



**ISLAMIC UNIVERSITY OF TECHNOLOGY**

# **Design and Analysis of a Wireless Embedded Safe and Secure Home System**

A Thesis Presented to

The Academic Faculty

By

MD.OMAR FAROQUE (102412)

MD.SOLAIMAN (102414)

MD.RAQUIBZZAMAN (102416)

*A thesis submitted in partial fulfillment of the requirement for the degree of  
Bachelor of Science in Electrical & Electronic Engineering.*

Academic Year: 2013-2014

# **Design and Analysis of a Wireless Embedded Safe and Secure Home System**

Approved By:

.....

Prof. Dr. Md. Shahid Ullah

Head of the Department,

Department of Electrical and Electronic Engineering,

Islamic University of Technology.

.....

Dr. Khondokar Habibul Kabir

Assistant professor,

Department of Electrical and Electronic Engineering,

Islamic University of Technology.

# Declaration of Authorship

We, Md.Omar Faroque (102412) Md.Solaiman (102414) and Md. Raquibzzaman (102416), declare that this thesis titled “**Design and Analysis of a Wireless Embedded Safe and Secure Home System**” and the works presented in it are our own. We confirm that

- This work has been done for the partial fulfillment of BSc. in EEE.
- Any part of this thesis has not been submitted anywhere else for obtaining degree.

**Submitted By:**

---

Md. Omar Faroque (102412)

---

Md. Solaiman (102414)

---

Md. Raquibuzzaman(102416)

## **ACKNOWLEDGEMENTS**

We are using this opportunity to express our sincere gratitude to everyone who supported us throughout the BSC project. We would like to thank our thesis advisor Dr. Khondokar Habibul Kabir, for his inspiring guidance, invaluable constructive criticism and friendly advice during the project work. We are sincerely grateful to them for sharing his truthful and illuminating views on a number of issues related to the project.

Additionally, we are thankful to all other respected teachers with whom we were able to collaborate with and whose efforts made our educational and research experiences that much more enjoyable and fruitful.

## **“Design and Analysis of a Wireless Embedded Safe and Secure Home System”**

### **Abstract**

Secured and safe home system is needed for the current condition in Bangladesh. In this paper, we present the design and analysis of an embedded home security system. There are a lot of safe and secured systems are present. Here we want to make a system for the middle class people in Bangladesh. Most of the systems are present for the rich people but here we want to design for the middle class people so that they can afford to apply the system within their financial ability. So our main goal is to design and analysis of a wireless embedded safe and secure home system at a low cost than the system present at high cost for the rich people. The system contains Password protected Door Lock system, motion detection system and fire alarm and leakage gas sensors.

### **Keywords**

**LCD**- Liquid crystal display, **RTC**- Real Time Clock, **PIR**- Passive Infrared  
**LED**- Light Emitting Diode, **LM**-Linear Monolithic, **ppm**-Parts per million

# Contents

List of figures.....	7
List of Flow Charts.....	8
List of tables .....	8
<b>Chapter 1</b> .....	9
Introduction .....	9
<b>Chapter 2</b> .....	12
Design of safe and secure home security system (Hardware).....	12
2.1 Arduino Mega 2560 .....	13
2.2 Micro SD shield: .....	16
2.3 Micro SD card and adapter: .....	17
2.4 RTC module:.....	18
2.5 Servo motor: .....	20
2.6 PIR Sensor: .....	22
2.7 Temperature Sensor (LM-35):.....	23
2.8 Gas sensor (MQ-2): .....	24
2.9 LCD Display:.....	26
2.10 Keypad: .....	27
2.11 Exhaust fan:.....	28
2.12 Buzzer.....	28
2.13 LED .....	29
2.14 Pushbutton.....	29
2.15 Potentiometer:.....	30
2.17 D400 transistor.....	31
2.18 Diagram showing connections of equipment with ATMEGA 2560 pins for designing the hardware of our project.....	32
<b>Chapter 3</b> .....	34
Design of safe and secure home security system (SOFTWARE) .....	34
3.1 PRIMARY DOOR ENTRANCE FLOW CHART:.....	35
3.2 SECONDARY DOOR ENTRANCE FLOW CHART:.....	36
3.3 FIRE DETECTION BY SMOKE SENSOR FLOW CHART:.....	37

