**BANGLADESH UNIVERSITY OF ENGINEERING AND TECHNOLOGY**

**Department of Electrical and Electronic Engineering**

| | | | |
|---|---|---|---|
| **Course No.** | **:** EEE 310 | **Group No** | **:** 03 |
| **Course Title** | **:** Communication Laboratory | **Section** | **:** C |

**Project Report on,**

# Intelligent Gesture Based Security System

**Level :** 3 **Term :** 1

| **Name** | **Student ID** |
|---|---|
| Fariha Ferdousi | 1706143 |
| Nausin Tabassum Kudrot | 1706158 |
| Sadia Afrose | 1706161 |
| Raisa Mashtura | 1706163 |
| Subah Karnine | 1706174 |

## Contents

## Introduction:

The main objective of this project is to design a security system that can be unlocked simply by means of a stored hand gesture pattern. The user of the security system can save a gesture pattern. Afterwards, if the user is able to recreate the gesture, the systems recognizes it and unlocks. A security system like this can be used in homes, offices, hospitals etc.

The reasons for coming up with this project is:

1) **Safe alternative to retina scan or finger print based security system**
   During a pandemic situation like in the current statusquo, minimizing human contact with machines is a goal
2) **Easy implementation in real life**
   Much less advanced technology needed compared to other security systems. Its also cost effective.
3) **User friendly Security system**
   User can set and reset password whenever needed.
4) **Convenient for Disabled people**
   With hand gesture recognition, interaction with computer becomes easier.

## Brief Overview

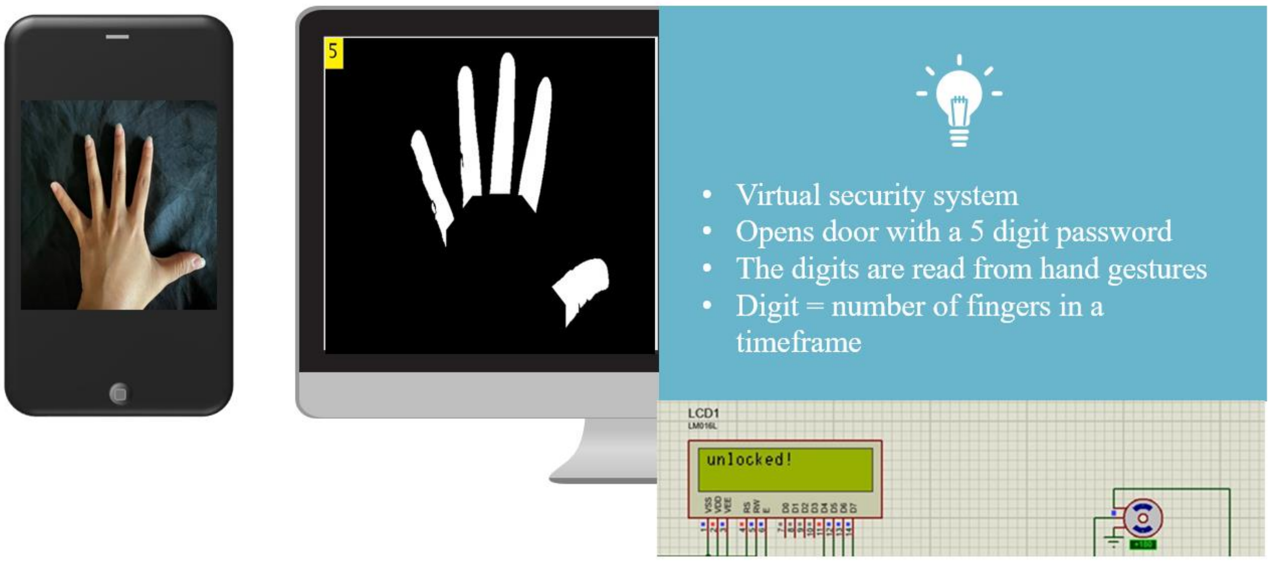
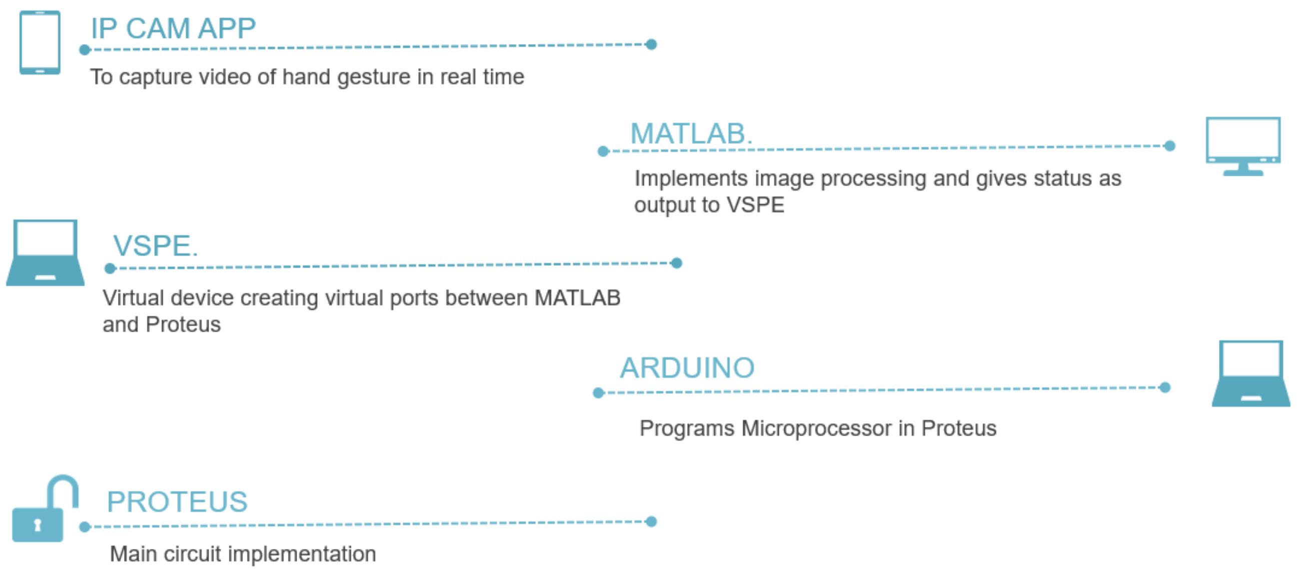

