



Instrumentation & Measurement Lab Proposal

|  |  |  |
| --- | --- | --- |
| **PROJECT**  **TITLE** | **:** |  |
|  |  |  |
| **Batch** | **:** |  |
| **Section** | **:** |  |
|  |  | **NAME ROLL NO** |

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |

CONTENT TABLE:

OVERVIEW:

An introduction to the project, its working and applications.

COMPONENTS:

List of components used in the project.

CIRCUIT DIAGRAM:

Circuit diagram of project

EXPLAINATION:

How the whole circuit works and explanation of code

CODE:

Arduino code for project.

OUTPUT:

Hardware output for project.

CONCLUSION:

Conclusion of project.



**INSTRUMENTATION AND MEASUREMENT LAB PROJECT REPORT**

PROJECT NAME: Fingerprint Door Lock system using Arduino

# OVERVIEW:

The Fingerprint Door Lock System using Arduino is a secure and reliable security system that utilizes the FPM10A DY50 fingerprint scanner, an Arduino Nano board, a 5V relay, a solenoid 12V lock, a 16X2 LCD, a 12V AC-DC adapter, and jumper wires. The system is designed to provide enhanced security by allowing access only to authorized individuals whose fingerprints are stored in the system's database.

# Components:

* Arduino Nano
* Finger Print Scanner (FPM10A DY50)
* 5V relay
* Solenoid 12V lock
* A 12V battery
* 16X2 LCD
* I2C Module
* Jumper wires

# CIRCUIT DIAGRAM:

# 

Explaination:

The new code for the Fingerprint Door Lock System using Arduino includes an LCD display to provide feedback to the user. The LiquidCrystal\_I2C library is used to communicate with the LCD display. The setup function initializes the fingerprint scanner and verifies the password. If the password is not verified, the system will not proceed. The LCD display is initialized and a welcome message is displayed. The loop function checks if a valid fingerprint is detected by calling the getFingerPrint function. If a valid fingerprint is detected, the LCD display shows a message indicating that the door is unlocked and welcomes the user. The relay is switched on, the lock is unlocked for 5 seconds, and then the relay is switched off again. The getFingerPrint function returns the ID number of the fingerprint if it is valid. The function first gets an image from the fingerprint scanner, converts it to a template, and then searches for a match in the database. If a match is found, the function returns the ID number of the fingerprint. If no match is found, the function returns an error message on the LCD display.

CODE:

For scaning finger:

#include <Adafruit\_Fingerprint.h>

#if (defined(\_\_AVR\_\_) || defined(ESP8266)) && !defined(\_\_AVR\_ATmega2560\_\_)

// For UNO and others without hardware serial, we must use software serial...

// pin #2 is IN from sensor (GREEN wire)

// pin #3 is OUT from arduino (WHITE wire)

// Set up the serial port to use softwareserial..

SoftwareSerial mySerial(2, 3);

#else

// On Leonardo/M0/etc, others with hardware serial, use hardware serial!

// #0 is green wire, #1 is white

#define mySerial Serial1

#endif

Adafruit\_Fingerprint finger = Adafruit\_Fingerprint(&mySerial);

uint8\_t id;

void setup()

{

Serial.begin(9600);

while (!Serial); // For Yun/Leo/Micro/Zero/...

delay(100);

Serial.println("\n\nAdafruit Fingerprint sensor enrollment");

// set the data rate for the sensor serial port

finger.begin(57600);

if (finger.verifyPassword()) {

Serial.println("Found fingerprint sensor!");

} else {

Serial.println("Did not find fingerprint sensor :(");

while (1) { delay(1); }

}

Serial.println(F("Reading sensor parameters"));

finger.getParameters();

Serial.print(F("Status: 0x")); Serial.println(finger.status\_reg, HEX);

Serial.print(F("Sys ID: 0x")); Serial.println(finger.system\_id, HEX);

Serial.print(F("Capacity: ")); Serial.println(finger.capacity);

Serial.print(F("Security level: ")); Serial.println(finger.security\_level);

Serial.print(F("Device address: ")); Serial.println(finger.device\_addr, HEX);

Serial.print(F("Packet len: ")); Serial.println(finger.packet\_len);

Serial.print(F("Baud rate: ")); Serial.println(finger.baud\_rate);

}

uint8\_t readnumber(void) {

uint8\_t num = 0;

while (num == 0) {

while (! Serial.available());

num = Serial.parseInt();