3.4 FIRE DETECTION BY HEAT SENSOR FLOW CHART: .....	38
3.5 GAS LEAKAGE DETECTION FLOW CHART: .....	39
3.6 PIR SENSOR DETECTION FLOW CHART: .....	40
<b>Chapter 4</b> .....	<b>41</b>
Analysis .....	41
4.1 Implementation of the equipment in designing our project: .....	41
4.2 Data Analysis .....	55
4.3 Cost analysis .....	58
<b>Chapter 5</b> .....	<b>59</b>
Conclusion & Future work .....	59
5.1 Conclusion .....	59
5.2 Future work .....	60
Appendix .....	61
References .....	86

## List of figures

Figure 1.1: A simple residence in Bangladesh.....	9
Figure 1.2: System overview .....	10
Figure 2.1 : Arduino Mega 2560 .....	13
Figure 2.2: Micro SD shield .....	16
Figure 2.3: Micro SD card.....	17
Figure 2.4: Micro SD adapter .....	18
Figure 2.5: RTC module .....	18
Figure 2.6: Servo motor .....	20
Figure 2.7: Variable Pulse Width Control Servo position.....	21
Figure 2.8: PIR Sensor .....	22
Figure 2.9 LM-35 .....	23
Figure 2.10: MQ-2 Gas sensor (Top view).....	24
Figure 2.11 MQ-2 Gas sensor (Bottom view).....	24
Figure 2.12: LCD Display.....	26
Figure 2.13: Keypad .....	27
Figure 2.14: Keypad internal structure .....	27
Figure 2.15: Exhaust fan.....	28
Figure 2.16: Buzzer.....	28
Figure 2.17: LED .....	29
Figure 2.18: Pushbutton.....	29
Figure 2.19: Potentiometer.....	30
Figure 2.20: D400 transistor .....	31
Figure 4.1: Displayed data at COM port.....	55
Figure 4.2 extracted data from memory card.....	56
Figure 4.3 Simulating results for fire and leakage gas detection.....	57



## List of Flow Charts

Flow Chart 1: PRIMARY DOOR ENTRANCE FLOW CHART .....	35
Flow Chart 2: SECONDARY DOOR ENTRANCE FLOW CHART .....	36
Flow Chart 3: FIRE DETECTION BY SMOKE SENSOR FLOW CHART .....	37
Flow Chart 4: FIRE DETECTION BY HEAT SENSOR FLOW CHART.....	38
Flow Chart 5: GAS LEAKAGE DETECTION FLOW CHART.....	39
Flow Chart 6: PIR SENSOR DETECTION FLOW CHART .....	40

## List of tables

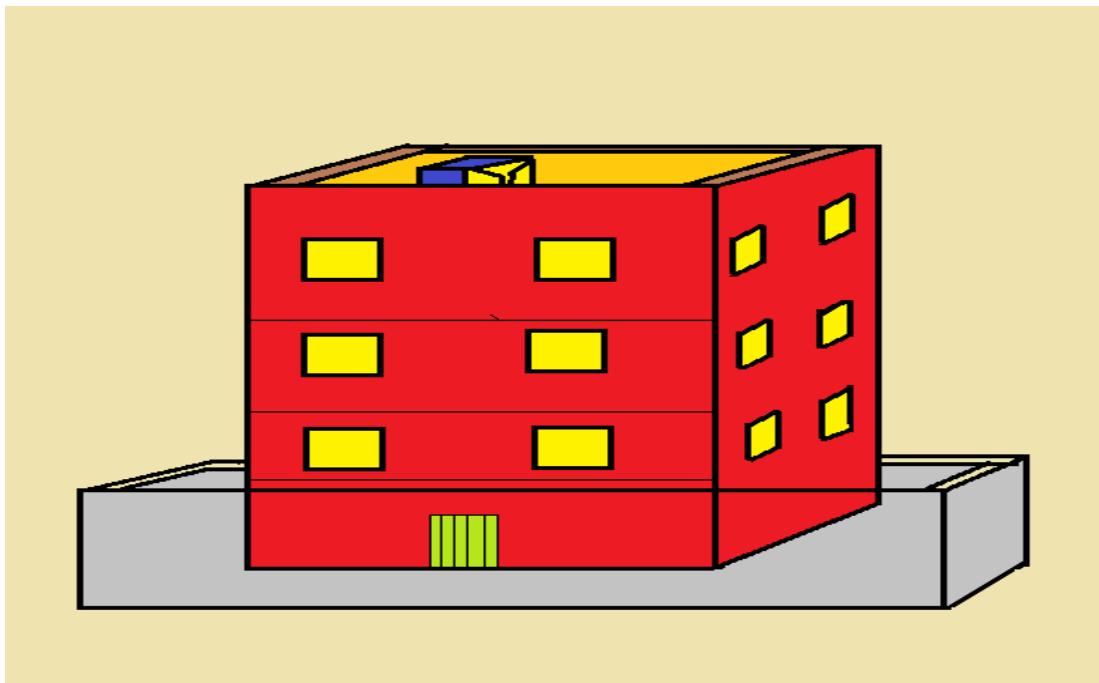
Table 1: Features of ATmega2560 .....	14
Table 2: Features of MQ-2 .....	25
Table 3: PIN configuration of LCD .....	26
Table 4: Cost comparison between ADT and our security system. ....	58

