**MINI PROJECT REPORT**

**On**

**Door Locking System with Fingerprint Sensor**

**Submitted by**

**Deeksha kashyap**

**171500091**

**Abhishek Raghav**

**181599001**

Department of Computer Engineering & Applications

**Institute of Engineering & Technology**



**GLA University**

**Mathura- 281406, INDIA**

**2019**

**Department of computer Engineering and Applications**

**GLA University, Mathura**

**17 km. Stone NH#2, Mathura-Delhi Road, P.O. – Chaumuha,**

**Mathura – 281406**



**Declaration**

I hereby declare that the work which is being presented in the Mini Project Training “Door Locking System with fingerprint sensor**”,** in partial fulfillment of the requirements for Mini Project Training viva voce, is an authentic record of my own work carried under the supervision of “GLA University”**.**

Signature of Candidate:

Name of Candidate:

Roll. No. :

Course:

Year:

Semester:

**ACKNOWLEDGEMENT**

The success and final outcome of this project required a lot of guidance and assistance from many people and I am extremely privileged to have got this all along the completion of my project. All that I have done is only due to such supervision and assistance and I would not forget to thank them.

I respect and thank **Mr. Mandeep Sir**, for providing me an opportunity to do the project and giving me all support and guidance which made me complete the project duly. I am extremely thankful to him for providing such a nice support and guidance, although he had busy schedule managing the corporate affairs.

I own my deep gratitude to our project guide **Mr. Mandeep** who took keen interest on my project work and guided me all along, till the completion of our project work by providing all the necessary information for developing a good project.

I would like to thank my team members for the operation extended to us throughout the project. After doing this project I can confidently say that this experience has not only enriched me with technical knowledge but also has unparsed the maturity of thought and vision. The attributes required in being a successful professional.

Deeksha kashyap

Abhishek Raghav

**Abstract**

Security of valuables is as paramount as their acquisition. Valuables ranging from human lives to expensive resources and sensitive data need to be tightly secured. In this present day when armed robbery has gotten more sophisticated particularly in developing countries, there is the need for tighter security means, and one of the most secured technologies that can be employed is biometrics, finger print door lock to be precise. Biometrics is the science and technology of measuring and analyzing biological data, biometrics measures and analyses human body characteristics such as DNA, fingerprint, eye retina and iris, voice pattern, facial pattern and hand measurement.

The software that drives the microcontroller was done using the C language on MPLAB compiler, the coding was segmented into various modules; first, the module that drives the LCD screen, next is the module that drives the finger print scanner, this instructs the scanner to first register users and allow the inputted finger print to be compared with the pre-registered finger prints. The pre-registered finger prints are saved on the IC registers of the Micro controller. An alert is sounded whenever a fingerprint that is not found in the Microcontroller memory is placed on the scanner as an intruder.

The construction of this project was done in three different stages, the writing of the code (driver) which controls the Microcontroller using C language, the implementation of the whole project on a solder-less experiment board, the soldering of the circuits on Vero-boards and the coupling of the entire project to the casing. The implementation of this project was done on the breadboard as a prototype, the power supply was first derived from a bench power supply in the electronics laboratory, in all the development guaranteed security for illegal intrusion of illegal entity to room, the mechanism can be implemented in a broader sense on a door where a there is restriction of access.

**Contents**

Acknowledgments

Abstract

1. **Introduction**  1

1.1 Circuit and Working of Arduino Fingerprint Sensor Lock……………………..1

## 1.2 **Software of Arduino Fingerprint Sensor Lock………………………………….3**

**2. Programs 5**

2.1 Finger print program………………………………………………………….....5

2.2Finger print sensor program……………………………………………………..12

2.3Servo motor program..…………………………………………………………..17

**3. Conclusion**   **21**

**4. References 21**

**INTRODUCTION**

**The project “Arduino Fingerprint Sensor Lock” is simply fabricated around Arduino board. Not only to ensure door security, this project also make certain to assist in fields like forensics, crime investigation, personal identification, attendance system and there is a lot more.**

