בית ספר לאטיני – ריינה

מערכת אבטחה לבית

פרויקט גמר במגמת הנדסת אלקטרוניקה ומחשבים
בהתמחות מערכת אלקטרוניקה בחלופה מערכות תקשורת
סמל שאלון 841387

מאת:
חליל ורור

מספר תעודת זהות:

בהנחיית:
אשרף חוריה
טוני חורי

שנה"ל תש"פ

תוכן עניינים:

דף השער.....1
תוכן עניינים.....2
תקציר פרויקט.....3
רשימת חלקית.....4
סכימה מלבנית.....7
הסבר סכימה מלבנית.....8
סכימה חשמלית.....9
הסבר סכימה חשמלית.....10
טבלת הדקים.....12
הסבר על כל מודול.....13
פרוטוקולי תקשורת.....15
יומן התקדמות.....18
תוכניות בדיקה.....21
תוכנה סופית.....29
הסבר פקודות מיוחדות.....36
הסבר פונקציות.....39
רפלקציה אישית.....41
מראה מקום.....42
נספחים.....43
README.....52

תקציר פרויקט:

הפרויקט מדמה מערכת אבטחה לבית חכם, והפרויקט מנוהל ע״י בקר ארדואינו אונו (ATMega328).

בפרויקט נמצא מסך (LCD) שמציג כאשר יש אפשרות לצלצול הפעמון (Speaker) באמצעות הכפתור (Push Button) ויש חיישן טביעת אצבע (Fingerprint Nano) שיסרוק את האצבע של המשתמש ואם טביעת האצבע מוגדרת, נעילת הדלת (Magnetic lock) תבוטל ותוצג הודעה על המסך שהדלת נפתחה ואחרי כמה שניות יופיע על המסך תזכורת לסגור את הדלת, ויש אפשרות להוספת ולמחיקת טביעות אצבע בעזרת המסך והכפתורים, ואם מישהו ינסה לפרוץ את הדלת - תופעל האזעקה.

מטרת הפרויקט היא לעשות מערכת אבטחה לבית שתשלב את פתיחת דלת באמצעות טביעת אצבע ואזעקה ופעמון הדלת במערכת אחת. זאת על מנת שהמערכת תעבוד בדרך יותר איכותית ואמינה, עם ממשק משתמש קל שכולל בתוכו תפריטים, שדרכם המשתמש ילמד מהר את המערכת.

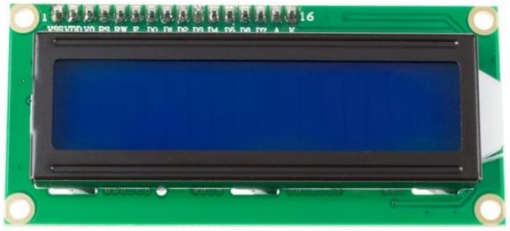
רשימת חלקים:

כפתורים

(Push Button Switches)

מסך

(16x2 LCD)



רמקול

(Speaker)

חיישן טביעת האצבע

(Adafruit Fingerprint Sensor)

מְמסָר

(Relay)

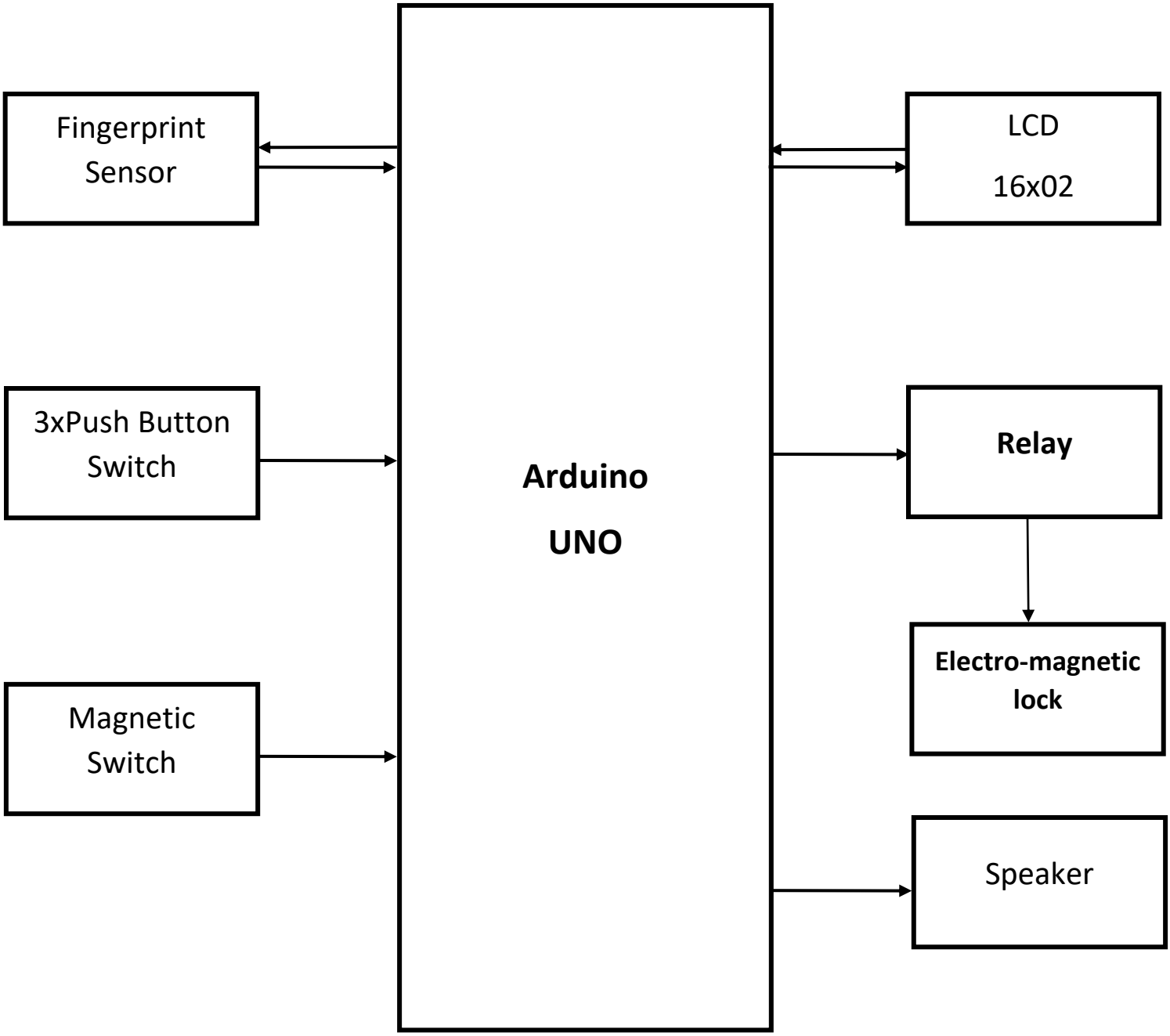
מפסק מגנטי

(Magnetic Switch)

מנעול מגנטי

(Magnetic Lock)

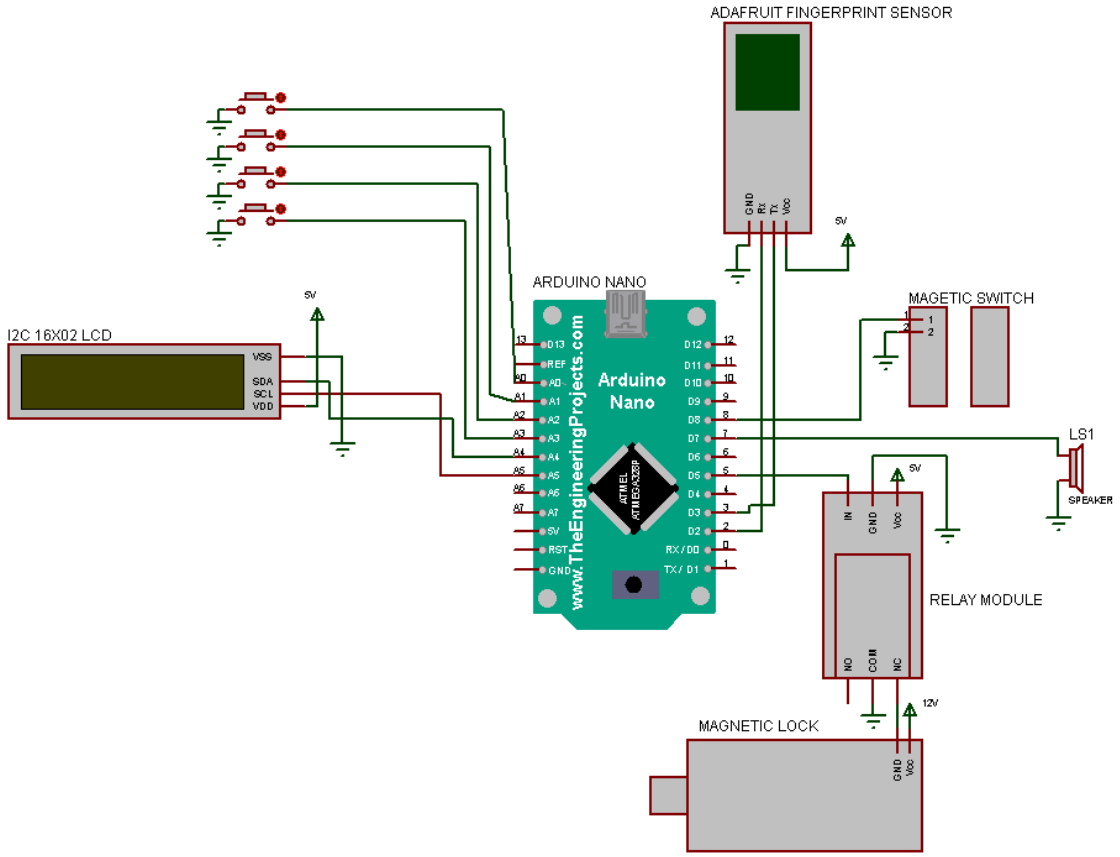
סכימה מלבינית:



הסבר סכימה מלבנית:

- חיישן טביעת אצבע (Fingerprint Sensor): ישנם שני חצים בשני הכיוונים כי הוא מתקשר על הארדוינו שולח ומקבל מידע, והוא קלט למערכת כיוון שהוא קורא טביעות אצבע ומחזיר תוצאות לארדוינו.
- כפתורים (Push Button Switches): יוצא חץ בכיוון הארדוינו כי הם קולטים ומכניסים מידע.
- מפסק מגנטי (Magnetic Switch): יוצא חץ בכיוון הארדוינו כי הוא קולט ומכניס מידע.
- מסך (I^2C LCD): ישנם שני חצים בשני הכיוונים מכיוון שהוא מתקשר על הארדוינו, שולח ומקבל מידע, והוא פלט למערכת מאחר והוא מוציא מידע למשתמש.
- ממסר (Relay): נכנס חץ מהארדוינו לממסר כי הוא פולט ומקבל מידע מהאדוינו.
- רמקול (Speaker): נכנס חץ מהארדוינו לרמקול כי הוא פולט ומקבל מידע מההארדוינו.
- מנעול מגנטי (Magnetic Switch): נכנס חץ מהממסר לרמקול מכיוון שהוא פולט ומקבל מידע מהממסר.