# Chapter 1

## Introduction

We want to design and analysis of a wireless embedded safe and secure home system for Bangladeshi middle class people. We gave emphasize on the cost for the system as the present systems are very much costly for the middle class people and only affordable by the rich people. Our target is to set a home security system in a way such that it will be easily affordable for the middle class people of Bangladesh.

The available security service system providers around the world view of solving the home security problems is slightly different from ours. Most of the cases the surroundings which they are mainly consider is very much different from Bangladesh. Home Security Systems is essential to secure properties .The era in which we are living is more vulnerable to robbery and anti-social actions. Trespassing any residence has become much easier for criminals by using hi-tech gadgets. In a situation like this, it has become very important to set up technically innovative home security systems in our houses.



**Figure 1.1: A simple residence in Bangladesh**

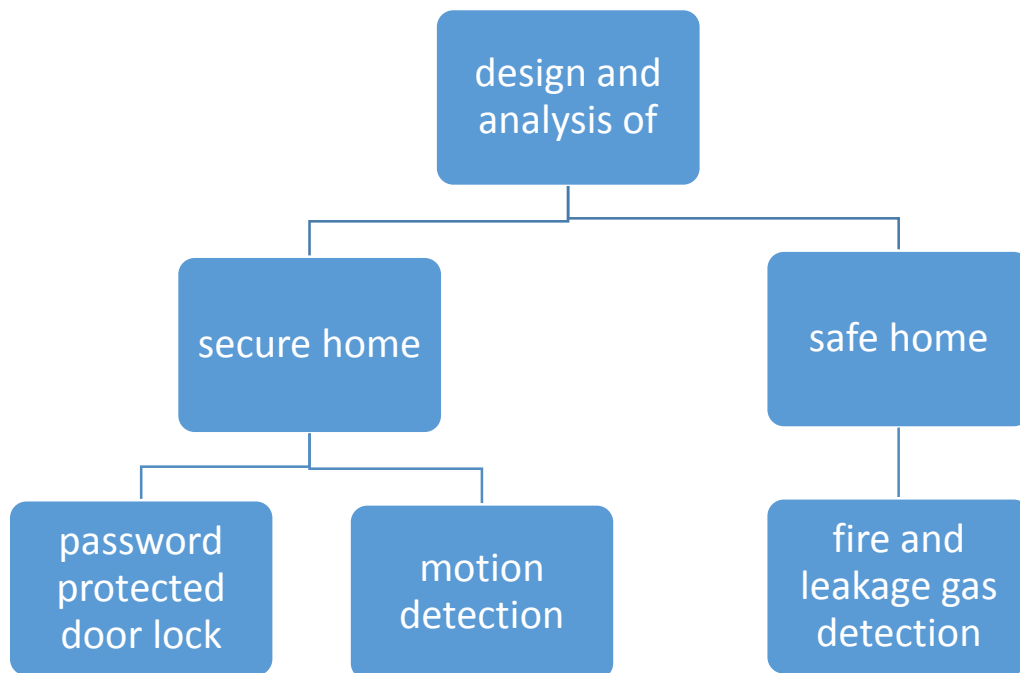
A typical home in Bangladesh is taken as a sample (Figure 1). Considering its structure we analyse the ways of trespassing and tried to make our security system accordingly.