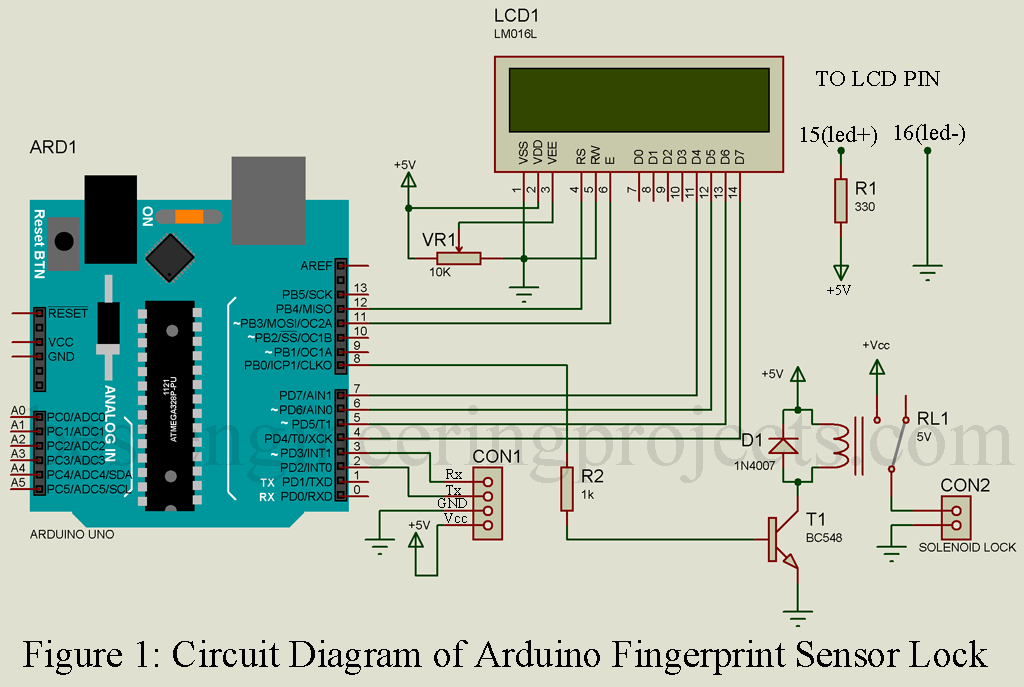
Recently, there has been recorded tremendous increase in the crime rate everywhere in the world. This issue is turning more severe every day. To get away with this problem, we decided to take help from technology and there this project “Arduino Fingerprint Sensor Lock” developed. We know the saying very well- ‘Prevention is better than cure’, rather than to face the loss it is much better to take necessary actions to eradicate that issue before it happens.

The project helps us to implement the fact.  The reason behind the fact that project has gained so much popularity in a short interval is mostly because of its simplicity and attractive feature. Today, fingerprint project is linked with security and major task, later it may be employed as fingerprint based driving license, bank accounts operation and so on. ‘Matching Algorithm’ is the main principle of this project where specified templates of fingerprints are initially stored. Then, the fingerprint of user is compared with the pre-stored templates of fingerprints. It verifies authentication process.

The old practice of using a simple key to unlock a door is time consuming as well as less secure. Replacing those methods with fingerprints, we get access inside a house/room just by placing the correct finger on the sensor. However, only authorized people can open the door because of the special fingerprint technique. If the fingerprint matches with any one of the image from database, the door unlocks and the LCD displays a welcome message along with that person’s name.

**Circuit and Working of Arduino Fingerprint Sensor Lock**

12V power is the main source of energy supply required for this system, which is given to the VINpin of Arduino board. The solenoid electric lock itself consumes 12V supply; however Arduino microcontroller (MCU) requires only 5V which can be easily supplied from the inbuilt 5V regulator from the Arduino Uno Board. And, the other common 12V supply is externally supplied to the system.



### ****(a). Arduino Uno MCU board | Arduino Fingerprint Sensor Lock****

Arduino Uno MCU board is based on ATmega328/ATmega328P acts like a CPU of the system “Arduino Fingerprint Sensor Lock” the figure of which is shown in figure 1. This board comprises multiple features. There are 14 digital input/output (I/O) pins, six analogue inputs, 32k flash memory, 16MHz crystal oscillator, a USB connection, power jack, ICSP header and reset button. We can use any of its features through Arduino IDE software through proper programming

### ****(b). Fingerprint sensor module | Arduino Fingerprint Sensor Lock****

The RX and TX pin of fingerprint sensor module R305 is connected across D3 and D2 pin of Arduino board respectively as shown in circuit diagram (figure 1). Since this module is constructed using UART technology, it is easy to interface sensor directly with the MCU or also to the PC using max232/USB serial adaptor.  The information collected from the fingerprint can be collected in the module. During the process of identification, the data can be configured in either 1:1 or 1: N module. In order to ensure serial communication, two pins of R305 sensor; TX and RX are connected across digital pins 2 and 3 of Arduino Uno.