סכימה חשמלית:



הסבר סכימה חשמלית:

רגלי הארדויני:

D2 – מחוברת לRx של חיישן טביעת האצבע.

D3 – מחוברת לTx של חיישן וביעת האצבע.

D5 – מחוברת לIN של המְמסָר.

D7 – מחוברת לרמקול.

D8 – מחוברת למפסק המגנטתי.

A0 – A3 – מחוברות לכפתורים.

A4 – מחוברת לSDA של המסך.

A5 – מחוברת לSCL של המסך.

רגלי הממסר:

Vcc – מחוברת למקור מתח 5V.

GND – מחוברת לערכה.

NI – מחוברת לD5 של הארדוינו.

NC – מחוברת לGND של המנעול המגנטי.

COM – מחוברת לערכה.

רגלי המנעול המגנטי

Vcc – מחוברת למקור מתח 12V.

GND – מחוברת לNC של הממסר.

רגלי הרמקול:

(+) – מחוברת לD7 של הארדוינו.

(-) – מחוברת לארכה.

רגלי המפסק המגנטי:

(1) – מחוברת לD8 של הארדוינו.

(2) – מחוברת לארכה.

רגלי חיישן טביעת האצבע:

Vcc – מחוברת למקור מתח 5V.

Rx – מחוברת לD2 של הארדוינו.

Tx – מחוברת לD3 של הארדוינו.

GND – מחוברת לערכה.

רגלי המסך:

VDD – מחוברת למקור מתח 5V.

SCL – מחוברת לA5 של הארדוינו.

SDA – מחוברת לA4 של הארדוינו.

VSS – מחוברת לערכה.

טבלת הדקים:

שם הרכיב	כיוון זרימת המידע	הדקים בכרטיס הארדוינו
מפסק מגנטי (Magnetic Switch)	כניסה (Input)	D8 כניסה דיגיטאלית (Digital Input)
מִמסָר (Relay)	יציאה (Output)	D5 יציאה דיגיטאלית (Digital Output)
רמקול (Speaker)	יציאה (Output)	D7 יציאה דיגיטאלית (Digital Output)
חיישן טביעת אצבע (Fingerprint Sensor)	תקשורת (UART)	TxD – D2, RxD – D3 (Protocol UART)
מסך (16x2 LCD)	תקשורת (I^2C)	SDA – A4, SCL – A5 (Protocol I^2C)
קפתורים (Push Button Switches)	כניסה (Input)	A0 – A3 כניסה דיגיטאלית (Digital Input)

הסבר על כל מודול:

חיישן טביעת אצבע (Adafruit Fingerprint):

הוא חיישן אופטי שמכיל שבב בעל עוצמה גבוהה שמטפל בציור תמונות, חישוב מידע, חיפוש ומציאת תכונות. הוא מתחבר ומתקשר לכל בקר או מערכת באמצעות תקשורת טורית בכדי לשלוח מידע ופקודות. חיישן זה יכול לאחסן עד 162 טביעת אצבע בזיכרון שלו.

מסך (LCD 16x2 I²C):

השם LCD , הוא קיצור של Liquid Crystal Display ובעברית: "תצוגת גביש נוזלי", התצוגה איננה מקרינה אור (כמו ב- LED או תצוגת 7 Segments), אלא משתמשת באור הסביבה כדי למסור מידע. ה-LCD היא תצוגה אלפאנומרית, שניתן להציג בה אותיות, סימנם וספרות. התצוגה בנויה משלושים ושתיים משבצות המסודרות בשש עשרה עמודות ושתי שורות, כל משבצת בנויה ממטריצת פיקסלים המסודרים בצורת 5 עמודות ו- 7 שורות, הארת הפיקסלים המתאימים גורמת להצגת צורת האות או הספרה המבוקשת על התצוגה.

תצוגה זו כוללת בדרך כלל 2 מעלים משולבים אשר מבקרים את פעולת הרכיב, וכוללים מעגלים אשר יודעים להתחבר אל מיקרו בקר ולקבל ממנו פקודות ונתונים. כמו כן, קיימים מעגלי זיכרון מסוג RAM ו ROM אשר יודעים להציג את התווים הנשלחים מהמיקרו במקום הרצוי בתצוגה.

ממסר (Relay):

ממסר או מגען, הינו התקן אשר מכיל את המפסק ואת הסליל. כאשר זורם זרם בסליל, הסליל יוצר שדה מגנטי, הגורם לתנועה שמועברת למפסק, והמפסק עובר ממצב "רגיל" למצב "מופעל". קיימת הפרדה גלוונית (שאינה מאפשרת מעבר זרם חשמלי) בין חיבורי הסליל לחיבורי המפסק, כך שהתקן הזה, מאפשר למשל חיבור וניתוק של מעגלי מתח רשת על ידי מעגלי מתח נמוך, תוך שמירת הבידוד ביניהם.

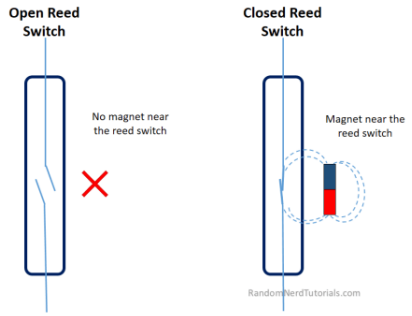
רמקול (Speaker):

רמקול בסיסי מורכב ממגנט קבוע ומסליל של מוליך חשמלי החופשי לנוע בתוך השדה המגנטי הקבוע של המגנט. אל הסליל מחוברת דיאפרגמה - (ממברנה דקה, צפידה למחצה) בצורה המזכירה חרוט רחב בסיס. שולי הממברנה מחוברים אל מסגרת מתכתית בחיבור גמיש שאינו מונע את תנועתה. המגנט הקבוע מחובר אל המסגרת המתכתית בחיבור קשיח. כאשר זרם חשמלי עובר בסליל, מושרה מתוך הסליל שדה אלקטרומגנטי שעוצמתו וכיוונו יחסיים לעוצמת הזרם החשמלי ולכיוונו. שדה אלקטרומגנטי זה, גורם לתנועה של הסליל ביחס למגנט הקבוע. תנועה זו נגרמת מהכוחות הפועלים בין השדה המגנטי הקבוע והשדה האלקטרומגנטי המושרה מתוך הסליל. תנועת הסליל גורמת לתנועה של הממברנה המחוברת אליו, וזו דוחפת את האוויר שמסביבה בכיוון ובעוצמה יחסיים לכיוונו ולעוצמתו של הזרם החשמלי העובר בסליל. מכאן, שכאשר הזרם החשמלי העובר בסליל מתנודד, הוא יגרום לתנודה של הממברנה בקצב ובעוצמה התואמים לקצב ועוצמת תנודות הזרם, ולהרעדת האוויר סביב הממברנה. תנודות האוויר המופקות באופן זה הן, בעצם, גלי קול או צלילים.

מפסק מגנטי (Magnetic switch):

הוא בעצם מתג קנה (מתג הקנה הוא מתג חשמלי המופעל על ידי שדה מגנטי מיושם) הכלול במעטפת פלסטיק כך שניתן ליישם אותו בקלות בדלת, בחלון או במגירה, וזאת כדי לגלות אם הדלת פתוחה או סגורה.

המעגל החשמלי נסגר כאשר המגנט נמצא קרוב למתג (פחות מ- 13 מ״מ). ומאידך, כאשר המגנט רחוק מהמתג, המעגל יפתח.



מנעול מגנטי (Magnetic Lock):

הוא מנעול הבנוי מסליל ,מגנט וקפיץ, המגנט מובנה על ראש המנעול והסליל נמצא סביב המגנט מאחוריו, והקפיץ מושך את ראש המנעול בחוץ כדי לשמור על המנעול במצב נעילה.

כאשר מעבירים זרם בסליל נוצר שדה אלקטרומגנטי שימשך על ידי ראש המנעול אחורה והנעילה תבוטל.

פרוטוקולי תקשורת:

תקשורת טורית (UART):

UART (Universal Asynchronous Receiver Transmitter) הוא פרוטוקול תקשורת נפוץ לתקשורת טורית אסינכרונית.

הפרוטוקול מאפשר שליטה על הפרמטרים האלה:

- קצב העברת הנתונים: מסיבות היסטוריות, הקצבים האפשריים הם 50, 300, 1200, 2400, 4800, 9600, 14400, 19200, 18800, 57600, 115200 סיביות לשנייה. אבל כאשר לא משתמשים בבקר UART קני, ניתן להשתמש בקל קצב רצוי בהתאמה למגבלות הפיזיקאליות.
- כמות סיביות המידע בכל מסגרת: ניתן להשתמש ב 7, 6, 5, 8 סיבית בכל מסגרת.
- הוספת סיבית בדיקה (זוגיות): יש אפשרות להוסיף לכל מסגרת סיבית שמכילה אם מספר ה bit 1 במסגרת הוא זוגי או לא זוגי כדי לתפוס את החריגות בשליחה, וניתן לציין אם המשמעות של הסיבית 1 תהיה זוגית או לא זוגית.
- משך סימבול הסיום: ניתן לקבוע את משך הזמן של שידור סימבול הסיום לאורך של 1, 1.5 או 2 סביות.

ניתן לחשב את קצב המידע על פי הנוסחה:

$$DataRate = BuadRate \cdot \frac{D}{D + P + SB + 1}$$

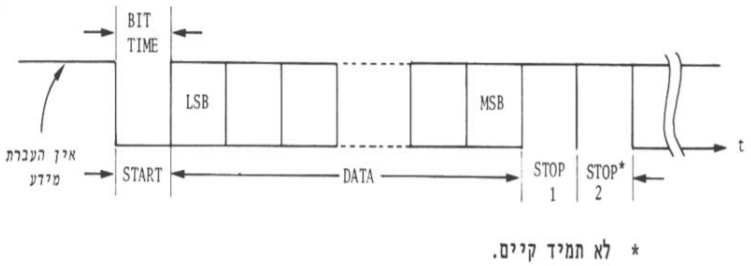
כך ש:

SB אורך סימבול הסיום

P קיום סיבית זוגיות (0 אם לא קיימת, 1 אם קיימת)

D כמות סיביות המידע בכל מסגרת.

מסגרת UART:



תקשורת I^2C:

תקן IIC (Inter-Integrated Circuit) נועד לאפשר תקשורת טורית בין מכשיר MASTER ומכשירי SLAVE. התקן הזה לא מגדיר את המידע אלא את אופן ותזמון השידור והקליטה שלו.