## ➤ Scenario

The total system is divided by two categories

- Safe home
  - Fire detection
  - Leakage gas detection
- Secured home
  - password protected door lock
  - motion detection

So the system looks like:



**Figure 1.2: System overview**

There is one main entrance and there are secondary entrances for each flat. The main entrance is taken care of being installed the password protected door lock. For each flat switching system are attached with the password protected door.

- Each residents will have a password to open the door.
- The module which will be connected to the front door has its switching option as well as it will've a storage device system that will keep the information who entered through the gate last.
- There will be an emergency (forcefully entered) password facility provided with the system that will help the security purpose of the dwellers too.

The motion detection security is also installed with it for detecting the motion of a human body.

It will detect the motion of human and will take a picture with the cc camera attached with the motion detection device. The images will be stored in the storage device.

This two system – the password protected door lock and the motion detection system are the part of security purpose.

For the safety purpose the fire detection and leakage gas detection systems are installed inside the rooms.

- For fire detection heat sensor and smoke sensor have been used
- For Gas leakage detection gas sensor (MQ-2) has been used

The total systems device is implemented. The process of making a device for safe and secured home system will be described in the later chapters.

## Chapter 2

### Design of safe and secure home security system

#### (Hardware)

We want to design an embedded safe and secure home system for Bangladeshi middle class people. To design the required device we needed many kind of hardware. In this chapter we are discussing about the equipment used for making our device. The following equipment were used to design our safe and secure home security system.

- **Microcontroller (Arduino Mega 2560):** Arduino Mega 2560 has 54 digital input/output pins. It has various built in functions. Its programming language is easy to understand.
- **Memory Storage device:** Memory Storage device consists of two parts- Micro SD card and Micro SD shield. We used it to store necessary data and information.
- **RTC module:** We used RTC module module to synchronize our device with real time clocking.
- **Servo motor:** We used particularly servo motor to build the prototype of our door lock system.
- **Motion detection sensor (PIR sensor):** We used PIR sensor since it only detects human infrared rays.
- **Temperature sensor (LM-35):** We used LM-35 because it gives accurate reading of temperature and easy to use.
- **Gas sensor module (MQ-2):** We used MQ-2 gas sensor module because it detects combustible gas and smoke.
- **Transistor (D400):** We used D400 transistor as the switching device to control the current flow thorough the Exhaust fan.
- **Display Device (LCD):** We used LCD as a display device as it is cheap and easily programmable for our device.
- **Keypad:** Keypad is used to give inputs into our device

- **Exhaust fan:** Exhaust fan is used as a prototype device to extract out the leaked gas from inside to outer environment.
- **Buzzer:** Buzzer is used to make a sound as an alarm.
- **LED:** LED is used to represent the change of situation.
- **Pushbutton:** We used Pushbutton to select the mode of operation.
- **Potentiometer:** Potentiometer is used to control the contrast of the LCD display.

We discussed about the equipment shortly above. We will now discuss about the detail information about the equipment with their characteristics, features, pin configuration and connections.

## 2.1 Arduino Mega 2560



Figure 2.1 : Arduino Mega 2560

- **Overview**

The Arduino Mega 2560 is a microcontroller board based on the ATmega2560 [1]. 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 a AC-to-DC adapter or battery to get started. The Mega is compatible with most shields designed for the Arduino Duemilanove or Diecimila.

The Mega 2560 is an update to the Arduino Mega, which it replaces.

The Mega2560 differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the ATmega16U2 (ATmega8U2 in the revision 1 and revision 2 boards) programmed as a USB-to-serial converter. This board has the following new features:

- **Summary:**

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

**Table 1: Features of ATmega2560**

- **Power:**

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

External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can be inserted in the Gnd and Vin pin headers of the POWER connector.

The board can operate on an external supply of 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may be unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts.