5
• Virtual security system
• Opens door with a 5 digit password
• The digits are read from hand gestures
• Digit = number of fingers in a timeframe
LCD1
LM016L
unlocked!

Fig1. The images show a snapshot of our hand taken from a mobile phone camera and the image processing output.

The picture shows a mobile phone acting as an IP camera. It also shows the output of an image processing

MATLAB program showing the number of fingers it has counted. On the bottom right we see that when the password matches with the saved password, the door gets unlocked with the help of a servo motor.

## **Software used:**



IP CAM APP
To capture video of hand gesture in real time

MATLAB.
Implements image processing and gives status as output to VSPE

VSPE.
Virtual device creating virtual ports between MATLAB and Proteus

ARDUINO
Programs Microprocessor in Proteus

PROTEUS
Main circuit implementation

## **Hand Gesture Recognition:**

This is an example of HGR used for device control. We use the following Matlab package which enables us to play the IP camera video when the IP address is provided. This enables us to counts number of fingers in real time.

We also downloaded an app in our mobile phones to act as an IP camera.

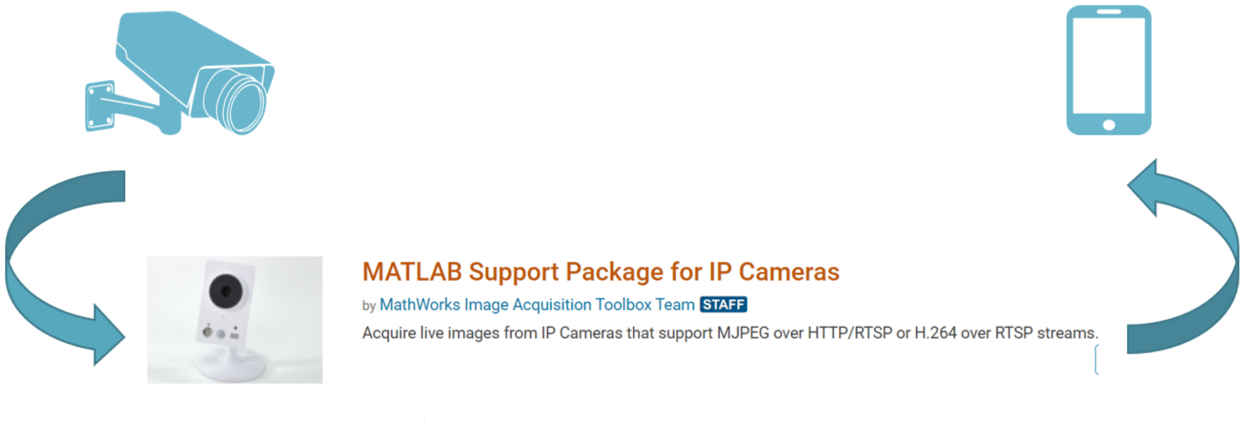

MATLAB Support Package for IP Cameras
by MathWorks Image Acquisition Toolbox Team STAFF
Acquire live images from IP Cameras that support MJPEG over HTTP/RTSP or H.264 over RTSP streams.

Fig2. Package used

## Flowchart of the process:

We have followed a number of steps to ensure maximum accuracy and minimum runtime.