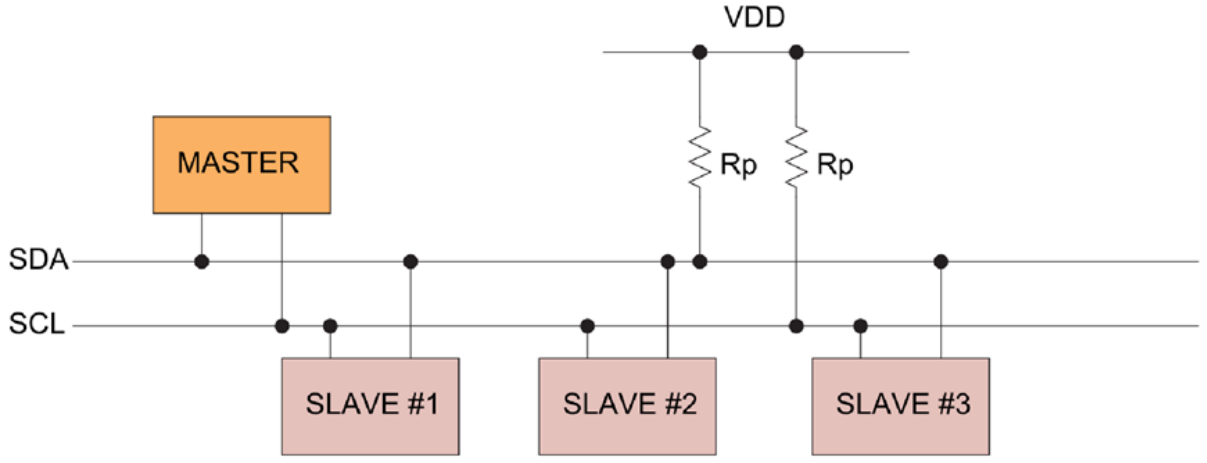
יש כמה קצבי שידור והם :

- Standard Mode – 100 Kbit/s
- Fast Mode – 400 Kbit/s
- Fast Mode Plus – 1 Mbit/s
- High Speed Mode – 3.4 Mbit/s
- Ultra Fast Mode – 5 Mbit/s

המרחק המומלץ בין מכשירים הוא עד 10 ft. התקן מאפשר זיהוי ההתנגשות וסנכרון שעון.

ההתקן מבוסס על העברת מידע באמצעות שני קווי תקשורת דו כיווניים :

- SCL – Serial Clock Line
- SDA – Serial Data Line



הפרוטוקול משתמש בכתובות 7 bits (7-bit address) אז ניתן להשתמש ב127 Slaves ביותר, אבל סכום הקיבול המקסימלי המותר בקו התקשורת הוא 400pF.

מכשיר הMaster תמיד מתחיל את התקשורת באמצעות העברת קו הSDA לאורכה בזמן שקו הSCL מחובר למתח הגבוה (בדרך כלל .(5V

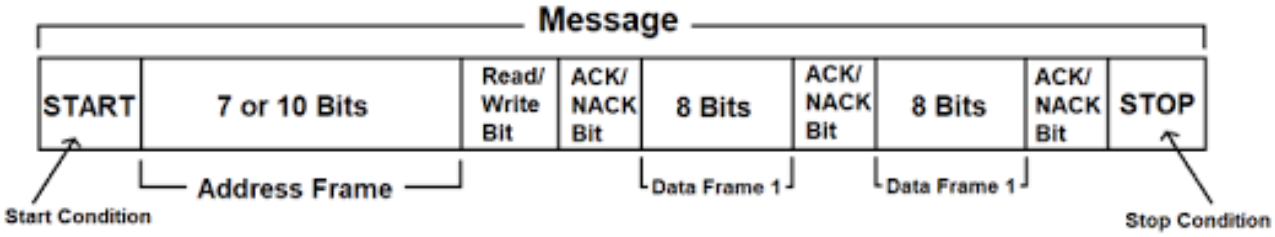
אחרי שהMaster מתחיל את התקשורת הוא שולח את כתובת ה 7bit של מכשיר הSlave שרוצה לקשר איתו ועוד bit אחת שקובעת אם הMaster רוצה לכתוב או לקרוא מהSlave . אם הוא שולח "1" אז הוא קורא מהSlave ו אם "0" הוא שולח ל.Slave

אחר כך הMaster מחכה שהSlave יחזיר ACK שהוא "0" ובמידה והיא אומרת שהכל בסדר וניתן להמשיך בתקשורת. אם הSlave מחזיר NACK שהוא "1" אז יש בעיה שהתרחשה בהעברת המידע ואי אפשר להמשיך בתקשורת.

אם הMaster קיבל את הACK הוא שולח כתובת 8bit של המקום בזיכרון שהוא רוצה לכתוב בו או לקרוא ממנו ומחכה לעוד ACK מה.Slave

אחרי שהMaster מקבל את הACK אם הוא רוצה לקרוא מידע אז הSlave שולח מידע באורך 8bit והMaster מחזיר לו ACK. אם הMaster רוצה לשלוח מידע אז הוא שולח 8bit של מידע ומחכה ל.ACK

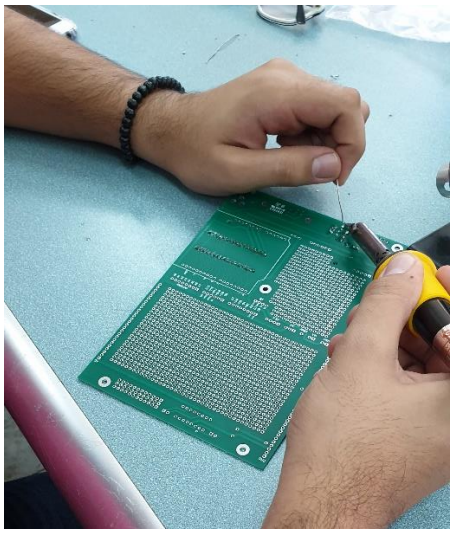
בסוף הMaster מסיים את התקשורת באמצעות העברת את הSDA מאורכה למתח הגבוה בזמן ש הSCL נמצא במתח הגבוה.

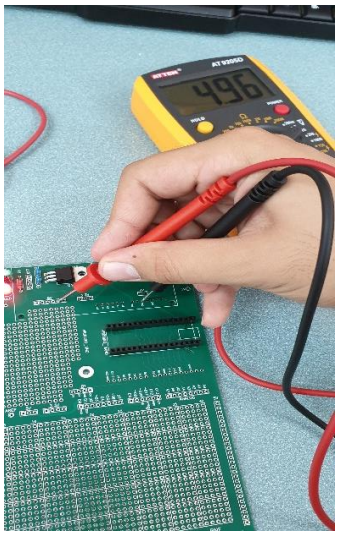


יומן התקדמות:

תאריך 12.10.2019 :

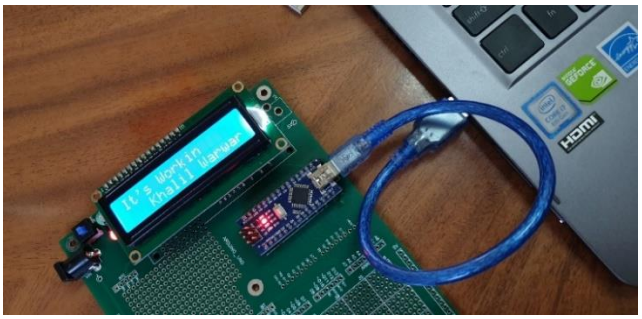
הלחמתי את הרכיבים הבסיסיים של הכרטיס ובדקתי את המתח של ההדקים.





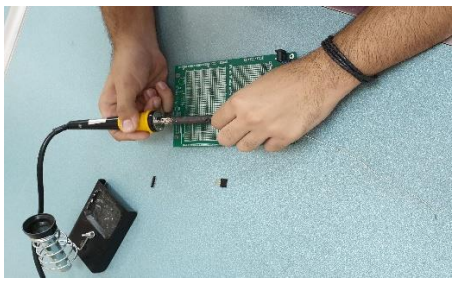
תאריך 13.10.2019 :

כתבתי תוכנה בסיסית למשך וניסיתי אותה.



תאריך 26.10.2019 :

הלחמתי את הדקים על הכרטיס.

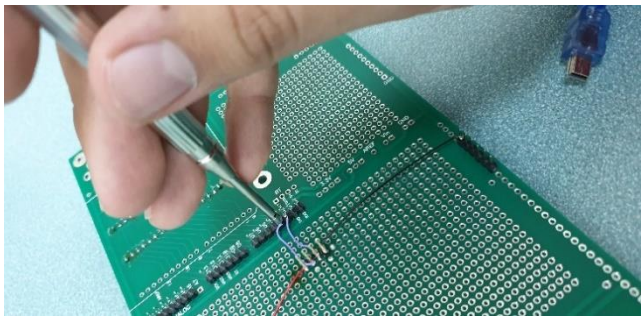


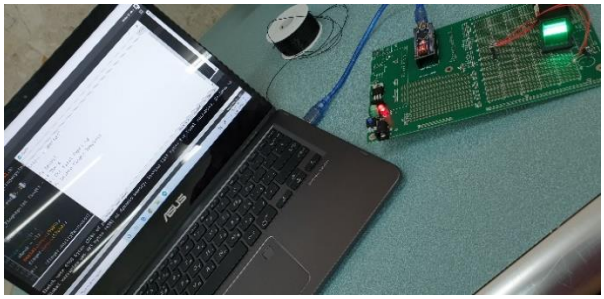
תאריך 29.10.2019 :

כתבתי תוכנה בסיסית לחיישן טביעת האצבע.

תאריך 2.11.2019 :

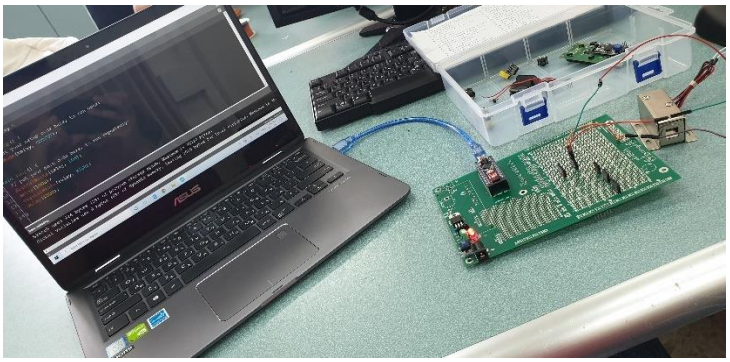
עשיתי חיבור לחיישן טביעת האצבע ובדקתי אותו עם התוכנה.





