Microcontroller Based Current Flow Control Using Biometric Finger Recognition

By

Debproshad Das
Id: 2012-1-60-013

Mohaiminul Ahmed
Id: 2012-1-65-012

Supervised By

Md. Shamsujjoha
Senior Lecturer
Department of Computer Science & Engineering
East West University

A Project Submitted in Partial Fulfillment of the Requirements for the Degree of Master of Science in Computer Science and Engineering

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
EAST WEST UNIVERSITY

Summer, 2016

ABSTRACT

This project presets a current flow control mechanism using a Finger Print Sensor (FPS) and a microcontroller. The proposed system starts with enrolling the finger prints of authenticated user. Then, the users her/his figure in the FPS module to store data in its SRAM. As soon as the enrolling completed the proposed system displays a successful massage and the administrator manually closed the enroll system.

After that whenever the user wants to flow current in any particular circuit, he have to press the FPS module again for authentication. If finger print matches with any of the stored finger prints then the current flow will start (in our case, we used an energy saving bulb to show the current flow). Display will give a massage “lights on”.

To end the current flow user have to press the FPS module again and if the finger print matches with any of the stored finger prints then the current flow will stop. Display will give a massage “lights off.”

DECLARATION

We hereby, declare that all the work presented in this project is the outcome of the investigation ad research performed by us (Mohaiminul Ahmed and Debproshad Das) under the supervision of Md. Shamsujjoha, Senior Lecturer, Department of Computer Science & Engineering, East West University, Dhaka, Bangladesh. We also declare that neither it nor part of it has been submitted for the requirement of any degree or diploma or for any other except for publication.

Signature

Debproshad Das

Mohaiminul Ahmed

LETTER FOR ACCEPTANCE

This Project entitled "**Microcontroller Based Current Flow Control Using Biometric Finger Recognition**" submitted by Debproshad Das (ID: 2012-1-60-013) and Mohaiminul Ahmed (ID: 2012-1-65-012), to the Department of Computer Science and Engineering, East West University, Dhaka, Bangladesh is accepted by the department in partial fulfillment of requirements for the Award of the Degree of Bachelor of Science in Computer Science and Engineering on Summer, 2016.

Supervisor

_____

Md. Shamsujjoha

Senior Lecturer,
Department of Computer Science and Engineering,

East West University, Dhaka, Bangladesh

Chairperson

_____

Dr. Mozammel Huq Azad Khan

Professor & Chairperson,
Department of Computer Science and Engineering,

East West University, Dhaka, Bangladesh

ACKNOWLEDGEMENT

First, we are thankful and expressing our gratefulness to Almighty who offers us divine blessings, patience, mental and psychical strength to complete this project. The progression of this thesis could not possibly be carried out without the help of several people who, directly or indirectly, are responsible for the completion of this work. We deeply indebted to our project supervisor Mr. Md. Shamsujjoha. His scholarly guidance, especially for his tolerance with our persistent bothers and unfailing support. He gives us the freedom to pursue aspects of reversible fault tolerant computing which we found interesting and compelling. This helped our project to achieve its desired goals.

We wish to thank the great people of Department CSE at East West University. A special thank goes to all faculties for their well-disposed instructions and Encouragements.

Finally, we would like to thank our friends and family. Their continued tolerance with our moods and tendency to disappear for weeks at a time gave us a much needed break from the world computing.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

The technology is growing day by day. Now a day, we cannot think a day without any technology. It is making our life easier and faster. Everything is being automatic. Very soon, the world will be known as an automatic world. It is just the beginning of new era of modern science.

As the dependency on machines rising, it has become very important to make machines more intelligent and to ensure security on access mechanisms. The security system kicks in and rescues the system from unauthorized abuse. As the systems are growing sharper, the necessity of more complex security is increasing.
When you have something to keep secure from unwanted people or system, you have to impose some sort of security technologies. With the progress of technology, we now have so many types of security systems. Biometric security, password security, pattern security are some examples.
More days are passing by as security systems are getting stronger and stronger. In an automatic system, strong security is must. When an automatic system itself is used for security, the necessity is beyond mentioning.

Today, for making complex security, Biometric security system become popular day by day. To keep this in our mind, we make a complex security system that will sure ensure the high security by controlling current. This device is intelligent, faster, and most important it is safer. The device we make will take finger print from user and store it. After that when the user gives finger print again to the finger print sensor, it will matched with its stored data. If the given fingerprint matches with its stored finger print then it will let the circuit open and allow the current to flow. On the other hand if the given fingerprint does not match then it will not allow the current to flow.

To make this device we use some devices like Arduino Mega2560, Finger Print Sensor and a 5Volts 4 Channel Relay. In the rest of the chapters we will discuss more about the entire project.

1.1 OBJECTIVES

Objectives of making this system are:

- To control electricity by biometric way
- To make intelligent security system
- To make security system safer and efficient
- To make a complex security system
- To make life easier and faster

CHAPTER 2

BACKGROUND STUDIES

In traditional electronic security system, current flow is controlled by switch. But now with more technology we have, those switching systems to control the current flows are gone under by new emerging password or pattern system. Password and patterns are now become very common terms to controlling current flow for security concerns. As it become common terms to control current flow, hackers are now more interested to break these systems. And with the blessings of the technology they are breaking these systems frequently. So the demand is rising again for new technologies to secure the electricity flow.

For these reasons we introduce a new security system to control the current flow by biometric way.

Suppose in a research lab, there are many important documents, computers and various research equipment which are controlled by electricity. It may possible that the entrance system off the lab is protected by password or pattern based security system. But it could be breakable and could get access to the lab. Now if we use biometric system to control the current flow of that lab then someone may entered there but he cannot do any harms to important documents or equipment. Again we can give another example to show the use of this device. This device could be use in the car. If we fixed this device with the lock of the engine then when user press the key of the engine to start but it could not start if he or she does not give his or her finger print for authenticate. This will also ensure the security of the car safer, more smartly than any security system ever.

CHAPTER 3

EQUIPMENT DESCRIPTION

This chapter introduces the hardware system of the project. The hardware system is divided into six different major parts to make the explanation easier.