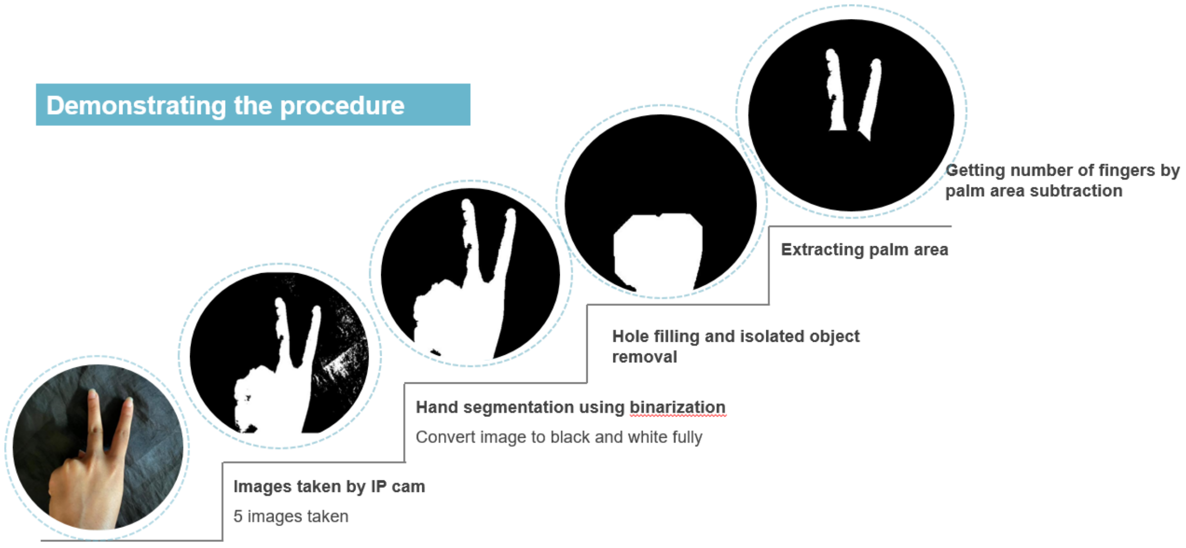
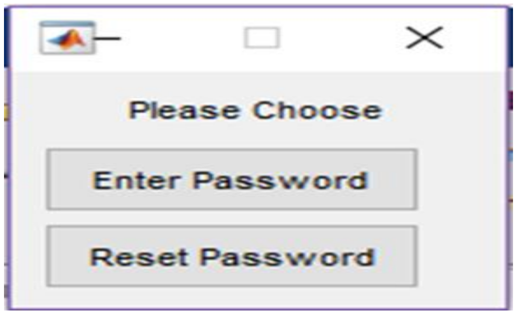

Demonstrating the procedure
Images taken by IP cam
5 images taken
Hand segmentation using binarization
Convert image to black and white fully
Hole filling and isolated object removal
Extracting palm area
Getting number of fingers by palm area subtraction

Fig3. demonstration of the steps

These steps were carried out to ensure maximum accuracy while counting the number of fingers in an instance by image processing.

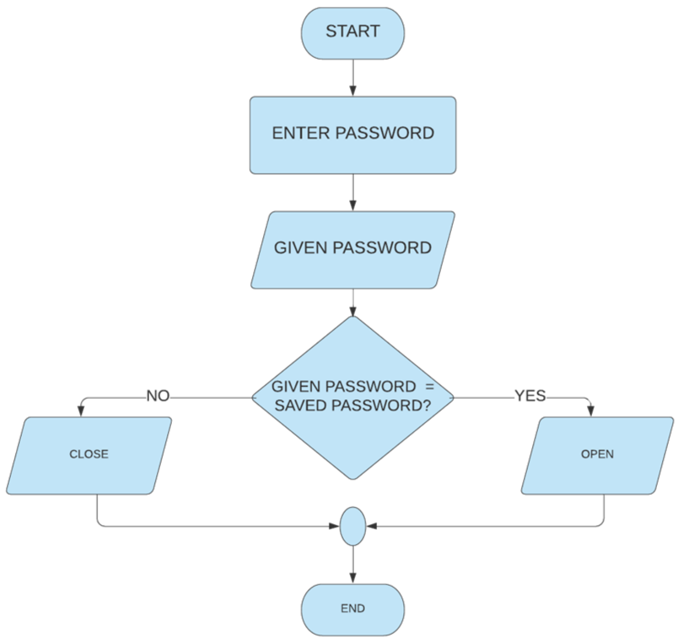
- At first the image is binarized into a b/w image. This is done by turning the background black and the object of interest white. If the background is white or lighter than the image, a simple complement function can modify the code.
- The binarized image has some holes and small isolated areas. Keeping these would cause disturbances in reading output. So at first holes in the image are filled and then small object in the range selected are removed.
- After extracting the clean binary image the next step is to remove the palm area. The image is eroded around a disk fixed on the palm. This extracts the outline for the palm which is then dilated.
- The dilated palm area is subtracted from the binarized image to leaving only the fingers with some isolated areas. Those small isolated areas are again cleaned.
- The output image of the image processing is a clean binary image showing only the fingers separated. These separated regions are counted and returned as the total number of fingers shown.

## Running Matlab:



A small pop up window welcomes the user and asks for the input. The input is taken in five images snapped in an interval of 6 seconds. The output of that image is a 5 element array used as the password.

A flowchart showing the operations of the two settings are shown below.