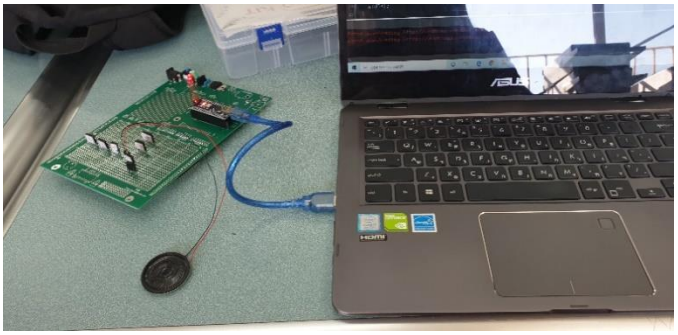
תאריך 4.11.2019 :

כתבתי תוכנה בסיסית למִמסָר ולמנעול המגנטי ובדקתי אותה.



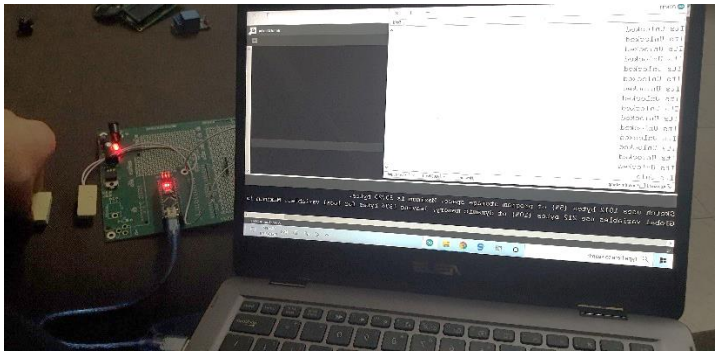
תאריך 5.11.2019 :

כתבתי תוכנה בסיסית לרמקול ובדקתי אותה.



תאריך 16.11.2019 :

כתבתי תוכנה בסיסית למפסק המגנטי ובדקתי אותה.

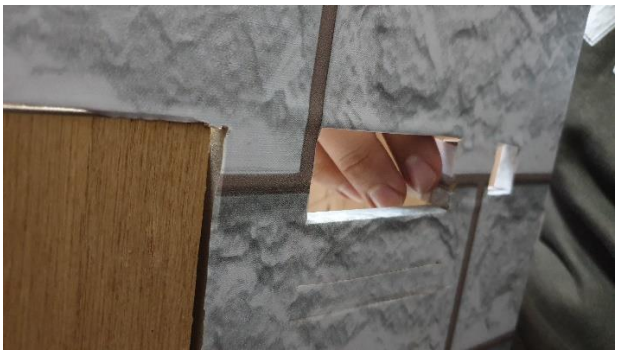


תאריך 25.2.2020 :

עשיתי את הבית אצל נגר.

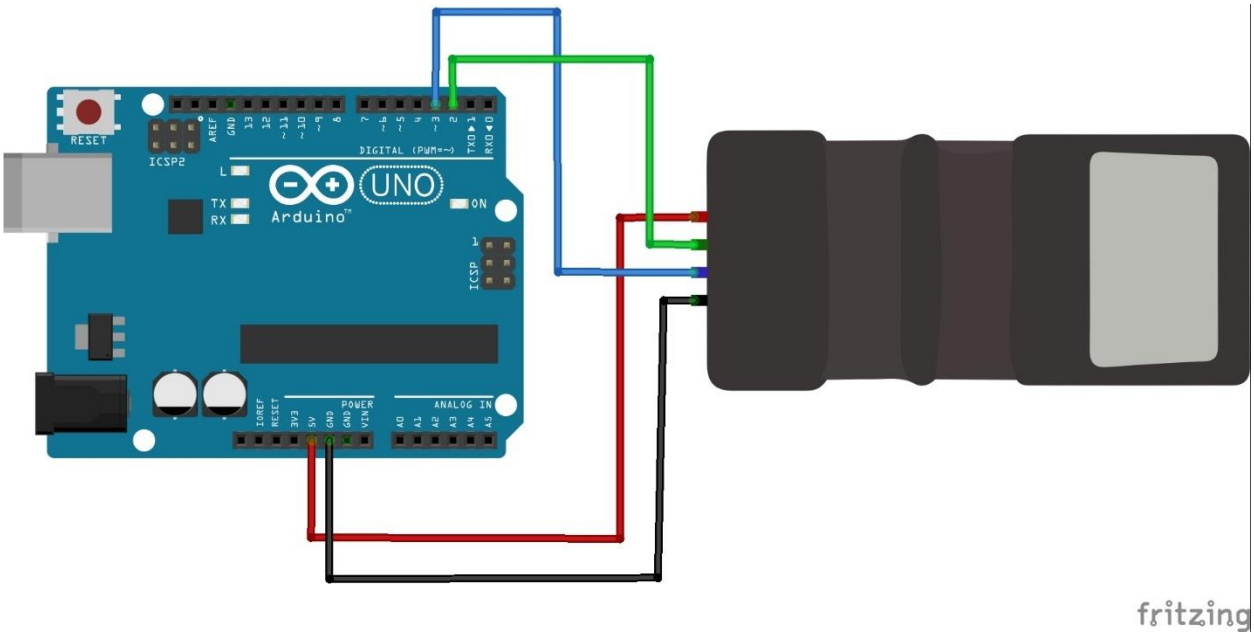
תאריך 2.3.2020 :

קישטתי את הבית.



תוכניות בדיקה:

חיישן טביעת האצבע:



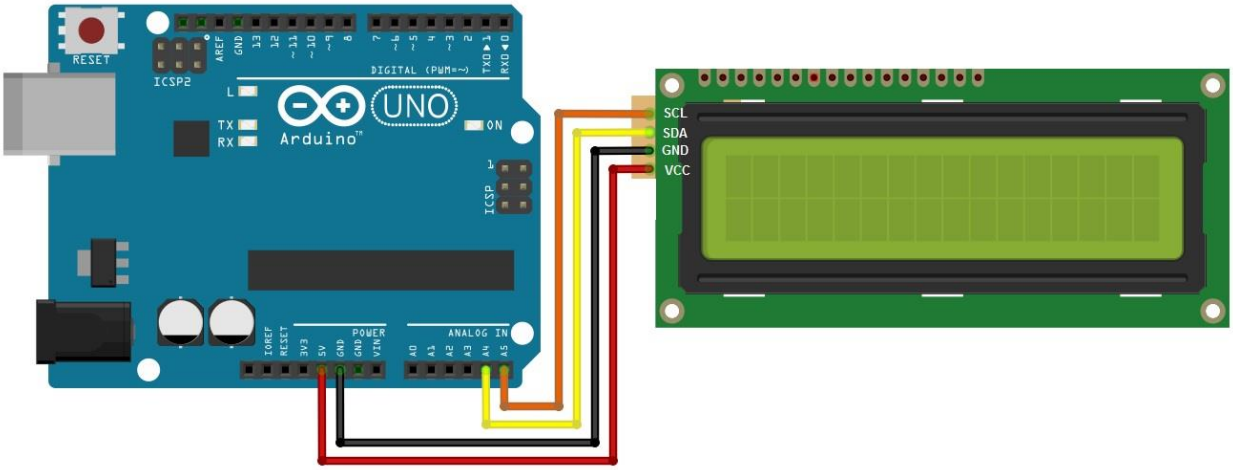

```
#include <SoftwareSerial.h>
#include <Adafruit_Fingerprint.h>

SoftwareSerial ser(3, 2);

Adafruit_Fingerprint finger = Adafruit_Fingerprint(&ser);

int check;

void setup() {
  check = -1;
  Serial.begin(9600);
  finger.begin(57600);
a: ;
  if (finger.verifyPassword()) {
    Serial.println("Sensor Connected!");
  } else {
    Serial.println("Can't Find Sernsor");
    delay(2000);
    goto a;
  }
  Serial.println("\nE: Enroll\nC: Check\nF: Get First Empty Id\nD: Delete Finger Template");
}

void loop() {
  if (Serial.available()) {
    char in = Serial.read();
    if (in == 'E') {
      enroll();
      Serial.println("\n\n\nE: Enroll\nC: Check\nF: Get First Empty Id\nD: Delete Finger Template");
    } else if (in == 'C') {
      scan();
      Serial.println("\n\n\nE: Enroll\nC: Check\nF: Get First Empty Id\nD: Delete Finger Template");
```

```
    } else if (in == 'F') {
      Serial.print(getEmptyID());
      Serial.println(" Is Empty");
      Serial.println("\n\n\nE: Enroll\nC: Check\nF: Get First Empty Id\nD: Delete Finger Template");
    } else if (in == 'D') {
      deleteTemplate();
      Serial.println("\n\n\nE: Enroll\nC: Check\nF: Get First Empty Id\nD: Delete Finger Template");
    }
  }
}

void getImage2Tz(int times = 1) {
  for (int i = 1; i <= times; i++) {
    check = -1;
    while (check != FINGERPRINT_OK) {
      check = finger.getImage();
      if (check == FINGERPRINT_NOFINGER) {
        Serial.println("Put Your Finger On The Sensor");
        while (check == FINGERPRINT_NOFINGER) {
          check = finger.getImage();
        }
      }
      switch (check) {
        case FINGERPRINT_OK:
          Serial.println("Image taken");
          break;
        case FINGERPRINT_PACKETRECIEVEERR:
          Serial.println("Communication error");
          break;
        case FINGERPRINT_IMAGEFAIL:
          Serial.println("Imaging error");
          break;
        default:
          Serial.println("Unknown error");
          break;
      }
      if (check == FINGERPRINT_OK) {
        check = finger.image2Tz(i);
        switch (check) {
          case FINGERPRINT_OK:
            Serial.println("Image converted");
            break;
          case FINGERPRINT_IMAGEMESS:
            Serial.println("Image too messy");
            break;
          case FINGERPRINT_PACKETRECIEVEERR:
            Serial.println("Communication error");
            break;
          case FINGERPRINT_FEATUREFAIL:
            Serial.println("Could not find fingerprint features");
            break;
          case FINGERPRINT_INVALIDIMAGE:
            Serial.println("Could not find fingerprint features");
            break;
          default:
            Serial.println("Unknown error");
            break;
        }
      }
      Serial.println("Remove Your Finger");
      while (finger.getImage() != FINGERPRINT_NOFINGER && times == 2);
    }
  }
}

```

```
bool createStoreModel(int id) {
  Serial.println("Creating Model");
  check = finger.createModel();
  bool res = false;
  do {
    switch (check) {
      case FINGERPRINT_OK:
        Serial.println("Prints matched!");
        break;
      case FINGERPRINT_PACKETRECIEVEERR:
        Serial.println("Communication error");
        res = !res;
        break;
      case FINGERPRINT_ENROLLMISMATCH:
        Serial.println("Fingerprints did not match");
        break;
      default:
        Serial.println("Unknown error");
        res = !res;
        break;
    }
  } while (res);
  if (check == FINGERPRINT_OK) {
    Serial.println("Storing Model");
    finger.storeModel(id);
    res = false;
    do {
      switch (check) {
        case FINGERPRINT_OK:
          Serial.println("Stored!");
          break;
        case FINGERPRINT_PACKETRECIEVEERR:
          Serial.println("Communication error");
          res = !res;
          break;
        case FINGERPRINT_BADLOCATION:
          Serial.println("Could not store in that location");
          break;
        case FINGERPRINT_FLASHERR:
          Serial.println("Error writing to flash");
          res = !res;
          break;
        default:
          Serial.println("Unknown error");
          break;
      }
    } while (res);
    if (check == FINGERPRINT_OK)
      return true;
    else
      return false;
  } else return false;
}

void enroll() {
  int id = readId();

  do {
    getImage2Tz(2);
  } while (!createStoreModel(id));

}

void scan() {
  getImage2Tz();
```