}

return num;

}

void loop() // run over and over again

{

Serial.println("Ready to enroll a fingerprint!");

Serial.println("Please type in the ID # (from 1 to 127) you want to save this finger as...");

id = readnumber();

if (id == 0) {// ID #0 not allowed, try again!

return;

}

Serial.print("Enrolling ID #");

Serial.println(id);

while (! getFingerprintEnroll() );

}

uint8\_t getFingerprintEnroll() {

int p = -1;

Serial.print("Waiting for valid finger to enroll as #"); Serial.println(id);

while (p != FINGERPRINT\_OK) {

p = finger.getImage();

switch (p) {

case FINGERPRINT\_OK:

Serial.println("Image taken");

break;

case FINGERPRINT\_NOFINGER:

Serial.println(".");

break;

case FINGERPRINT\_PACKETRECIEVEERR:

Serial.println("Communication error");

break;

case FINGERPRINT\_IMAGEFAIL:

Serial.println("Imaging error");

break;

default:

Serial.println("Unknown error");

break;

}

}

// OK success!

p = finger.image2Tz(1);

switch (p) {

case FINGERPRINT\_OK:

Serial.println("Image converted");

break;

case FINGERPRINT\_IMAGEMESS:

Serial.println("Image too messy");

return p;

case FINGERPRINT\_PACKETRECIEVEERR:

Serial.println("Communication error");

return p;

case FINGERPRINT\_FEATUREFAIL:

Serial.println("Could not find fingerprint features");

return p;

case FINGERPRINT\_INVALIDIMAGE:

Serial.println("Could not find fingerprint features");

return p;

default:

Serial.println("Unknown error");

return p;

}

Serial.println("Remove finger");

delay(2000);

p = 0;

while (p != FINGERPRINT\_NOFINGER) {

p = finger.getImage();

}

Serial.print("ID "); Serial.println(id);

p = -1;

Serial.println("Place same finger again");

while (p != FINGERPRINT\_OK) {

p = finger.getImage();

switch (p) {

case FINGERPRINT\_OK:

Serial.println("Image taken");

break;

case FINGERPRINT\_NOFINGER:

Serial.print(".");

break;

case FINGERPRINT\_PACKETRECIEVEERR:

Serial.println("Communication error");

break;

case FINGERPRINT\_IMAGEFAIL:

Serial.println("Imaging error");

break;

default:

Serial.println("Unknown error");

break;

}

}

// OK success!

p = finger.image2Tz(2);

switch (p) {

case FINGERPRINT\_OK:

Serial.println("Image converted");

break;

case FINGERPRINT\_IMAGEMESS:

Serial.println("Image too messy");

return p;

case FINGERPRINT\_PACKETRECIEVEERR:

Serial.println("Communication error");

return p;

case FINGERPRINT\_FEATUREFAIL:

Serial.println("Could not find fingerprint features");

return p;

case FINGERPRINT\_INVALIDIMAGE:

Serial.println("Could not find fingerprint features");

return p;

default:

Serial.println("Unknown error");

return p;

}

// OK converted!

Serial.print("Creating model for #"); Serial.println(id);

p = finger.createModel();

if (p == FINGERPRINT\_OK) {

Serial.println("Prints matched!");

} else if (p == FINGERPRINT\_PACKETRECIEVEERR) {

Serial.println("Communication error");

return p;

} else if (p == FINGERPRINT\_ENROLLMISMATCH) {

Serial.println("Fingerprints did not match");

return p;

} else {

Serial.println("Unknown error");

return p;

}

Serial.print("ID "); Serial.println(id);

p = finger.storeModel(id);

if (p == FINGERPRINT\_OK) {

Serial.println("Stored!");

} else if (p == FINGERPRINT\_PACKETRECIEVEERR) {

Serial.println("Communication error");

return p;

} else if (p == FINGERPRINT\_BADLOCATION) {

Serial.println("Could not store in that location");

return p;

} else if (p == FINGERPRINT\_FLASHERR) {

Serial.println("Error writing to flash");

return p;

} else {

Serial.println("Unknown error");

return p;

}

return true;

}

For Door lock and open:

#include <Adafruit\_Fingerprint.h>

#include <Wire.h>

#include <LiquidCrystal\_I2C.h>

SoftwareSerial mySerial(2, 3); //Serial Communications

LiquidCrystal\_I2C lcd(0x27, 16, 2);