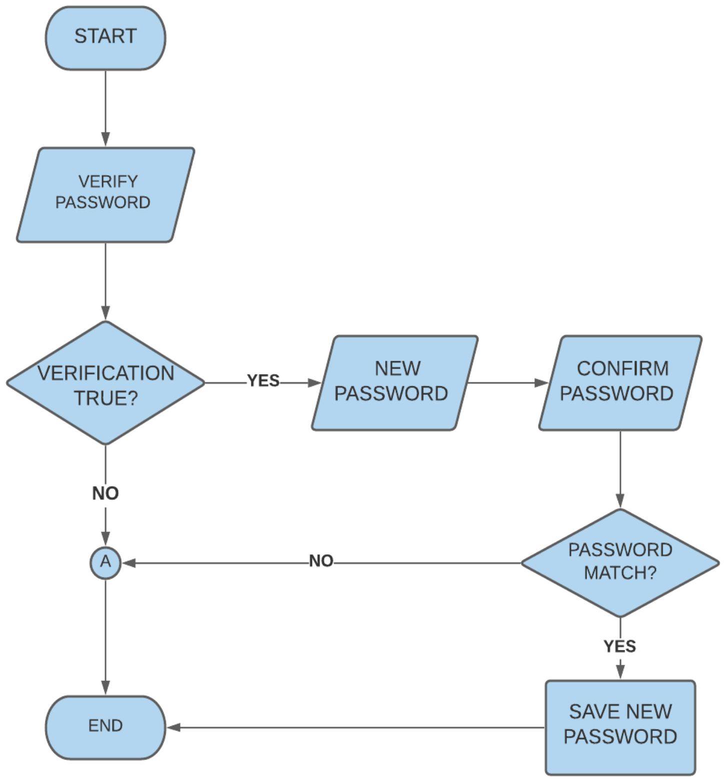


Fig4: flowchart of the input arguments

## Limitations of Image Processing:

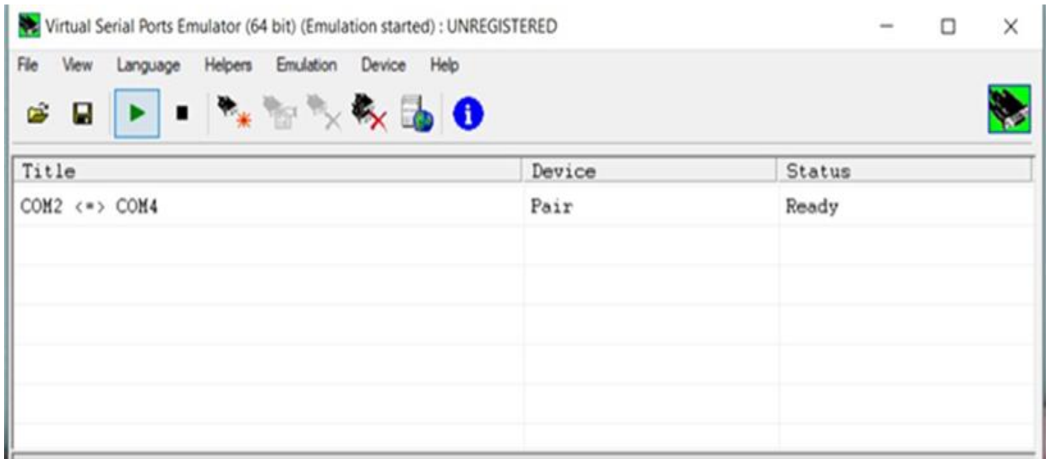
Some limitations are faced when implementing image processing. Like:

- Disturbances due to background object.
- Incorrect binarization due to shadow or bright splashes of light.
- Unable to read image taken at an angle.
- Doesn't take into account how the fingers are being shown, only the number of it.
- Doesn't recognize other arbitrary gesture.

The parameters for the processing was done to ensure maximum accuracy and minimize these limitations.

## Virtual Serial Port Emulator (VSPE):

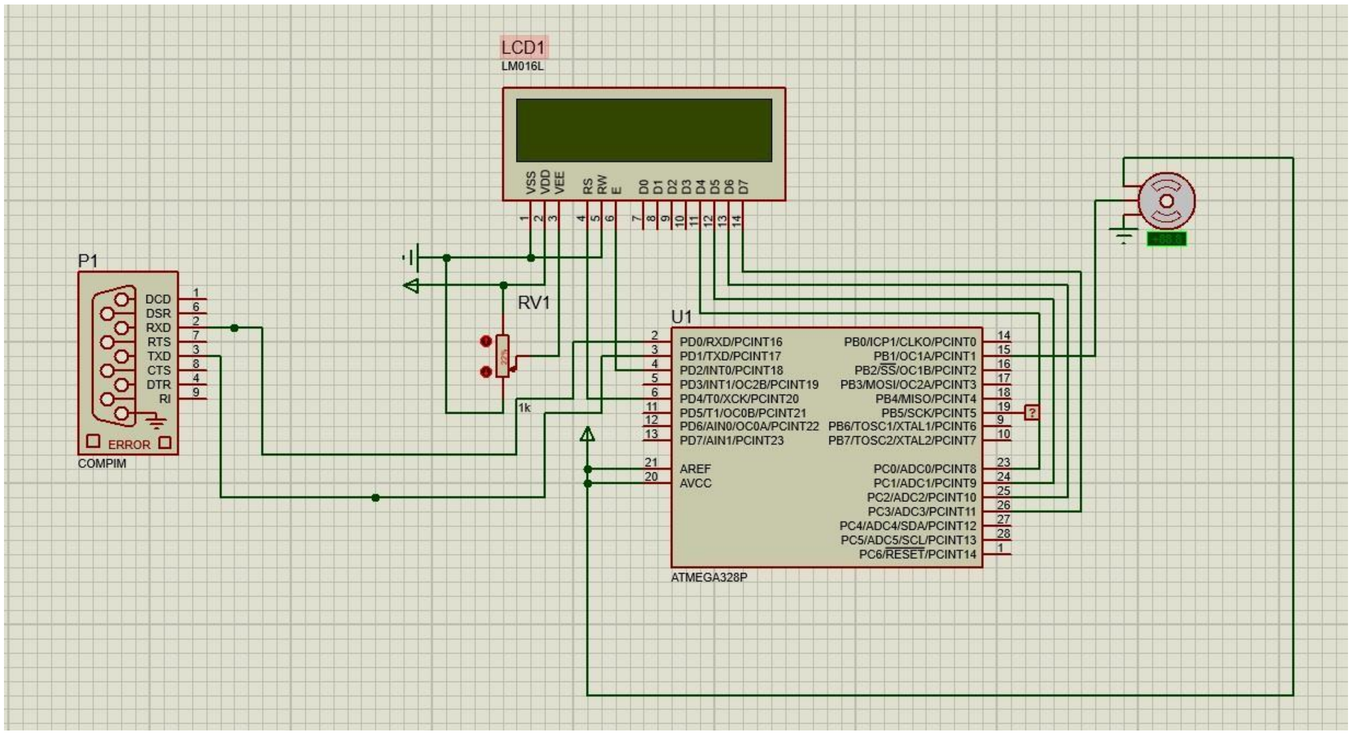
VSPE is a software which is able to create various virtual devices to transmit/receive data. After setting up the MATLAB code and the Arduino code in the microprocessor of Proteus comes the task of establishing connection between them. If the hardware part of the project was done physically than it would've been a matter of plugging the USB ports. But since it is done completely through softwares the connection had to be established through such. That is when VSPE comes. VSPE is used to create ports for MATLAB and Proteus. From the pairing option we select COM2 for the MATLAB and COM4 for the COMPIM component of our circuit in Proteus. After successfully connecting the two softwares, we can now send the data from MATLAB to Proteus.