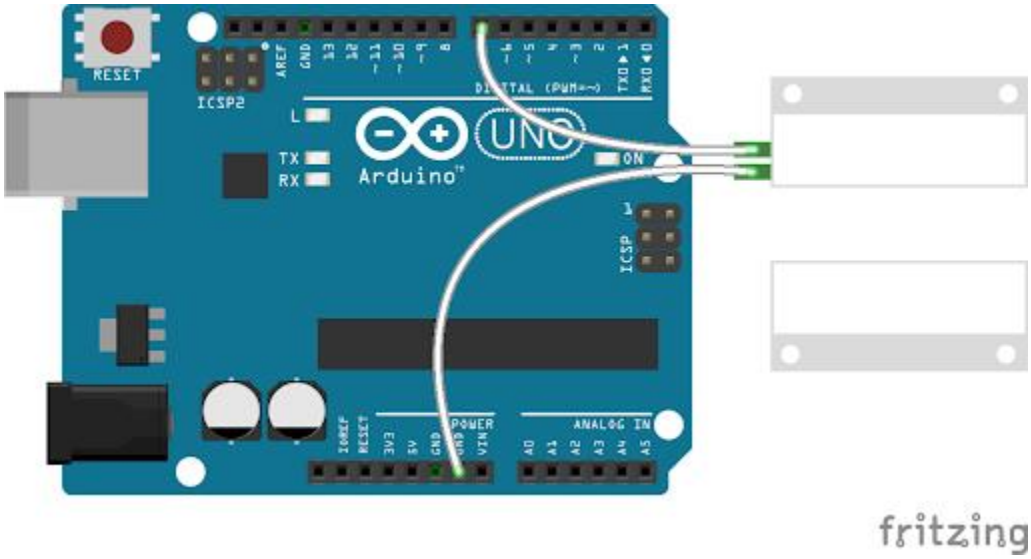
```
  check = finger.fingerFastSearch();
  switch (check) {
    case FINGERPRINT_OK:
      Serial.print("Found a print match!\nFinger Id = ");
      Serial.println(finger.fingerID);
      break;
    case FINGERPRINT_PACKETRECIEVEERR:
      Serial.println("Communication error");
      break;
    case FINGERPRINT_NOTFOUND:
      Serial.println("Did not find a match");
      break;
    default:
      Serial.println("Unknown error");
      break;
  }
}

int getEmptyID() {
  finger.getTemplateCount();
  if (finger.templateCount < 127) {
    for (int i = 1; i <= 127; i++) {
      if (finger.loadModel(i) != FINGERPRINT_OK) {
        return i;
        break;
      }
    }
  } else return -1;
}

void deleteTemplate() {
  int id = readId();
  check = finger.deleteModel(id);

  if (check == FINGERPRINT_OK) {
    Serial.println("Deleted!");
  } else
    Serial.println("Failed To Delete");
}

int readId() {
  Serial.println("Enter Finger Id 1-127");
  int id = 0;
  while (id == 0) {
    if (Serial.available()) {
      id = Serial.parseInt();
    }
  }
  return id;
}
```

מסך :



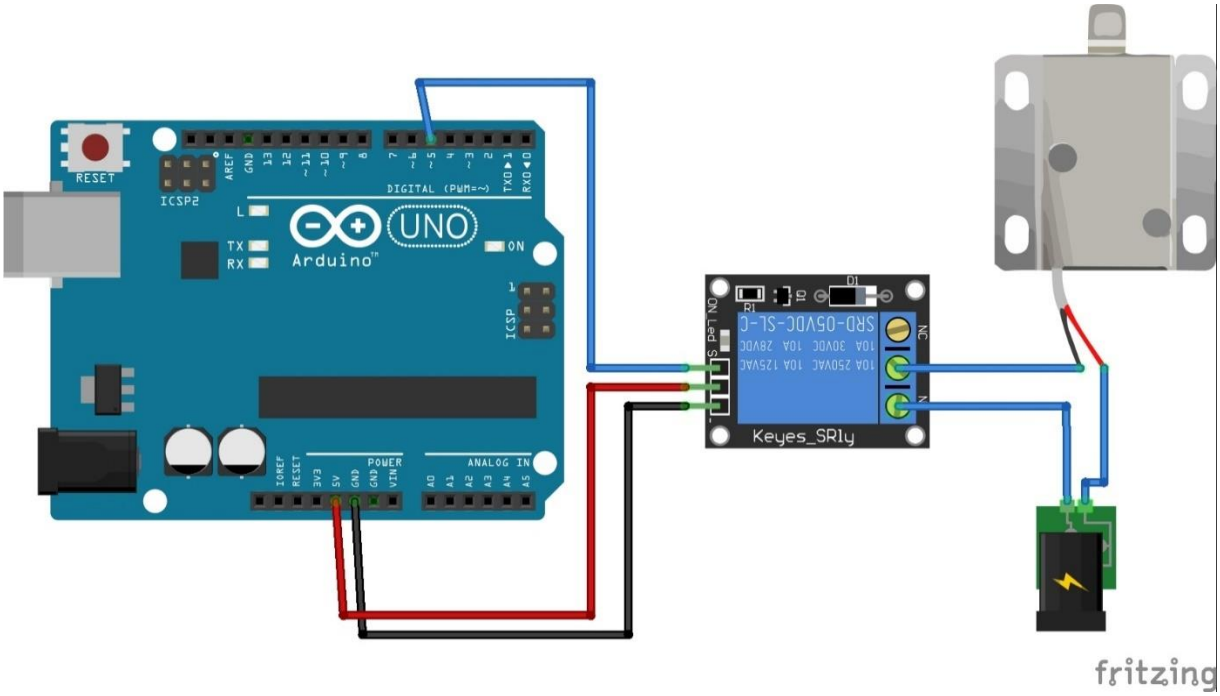

```
1.  #include<Wire.h>
2.  #include<LiquidCrystal_I2C.h>
3.
4.  LiquidCrystal_I2C lcd(0x27, 16, 2);
5.
6.  void setup() {
7.    lcd.begin();
8.
9.    lcd.print("It's Workin");
10.   lcd.setCursor(3, 1);
11.   lcd.print("Khalil Warwar");
12. }
13.
14. void loop() {
15.
16. }
```

המפסק המגנטי :



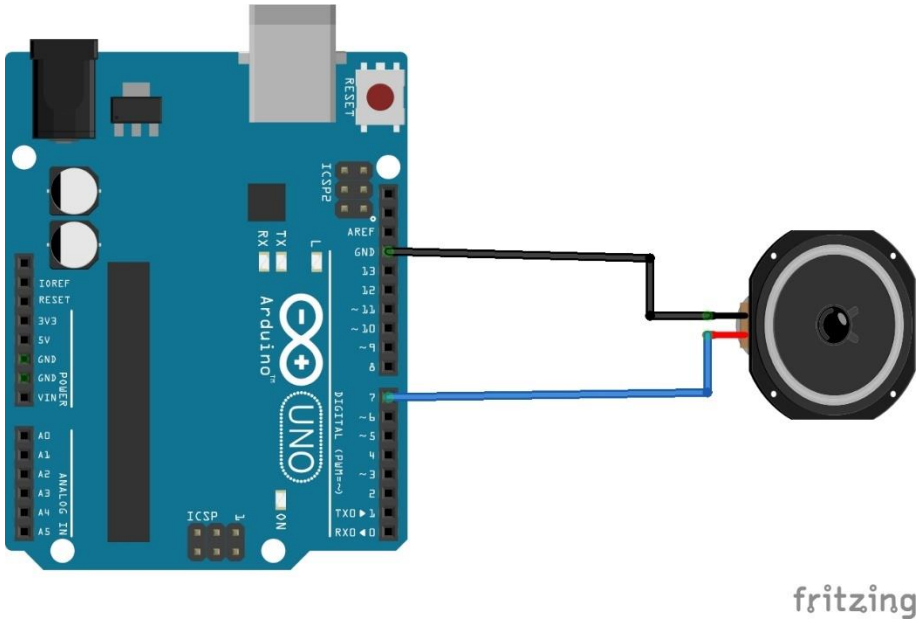

```
1.  #define mg 7
2.
3.  void setup() {
4.    pinMode(mg, INPUT_PULLUP);
5.    Serial.begin(9600);
6.  }
7.
8.  void loop() {
9.    Serial.println(digitalRead(mg));
10. }
```

המִמסָר והמנעול המגנטי :




```
1.  #define relay 5
2.
3.  void setup() {
4.    pinMode(relay, OUTPUT);
5.  }
6.
7.  void loop() {
8.    digitalWrite(relay, LOW);
9.    delay(1000);
10.   digitalWrite(relay, HIGH);
11.   delay(1000);
12. }
```

המִמסָר והמנעול המגנטי :




```
void setup() {
  pinMode(7, OUTPUT);

}

void loop() {
  int i = 200;
  while(i < 800) {
    i++;
    tone(7, i);
    delay(5);
  }
  delay(100);
}
```

