Title

Fingerprint scanners are biometric security systems. They are used in police stations, security industries, smartphones,and other mobile devices. This lab  uses Fingerprint sensor AS608 to Enroll Key and Verify Key .

Hardware Required

·        Arduino

·        Finger print module

·        LCD 16x2

·        Keypad 4x4

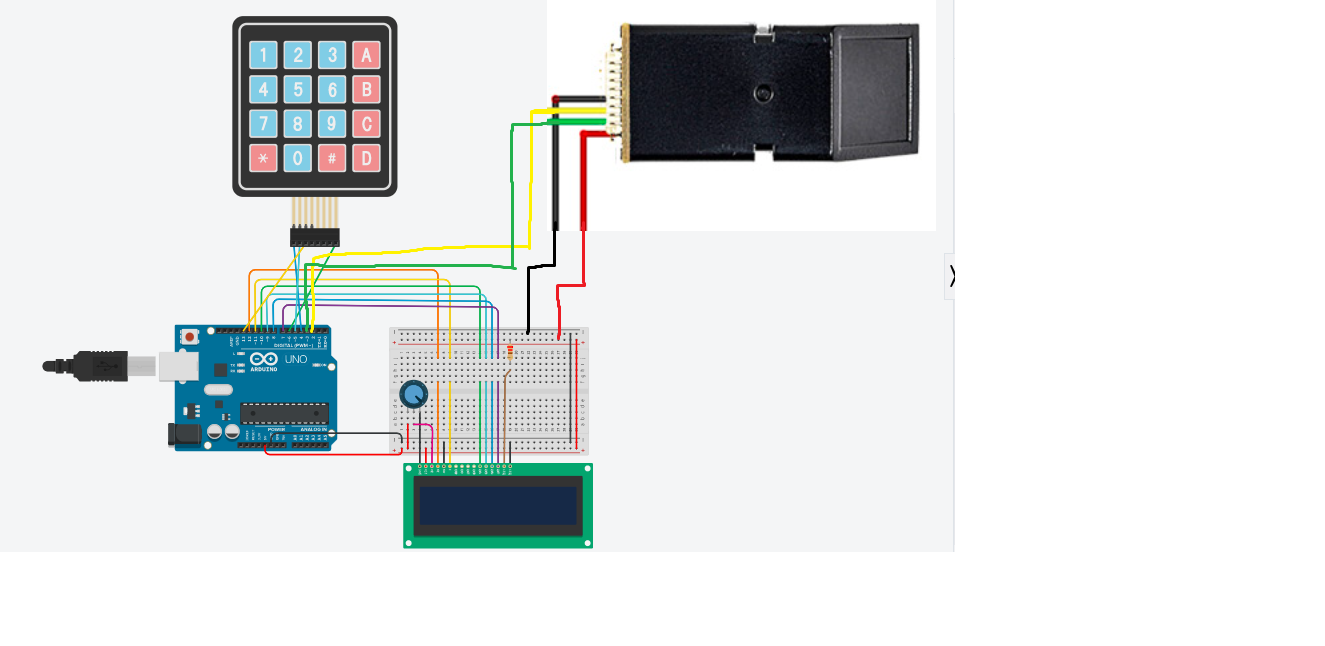
Theory

The program uses several libraries, including **LiquidCrystal, Keypad, SoftwareSerial, and Adafruit\_Fingerprint.** The **LiquidCrystal library** is used to control an LCD display, the **Keypad library** is used to interface with a keypad, the **SoftwareSerial library** is used to create a software serial port, and the **Adafruit\_Fingerprint library** is used to communicate with a fingerprint scanner.

The program has **two main functions**: **setup** and **loop** . The program also includes several **helper functions** such as **getFingerprintEnrol**l, **verifyFingerPrint**, **emptyFingerDB**, and **printInteger**. These functions are used to perform specific tasks such as enrolling fingerprints, verifying fingerprints, emptying the fingerprint database, and printing integers on the LCD display.