- **Communication:**

The Arduino Mega2560 has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmega2560 provides four hardware UARTs for TTL (5V) serial communication. An ATmega16U2(ATmega 8U2 on the revision 1 and revision 2 boards) on the board channels one of these over USB and provides a virtual com port to software on the computer (Windows machines will need a .inf file, but OSX and Linux machines will recognize the board as a COM port automatically). The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the board. The RX and TX LEDs on the board will flash when data is being transmitted via the ATmega8U2/ATmega16U2 chip and USB connection to the computer (but not for serial communication on pins 0 and 1).

A Software Serial library allows for serial communication on any of the Mega2560's digital pins.

The ATmega2560 also supports TWI and SPI communication. The Arduino software includes a Wire library to simplify use of the TWI bus.

- **Programming:**

The Arduino Mega can be programmed with the Arduino software.

The ATmega2560 on the Arduino Mega comes preburned 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. It's also possible to bypass the bootloader and program the microcontroller through the ICSP (In-Circuit Serial Programming) header using Arduino ISP or similar.

- **USB Overcurrent Protection:**

The Arduino Mega2560 has a resettable polyfuse that protects your computer's USB ports from shorts and overcurrent. Although most computers provide their own internal



protection, the fuse provides an extra layer of protection. If more than 500 mA is applied to the USB port, the fuse will automatically break the connection until the short or overload is removed.

- **Physical Characteristics and Shield Compatibility:**

The maximum length and width of the Mega2560 PCB are 4 and 2.1 inches respectively, with the USB connector and power jack extending beyond the former dimension. Three screw holes allow the board to be attached to a surface or case. Note that the distance between digital pins 7 and 8 is 160 mil (0.16"), not an even multiple of the 100 mil spacing of the other pins.

The Mega2560 is designed to be compatible with most shields designed for the Uno, Diecimila or Duemilanove. Digital pins 0 to 13 (and the adjacent AREF and GND pins), analog inputs 0 to 5, the power header, and ICSP header are all in equivalent locations. Further the main UART (serial port) is located on the same pins (0 and 1), as are external interrupts 0 and 1 (pins 2 and 3 respectively). SPI is available through the ICSP header on both the Mega2560 and Duemilanove / Diecimila. *It's important to note that I<sup>2</sup>C is not located on the same pins on the Mega (20 and 21) as the Duemilanove / Diecimila (analog inputs 4 and 5).*

## 2.2 Micro SD shield:



Figure 2.2: Micro SD shield

The communication between the microcontroller and the SD card uses SPI, which takes place on digital pins 11, 12, and 13 (on most Arduino boards) or 50, 51, and 52 (Arduino Mega). Additionally, another pin must be used to select the SD card. This can be the hardware SS pin - pin 10 (on most Arduino boards) or pin 53 (on the Mega) - or another pin specified in the call to `SD.begin()`. Note that even if the hardware SS pin doesn't, it must be left as an output or the SD library won't work [2].

- **Examples how the SD library works:**

- **Data logger:** Log data from three analog sensors to a SD card using the SD library
- **Dump File:** Read a file from a SD card using the SD library and send it over the serial port
- **Files:** Create and destroy a file on a SD card
- **Read Write:** Read and write data to and from a file on a SD card
- **Card Info:** Get information about a SD card

## 2.3 Micro SD card and adapter:



Figure 2.3: Micro SD card



**Figure 2.4: Micro SD adapter**

The SD library allows for reading from and writing to SD cards, e.g. on the Arduino sd card shield. It is built on sdfatlib by William Greiman. The library supports FAT16 and FAT32 file systems on standard SD cards and SDHC cards. It uses short 8.3 names for files. The file names passed to the SD library functions can include paths separated by forward-slashes, /, e.g. "directory/filename.txt". Because the working directory is always the root of the SD card, a name refers to the same file whether or not it includes a leading slash (e.g. "/file.txt" is equivalent to "file.txt"). As of version 1.0, the library supports opening multiple files.

Adapter was mainly used as the sd module on the sd shield was large then the sd card which was intend to use for the device.

## **2.4 RTC module:**



**Figure 2.5: RTC module**

- **Overview:**

The DS3231 is a low-cost, extremely accurate I<sup>2</sup>C real-time clock (RTC) with an integrated temperature-compensated crystal oscillator (TCXO) and crystal. The device incorporates a battery input, and maintains accurate timekeeping when main power to the device is interrupted. The integration of the crystal resonator enhances the long-term accuracy of the device as well as reduces the piece-part count in a manufacturing line. The DS3231 is available in commercial and industrial temperature ranges, and is offered in a 16-pin, 300-mil SO package [3].