3.1 EQUIPMENT LIST

a) Processing Unit- Arduino Mega 2560
b) Finger Print Sensor module (FPS)
c) 4-channel 5V relay module
d) Display with IC Rail
e) Energy saving bulb
f) Arduino IDE

Apart from this breadboard, resistor and wires are also used.

3.1.1 PROCESSING UNIT

The Arduino Mega 2560 is a microcontroller board based on the ATmega2560. It has 54 digital input/output pins of which 15 can be used as PWM outputs, 16 analog inputs, 4 UARTs (hardware serial ports), a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller, simply connect it to a computer with a USB cable or power it with an AC-to-DC adapter or battery to get started. The Mega 2560 board it compatible with most shields designed for the Uno and the former boards Duemilanove or Diecimila. The Mega 2560 is an update to the Arduino Mega, which it replaces.

The ATmega2560 on the Mega 2560 comes preprogrammed with a bootloader that allows you to upload new code to it without the use of an external hardware programmer. It communicates using the original STK500 protocol (reference, C header files).

The Mega 2560 can be powered via the USB connection or with an external power supply. The power source is selected automatically.

The ATmega2560 has 256 KB of flash memory for storing code (of which 8 KB is used for the bootloader), 8 KB of SRAM and 4 KB of EEPROM (which can be read and written with the EEPROM library).

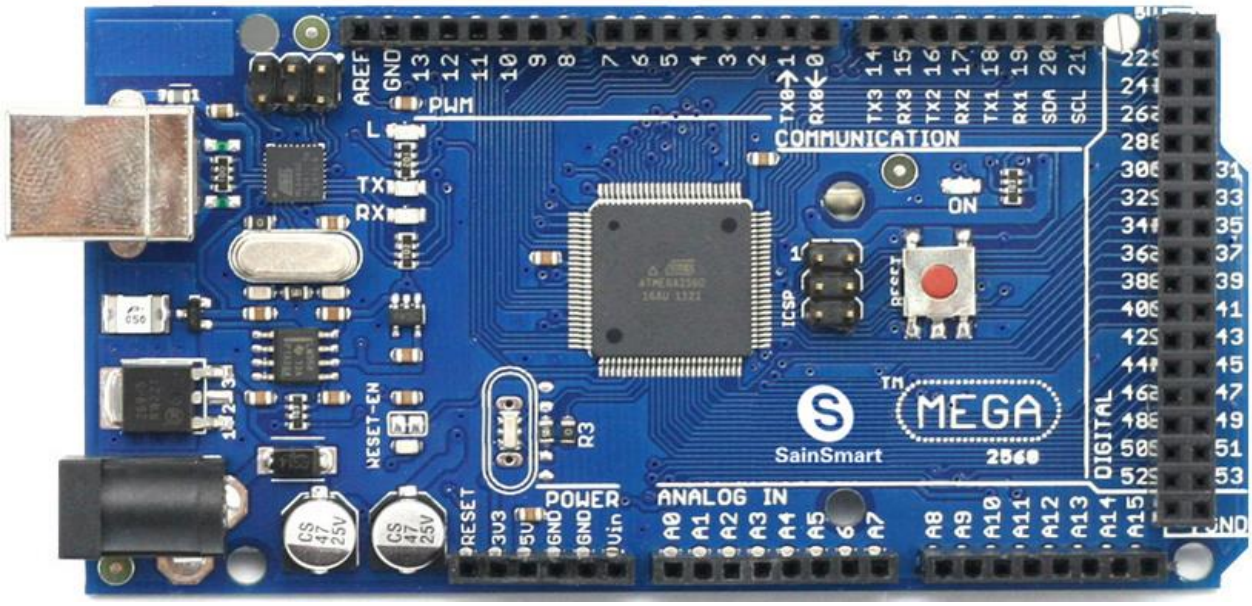

Figure 1: Arduino mega 2560

Technical Specification:

Microcontroller	ATmega2560
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	54 (of which 15 provide PWM output)
Analog Input Pins	16
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	256 KB of which 8 KB used by bootloader
SRAM	8 KB
EEPROM	4 KB
Clock Speed	16 MHz
Length	101.52 mm
Width	53.3mm
Weight	37g

3.1.2 FINGER PRINT SENSOR MODULE (FPS)

This is the updated version of the GT-511 which has an increased memory capacity. The module can store up to 200 different fingerprints and is now capable of 360° recognition.

The module itself does all of the heavy lifting behind reading and identifying the fingerprints with an on-board optical sensor and 32-bit CPU. To get started, just register each fingerprint that you want to store by sending the corresponding command and pressing your finger against the reader three times. The fingerprint scanner can store different fingerprints and the database of prints can even be downloaded from the unit and distributed to other modules. As well as the fingerprint "template," the analyzed version of the print, you can also retrieve the image of a fingerprint and even pull raw images from the optical sensor!

The module is small and easy to mount using two mounting tabs on the side of the sensor. The on-board JST-SH connector has four signals: Vcc, GND, Tx, Rx.

Dimensions: 37 x 17 x 9.5 mm

Features:

- High-Speed, High-Accuracy Fingerprint Identification using the SmackFinger 3.0 Algorithm
- Download Fingerprint Images from the Device
- Read and Write Fingerprint Templates and Databases
- Simple UART protocol (Default 9600 baud)
- Capable of 1:1 Verification and 1:N Identification

Figure 2: Finger print sensor

Technical Specifications:

CPU	ARM Cortex M3 Core
Sensor	Optical Sensor
Effective area of the sensor	14 x 12.5mm
Image size	202 x 258 pixels
Resolution	150 dpi
Maximum number of finger prints	200
Matching mode	1:1, 1:N

Size of template	496 Bytes (template) + 2 Bytes (checksum)
Communication interface	UART, default baud rate = 9600 bps after power
False acceptance rate	< 0.001%
False rejection rate	< 0.1%
Enrollment time	< 3 sec
Identification time	< 1 sec
Operating voltage	DC 3.3 ~ 6V
Operating Current	< 130 mA
Operating temperature	-20°C to +60°C
Operating humidity	20% to 80%
Storage temperature	-20°C to +60°C
Storage humidity	10% to 80%

3.1.3 4-Channel 5V relay module

Four 5V relay together in a single board to control 4 loads simultaneously. The relays can be easily switched by microcontroller or arduino.