תוכנה סופית:

```
#include <SoftwareSerial.h>
#include <Adafruit_Fingerprint.h>
#include <LiquidCrystal_I2C.h>

#define fp_tx 2
#define fp_rx 3

#define ok_bu A2
#define nxt_bu A1
#define prv_bu A0

#define in_bu A3

#define relay 5

#define speaker 7

#define mg_sw 8

SoftwareSerial ser(fp_rx, fp_tx);
Adafruit_Fingerprint finger = Adafruit_Fingerprint(&ser);

LiquidCrystal_I2C my_lcd(0x27, 16, 2);

byte arrow[] = {
  B00000,
  B00000,
  B10001,
  B10001,
  B01010,
  B01010,
  B00100,
  B00000
};

const int e5 = 659, d5s = 622, b4 = 493;
const int d5 = 587, c5 = 523, a4 = 440;
const int c4 = 261, e4 = 329, g4s = 415;

unsigned long timer;
int check;
int ft;

void lcd(String l1 = "", String l2 = "") {
  my_lcd.clear();
  my_lcd.setCursor(0, 0);
  my_lcd.print(l1);
  my_lcd.setCursor(0, 1);
  my_lcd.print(l2);
}

void furElise() {
  ft = 300;
  bool s = false;
  for (int ii = 0 ; ii < 2; ii++) {
    for (int i = 0; i < 2; i++) {
      note(e5);
      note(d5s);
    }
    note(e5);
    note(b4);
```

```
    note(d5);
    note(c5);
    note(a4);
    note(c4);
    note(e4);
    note(a4);
    note(b4);
    if (!s) {
      s = !s;
      note(e4);
      note(g4s);
      note(b4);
      note(c5);
    } else {
      s = !s;
      note(e4);
      note(c5);
      note(b4);
      note(a4);
    }
    note(e4);
  }
}

void note(int n) {
  tone(speaker, n);
  delay(ft);
  if (digitalRead(mg_sw) || digitalRead(ok_bu) == LOW) {
    ft = 0;
  }
  if (!digitalRead(in_bu)) {
    ft = 0;
    unlock();
  }
}

void lcdHome() {
  lcd("Ring The Bell");
  my_lcd.write(0);
}

void unlock() {
  timer = millis() + 5000;
  digitalWrite(relay, LOW);
  tone(speaker, 50);
  delay(2000);
  lcd("Lock The Door");
  my_lcd.write(0);
  while (timer > millis() && digitalRead(prv_bu) && !digitalRead(mg_sw)) {
    yield();
  }
  if (digitalRead(mg_sw)) {
    lcd("Don't Forget To", "Close The Door!");
    delay(500);
  }
  lock();
  while (digitalRead(mg_sw)) {
    yield();
  }
  lcdHome();
  delay(500);
}

void lock() {
  digitalWrite(relay, HIGH);
```

```
  noTone(speaker);
}

int getEmptyID() {
  finger.getTemplateCount();
  if (finger.templateCount < 127) {
    for (int i = 1; i <= 127; i++) {
      if (finger.loadModel(i) != FINGERPRINT_OK) {
        return i;
        break;
      }
    }
  } else return -1;
}

bool getImg2Tz(int times = 1) {
  check = finger.getImage();
  if (check != FINGERPRINT_NOFINGER) {
    if (check == FINGERPRINT_OK) {
      check = finger.image2Tz(times);
    }
    else check = -1;
    lcd("Remove Your", "Finger");
    while (finger.getImage() != FINGERPRINT_NOFINGER)yield();
    return true;
  } else {
    return false;
  }
}

void enroll(bool noCancel = false) {
  int id = getEmptyID();
  if (id == -1) {
    lcd("Storage Full", "Cant Store More");
    delay(3000);
  } else {
    check = -1;
    while (check != FINGERPRINT_OK) {
      for (int i = 1; i <= 2; i++) {
        check = -1;
        if (!noCancel) {
          lcd("Put Your Finger", "         Cancel");
        } else {
          lcd("Enrolling", "Put Your Finger");
        }
        while (check != FINGERPRINT_OK) {
          getImg2Tz(i);
          if (check == -1) {
            if (!noCancel) {
              lcd("Put Your Finger", "         Cancel");
            } else {
              lcd("Enrolling", "Put Your Finger");
            }
          }
          if (digitalRead(ok_bu) == LOW && !noCancel) {
            goto e;
          }
        }
      }
      check = finger.createModel();
      if (check == FINGERPRINT_OK) {
        check = finger.storeModel(id);
      }
    }
    lcd("Finger Stored", "Your Id Is " + String(id));
```

```
    delay(3000);
  }
e:;
  if (!noCancel) {
    lcd("Enroll", "  >          OK");
  }
}

void setup() {
  pinMode(relay, OUTPUT);
  digitalWrite(relay, HIGH);

  pinMode(speaker, OUTPUT);

  pinMode(ok_bu, INPUT_PULLUP);
  pinMode(nxt_bu, INPUT_PULLUP);
  pinMode(prv_bu, INPUT_PULLUP);
  pinMode(mg_sw, INPUT_PULLUP);
  pinMode(in_bu, INPUT_PULLUP);

  my_lcd.begin();
  finger.begin(57600);

  my_lcd.createChar(0, arrow);

  if (!finger.verifyPassword()) {
    lcd("Sensor Error:", "Fingerprint");
    while (!finger.verifyPassword()) {
      yield();
    }
  }
  finger.getTemplateCount();
  if (finger.templateCount < 1) {
    digitalWrite(relay, LOW);
    enroll(true);
    digitalWrite(relay, HIGH);
  }
  lcdHome();
}

bool checkFinger() {
  if (getImg2Tz()) {
    if (finger.fingerFastSearch() == FINGERPRINT_OK) {
      lcd("Door Unlocked", "ID = " + String(finger.fingerID));
      return true;
    } else {
      lcd("Unknown Finger");
      delay(2000);
      lcdHome();
      return false;
    }
  } else {
    return false;
  }
}

void refMenu(int p) {
  switch (p) {
    case 1:
      lcd("Enroll", "  >          OK");
      break;
    case 2:
      lcd("Remove", "< >          OK");
      break;
    case 3:
```

```
      lcd("Factory Reset", "< >          OK");
      break;
    case 4:
      lcd("Exit Menu", "<            OK");
  }
}

void deleteTemplate(int id, bool withConf = true) {
  check = finger.deleteModel(id);
  if (withConf) {
    if (check == FINGERPRINT_OK) {
      lcd("Deleted", "Successfully");
    } else {
      lcd("Delete Failed", "Try Again");
    }
    delay(2000);
  }
}

void remo() {
  finger.getTemplateCount();
  int len = finger.templateCount;
  if (len == 1) {
    lcd("There Is One Id", "Cant Delete It");
    delay(2000);
    lcd("Remove", "< >          OK");
  } else {
    int used[127], n = 0;
    for (int i = 1; i <= 127; i++) {
      if (finger.loadModel(i) == FINGERPRINT_OK) {
        used[n] = i;
        n++;
      }
    }
    n = 0;
    timer = millis() + 6000;
    lcd("Delete #" + String(used[n]), "X >          OK");
    while (timer > millis()) {
      if (digitalRead(prv_bu) == LOW) {
        if (n == 0) {
          timer = 0;
          break;
        } else if (n == 1) {
          n--;
          timer = millis() + 6000;
          lcd("Delete #" + String(used[n]), "X >          OK");
        } else {
          n--;
          timer = millis() + 6000;
          lcd("Delete #" + String(used[n]), "< >          OK");
        }
        delay(400);
      } else if (digitalRead(nxt_bu) == LOW) {
        if (!(n == len - 1)) {
          if (n == len - 2) {
            n++;
            lcd("Delete #" + String(used[n]), "<            OK");
          } else {
            n++;
            lcd("Delete #" + String(used[n]), "< >          OK");
          }
        }
        timer = millis() + 6000;
        delay(400);
      }
```

```
322.      else if (digitalRead(ok_bu) == LOW) {
323.        timer = millis() + 6000;
324.        deleteTemplate(used[n]);
325.        delay(2000);
326.        remo();
327.      }
328.    }
329.  }
330. }
331.
332. void menu() {
333.   bool mloop = true;
334.   int page = 1;
335.   refMenu(page);
336.   timer = millis() + 6000;
337.   while (mloop) {
338.     if (digitalRead(prv_bu) == LOW && page > 1) {
339.       timer = millis() + 6000;
340.       page--;
341.       refMenu(page);
342.       delay(400);
343.     }
344.     else if (digitalRead(nxt_bu) == LOW && page < 4) {
345.       timer = millis() + 6000;
346.       page++;
347.       refMenu(page);
348.       delay(400);
349.     } else if (digitalRead(ok_bu) == LOW) {
350.       timer = millis() + 6000;
351.       if (page == 1) {
352.         enroll();
353.         timer = millis() + 6000;
354.         delay(400);
355.       } else if (page == 2) {
356.         remo();
357.         finger.getTemplateCount();
358.         if (finger.templateCount < 1) {
359.           digitalWrite(relay, LOW);
360.           enroll(true);
361.           digitalWrite(relay, HIGH);
362.         } else {
363.           timer = millis() + 6000;
364.         }
365.         delay(400);
366.       } else if (page == 3) {
367.         delay(400);
368.         lcd("Are You Sure?", "Cancel      Yes");
369.         timer = millis() + 5000;
370.         while (timer > millis()) {
371.           if (digitalRead(ok_bu) == LOW) {
372.             for (int i = 1; i < 128; i++) {
373.               deleteTemplate(i, false);
374.             }
375.             digitalWrite(relay, LOW);
376.             enroll(true);
377.             digitalWrite(relay, HIGH);
378.           } else if (digitalRead(prv_bu) == LOW) {
379.             break;
380.           }
381.         }
382.       } else if (page == 4) {
383.         mloop = false;
384.       }
385.     }
386.     if (millis() > timer) {
```

```
      mloop = false;
    }
    yield();
  }
}

void loop() {
  if (!digitalRead(in_bu)) {
    unlock();
  }
  if (digitalRead(mg_sw)) {
    bool st = false;
    while (!st) {
      int i = 200;
      while (i < 800 && !st) {
        i++;
        tone(7, i);
        delay(5);
        if (checkFinger())st = true;
      }
      delay(100);
    }
    lcdHome();
    noTone(7);
  }

  if (checkFinger()) {
    unlock();
  }

  if (digitalRead(prv_bu) == LOW) {
    lcd("Ringing", "          STOP");
    my_lcd.write(0);
    furElise();
    noTone(speaker);
    lcdHome();
  }

  if (digitalRead(ok_bu) == LOW) {
    delay(500);
    if (digitalRead(ok_bu) == LOW) {
      timer = millis() + 6000;
      lcd("Ready To Scan", "Settings");
      while (finger.getImage() == FINGERPRINT_NOFINGER) {
        if (!digitalRead(in_bu)) {
          unlock();
          break;
        }
        if (timer < millis()) {
          break;
        } else {
          yield();
        }
      }
      if (checkFinger()) {
        digitalWrite(relay, LOW);
        menu();
        digitalWrite(relay, HIGH);
      }
      lcdHome();
    }
  }
}
```

הסבר פקודות מיוחדות:

millis() :

הפקודה תחזיר כמה זמן עבר מהפעלת הבקר.

פקודות לLCD:

begin() :

משתמשים בפקודה לאתחול המסך

clear() :

משתמשים בפקודה כדי למחוק t, כל מה שכתוב על המסך.

setCursor(arg1,arg2) :

משתמשים בפקודה כדי לשנות את מקום הסמן במסך, והיא מקבלת שני משתנים. הראשון הוא מספר העמודה והשני הוא מספר השורה.

print(arg1) :

משתמשים בפקודה כדי להציג טקסט על המסך, מתקבל משתנה אחד שהוא הטקסט.

createChar(arg1, arg2) :

משתמשים בפקודה כדי ליצר תו או צורה שלא מוגדר בספריה של המסך, מקבל שני משתנים הראשון הוא שם הצורה ובשני הוא מערך שבו הנתונים של התו או צורה.

write(arg1) :

משתמשים בפקודה כדי להדפיס את התו או את התצורה שעשינו בעזרת פקודת createChar, והיא מקבלת משתנה אחד שהוא שם התו שרוצים להדפיס.