Virtual Serial Ports Emulator (64 bit) (Emulation started) : UNREGISTERED
File View Language Helpers Emulation Device Help

| Title | Device | Status |
|---|---|---|
| COM2 <=> COM4 | Pair | Ready |

## Schematic Diagram :

The hardware part of this project was simulated using proteus. The schematic diagram is shown below:



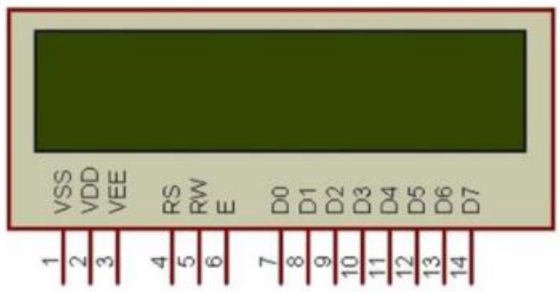
The list of components used in this project is given below:

- Microprocessor, ATEMEGA328P : The microprocessor is programmed in a way so that it can rotate the servo motor and display message on the LCD monitor by reading the input from the serial port. It is programmed using Arduino IDE.

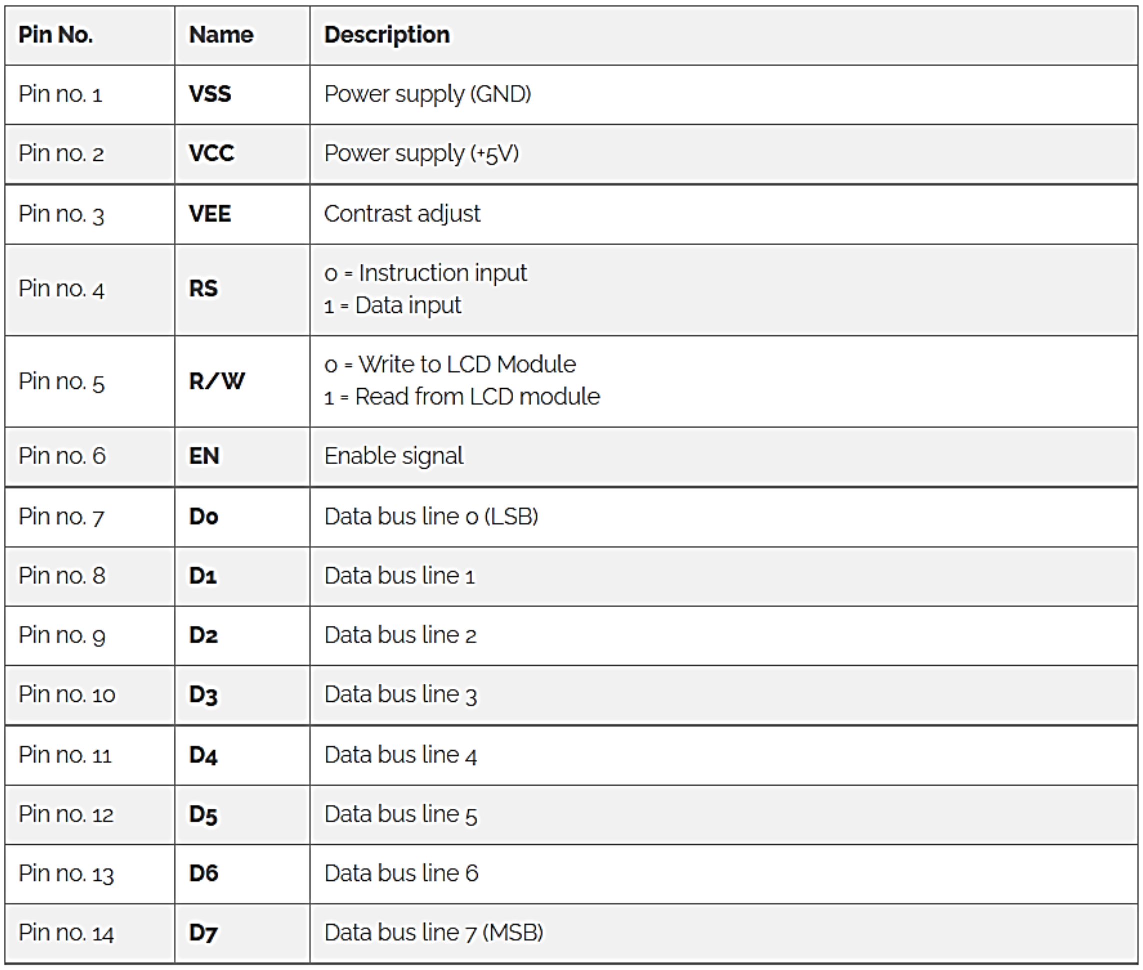
- COMPIM : COMPIM is used to model physical COM interfaces in Proteus.It is used to establish serial connection between MATLAB and Proteus through VSPE.