- 5V 4-Channel Relay interface board, and each one needs 15-20mA Driver Current
- Equiped Equipped with high-current relay, AC250V 10A ; DC30V 10A
- Standard interface that can be controlled directly by microcontroller (Arduino , 8051, AVR, PIC, DSP, ARM, ARM, MSP430, TTL logic active low)
- Opto-isolated inputs
- Indication LED's for Relay output status

Figure 3: 4-Channel 5V relay

Technical Specifications:

Rated coil Voltage	5V DC
Operating Frequency	50/60HZ
Nominal Current	10AAC, 10A at 28V DC
Maximum Switching Voltage	250V AC, 28V DC
Power Required by the board	VCC = 5V DC(For relay coils),5V DC(For energizing every relay individually)
4 relay size	75*55*20mm

3.1.4 DISPLAY WITH IC RAIL

16 Pin 20x4 LCD Display. All you have to do is, solder male connectors with it and connect it to your breadboards, trainer boards and so on.

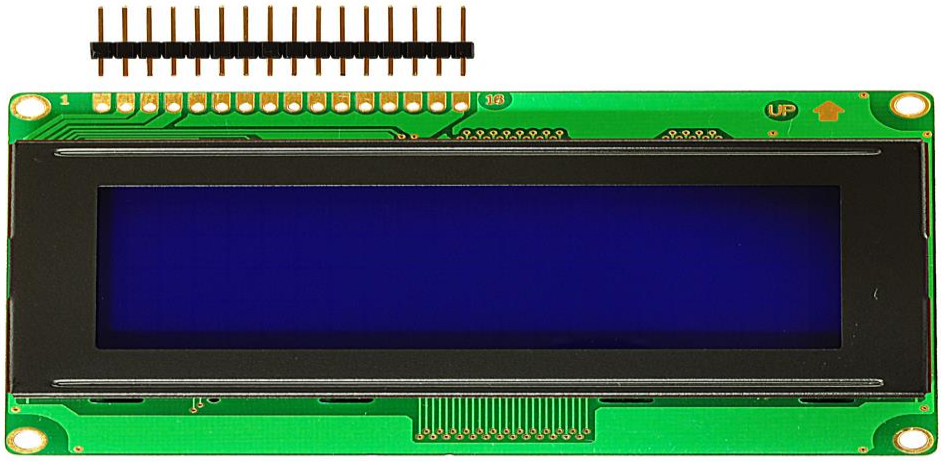

Figure 4: Display with IC Rail

Technical Specification:

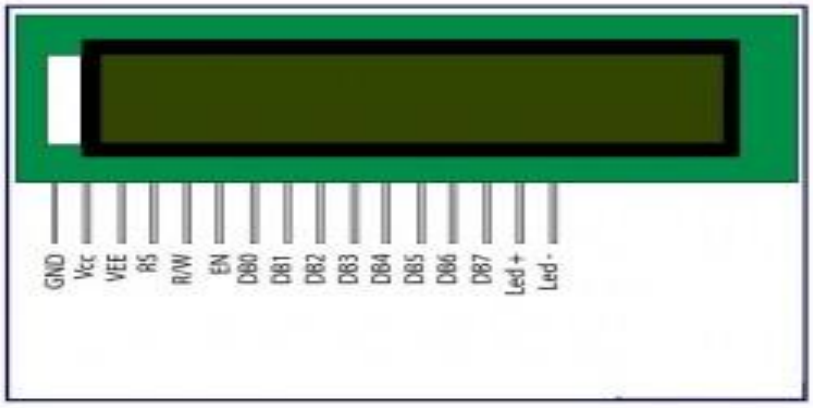

Figure 5: Technical specification of display pins

3.1.5 ENERGU SAVING BULB

Compared to traditional incandescent, energy-efficient light bulbs such as halogen incandescent, compact fluorescent lamps (CFLs), and light emitting diodes (LEDs) have the following advantages: Typically use about 25%-80% less energy than traditional incandescent.

Figure 6: 5W Energy saving bulb

3.1.6 ARDUINO IDE

The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. It runs on Windows, Mac OS X, and Linux. The environment is written in Java and based on Processing and other open-source software.

This software can be used with any Arduino board.

Programs written using Arduino Software (IDE) are called sketches. These sketches are written in the text editor and are saved with the file extension .ino. The editor has features for cutting/pasting and for searching/replacing text. The message area gives feedback while saving and exporting and also displays errors. The console displays text output by the Arduino Software (IDE), including complete error messages and other information. The bottom righthand corner of the window displays the configured board and serial port. The toolbar buttons allow you to verify and upload programs, create, open, and save sketches, and open the serial monitor.

The Arduino Integrated Development Environment - or Arduino Software (IDE) - contains a text editor for writing code, a message area, a text console, a toolbar with buttons for common functions and a series of menus. It connects to the Arduino and Genuino hardware to upload programs and communicate with them.

All the logics and ideas were turned into an algorithm and it was converted to some codes in Arduino Platform. The logics are mostly written in mostly “C” because are syntaxes are little similar. But the codes for taking readings from ports and sending signals to them are different. As a language, Arduino also have some header files containing many libraries too.

3.2 EXTRA TOOLS

- Male to male wire
- Male to female wire
- 1 KΩ resistor
- Breadboard
- JST SH Jumper 4 Wire Assembly

CHAPTER 4

PROPOSED WORK

This is the most crucial part of this project. For the successful completion of this project, it is very important for the hardware devices to interact with each other and work in the proper synchronization. The Finger Print Sensor (FPS) module is used as the interface for the user enrolls and authentication. Display will help user to make these processes much easier by displaying messages. Relay will convert 220 volt current to 5 volt to make the process easier. The Arduino will process the whole thing and the relay will turn the light on and off.

4.1 DESCRIPTION

The "**Microcontroller Based Current Flow Control Using Biometric Finger Recognition**" is designed to enable user to control the electricity flow by Finger Print Scanner (FPS) module automatically.