Circuit

*(Block diagram and Circuit diagram shows the wire connection).*



To wire your LCD screen to your board, connect the following pins:

·        LCD RS pin to digital pin 12

·        LCD Enable pin to digital pin 11

·        LCD D4 pin to digital pin 10

·        LCD D5 pin to digital pin 9

·        LCD D6 pin to digital pin 8

·        LCD D7 pin to digital pin 7

·        VSS  pin to pin Ground (-)

·        VDD pin to pin +5 Volt(+)

To wire your Keypad to your board, connect the following pins:

·        Keypad ROWS to digital pin 4,5,13

·        Keypad COLS to digital pin 6

To wire your Fingerprint sensor to your board, connect the following pins:

·        Fingerprint 3V3 pin to pin 3V3 volt

·        Fingerprint TX pin to digital pin 3

·        Fingerprint GX pin to digital pin 22

·        Fingerprint GND pin to pin GND

Code

#include <LiquidCrystal.h>

#include <Keypad.h>

#include <SoftwareSerial.h>

#include <Adafruit\_Fingerprint.h>

SoftwareSerial mySerial(2, 3);

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

//keypad

const byte ROWS = 4;

const byte COLS = 4;

char hexaKeys[ROWS][COLS] = {

  {'A', '2', '3', 'A'},

  {'B', '5', '6', 'B'},

  {'C', '8', '9', 'C'},

  {'\*', '0', '#', 'D'}

};

byte rowPins[ROWS] = {4, 5, 13};

byte colPins[COLS] = {6};

Keypad customKeypad = Keypad(makeKeymap(hexaKeys), rowPins, colPins, ROWS, COLS);

//init ID

int id = 1;

//lcd

const int rs = 12, en = 11, d4 = 10, d5 = 9, d6 = 8, d7 = 7;

LiquidCrystal lcd(rs, en, d4, d5, d6, d7);

void setup() {

  Serial.begin(9600);

  finger.begin(57600);

  if (finger.verifyPassword()) {

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

  } else {

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

    while (1) {

      delay(1);

    }

  }

  lcd.begin(16, 2);

  //check ID has fingerPrint or not

  while (finger.loadModel(id) == 0) {

    id++;

  }

  delay(1000);

}

char IdString[100];

void loop() {

  char customKey;

  customKey = customKeypad.getKey();

  lcd.clear();

  lcd.print("A:Enroll");

  lcd.setCursor(0, 1);

  lcd.print("B:Verify");

  lcd.setCursor(9, 0);

  lcd.print("C:Empty");

  while (customKey == '\0') {

    customKey = customKeypad.getKey();

  }

  Serial.println("custom key");

  Serial.print(customKey);

  lcd.clear();

  if (customKey == 'A') {

    lcd.print("Enroll mode");

    delay(2000);

    lcd.clear();

    lcd.print("Enrolling ID #");

    lcd.setCursor(0, 1);

    printInteger(id);

    delay(2000);

    while (!getFingerprintEnroll());

  } else if (customKey == 'B') {

    lcd.print("Verify mode");

    delay(2000);

    while (!verifyFingerPrint());

  }

  else if (customKey == 'C') {

    emptyFingerDB();

  }

}

uint8\_t getFingerprintEnroll() {

  int p = -1;

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

  while (p != FINGERPRINT\_OK) {

    p = finger.getImage();

    switch (p) {

      case FINGERPRINT\_OK:

        Serial.println("Image taken");

        lcd.clear();

        lcd.print("Image taken");

        break;

      case FINGERPRINT\_NOFINGER:

        lcd.clear();

        lcd.print("Detecting");

        lcd.setCursor(0, 1);

        lcd.print("finger ID ");

        for (int i = 0; i < 3; i++) {

          if (IdString[i] >= '0' && IdString[i] <= '9')

            lcd.print(IdString[i]);

          else break;

        }

        lcd.print("...");

        break;

      case FINGERPRINT\_PACKETRECIEVEERR:

        Serial.println("Communication error");

        break;

      case FINGERPRINT\_IMAGEFAIL:

        Serial.println("Imaging error");

        break;

      default:

        Serial.println("Unknown error");

        break;

    }

  }

  delay(2000);

  // OK success!

  p = finger.image2Tz(1);

  switch (p) {

    case FINGERPRINT\_OK:

      Serial.println("Image converted");

      lcd.clear();

      lcd.print("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;

  }

  delay(2000);

  p = finger.fingerSearch();

  if (p == FINGERPRINT\_OK) {

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

    lcd.clear();

    lcd.print("Found a print");

    lcd.setCursor(0, 1);

    lcd.print("Match ID ");

    printInteger(finger.fingerID);

    delay(2000);

    lcd.clear();

    lcd.print("Enroll again");

    delay(2000);

    return false;