Adafruit\_Fingerprint finger = Adafruit\_Fingerprint(&mySerial);

#define Relay 4

#define buzzer 5

void setup()

{

Serial.begin(9600);

lcd.init();

lcd.backlight();

finger.begin(57600);

pinMode(Relay, OUTPUT);

digitalWrite(Relay, LOW);

delay(5);

pinMode(buzzer,OUTPUT);

digitalWrite(buzzer,LOW);

delay(5);

if (finger.verifyPassword()) {

lcd.setCursor(0,0);

lcd.print("  Finger Print ");

Serial.println("  Finger Print ");

lcd.setCursor(0, 1);

lcd.print("Sensor Connected");

Serial.println("Sensor Connected");

delay (3000);

} else {

lcd.setCursor(0,0);

lcd.print("Unable to found   ");

Serial.println("Unable to found   ");

lcd.setCursor(0,1);

lcd.print("Sensor");

Serial.println("Sensor");

delay(2000);

lcd.setCursor(0, 0);

lcd.print("Please Check    ");

Serial.println("Please Check    ");

lcd.setCursor(0,1);

lcd.print("  Sensor Wiring ");

Serial.println("  Sensor Wiring ");

while (1) {

delay(1); }

}

}

void loop()

{

lcd.setCursor(0,0);

lcd.print("Place finger...");

Serial.println("Place finger...");

lcd.setCursor(0,1);

lcd.print("   start scan    ");

Serial.println("   start scan    ");

getFingerprintID();

delay(50);            //don't ned to run this at full speed.

}

uint8\_t getFingerprintID() {

uint8\_t p = finger.getImage();

if(p == FINGERPRINT\_NOFINGER){

return p;

}

else if(p != FINGERPRINT\_OK){

lcd.setCursor(0,0);

lcd.print("Scan Error      ");

Serial.println("Scan Error      ");

lcd.setCursor(0,1);

delay(2000);

return p;

}

p = finger.image2Tz();

if(p != FINGERPRINT\_OK){

lcd.setCursor(0,0);

lcd.print("Processing Error");

Serial.println("Processing Error");

lcd.setCursor(0,1);

lcd.print(" Try Again ");

Serial.println(" Try Again ");

return p;

}

p = finger.fingerSearch();

if (p == FINGERPRINT\_OK ) {

lcd.clear();

lcd.setCursor(0, 0);

lcd.print("  Door Unlocked");

Serial.println("  Door Unlocked");

lcd.setCursor(0, 1);

lcd.print("    Welcome");

Serial.println("    Welcome");

digitalWrite(buzzer,LOW);

digitalWrite(Relay,HIGH);

delay(5000);

digitalWrite(Relay,LOW);

} else if (p == FINGERPRINT\_PACKETRECIEVEERR) {

lcd.setCursor(0,0);

lcd.print("Comm Error      ");

Serial.println("Comm Error      ");

lcd.setCursor(0,1);

lcd.print("                ");

Serial.println("                ");

delay(2000);

return p;

} else if (p == FINGERPRINT\_NOTFOUND) {

lcd.setCursor(0,0);

lcd.print("Access Denied   ");

lcd.setCursor(0, 1);

lcd.print("Door is Lock.");

Serial.println("Door is Lock.");

digitalWrite(buzzer,HIGH);

delay(3000);

digitalWrite(buzzer,LOW);

//delay(2000);

return p;

} else {

lcd.setCursor(0,0);

lcd.print("Error in matching");

Serial.println("Error in matching");

lcd.setCursor(0,1);

lcd.print("Not Valid Finger");

Serial.println("Not Valid Finger");

delay(2000);

return p;

}

digitalWrite(Relay,LOW);

digitalWrite(buzzer,LOW);

return finger.fingerID;

}

RESULT:

CONCLUSION:

In summary, the Fingerprint Door Lock System utilizing Arduino showcased a successful integration of biometric technology for secure access control. The project effectively demonstrated the accurate and reliable authentication of users through fingerprint recognition. The Arduino platform provided a versatile and cost-effective foundation for the implementation of the system. The project's success implies its potential application in enhancing security for various environments, offering a convenient and efficient alternative to traditional lock-and-key mechanisms. Overall, the Fingerprint Door Lock System with Arduino showcased the practicality and feasibility of employing biometrics for access control in real-world scenarios.