, 1); lcd.print("Voting Machine");
  delay(2000);
  lcd.clear();
}

void setup() {
  pinMode(S1, INPUT_PULLUP);
  pinMode(S2, INPUT_PULLUP);
  pinMode(S3, INPUT_PULLUP);
  pinMode(S4, INPUT_PULLUP);
  pinMode(S5, INPUT_PULLUP);

  lcd.begin(16, 2);
  showWelcomeAnimation();
}

void thankYouMessage() {
```

```
  lcd.clear();
  lcd.setCursor(2, 0);
  lcd.print("Thank you for");
  lcd.setCursor(4, 1);
  lcd.print("voting!");
  delay(5000); // Show message for 5 seconds
  lcd.clear();
  lcd.setCursor(1, 0);
  lcd.print("Please Wait...");
  delay(1000);
  lcd.clear();
}

void showAllVotes() {
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("MO:");
  lcd.print(vote1);
  lcd.setCursor(8, 0);
  lcd.print("VA:");
  lcd.print(vote2);
  lcd.setCursor(0, 1);
  lcd.print("KA:");
  lcd.print(vote3);
  lcd.setCursor(8, 1);
  lcd.print("DA:");
  lcd.print(vote4);
  delay(3000);
}

void showWinner() {
  int totalVotes = vote1 + vote2 + vote3 + vote4;
  lcd.clear();

  if (totalVotes == 0) {
    lcd.setCursor(2, 0);
    lcd.print("No Voting...");
    delay(2000);
    lcd.clear();
    return;
  }
```

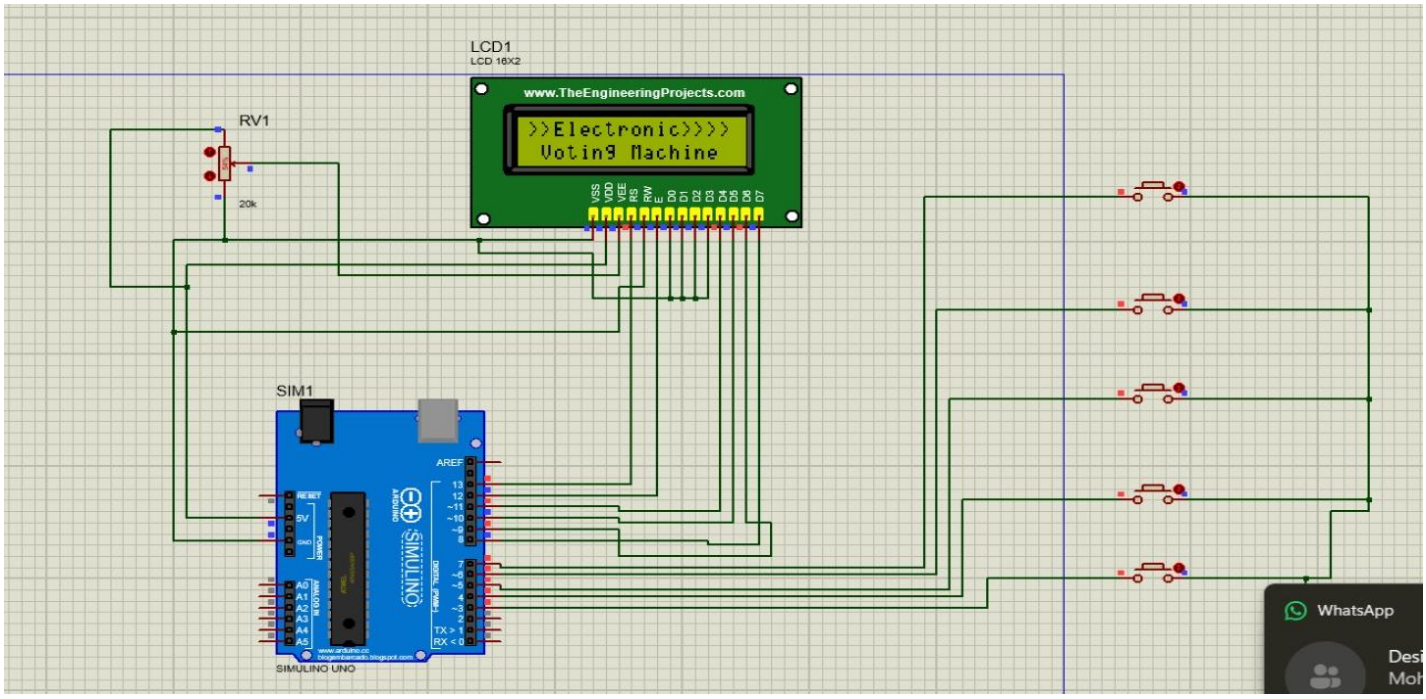
```
  String winner;
  int maxVotes = vote1;
  winner = "MOHIT";

  if (vote2 > maxVotes) {
    maxVotes = vote2;
    winner = "VARU";
  }
  if (vote3 > maxVotes) {
    maxVotes = vote3;
    winner = "KAI";
  }
  if (vote4 > maxVotes) {
    maxVotes = vote4;
    winner = "DARSH";
  }

  int count = 0;
  if (vote1 == maxVotes) count++;
  if (vote2 == maxVotes) count++;
  if (vote3 == maxVotes) count++;
  if (vote4 == maxVotes) count++;

  lcd.clear();
  if (count > 1) {
    lcd.setCursor(4, 0); lcd.print("TIE");
    lcd.setCursor(1, 1); lcd.print("No Clear Winner");
  } else {
    lcd.setCursor(0, 0);
    lcd.print(winner + " Wins!");
    lcd.setCursor(0, 1);
    lcd.print("Votes: ");
    lcd.print(maxVotes);
    lcd.print("/");
    lcd.print(totalVotes);
  }

  delay(4000);
  lcd.clear();
}
```

```
void loop() {
  if (!isOn) return;

  // Don't show votes while voting — LCD stays blank

  if (!voteInProgress) {
    if (digitalRead(S1) == LOW) {
      vote1++;
      voteInProgress = true;
      thankYouMessage();
    }
    else if (digitalRead(S2) == LOW) {
      vote2++;
      voteInProgress = true;
      thankYouMessage();
    }
    else if (digitalRead(S3) == LOW) {
      vote3++;
      voteInProgress = true;
      thankYouMessage();
    }
    else if (digitalRead(S4) == LOW) {
      vote4++;
      voteInProgress = true;
      thankYouMessage();
    }
  }

  if (voteInProgress) {
    if (digitalRead(S1) == HIGH && digitalRead(S2) == HIGH &&
        digitalRead(S3) == HIGH && digitalRead(S4) == HIGH) {
      voteInProgress = false;
    }
    return;
  }

  if (digitalRead(S5) == LOW) {
    unsigned long startTime = millis();
    while (digitalRead(S5) == LOW);
    unsigned long duration = millis() - startTime;
```

```
    if (duration >= 3000) {
      lcd.clear();
      lcd.setCursor(4, 0);
      lcd.print("Powering Off");
      delay(1000);
      lcd.noDisplay();
      isOn = false;
      return;
    }
    else if (duration >= 2000) {
      lcd.clear();
      lcd.setCursor(3, 0);
      lcd.print("Votes Reset");
      vote1 = vote2 = vote3 = vote4 = 0;
      delay(2000);
      lcd.clear();
    }
    else {
      showAllVotes();
      showWinner();
    }
  }
}
```

9.Appendix

9.1 Proteus Design



9.2 Circuit Diagram