  }

  p = 0;

  while (p != FINGERPRINT\_NOFINGER) {

    p = finger.getImage();

  }

  p = -1;

  lcd.clear();

  lcd.print("Place again");

  delay(2000);

  while (p != FINGERPRINT\_OK) {

    p = finger.getImage();

    switch (p) {

      case FINGERPRINT\_OK:

        Serial.println("Image taken");

        lcd.clear();

        lcd.print("Image taken");

        break;

      case FINGERPRINT\_NOFINGER:

        Serial.print(".");

        lcd.print(".");

        break;

      case FINGERPRINT\_PACKETRECIEVEERR:

        Serial.println("Communication error");

        break;

      case FINGERPRINT\_IMAGEFAIL:

        Serial.println("Imaging error");

        break;

      default:

        Serial.println("Unknown error");

        break;

    }

  }

  delay(2000);

  p = finger.image2Tz(2);

  switch (p) {

    case FINGERPRINT\_OK:

      Serial.println("Image converted");

      lcd.clear();

      lcd.print("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;

  }

  delay(2000);

  // OK converted!

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

  p = finger.createModel();

  if (p == FINGERPRINT\_OK) {

    Serial.println("Prints matched!");

    lcd.clear();

    lcd.print("Prints matched!");

    delay(1000);

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

    Serial.println("Communication error");

    return p;

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

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

    lcd.clear();

    lcd.print("Not match");

    delay(1000);

    return false;

  } else {

    Serial.println("Unknown error");

    return p;

  }

  p = finger.storeModel(id);

  if (p == FINGERPRINT\_OK) {

    Serial.println("Stored!");

    id++;

    lcd.clear();

    lcd.print("Success");

    lcd.setCursor(0, 1);

    lcd.print("Enrolling");

    delay(2000);

    lcd.clear();

  } 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;

}

uint8\_t verifyFingerPrint() {

  int p = -1;

  while (p != FINGERPRINT\_OK) {

    p = finger.getImage();

    switch (p) {

      case FINGERPRINT\_OK:

        Serial.println("Image taken");

        lcd.clear();

        lcd.print("Image taken");

        break;

      case FINGERPRINT\_NOFINGER:

        lcd.clear();

        lcd.print("Detecting");

        lcd.setCursor(0, 1);

        lcd.print("finger");

        lcd.print("...");

        break;

      case FINGERPRINT\_PACKETRECIEVEERR:

        Serial.println("Communication error");

        break;

      case FINGERPRINT\_IMAGEFAIL:

        Serial.println("Imaging error");

        break;

      default:

        Serial.println("Unknown error");

        break;

    }

  }

  p = finger.image2Tz();

  switch (p) {

    case FINGERPRINT\_OK:

      Serial.println("Image converted");

      lcd.clear();

      lcd.print("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;

  }

  delay(2000);

  p = finger.fingerSearch();

  if (p == FINGERPRINT\_OK) {

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

    lcd.clear();

    lcd.print("Found a print");

    lcd.setCursor(0, 1);

    lcd.print("Match ID ");

    printInteger(finger.fingerID);

    delay(2000);

    lcd.clear();

    lcd.print("Success");

    lcd.setCursor(0, 1);

    lcd.print("Verify");

    delay(2000);

    lcd.clear();

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

    Serial.println("Communication error");

    return p;

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

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

    lcd.clear();

    lcd.print("Not exist");

    delay(2000);

    lcd.clear();

    lcd.print("Back to menu...");

    delay(2000);

    return true;

  } else {

    Serial.println("Unknown error");

    return p;

  }

  return true;

}

void emptyFingerDB() {

  id = 1;

  finger.emptyDatabase();

  lcd.clear();

  lcd.print("Empty DB success");

  delay(2000);

}

void printInteger(int number) {

  char string[100];

  sprintf(string, "%d", number);

  for (int i = 0; i < 3; i++) {

    if (string[i] >= '0' && string[i] <= '9')

      lcd.print(string[i]);

    else break;

  }

}

Demonstrations

*(Demonstrations with photos of experiments)*

Link video experiments

<https://drive.google.com/drive/folders/1x_x5NHqeSv7vpSS66nGyDlxVjRyAKeEO?usp=sharing>

References

<https://www.instructables.com/How-to-Setup-Fingerprint-Sensor-With-Arduino/>

<https://www.youtube.com/watch?v=gj2bc-u6W-g&t=47s>