At first when we connect Arduino Mega 2560 board with USB or power adapter then display will start up with a message "*WELCOME*".

If user wants to enroll a finger then enroll function have to open manually from the Arduino IDE software. After that we have to load .ino file to the board and start the system again. Now display will show "*WELCOME*" message again. After 5 sec delay display will print "*Press a finger to enroll*" message. Then user will press a finger to FPS module for enroll.

In the enroll process the user will press his finger three times in order to successful enrolling. First time display will show "*Press a finger to enroll*" message. After pressing the finger first time, display will give "*Remove Finger*" message if the finger enroll successfully first time or display will print "*Failed to capture first finger*" if the finger cannot enroll successfully first time.

When first time finger input is successful display will give a message "*Press same finger again*". After pressing the finger second time display will give "*Remove finger*" message if the finger enroll successful second time or display will print "*Failed to capture second finger*" if enroll is not successful.

When second time finger is successful display will give a message "*Press same finger yet again*". After pressing the finger display will give "*Remove finger*" message again and finally display will show "*Enrolling Successful*" to ensure successful enrolment. But if third time finger input is not successful then display will give "*Failed to capture third finger"* message.

If the enrolling process is failed the display will print "*Enrolling failed with error code*" message. It means that the finger which is pressed three times is not matched each other. If the first time finger input is not matched second and third time input finger then display will show code 1. If the second time finger input is not matched first and third time input finger then display will show code 2. If the third time finger input is not matched second and first time input finger then display will show code 3.

If the user does not want to enroll his finger or finger of a particuler user is already stored then he can directly access to authentication process.

Authentication process will start by giving a message "*Please press a finger*" in the display. Now the user presses a finger to FPS module for authentication. If the input finer is matched with any of the stored finger then display will show the matched id and after few seconds microcontroller will let the current flow through the relay and turn the light on. At this moment display will show "*Light on*" message. If the input finger is not matched then display will give "*Finger not found*" message.

After few second the display will show "*Please press a finger*" message again. Then if the user wants to cut off the flow of current through the relay and turn off the light then he will input the finger again. Now if the finger matched again with the stored any of the fingers, the current will cut off and the light will turn off by displaying a message "*Light off*". In that case if the finger print does not match then display will show "*Finger not found*" message again.

The process will describe through a flow given below:

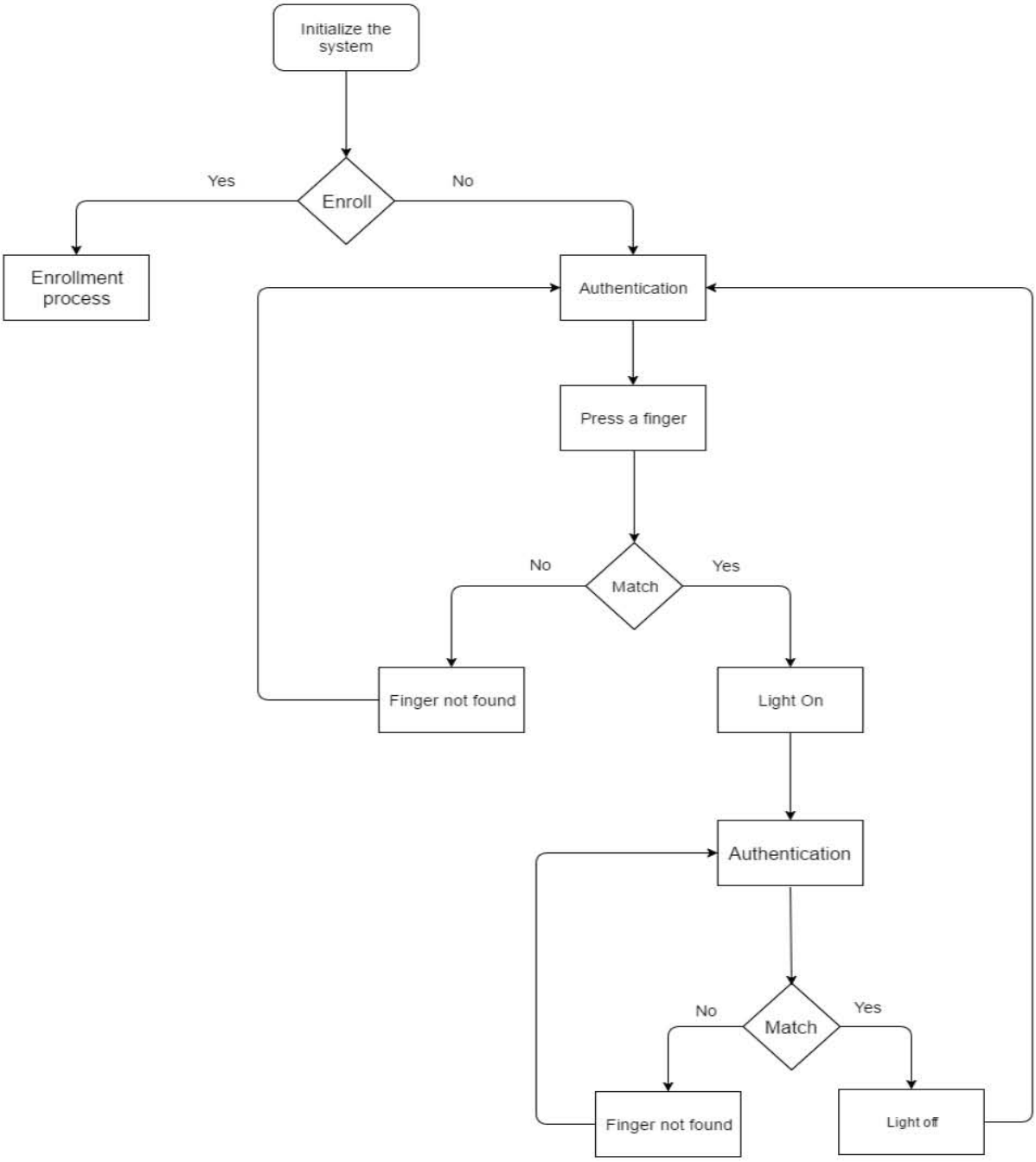

Figure 7: Flowchart of "**Microcontroller Based Current Flow Control Using Biometric Finger Recognition**"

4.2 CONNECTIONS

A Finger Print Sensor (FPS) module is used for finger recognition, one 4 channel 5 volt relay is used for controlling current, one microcontroller for controlling all the devices and one display to view message.
The connection diagrams are mentioned below.