The RTC maintains seconds, minutes, hours, day, date, month, and year information. The date at the end of the month is automatically adjusted for months with fewer than 31 days, including corrections for leap year. The clock operates in either the 24-hour or 12-hour format with an AM/PM indicator. Two programmable time-of-day alarms and a programmable square-wave output are provided. Address and data are transferred serially through an I<sup>2</sup>C bidirectional bus [4].

A precision temperature-compensated voltage reference and comparator circuit monitors the status of V<sub>CC</sub> to detect power failures, to provide a reset output, and to automatically switch to the backup supply when necessary. Additionally, the RST pin is monitored as a pushbutton input for generating a  $\mu$ P reset.

- **Description of the module:**

- **Size:** 38mm (length) \* 22mm (W) \* 14mm (height)
- **Operating voltage :**3.3 - 5.5 V
- **Clock chip:** high-precision clock chip DS3231
- **Clock Accuracy :**0-40 degree range, the accuracy 2ppm, the error was about 1 minute
- Calendar alarm clock with two
- Programmable square-wave output
- Real time clock generator seconds, minutes, hours, day, date, month and year timing and provide valid until the year 2100 leap year compensation
- Chip temperature sensor comes with an accuracy of  $\pm 3$  degree

- Memory chips: AT24C32 (storage capacity 32K)
- IIC bus interface, the maximum transmission speed of 400KHz (working voltage of 5V)
- Can be cascaded with other IIC device, 24C32 addresses can be shorted A0/A1/A2 modify default address is 0x57
- With rechargeable battery LIR2032, to ensure the system after power failure, the clock move any natural normal

### **Wiring instructions (with Mega):**

SCL → 21

SDA → 20

VCC → 5V

GND → GND

## **2.5 Servo motor:**



**Figure 2.6: Servo motor**

### **• What's Inside the Servo?**

To fully understand how the servo works, you need to take a look under the hood. Inside there is a pretty simple set-up: a small DC motor, potentiometer and a control circuit. The motor is attached by gears to the control wheel. As the motor rotates, the potentiometer's

resistance changes, so the control circuit can precisely regulate how much movement there is and in which direction. When the shaft of the motor is at the desired position, power supplied to the motor is stopped. If not, the motor is turned in the appropriate direction. The desired position is sent via electrical pulses through the signal wire. The motor's speed is proportional to the difference between its actual position and desired position. So if the motor is near the desired position, it will turn slowly, otherwise it will turn fast [5]. This is called proportional control. This means the motor will only run as hard as necessary to accomplish the task at hand, a very efficient little guy.

- **How is the Servo Controlled?**

Servos are controlled by sending an electrical pulse of variable width, or pulse width modulation (PWM), through the control wire. There is a minimum pulse, a maximum pulse and a repetition rate. A servo motor can usually only turn  $90^\circ$  in either direction for a total of  $180^\circ$  movement. The motor's neutral position is defined as the position where the servo has the same amount of potential rotation in the both the clockwise or counter-clockwise direction. The PWM sent to the motor determines position of the shaft, and based on the duration of the pulse sent via the control wire the rotor will turn to the desired position. The servo motor expects to see a pulse every 20 milliseconds (ms) and the length of the pulse will determine how far the motor turns. For example, a 1.5ms pulse will make the motor turn to the  $90^\circ$  position. Shorter than 1.5ms moves it to  $0^\circ$  and any longer than 1.5ms will turn the servo to  $180^\circ$ , as diagramed below.