- LCD Display, LM016L: It is a character based LCD monitor which can display maximum 2 lines including 16 characters per line.

  The schematic model of LM016L is shown below:



The following table describes the functions of different Pins of this model:



| Pin No. | Name | Description |
|---|---|---|
| Pin no. 1 | **VSS** | Power supply (GND) |
| Pin no. 2 | **VCC** | Power supply (+5V) |
| Pin no. 3 | **VEE** | Contrast adjust |
| Pin no. 4 | **RS** | 0 = Instruction input<br>1 = Data input |
| Pin no. 5 | **R/W** | 0 = Write to LCD Module<br>1 = Read from LCD module |
| Pin no. 6 | **EN** | Enable signal |
| Pin no. 7 | **D0** | Data bus line 0 (LSB) |
| Pin no. 8 | **D1** | Data bus line 1 |
| Pin no. 9 | **D2** | Data bus line 2 |
| Pin no. 10 | **D3** | Data bus line 3 |
| Pin no. 11 | **D4** | Data bus line 4 |
| Pin no. 12 | **D5** | Data bus line 5 |
| Pin no. 13 | **D6** | Data bus line 6 |
| Pin no. 14 | **D7** | Data bus line 7 (MSB) |

- Logic Probe: A **logic probe** is a test probe used for analyzing and troubleshooting the logical states (boolean 0 or 1) of a digital circuit.
- Motor- PWMSERVO : This motor is used for unlocking the door. For appropriate inputs the motor will rotate 180 degrees from its initial position in order to unlock the door.
- POT- HG: The potentiometer is **used to adjust the bias level of the LCD** i.e. the contrast of the LCD display.

## Circuit connections:

- The serial pins named TXD(pin 3) and RXD(pin 2) pins are connected to the Atmega and used for USB programs and communicating with it.
- Compim's data receiver(pin 2) and transfer pins(pin 3) were connected to ATmega's corresponding pins accordingly. Compim's physical port was set to COM4 and Baud Rate 9600. Compim captures the serial data from MATLAB and transfers it to ATmega's serial port.
- A logic probe is placed to pin 19 to display if the password is correct or incorrect.
- 5V DC Power is supplied to AVCC pin.
- A servo motor is connected to digital pin 15 for opening and closing door.
- For connecting LCD to the board VDD is connected to 5V dc voltage, Vss and RW(read/write) pin to ground as LCD is just writing data. VEE pin connected to potentiometer to adjust the bias level of the lcd. E pin connected to digital pin 4 to enable signal and RS pin connected to pin 6 for receiving input data. D4,D5,D6 and D7 pins were connected to analog pins 23,24,25,26 and 27 accordingly for data bus line.

## Programming the Microprocessor:

The microproceesor takes input fom MATLAB through the COMPIM connected at pin 2 and pin 3 of ATMEGA328P.

The LCD monitor used in this project is programmed to display maximum 2 lines including 16 characters per line.

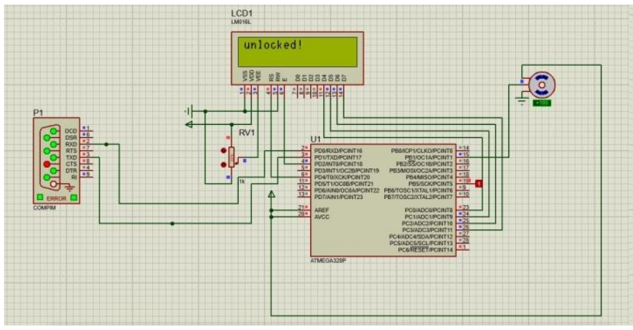
The Arduino code code reads inout from the serial port and performs the following function:

**If input is ‘1’ :**

- The servo motor attached at pin 15 of ATMEGA328P (pin 9 of Arduino) rotates by 180 degrees and thus unlocks the door.
- The LCD monitor displays ‘unlocked!’
- The servo motor remains at this position for 3 seconds and then rotates back to its initial position I.e. 0 degrees. Thus, the door becomes locked again.
- The LCD monitor displays ‘locked!’

**If input is any number other than ‘1’ :**

- The servo motor doesn’t rotate at all and thus the door remains locked.
- The LCD monitor displays ‘Wrong!’ ; ‘Try again!’ .