## ****Software of Arduino Fingerprint Sensor Lock****

The core section of the project; software part utilizes two different programs-enroll and fingerprint. getFingerprintEnroll (int, id), Adafruit\_Fingerprint (&mySerial) and getFingerprintEnroll(id) are some of the different functions syntax used in those programs. These are in-built functions found in library and they pass arguments when these functions are called at different locations of programs.

Once the enroll part of the program has been uploaded in the Arduino Uno, go through the Arduino IDE and then open the serial monitor by opening tabs like tools and then select serial monitor options. It is necessary to set the baud rate to a value lower than the serial monitor window to 38400. At the same time choose Newline option.

And now, one by one execute the instructions given on the serial monitor. Once you place a finger on the fingerprint module, type an ID number. It can be any whole number. Then when send key is entered, the corresponding ID number is transmitted to the main portion i.e. Arduino Uno form the serial monitor section. Thus sent information (fingerprint) is digitized and converted into storable form which is piled up in R305 module database.

This system can withstand a total of 200+ fingerprints which is remarkable. However, each fingerprint must have unique ID number assigned since this is the prime factor to be utilized in identification of the valid individual’s name. The serial monitor assists the client in an effective way. Every real-time information of when to place the finger on the sensing module and when it is okay to remove, is all provided by the serial monitor which makes this project more user-friendly.

If you prefer to debug the system without implementing LCD display, initially upload the fingerprint program and then set the same settings as mentioned above for the serial monitor configuration. Here again the serial monitor performs the guide function. This technique of implementing circuit is employed to make necessary comparisons between the current sensed fingerprints samples with the samples already stored in the database. The programming flexibility feature permits customer to amend necessary changes in names and ID number by changing the code to a slight extent as per the requirement.

One must be careful to ensure that the baud rate value listed in the program must be accurate. Its value does not affect the serial monitor but for sensitive device like R305 sensor, it must be precisely the value listed in the datasheet. However, this value may depend on the type of sensor used in the project. In the main code, this value are fed in the system as Serial.begin (38400) which represents the baud rate for serial monitor and finger.begin (57600) which represent the baud rate for sensor. The Arduino board must be reset beforehand to avoid any possible errors during validation of fingerprint.

**Program**

**Fingerprint Sensor Program**

#include <Adafruit\_Fingerprint.h>

SoftwareSerial mySerial(2, 3);

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); }

}

}

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;

}

}

**Finger Print Program**

#include <Adafruit\_Fingerprint.h>

SoftwareSerial mySerial(2, 3);

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

void setup()

{

Serial.begin(9600);

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

delay(100);

Serial.println("\n\nAdafruit finger detect test");

// set the data rate for the sensor serial port

finger.begin(57600);

delay(5);

if (finger.verifyPassword()) {

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

} else {

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

while (1) { delay(1); }

}

finger.getTemplateCount();

Serial.print("Sensor contains "); Serial.print(finger.templateCount); Serial.println(" templates");

Serial.println("Waiting for valid finger...");

}

void loop() // run over and over again

{

getFingerprintIDez();

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

}

uint8\_t getFingerprintID() {

uint8\_t p = finger.getImage();

switch (p) {

case FINGERPRINT\_OK:

Serial.println("Image taken");

break;

case FINGERPRINT\_NOFINGER:

Serial.println("No finger detected");

return p;

case FINGERPRINT\_PACKETRECIEVEERR:

Serial.println("Communication error");

return p;

case FINGERPRINT\_IMAGEFAIL:

Serial.println("Imaging error");

return p;

default:

Serial.println("Unknown error");

return p;

}

// OK success!

p = finger.image2Tz();

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!

p = finger.fingerFastSearch();

if (p == FINGERPRINT\_OK) {

Serial.println("Found a print match!");

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

Serial.println("Communication error");

return p;

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

Serial.println("Did not find a match");

return p;

} else {

Serial.println("Unknown error");

return p;

}

// found a match!

Serial.print("Found ID #"); Serial.print(finger.fingerID);

Serial.print(" with confidence of "); Serial.println(finger.confidence);

return finger.fingerID;

}

// returns -1 if failed, otherwise returns ID #

int getFingerprintIDez() {

uint8\_t p = finger.getImage();

if (p != FINGERPRINT\_OK) return -1;

p = finger.image2Tz();

if (p != FINGERPRINT\_OK) return -1;

p = finger.fingerFastSearch();

if (p != FINGERPRINT\_OK) return -1;

// found a match!