4.2.1 CONNECTION BETWEEN MICROCONTROOLER AND FPS

The FPS module has four pins which are all connected with Arduino board.

- First pin of FPS (Black wire) is connected to pin number 50 of Arduino.
- Second pin of FPS connected to pin number 51 of Arduino.
- Third pin is connected to the GND pin of the Arduino.
- Fourth pin of FPS is connected with the 5V pin of Arduino.

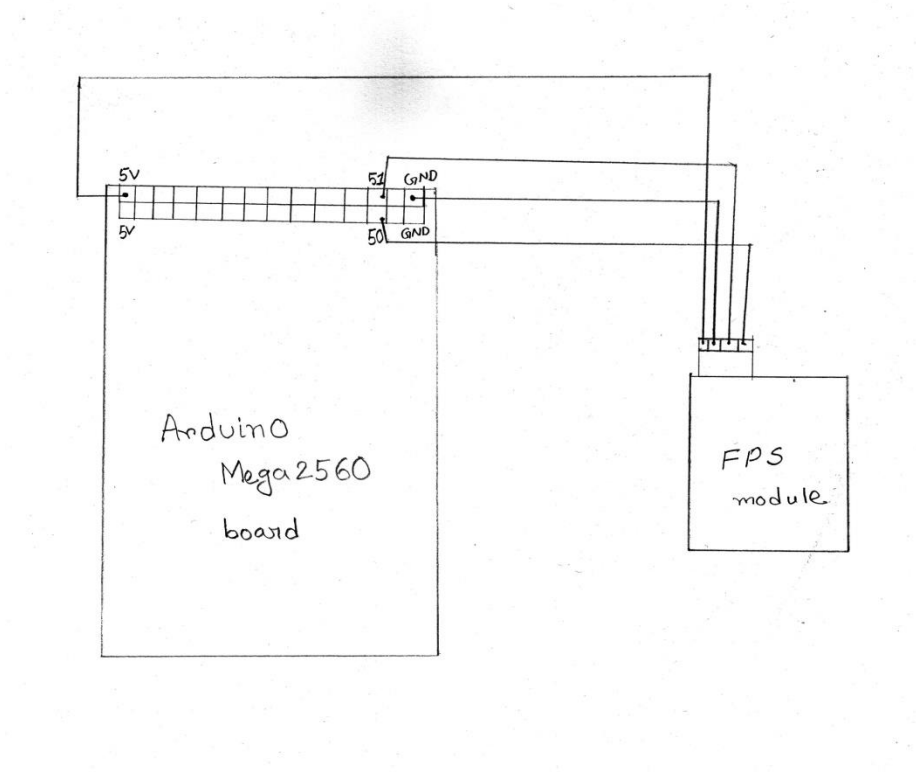

Figure 8: Connection between microcontroller and FPS

4.2.2 CONNECTIONS BETWEEN MICROCONTROLLER, RELAU, AND BULB

The relay has six pins from which three are connected with microcontroller. Every channel of the relay has three pins which are Normally Open (NO) pin, Common (COM) pin, Normally Closed (NC) pin.

- VCC pin of relay is connected 5V pin of Arduino board.
- GND pin of relay is connected GND pin of Arduino board.
- CH 4 pin of the relay is connected pin number 7 of Arduino.
- NO pin of channel 4 is connected (-ve) side of the bulb.
- COM pin of channel 4 is connected (+ve) side of the socket.
- (+ve) side of the bulb is connected (-ve) side of the socket.