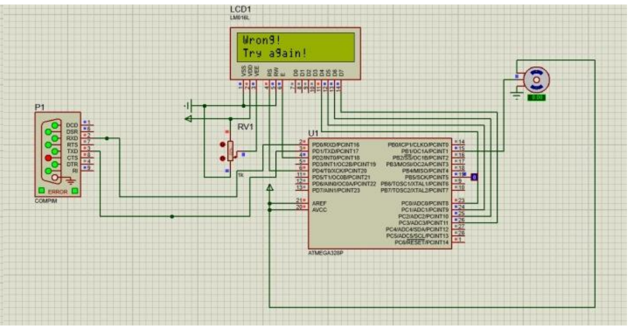
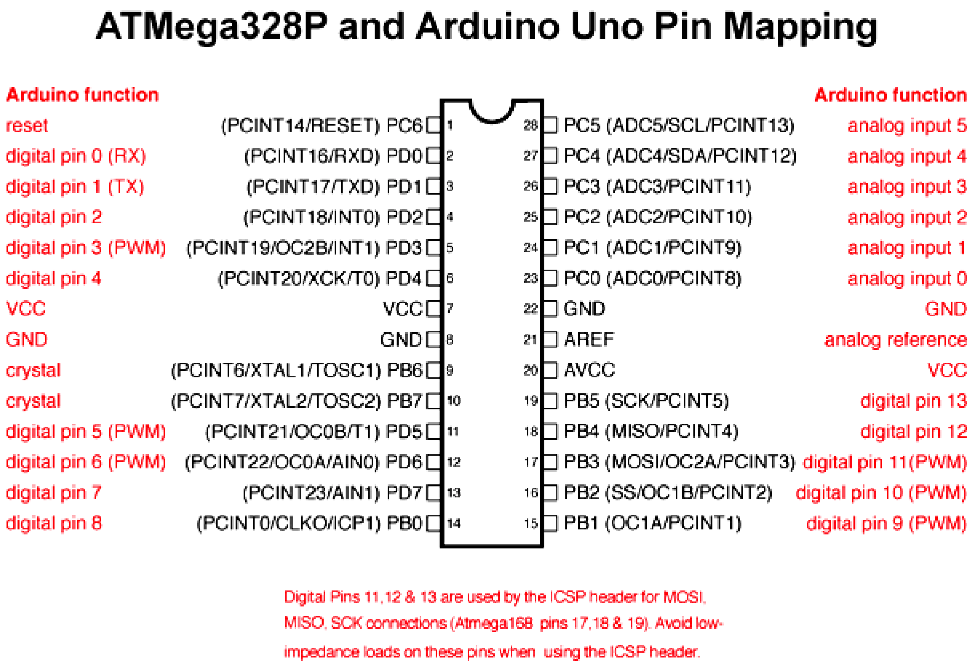


**fig5: If  correct password is entered**

**fig6: If incorrect password is entered**

The input and output pins of Arduino are mapped with that of ATMEGA328P using the diagram below:



ATMega328P and Arduino Uno Pin Mapping

| Arduino function | | Pin | Pin | | Arduino function |
|---|---|---|---|---|---|
| reset | (PCINT14/RESET) PC6 | 1 | 28 | PC5 (ADC5/SCL/PCINT13) | analog input 5 |
| digital pin 0 (RX) | (PCINT16/RXD) PD0 | 2 | 27 | PC4 (ADC4/SDA/PCINT12) | analog input 4 |
| digital pin 1 (TX) | (PCINT17/TXD) PD1 | 3 | 26 | PC3 (ADC3/PCINT11) | analog input 3 |
| digital pin 2 | (PCINT18/INT0) PD2 | 4 | 25 | PC2 (ADC2/PCINT10) | analog input 2 |
| digital pin 3 (PWM) | (PCINT19/OC2B/INT1) PD3 | 5 | 24 | PC1 (ADC1/PCINT9) | analog input 1 |
| digital pin 4 | (PCINT20/XCK/T0) PD4 | 6 | 23 | PC0 (ADC0/PCINT8) | analog input 0 |
| VCC | VCC | 7 | 22 | GND | GND |
| GND | GND | 8 | 21 | AREF | analog reference |
| crystal | (PCINT6/XTAL1/TOSC1) PB6 | 9 | 20 | AVCC | VCC |
| crystal | (PCINT7/XTAL2/TOSC2) PB7 | 10 | 19 | PB5 (SCK/PCINT5) | digital pin 13 |
| digital pin 5 (PWM) | (PCINT21/OC0B/T1) PD5 | 11 | 18 | PB4 (MISO/PCINT4) | digital pin 12 |
| digital pin 6 (PWM) | (PCINT22/OC0A/AIN0) PD6 | 12 | 17 | PB3 (MOSI/OC2A/PCINT3) | digital pin 11(PWM) |
| digital pin 7 | (PCINT23/AIN1) PD7 | 13 | 16 | PB2 (SS/OC1B/PCINT2) | digital pin 10 (PWM) |
| digital pin 8 | (PCINT0/CLKO/ICP1) PB0 | 14 | 15 | PB1 (OC1A/PCINT1) | digital pin 9 (PWM) |

Digital Pins 11,12 & 13 are used by the ICSP header for MOSI, MISO, SCK connections (Atmega168 pins 17,18 & 19). Avoid low-impedance loads on these pins when using the ICSP header.

## **Practical Implementation of this project:**

- A series of photo is taken by the IP camera and passed to a CPU connected to the security system
- The MATLAB code is installed in the CPU
- The MATLAB code analyses the input and passes necessary arguments to the microprocessor connected to the door circuit.
- The microprocessor analyses the MATLAB command and unlocks the door if correct password is provided as input.
-

## **Conclusion:**

The goal of our project was to design a useful and fully functional real-world product that efficiently translates the movement of hand to electrical signals that can control the home appliances. This would also help disabled people to control the electrical appliances more easily.

The gesture control automation system recognizes the hand positions and shows output onto a display and controls the electronic lock devices. The system was trained and tested for multiple users successfully. The proposed system has the advantage of low power consumption, simple hardware and hand gestures. It is also easy to operate and user friendly.