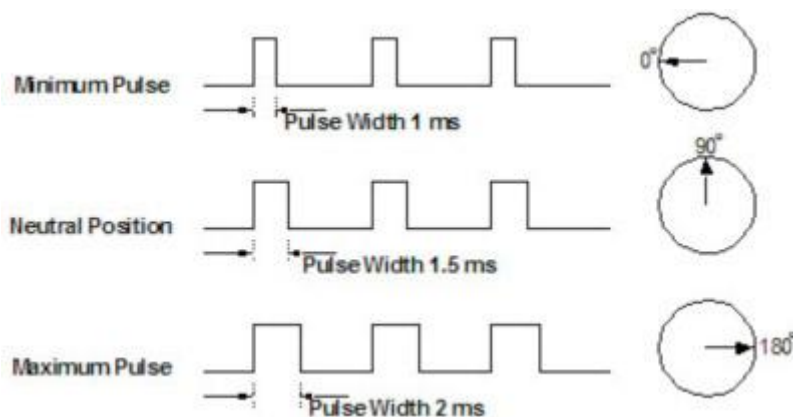


Figure 2.7: Variable Pulse Width Control Servo position

When these servos are commanded to move, they will move to the position and hold that position. If an external force pushes against the servo while the servo is holding a position, the servo will resist from moving out of that position. The maximum amount of force the servo can exert is called the torque rating of the servo. Servos will not hold their position forever though; the position pulse must be repeated to instruct the servo to stay in position.

## 2.6 PIR Sensor:



**Figure 2.8: PIR Sensor**

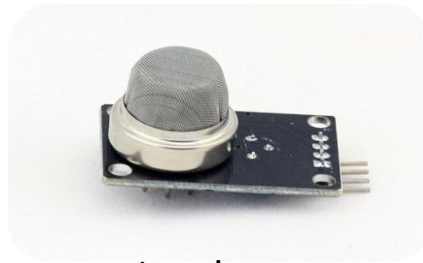
The modern world is filled with gadgets that get excited when they sense human motion. Automatic doors in elevators and shopping malls, burglar alarms at houses and shops, automatic lighting systems, electronic amenities in washrooms are just a few examples where human presence or absence puts the device into active or passive state. Passive Infrared Sensor (PIR), this small electronic device is the curious case for this Insight [7].

Every object that has a temperature above perfect zero emits thermal energy (heat) in form of radiation. We, Homo sapiens, radiate at wavelength of 9-10micrometers all time of the day. The PIR sensors are tuned to detect this IR wavelength which only emanates when a human being arrives in their proximity. The term “pyro electricity” means: heat that generates electricity (here, an electric signal of small amplitude). Since these sensors do not have an infrared source of their own, they are also termed as passive.



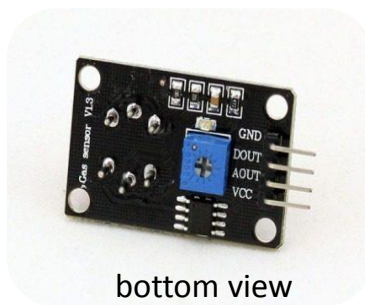


## 2.8 Gas sensor (MQ-2):



top view

Figure 2.10: MQ-2 Gas sensor (Top view)



bottom view

Figure 2.11 MQ-2 Gas sensor (Bottom view)

Sensitive material of MQ-2 gas sensor is  $\text{SnO}_2$ , which with lower conductivity in clean air. When the target combustible gas exist, The sensor's conductivity is more higher along with the gas concentration rising. Please use simple electrocircuit, Convert change of conductivity to correspond output signal of gas concentration [9].

- **Technical Data:**

Model No.		MQ-2	
Sensor Type		Semiconductor	
Standard Encapsulation		Bakelite (Black Bakelite)	
Detection Gas		Combustible gas and smoke	
Concentration		300-10000ppm ( Combustible gas)	
Circuit	Loop Voltage	$V_c$	$\leq 24V$ DC
	Heater Voltage	$V_H$	$5.0V \pm 0.2V$ AC or DC
	Load Resistance	$R_L$	Adjustable
Character	Heater Resistance	$R_H$	$31\Omega \pm 3\Omega$ (Room Tem.)
	Heater consumption	$P_H$	$\leq 900mW$
	Sensing Resistance	$R_s$	$2K\Omega - 20K\Omega$ (in 2000ppm $C_3H_8$ )
	Sensitivity	$S$	$R_s$ (in air)/ $R_s$ (1000ppm isobutene) $\geq 5$
	Slope	$\alpha$	$\leq 0.6(R_{5000ppm}/R_{3000ppm} CH_4)$
Condition	Tem. Humidity		$20^\circ C \pm 2^\circ C$ ; $65\% \pm 5\% R_H$
	Standard test circuit		$V_c: 5.0V \pm 0.1V$ ; $V_H: 5.0V \pm 0.1V$
	Preheat time		Over 48 hours