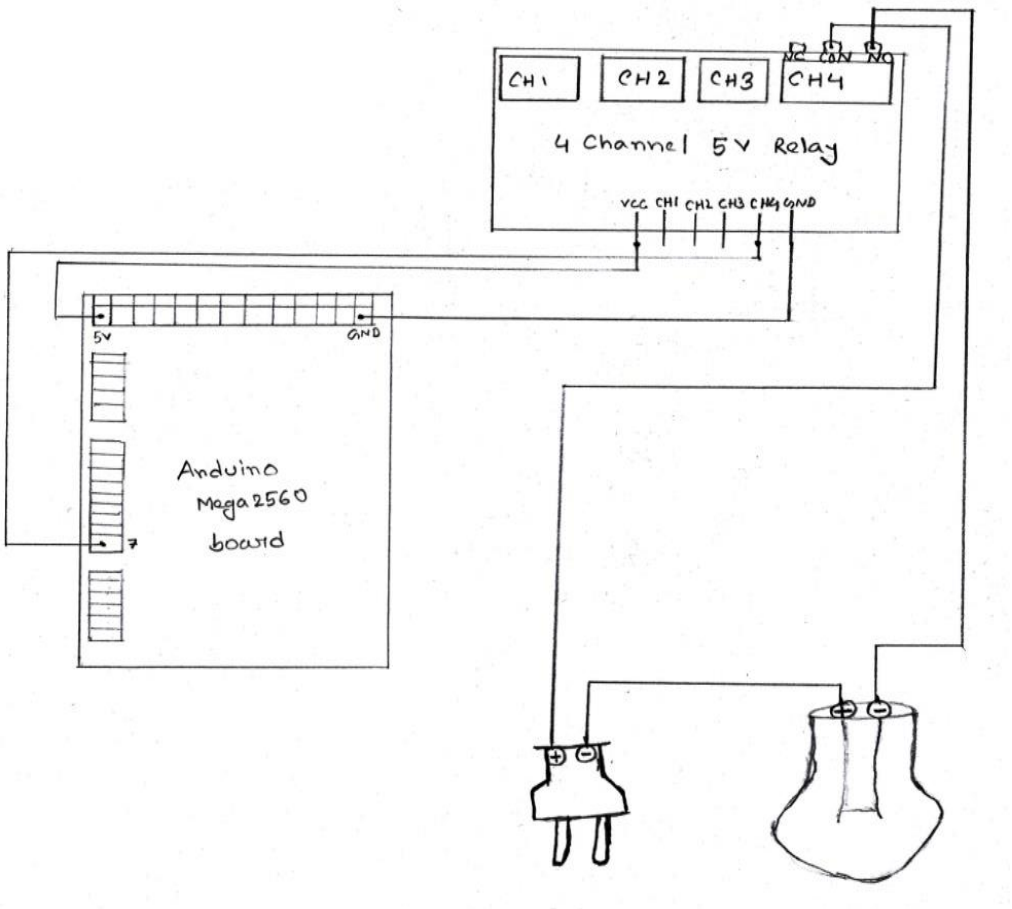

Figure 9: Connections between microcontroller, relay and bulb

4.2.3 CONNECTION BETWEEN DISPLAY AND MICROCONTROLLER

Display has sixteen pins from which 1, 2, 4, 5, 6, 11, 12, 13, 14, 16 number pins are connected to Arduino. Number 3 and 15 pin is connected with different register.

- Pin 1 of display connected to GND of Arduino.
- Pin 2 of display connected to 5V of Arduino.
- Pin 3 of display connected to one side of 2KΩ register and other side is connected to GND of Arduino.
- Pin 4 of display connected to 12 number pin of Arduino.
- Pin 5 of display connected to GND of Arduino.
- Pin 6 of display connected to 11 number pin of Arduino.
- Pin 11 of display connected to 5 number pin of Arduino.
- Pin 12 of display connected to 4 number pin of Arduino.
- Pin 13 of display connected to 3 number pin of Arduino.
- Pin 14 of display connected to 2 number pin of Arduino.
- Pin 15 of display connected to one side of 1KΩ register and other side is connected to 5V of Arduino.
- Pin 16 of display connected to GND of Arduino.