Serial.print("Found ID #"); Serial.print(finger.fingerID);

Serial.print(" with confidence of "); Serial.println(finger.confidence);

return finger.fingerID;

}

**Servo Motor Program**

#include <Adafruit\_Fingerprint.h>

#include <SoftwareSerial.h>

#include <Servo.h> //Add servo library

int getFingerprintIDez();

Servo servo1; //Define servo name / object

#define servoPin 9 //Define pin number to which servo motor is connected

#define durationTime 3000 //Define the time it remains in the open position of the door lock (miliseconds)

#define servoMin 0 //Open position

#define servoMax 90 // Closed position

SoftwareSerial mySerial(2, 3);

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

void setup()

{

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

Serial.begin(9600);

Serial.println("Adafruit finger detect test");

servo1.attach(servoPin); //Define pin number of the servo

servo1.write(servoMax); //The position of the servo at the start of the program

// 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);

}

Serial.println("Waiting for valid finger...");

}

void loop() // run over and over again

{

getFingerprintIDez();

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

}

uint8\_t getFingerprintID() {

uint8\_t p = finger.getImage();

switch (p) {

case FINGERPRINT\_OK:

Serial.println("Image taken");

break;

case FINGERPRINT\_NOFINGER:

Serial.println("No finger detected");

return p;

case FINGERPRINT\_PACKETRECIEVEERR:

Serial.println("Communication error");

return p;

case FINGERPRINT\_IMAGEFAIL:

Serial.println("Imaging error");

return p;

default:

Serial.println("Unknown error");

return p;

}

// OK success!

p = finger.image2Tz();

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!

p = finger.fingerFastSearch();

if (p == FINGERPRINT\_OK) {

Serial.println("Found a print match!");

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

Serial.println("Communication error");

return p;

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

Serial.println("Did not find a match");

return p;

} else {

Serial.println("Unknown error");

return p;

}

// found a match!

Serial.print("Found ID #"); Serial.print(finger.fingerID);

Serial.print(" with confidence of "); Serial.println(finger.confidence);

}

// returns -1 if failed, otherwise returns ID #

int getFingerprintIDez() {

uint8\_t p = finger.getImage();

if (p != FINGERPRINT\_OK) return -1;

p = finger.image2Tz();

if (p != FINGERPRINT\_OK) return -1;

p = finger.fingerFastSearch();

if (p != FINGERPRINT\_OK) return -1;

servo1.write(servoMin); //If the fingerprint is correct open the door lock

delay(durationTime); //Keep the lock open for the defined duration

servo1.write(servoMax); //take the lock OFF again

// found a match!

Serial.print("Found ID #"); Serial.print(finger.fingerID);

Serial.print(" with confidence of "); Serial.println(finger.confidence);

return finger.fingerID;

}

**CONCLUSION**

Fingerprint identification enhances the security of a vehicle and makes it possible only for some selected people to use the car. Thus by implementing this relatively cheap and easily available system on a car, one can ensure much greater security and exclusivity than that offered by a conventional lock and key. It can be deduced that the use of biometric security systems offers a much better and foolproof means of restricting the use of vehicles by unauthorized users. The developed prototype serves as an impetus to drive future research, geared towards developing a more robust and embedded real-time fingerprint based automatic door lock systems in vehicles.

**REFERENCES**

[1]LinHong.”AutomaticPersonalIdentificationUsingFingerprints”,Ph.D.Thesis, 1998. [2]D.Maio and D. Maltoni. Direct gray-scale minutiae detection in fingerprints IEEE Trans. Pattern Anal. And Machine Intel., 19(1):27-40, 1997. [3]Alessandro Farina, Zsolt M.Kovacs-Vajna, Alberto leone, Fingerprint minutiae extraction from skeletonized binary images, Pattern Recognition, Vol.32, No.4, pp877-889, 1999. [4]N. Ratha, S. Chen and A.K. Jain, ”Adaptive Flow Orientation Based Feature Extraction in Fingerprint Images”, Pattern Recognition, Vol. 28, pp. 1657-1672, November 1995. [5]Jain, A.K., Hong, L., and Bolle, R.(1997), On-Line Fingerprint Verification, IEEE Trans. On Pattern Anal and Machine Intell, 19(4), pp. 302-314.