פקודות לSpeaker:

tone(arg1, arg2) :

משתמשים בפקודה כדי להפעיל את הרמקול, המשתנה הראשון מקבל את מספר הרגל שמחובר אליה הרמקול והמשתנה השני מקבל באיזה תדירות יעבוד

הרמקול.

noTone(arg1) :

משתמשים בפקודה לכבות את הרמקול, ומתקבל מספר הרגל שמחובר אליו הרמקול.

פקודות לFingerprint:

begin(arg1) :

משתמשים בפקודה לאתחול החיישן והוא מקבל את קצב שידור.

verifyPassword() :

משתמשים בפקודה כדי לאמת את הסיסמה ואם החיישן מגיב והיא מחזירה ערך בוליאני אם הסיסמה נכונה או לא, והיא יכולה לקבל משתנה שבו את הסיסמה שהשתנתה.

getTemplateCount() :

הפקודה מחזירה מספר טביעות האצבע המוגדרות.

getImage() :

הפקודה נותנת לחיישן לצלם את האצבע שעל הזכוכית, ומחזירה אם הצליחה לצלם או לא.

image2Tz(arg1) :

הפקודה נותן לחיישן להמיר את תמונת האצבע לפורמט Tz, והיא מקבלת אות שישמר בו המידע, ומחזירה אם הצליחה להמיר או לא.

createModel() :

הפקודה נותנת לחיישן ליצר דגם של טביעת האצבע, ומחזירה אם הצליחה ליצר הדגם או לא.

storeModel(arg1) :

הפקודה נותנת לחיישן לשמור על הדגם של טביעת האצבע, והיא מקבלת את מספר טביעת האצבע, ומחזירה אם הצליחה לשמור את הדגם או לא.

loadModel(arg1) :

הפקודה נותנת לחיישן להוציא את הדגם של טביעת האצבע לram של החיישן, והיא מקבלת את מספר טביעת האצבע, ומחזירה אם הצליחה להוציא הדגם או לא.

deleteModel(arg1) :

הפקודה נותנת לחיישן למחוק את הדגם של טביעת האצבע, והיא מקבלת את מספר טביעת האצבע שרוצים למחוק, ומחזירה אם הצליחה למחוק את הדגם או לא.

fingerFastSearch() :

הפקודה נותנת לחיישן לבדוק את טביעת האצבע שצולמה מוכרת או לא והיא מחזירה את התוצאה.

שם הפונקציה	תפקיד הפונקציה	מה בפונקציה מקבלת	מה הפונקציה מחזירה	מימוש הפונקציה

lcd	משתמשים בפקודה כדי להציג טקסט על המשך.	מה להדפיס על כל שורה.		Void lcd(String l1, String l2)
				{
				}

furElise	מפעילה מנגינת fur elise.			Void furElise(){ }

note	מפעילה את הרמקול ובודקת אם צריך לכבות את הרמקול	התדירות שיופעל בה הרמקול.		Void note(int n)
				{
				}

lcdHome	מציגה ההדף הראשי על המשך.			Void lcdHome()
				{
				}

unlock	מבטל נעילת הדלת ומהציג הודעה על המשך.			Void unlock()
				{
				}

lock	מנעילה את הדלת.			Void lock()
				{
				}

getEmptyID			המיקום הראשון הלא תפוס.	Int getEmptyID()
				{
				}

getImg2Tz	מפקדת חיישן טביעת האצבע מצלם את האצבע וממיר	מספר הפעמים שצולמה בה האצבע.	אם הפעולה הצליחה.	Bool getImg2Tz(int times = 1)
				{
				}

	את התמונה לקובץ Tz.			

enroll	מגדירה טביעת אצבע חדשה.	אם אפשר לבטל את הגדרת האצבע.		Void enroll(bool noCancel = false)
				{
				}

checkFinger	מצלמת את האצבע שעל החיישן ובודקת אם טביעת האצבע מוכרת.		אם טביעת האצבע מוכרת או לא.	Bool checkFinger()
				{
				}

deleteTemplate	מוחקת טביעת אצבע מוגדרת.	הID של טביעת האצבע שצריך למחוק, ואם צריך להציג הודעת אזהרה או רק למחוק.		Void deleteTemplate
				(int id, bool withConf = true)
				{
				}

refMenu	מציגה את התפריט על המסך.	איזה עמוד מהתפריט להציג.		Void refMenu(int p) { }

remo	מתחילה תפריט מחיקת טביעות אצבע.			Void remo()
				{
				}

menu	מתחילה את התפריט הראשי.			Void menu()
				{
				}

רפלקציה אישית:

בפרויקט האישי שלי דמיינתי מערכת אבטחה לבית שמכילה מנעול לדלת, מנעול אשר ייפתח באמצעות טביעת אצבע של דיירי הבית, זאת בנוסף לאזעקה שתופעל במידה ומישהו זר ינסה להיכנס ולפרוץ את הדלת.

מאוד נהניתי לעשות את עבודתי, ובמיוחד נהנתי מחלק התכנות מאחר וכתיבת תוכנות זה אחד מהתחביבים שלי, ומעבר לזאת, הייתה לי את ההזדמנות ללמוד על רכיבים חדשים ועל פרוטוקולי תקשורת שלא ידעתי כיצד הם עובדים לפני כן.

החיישן שאהבתי ביותר הינו חיישן טביעת האצבע, הוא חיישן מעניין וניתן להשתמש בו בהרבה בפרויקטים שימושיים. הוא קל להשתמש ועובד בצורה מעניינת, ומעבר לכך, היה לי מאוד מהנה להשתמש בכל הרכיבים ביחד כדי להפוך אותם למערכת אחידה - שמצליחה לעבוד בצורה יעילה ומסקרנת.

במהלך כתיבת התוכנה, בעיה שעלתה הייתה האפשרות לגרום ללארדוינו לבצע יותר מפקודה אחת בו זמנית. דוגמא לזאת למשל, הייתה כאשר רציתי לגרום לתוכנה לחכות ללחיצת הכפתור ולקרוא את טביעת האצבע באותו זמן. בזמן הניסיון שלי לפענח זאת, גיליתי שזה דבר שאינו אפשרי בארדוינו, זאת מאחר והוא בנוי על מיקרו בקר (microcontroller). בסוג זה של מיקרו-בקרים, ריבוי גלים (multi-threading) אינו מתאפשר. עם זאת, בסוף הצלחתי לגרום לתוכנה לעבוד כמו שרציתי, על ידי ביצוע חלקים קטנים מהפקודות אחד אחרי השני ותזמון דברים בעזרת פקודת .()millis

מראה מקום:

Fingerprint Sensor:

- https://adafruit.github.io/Adafruit-Fingerprint-Sensor-Library/html/class_adafruit___fingerprint.html
- https://github.com/adafruit/Adafruit-Fingerprint-Sensor-Library

Relay:

- https://electronicshobbyists.com/relay-module-interfacing-with-arduino-arduino-relay-module/

I2C LCD:

- https://www.makerguides.com/character-i2c-lcd-arduino-tutorial/

Speaker:

- https://www.arduino.cc/en/Tutorial/toneMelody

Magnetic Lock: ד

- Datasheet

Magnetic switch:

- https://www.instructables.com/id/How-to-Use-a-Magnetic-Door-Switch-Sensor-With-Ardu/
- https://randomnerdtutorials.com/monitor-your-door-using-magnetic-reed-switch-and-arduino/

Protocol I2C:

- https://en.wikipedia.org/wiki/I%C2%B2C
- https://www.youtube.com/watch?v=6IAkYpmA1DQ

Protocol UART:

- https://en.wikipedia.org/wiki/Universal_asynchronous_receiver-transmitter

נספחים:

חיישן טביעת האצבע:

adafruit learning system

Overview

Secure your project with biometrics - this all-in-one optical fingerprint sensor will make adding fingerprint detection and verification super simple. These modules are typically used in safes - there's a high powered DSP chip that does the image rendering, calculation, feature-finding and searching. Connect to any microcontroller or system with TTL serial, and send packets of data to take photos, detect prints, hash and search. You can also enroll new fingers directly - up to 162 finger prints can be stored in the onboard FLASH memory.

We like this particular sensor because not only is it easy to use, it also comes with fairly straight-forward Windows software that makes testing the module simple - you can even enroll using the software and see an image of the fingerprint on your computer screen. But, of course, we wouldn't leave you a datasheet and a "good luck!" - we wrote a full Arduino library so that you can get running in under 10 minutes. The library can enroll and search so its perfect for any project (https://adafru.it/aRz). We've also written a detailed tutorial on wiring and use (https://adafru.it/clz). This is by far the best fingerprint sensor you can get.

- **Supply voltage:** 3.6 - 6.0VDC
- **Operating current:** 120mA max
- **Peak current:** 150mA max
- **Fingerprint imaging time:** <1.0 seconds
- **Window area:** 14mm x 18mm
- **Signature file:** 256 bytes
- **Template file:** 512 bytes
- **Storage capacity:** 162 templates
- **Safety ratings** (1-5 low to high safety)
- **False Acceptance Rate:** <0.001% (Security level 3)
- **False Reject Rate:** <1.0% (Security level 3)
- **Interface:** TTL Serial
- **Baud rate:** 9600, 19200, 28800, 38400, 57600 (default is 57600)
- **Working temperature rating:** -20C to +50C

© Adafruit Industries https://learn.adafruit.com/adafruit-optical-fingerprint-sensor Page 3 of 25

- **Working humidy:** 40%-85% RH
- **Full Dimensions:** 56 x 20 x 21.5mm
- **Exposed Dimensions** (when placed in box): 21mm x 21mm x 21mm triangular
- **Weight:** 20 grams

מסך:

Datasheet

I2C 1602 Serial LCD Module

Product features:

The I2C 1602 LCD module is a 2 line by 16 character display interfaced to an I2C daughter board. The I2C interface only requires 2 data connections, +5 VDC and GND to operate

For in depth information on I2C interface and history, visit: http://www.wikipedia/wiki/i2c

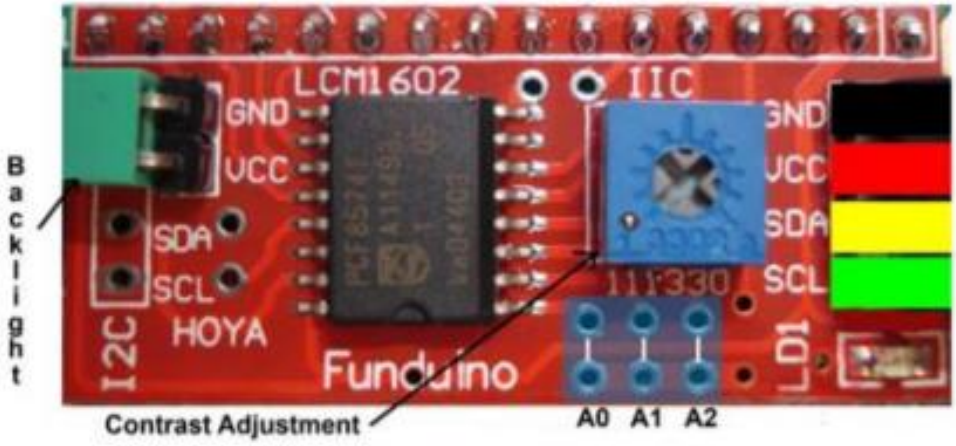
Specifications:

	2 lines by 16 character
I2C Address Range	0x20 to 0x27 (Default=0x27, addressable)
Operating Voltage	5 Vdc
Backlight	White
Contrast	Adjustable by potentiometer on I2c interface
Size	80mm x 36mm x 20 mm
Viewable area	66mm x 16mm

Power:

The device is powered by a single 5Vdc connection.