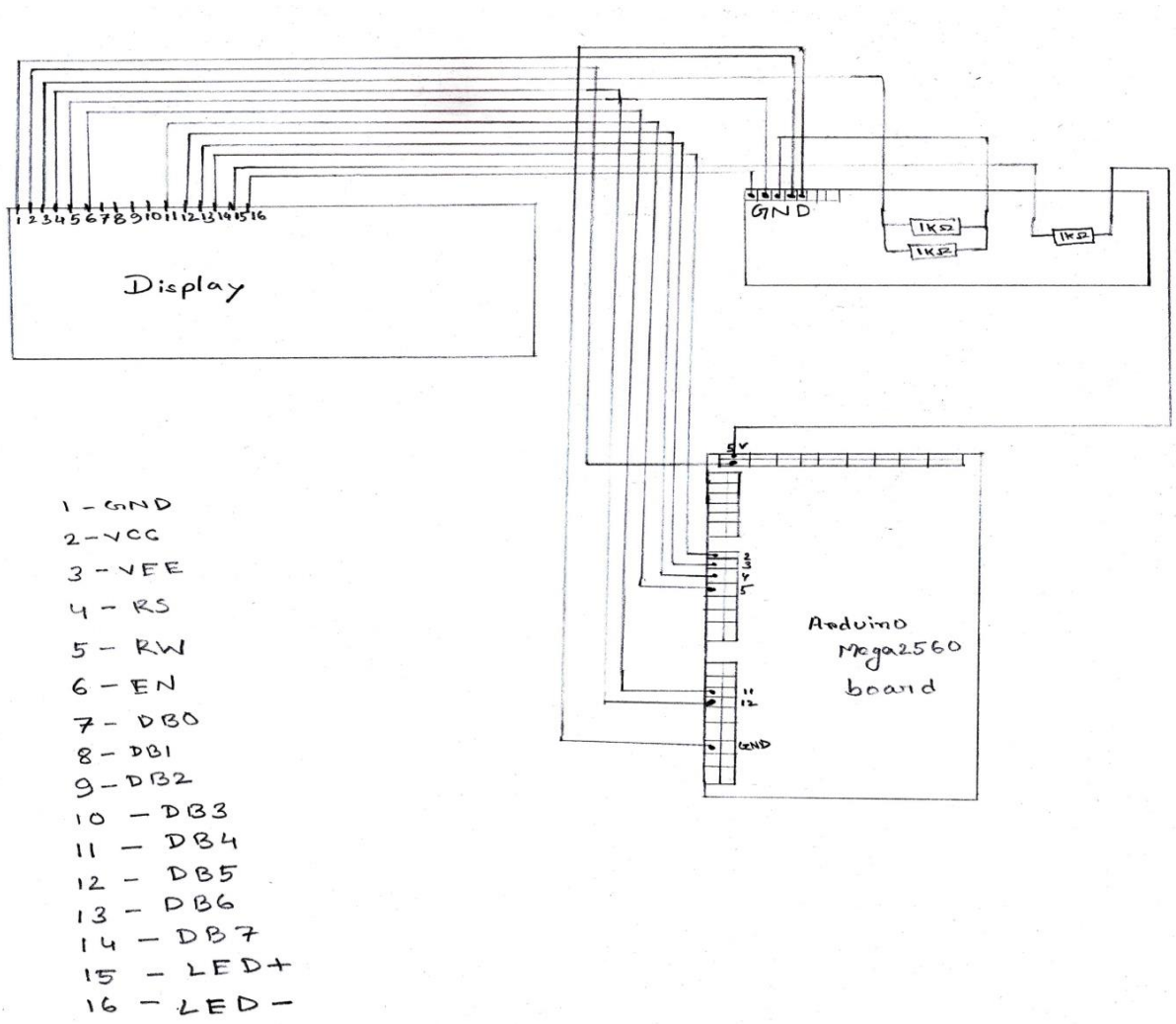

Figure 10: Connection between display and microcontroller

4.3 ARDUINO CODE

4.3.1 CODE

```
// include the library code:
#include "FPS_GT511C3.h"
#include "SoftwareSerial.h"
#include <LiquidCrystal.h>
#define RELAY4 7
int temp;

// initialize the library with the numbers of the interface pins

FPS_GT511C3 fps(50, 51);
// Hardware setup - FPS connected to:
//    digital pin 50(arduino rx, fps tx)
//    digital pin 51(arduino tx, fps rx)

LiquidCrystal lcd(12, 11, 5, 4, 3, 2);
/*The circuit:
 * LCD RS pin to digital pin 12
 * LCD Enable pin to digital pin 11
 * LCD D4 pin to digital pin 5
 * LCD D5 pin to digital pin 4
 * LCD D6 pin to digital pin 3
 * LCD D7 pin to digital pin 2
 * LCD R/W pin to ground
 * LCD VSS pin to ground
 * LCD VCC pin to 5V*/

void setup()
{
  // Open serial communications and wait for port to open:

  Serial.begin(9600);

  delay(100);

  //One of the simplest things to do is to turn on the LED that is inside the
device
```

```
  fps.Open();

  fps.SetLED(true);

  //set the CH4 pin of relay as output
  pinMode(RELAY4, OUTPUT);

  // set up the LCD's number of columns and rows:
  lcd.begin(20, 4);

  // set the cursor to column 4, line 1
  // (note: line 1 is the second row, since counting begins with 0):
  lcd.setCursor(4, 1);

  // Print a message to the LCD.
  lcd.print("WELLCOME !!");

  //delay 5 sec
  delay(5000);

  //fps.DeleteAll();

  //Enroll();
}

void Enroll()
{

  int enrollid = 0;
  bool usedid = true;
  while (usedid == true)
  {
    //Check for an ID slot that is unused until you find one
    usedid = fps.CheckEnrolled(enrollid);
    if (usedid == true) enrollid++;
  }

  //Start and enrollment with that ID
  fps.EnrollStart(enrollid);
```

```
lcd.clear();
lcd.setCursor(0, 0);

Serial.print("Press finger to Enroll #");
Serial.println(enrollid);

lcd.print("Press finger to");
lcd.setCursor(0, 1);
lcd.print(" Enroll # ");
lcd.print(enrollid);

//Wait for a finger press
while (fps.IsPressFinger() == false) delay(100);

//Capture an image of that finger in high resolution
bool bret = fps.CaptureFinger(true);

int iret = 0;
if (bret != false)
{
  Serial.println("Remove finger");

  //Clear all message from the LCD
  lcd.clear();

  lcd.setCursor(0, 1);
  lcd.print("Remove finger");

  //Call Enroll1 to use the image in memory
  fps.Enroll1();

  //Wait for finger to be removed
  while (fps.IsPressFinger() == true) delay(100);

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

  lcd.clear();
  lcd.setCursor(0, 2);
  lcd.print("Press same finger");
  lcd.setCursor(0, 3);
```

```
lcd.print("again");

//Wait for the same finger press
while (fps.IsPressFinger() == false) delay(100);

bret = fps.CaptureFinger(true);
if (bret != false)
{
 lcd.clear();
 lcd.setCursor(0, 1);

 Serial.println("Remove finger");
 lcd.print("Remove finger");

 //Call Enroll2 to use the image in memory
 fps.Enroll2();

 while (fps.IsPressFinger() == true) delay(100);
 Serial.println("Press same finger yet again");

 lcd.clear();
 lcd.setCursor(0, 2);
 lcd.print("Press same finger");
 lcd.setCursor(0, 3);
 lcd.print("yet again");

 while (fps.IsPressFinger() == false) delay(100);
 bret = fps.CaptureFinger(true);
 if (bret != false)
 {
  Serial.println("Remove finger");

  lcd.clear();
  lcd.setCursor(0, 1);
  lcd.print("Remove finger");

  //Call Enroll3 to use the image in memory
  iret = fps.Enroll3();

  if (iret == 0)
```

```
    {
      Serial.println("Enrolling Successfull");

      lcd.clear();
      lcd.setCursor(0, 1);
      lcd.print("     Enrolling     ");
      lcd.setCursor(0, 2);
      lcd.print("    Successfull    ");

      delay(5000);
    }
    else
    {
      Serial.print("Enrolling Failed with error code:");
      Serial.println(iret);

      lcd.clear();
      lcd.setCursor(0, 2);
      lcd.print("Enrolling Failed");
      lcd.setCursor(0, 3);
      lcd.print("with error code:");
      lcd.print(iret);

      delay(2500);
    }
  }
  else
  {
    Serial.println("Failed to capture third finger");

    lcd.clear();
    lcd.setCursor(0, 2);
    lcd.print("Failed to capture");
    lcd.setCursor(0, 3);
    lcd.print("third finger");

    delay(2500);
  }
}
else
```

```
    {
      Serial.println("Failed to capture second finger");

      lcd.clear();
      lcd.setCursor(0, 2);
      lcd.print("Failed to capture");
      lcd.setCursor(0, 3);
      lcd.print("second finger");

      delay(2500);
    }
  }
  else
  {
    Serial.println("Failed to capture first finger");

    lcd.clear();
    lcd.setCursor(0, 2);
    lcd.print("Failed to capture");
    lcd.setCursor(0, 3);
    lcd.print("first finger");

    delay(2500);
  }
}




  // run over and over
void loop()
{
  if (fps.IsPressFinger())
  {
    // Check to see if that image is in the database on the fps
    fps.CaptureFinger(false);

    int id = fps.Identify1_N();
```

```
if (id < 200)
{
  Serial.print("Verified ID:");
  Serial.println(id);

  lcd.setCursor(0, 3);
  lcd.print("Verified ID:");
  lcd.print(id);

  delay(2500);
  if (temp == false)
  {
    temp = true;

    lcd.clear();
    lcd.setCursor(0, 2);

    Serial.println("Light On");

    lcd.print("     Light On     ");
    //sets the bulb on
    digitalWrite(RELAY4, 1);
  }
  else
  {
    lcd.clear();
    temp = false;

    Serial.println("Light Off");

    lcd.setCursor(0, 2);
    lcd.print("     Light Off     ");
  //sets the bulb off
   digitalWrite(RELAY4, 0);
  }
}
else
{
  Serial.println("Finger not found");
```

```
      lcd.clear();
      lcd.setCursor(0, 1);
      lcd.print("Finger not found");
    }
  }
 else
  {
    Serial.println("Please press finger");

    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("Please press finger");
  }

delay(5000);
}
```

4.3.2 ESSENTIAL LIBRARIES

To establish the arduino code for "**Microcontroller Based Current Flow Control Using Biometric Finger Recognition**" three major libraries are used.