This system can be implemented in human-computer interaction since gestures are desired to play an important role. In future an RFID card can be used to identify the rooms and hence the system can be implemented for multiple rooms. This system can be used for sign language, robotics, games and various other applications with further modification.

# Appendix:

## MATLAB code for connecting VSPE:

```
doi = 1 ;

arduino=serial('COM2','BaudRate',9600); % create serial communication object

fopen(arduino); % initiate arduino communication

fprintf(arduino, '%s', char(doi)); % send answer variable content to arduino

fclose(arduino);
```

## ARDUINO CODE:

```
int solenoidPin = 13; //This is the output pin on the Arduino

int doi;

#include <Servo.h>

#include <LiquidCrystal.h>


// initialize the library by associating any needed LCD interface pin

// with the arduino pin number it is connected to

const int rs = 4, en = 2, d4 = A0, d5 = A1, d6 = A2, d7 = A3;

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


Servo myservo;  // create servo object to control a servo

// twelve servo objects can be created on most boards


int pos = 0;    // variable to store the servo position

void setup()

{

  Serial.begin(9600);

  myservo.attach(9);  // attaches the servo on pin 9 to the servo object

  pinMode(solenoidPin, OUTPUT); //Sets that pin as an output
  // set up the LCD's number of columns and rows:
 lcd.begin(16, 2); //the LCD has 2 lines, and can display 16 characters per line
}
void loop()
{
if(Serial.available()>0)
{
  doi = Serial.read();
  if (doi == 1)
  {
   //Serial.println(1);
   digitalWrite(solenoidPin, HIGH); //Switch Solenoid ON
   //delay(1000); //Wait .15 Second
    //digitalWrite(solenoidPin, LOW);
    for (pos = 0; pos <= 180; pos += 1) { // goes from 0 degrees to 180 degrees
   // in steps of 1 degree
   myservo.write(pos);              // tell servo to go to position in variable 'pos'
   delay(15);                       // waits 15ms for the servo to reach the position

 }
 lcd.print("unlocked!");
 delay(3000);
 for (pos = 180; pos>=0; pos -= 1) { // goes from 0 degrees to 180 degrees
   // in steps of 1 degree
   myservo.write(pos);              // tell servo to go to position in variable 'pos'
   delay(15);                       // waits 15ms for the servo to reach the position

 }
 digitalWrite(solenoidPin, LOW);
 lcd.clear();

 lcd.print("locked!");

 delay(3000);

 lcd.clear();



  }

 else

  {

    lcd.print("Wrong!");

    lcd.setCursor(0, 1);

    lcd.print("Try again!");

    delay(2000);

    lcd.clear();

  }

}

}
```

## MATLAB Code:

```
pass=xlsread('password.xlsx');

choice=menu('Please Choose', 'Enter Password', 'Reset Password');
switch choice
    case 1
        given_password=read_password();
        if given_password==pass
            status=1;
        else
            status=0;
        end
        disp(status);
    case 2
        verify_password=read_password();
        if verify_password==pass
            new_password1=read_password();
            new_password2=read_password();
            if new_password1==new_password2
                pass=new_password1;
                save_password(pass);
            else
                disp('password does not match')
                %break
            end
        else
            disp('password invalid')
        end
end

doi = status ;

arduino=serial('COM2','BaudRate',9600); % create serial communication object

fopen(arduino); % initiate arduino communication

fprintf(arduino, '%s', char(doi)); % send answer variable content to arduino

fclose(arduino);





function set_of_nof=read_password()
cam=ipcam('http://192.168.0.117:8080/video');
preview(cam);
videoframe=snapshot(cam);
frameSize=size(videoframe);
videoplayer=vision.VideoPlayer('Position',[100 100 [frameSize(2),
frameSize(1)]]);
runloop=true;

set_of_nof=zeros(1,5);
index=1;

%for i=1:5
while runloop
    img1=snapshot(cam);
    img1=rgb2gray(img1);
     img1=imresize(img1,[480,640]);

    %img2=imcomplement(imbinarize(img1));
    img2=(imbinarize(img1));
    img3=imfill(img2,'holes'); %filling holes
    img4=bwareaopen(img3,25000); %removing objects less than 10k size

    SE1=strel('disk',50);
    SE2=strel('disk',60);

    img4e=imerode(img4, SE1); %image erosion
    img4d=imdilate(img4e,SE2); %image dilation
    imgfo=img4-img4d; %getting fingers

    imgfo(imgfo==-1)=0;
    imgfo=logical(imgfo);
    imgfo=bwareaopen(imgfo,500); %removing objects lesser than 0.5 size

    CC=bwconncomp(imgfo); %connected component analysis
    nof=CC.NumObjects; %getting no. of fingers

    if index<6
        set_of_nof(index)=nof;
    else
        break;
    end

    index=index+1;

    imgfog=uint8(255.*imgfo);
    nofs=num2str(nof);

    %inserting no. of fingers in image

imgfogrgb=insertText(imgfog,[0,0],nofs,'Fontsize',30,'BoxColor','yellow','Box
Opacity',1,'TextColor','black');

    step(videoplayer,imgfogrgb);

    if index==6
        runloop=false;
    end

    runloop=isOpen(videoplayer);
    pause(6); %pause duration



end
%clear cam;
release(videoplayer);
end





function y=save_password(pass)
%writematrix(pass,'password.xlsx')
writematrix(pass,'password.xlsx')
end
```