Pinout Diagram:



Pin/Control Descriptions:

Pin #	Name	Type	Description
1	GND	Power	Supply & Logic ground
2	VCC	Power	Digital I/O 0 or RX (serial receive)
3	SDA	I/O	Serial Data line
4	SCL	CLK	Serial Clock line
A0	A0	Jumper	Optional address selection A0 - see below
A1	A1	Jumper	Optional address selection A1 - see below
A2	A2	Jumper	Optional address selection A2 - see below
Backlight		Jumper	Jumpered - enable backlight, Open - disable backlight
Contrast		Pot	Adjust for best viewing

Addressing:

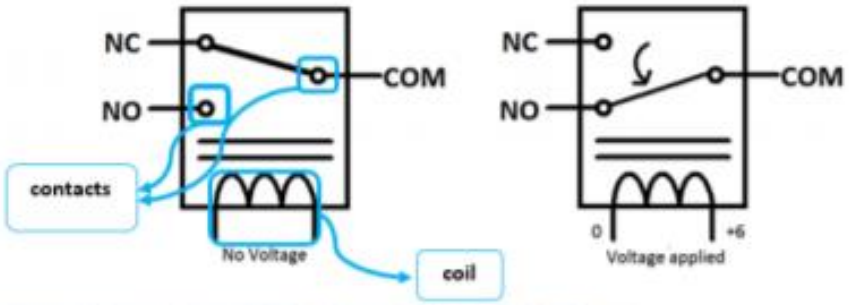
A0	A1	A2	Address
Open	Open	Open	0x27
Jumper	Open	Open	0x26
Open	Jumper	Open	0x25
Jumper	Jumper	Open	0x24
Open	Open	Jumper	0x23
Jumper	Open	Jumper	0x22
Open	Jumper	Jumper	0x21
Jumper	Jumper	Jumper	0x20

ממסר:

RELAY MODULES

RELAY WORKING IDEA

Relays consist of three pins normaly open pin , normaly closed pin, common pin and coil. When coil powerd on magntic field is generated the contacts connected to each other.



Relay modules 1-channel features

- Contact current 10A and 250V AC or 30V DC.
- Each channel has indication LED.
- Coil voltage 12V per channel.
- Kit operating voltage 5-12 V
- Input signal 3-5 V for each channel.
- Three pins for normally open and closed for each channel.

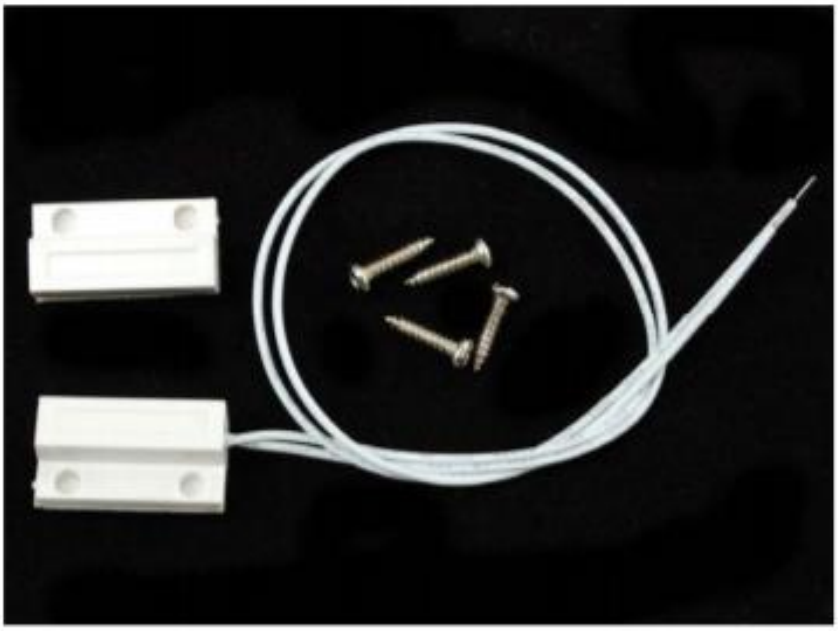
How to connect relay module with Arduino

As shown in relay working idea it depends on magnetic field generated from the coil so there is power isolation between the coil and the switching pins so coils can be easily powered from Arduino by connecting VCC and GND bins from Arduino kit to the relay module kit after that we choose Arduino output pins depending on the number of relays needed in project designed and set these pins to output and make it out high (5 V) to control the coil that allow controlling of switching process.

מפסק מגנטי:

Magnetic contact switch (door sensor)

PRODUCT ID: 375



Description

This sensor is essentially a reed switch, encased in an ABS plastic shell. Normally the reed is 'open' (no connection between the two wires). The other half is a magnet. When the magnet is less than 13mm (0.5") away, the reed switch closes. They're often used to detect when a door or drawer is open, which is why they have mounting tabs and screws. You can also pick up some double-sided foam tape from a hardware store to mount these, that works well without needing screws.

Technical Details

- Normally open reed switch
- ABS enclosure
- Rated current: 100 mA max
- Rated voltage: 200 VDC max
- Distance: 15mm max

Dimensions:

- Box size (each side): 29mm x 14mm x 9mm / 1.1" x 0.6" x 0.4"
- Cable Length: 305mm ± 12mm / 12" ± 0.5"
- Weight (per side): 5.4g

מנעול מגנטי:

Solenoids are basically electromagnets: they are made of a big coil of copper wire with an armature (a slug of metal) in the middle. When the coil is energized, the slug is pulled into the center of the coil. This makes the solenoid able to pull from one end.

This solenoid in particular is nice and strong, and has a slug with a slanted cut and a good mounting bracket. It's basically an electronic lock, designed for a basic cabinet or safe or door. Normally the lock is active so you can't open the door because the solenoid slug is in the way. It does not use any power in this state. When 9-12VDC is applied, the slug pulls in so it doesn't stick out anymore and the door can be opened.

The solenoids come with the slanted slug as shown above, but you can open it with the two Phillips-head screws and turn it around so its rotated 90, 180 or 270 degrees so that it matches the door you want to use it with.

TECHNICAL DETAILS

- 12VDC (you can use 9-12 DC volts, but lower voltage results in weaker/slower operation)
- Draws 650mA at 12V, 500 mA at 9V when activated
- Designed for 1-10 seconds long activation time
- Max Dimensions: 41.85mm / 1.64" x 53.57mm / 2.1" x 27.59mm / 11.08"
- Dimensions: 23.57mm / 0.92" x 67.47mm / 2.65" x 27.59mm / 11.08"
- Wire length: 222.25mm / 8.75"
- Weight: 147.71g

רמקול:

CUI INC

date 10/2010
page 1 of 2

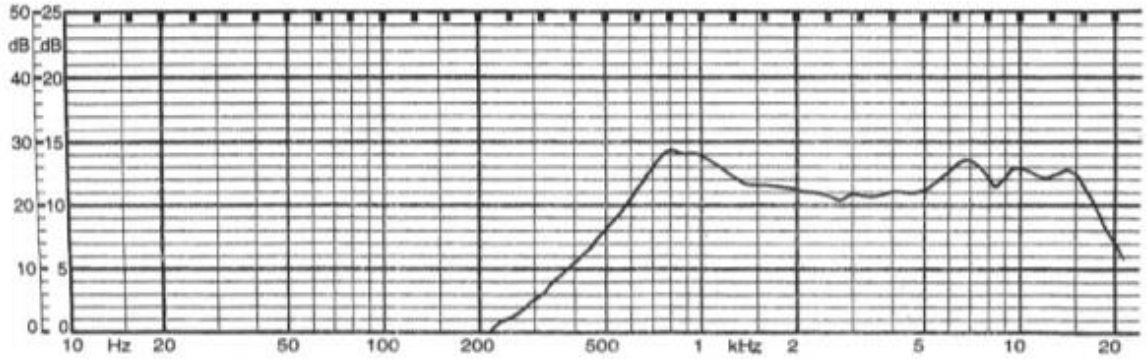
PART NUMBER: CLS0201MA-L152 **DESCRIPTION: SPEAKER**

SPECIFICATIONS

parameter	conditions/description	min	nom	max	units
nominal size	20 mm				
impedance	at 1 kHz, 1 V	6.8	8	9.2	Ω
resonant frequency	at 1 V	600	750	900	Hz
sound pressure level	1 W, 50 cm ave., at 0.8, 1, 1.2, 1.5 kHz	83	86	89	dB
response	10 dB max.	Fo		20,000	Hz
input power			0.5	1	W
operation	must be normal at program source		0.5		W
buzz, rattle, etc.	must be normal at sine wave		2		V dc
magnet	size: 8 x 1 mm				
load test	24 hours of white noise at		0.5		W
heat test	20 – 50% RH for 24 hours	58	60	62	°C
humidity test	90 – 95% RH for 24 hours	38	40	42	°C
RoHS	yes				

FREQUENCY RESPONSE CURVE

parameter	conditions/description
potentiometer range	50 dB
rectifier	RMS
lower limit frequency	20 Hz
wr. speed	100 mm/sec
zero level	60 dB



20050 SW 112th Ave. Tualatin, Oregon 97062 phone 503.612.2300 fax 503.612.2383 www.cui.com

README file:

Home Security System

A DIY home security system that consists of a fingerprint door lock, an alarm, and a doorbell built with Arduino NANO microController and other Arduino sensors & electronic modules.

First-time setup

When turning on the system for the first time a message will show telling that you need to enroll a fingerprint, you simply need to put your finger on the sensor and follow the instructions show on the screen.

After you enroll your first fingerprint you are good to go and the system will start working as intended to.

Adding and Removing Fingerprints and Factory Resting

In order to add a new fingerprint or remove an existing one, you need to enter the settings menu by clicking and holding the "ok" button for several seconds until the word "settings" show on the second line of the LCD, when it shows on the LCD you need to scan a registered fingerprint.

After the fingerprint authentication, a menu will show on the screen that you can navigate throw with the buttons and will let you do the given up actions.

License

The system was created by khalil warwar as a bagroot project in 2020

khalilwarwar2@gmail.com