- FPS_GT511C3.h
- SoftwareSerial.h
- LiquidCrystal.h

"FPS_GT511C3.h" Library:

Descriptions of the functions from FPS_GT511C3.h Library are given below:

- *int id = fps.Identify1_N();*
  Enrolling fingerprints is a little more complex. You have to do the following steps below in order, and check for errors after each one

  1. Check for an ID slot that is unused until you find one: *fps.CheckEnrolled(enrollid);*

  2. Start and enrollment with that ID: *fps.EnrollStart(enrollid);*

  3. Wait for a finger press: while*(fps.IsPressFinger() == false) delay(100);*

  4. Capture an image of that finger in high resolution: *fps.CaptureFinger(true);*

  5. Call Enroll1 to use the image in memory: *fps.Enroll1();*

6. Wait for finger to be removed: *while(fps.IsPressFinger() == true) delay(100);*

7. Wait for the same finger press: *while(fps.IsPressFinger() == false) delay(100);*

8. Call Enroll2 to use the image in memory: *fps.Enroll2();*

9. Call Enroll3 to use the image in memory: *fps.Enroll3();*

10. If the return value of Enroll3 is zero, the enrollment was successful

- *fps.Open();*
  One of the simplest things to do is to turn on the LED that is inside the device. It needs to be on to see the fingerprints anyways. I typically do this during setup so that all of the looping commands can see.

- *fps.SetLED(true);*
  Now that our FPS can see, we should check to see if there is a finger on it.

- *bool fingerpressed = fps.IsPressFinger();*
  Now that we know there is something to look at, let's capture an image of it. Note that the image is captured to the onboard memory for use with subsequent commands, and not returned to the Arduino. The Boolean parameter is used to enable or disable high resolution reading. In general you use high resolution when enrolling (adding finger print ids), and low resolution when doing everyone else because of speed.

- *fps.CaptureFinger(false);*
  Check to see if that image is in the database on the fps. If it is, this command will return the ID number of it (a number between 0 and 199).

"SoftwareSerial.h" Library:

Descriptions of the functions from SoftwareSerial.h library are given below:

- set the data rate for the SoftwareSerial port: *Serial.begin(9600);*
- Prints data to the transmit pin of the software serial port: *Serial.print();*
- Prints data to the transmit pin of the software serial port, followed by a carriage return and line feed: *Serial.println();*
- Pauses the program for the amount of time (in miliseconds) specified as parameter. (There are 1000 milliseconds in a second.): *delay();*

"LiquidCrystal.h" Library:

Descriptions of the functions from LiquidCrystal.h library are given below:

- set up the LCD's number of columns and rows: *lcd.begin(col, row);*
- set the cursor to column, line (note: line 1 is the second row, since counting begins with 0): *lcd.setCursor(col, line);*
- Print a message to the LCD: *lcd.print();*
- Clear all message from the LCD: *lcd.clear();*

CHAPTER 5

CONCLUSION AND FUTURE WORK

This world is evolving and so is the demand of safety for our data and other important things we have. "**Microcontroller Based Current Flow Control Using Biometric Finger Recognition**" can be a way to think about making our valuable things more secure in the future. This system has a unique level of security complexity that makes it standout of the ordinary systems.

5.1 CONCLUSION

The project presented a work to realize the smart living. Life can be easier and safer with the help of technology. As we are going forward with time, we must accept the necessity of ease and safety.

The system that has been developed in the project, user can control electricity flow by using Finger Print Sensor (FPS) module. Here users can store finger image and use it for security concern. As a lot of finger image can be stored (Up to 20), the system makes the security scheme more efficient and safer. Still as a prototype, "**Microcontroller Based Current Flow Control Using Biometric Finger Recognition**" may need to be worked out more but it can compete with its efficiency with any existing security system in the world.

5.2 FUTURE WORK

"**Microcontroller Based Current Flow Control Using Biometric Finger Recognition**" is still a prototype. It needs further improvements in its hardware section. All the parts used here are built to serve multiple purposes. If we want to make the system work industrially, we have to make purpose built parts.

Possible improvements may be a microcontroller board specially designed for finger print detection. A Finger Print Sensor (FPS) module with less cost and more efficiency to do the simple job. A switching system can be added in the relay board to make it more user friendly. These changes will make it a complete system and will make it more efficient, cost effective and safe. A 5V battery can be used as the power supply and it will have to be charging from wall socket. Then the power failure problem will be solved.

The application of the algorithm can be used in several purposes. With proper hardware, we can implement this security system on almost anything having current flow and it will be as strong as unbreakable. For example we can use this system in cars. Now a days cars are startup with key or switch but if we attached this system with key or switch then after pressing the switch or key the car will not start unless the driver authorize his finger print to the Finger Print Sensor. This will ensure more security of the car.

REFERENCES

- https://www.arduino.cc/en/Main/arduinoBoardMega2560
- https://www.arduino.cc/en/Guide/HomePage
- https://www.arduino.cc/en/Main/Software
- https://www.arduino.cc/en/Reference/softwareSerial
- https://www.arduino.cc/en/Reference/LiquidCrystal
- http://wordpress.hawleyhosting.com/ramblings/?p=375
- http://forum.arduino.cc/index.php?topic=270220.0
- https://www.techshopbd.com/product-categories/boards/1452/arduino-mega-2560-china-techshop-bangladesh
- https://www.techshopbd.com/product-categories/lcd/649/lcd-display-20x4-techshop-bangladesh
- https://www.techshopbd.com/product-categories/modules-98775/1447/4-channel-5v-relay-module-techshop-bangladesh
- http://www.arduino.cc/en/Tutorial/HomePage