**Table 2: Features of MQ-2**

## 2.9 LCD Display:



Figure 2.12: LCD Display

Pin	Label	Description
1	VSS	Ground
2	VCC	5V power supply input
3	VEE	Contrast adjust
4	RS	Data/command select (data=high, command=low)
5	R/W	Read/write select (read=high, write=low)
6	E	Enable
7	DB0	Data bit 0 (8 bit mode)
8	DB1	Data bit 1 (8 bit mode)
9	DB2	Data bit 2 (8 bit mode)
10	DB3	Data bit 3 (8 bit mode)
11	DB4	Data bit 4 (4 or 8 bit mode)
12	DB5	Data bit 5 (4 or 8 bit mode)
13	DB6	Data bit 6 (4 or 8 bit mode)
14	DB7	Data bit 7 (4 or 8 bit mode)
15	LED+	Backlight 5V power supply input
16	LED-	Backlight ground

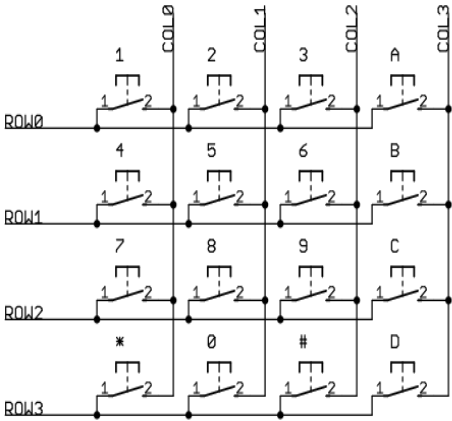
Table 3: PIN configuration of LCD

This Character LCD module is a 1602 LCM type with 2 rows of 16 columns. This model operates at 5V and has a blue backlight with white on blue text. It features a standard HD44780 compatible controller to simplify integration into your project and includes a strip of male header that can be soldered to the board as needed [10].

**2.10 Keypad:**



**Figure 2.13: Keypad**



**Figure 2.14: Keypad internal structure**

This is a membrane matrix keypad 4X4. The keys printed on the keypad are 0-9, A-D, \* and #.The lcd backpack supports multitap so you can use the keypad to enter all alphanumerical glyphs and all symbols on a PC keyboard.

## 2.11 Exhaust fan:



Figure 2.15: Exhaust fan

It was used for gas detection purpose. When leakage gas is detected it will then start to extract the gas to the outer environment.

## 2.12 Buzzer



Figure 2.16: Buzzer

A piezoelectric(buzzer) element may be driven by an oscillating electronic circuit or other audio signal source, driven with a piezoelectric audio amplifier. Sounds commonly used to

indicate that a button has been pressed are a click, a ring or a beep. It's normally used as an alarm in any kind of prototyped design.

### 2.13 LED

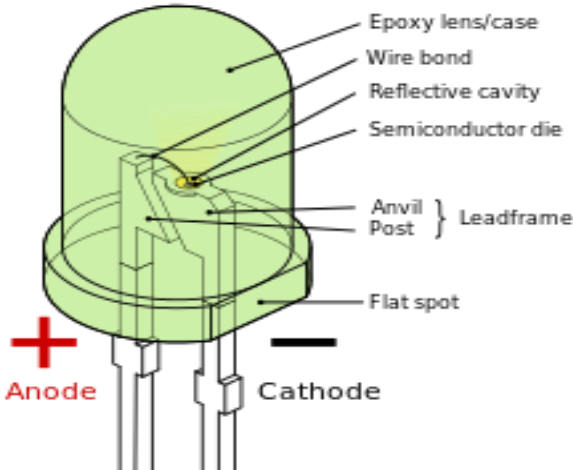


Figure 2.17: LED

light-emitting diode (LED) is a (direct band gap) semiconductor device. When an external voltage applied in such a direction so that the led is forward biased then the electrons are able to combine with the uncompensated donar atoms within the device and releasing the energy in the form of light which is called electroluminescence and the color is depended upon the energy band gap of the emitted photon

### 2.14 Pushbutton

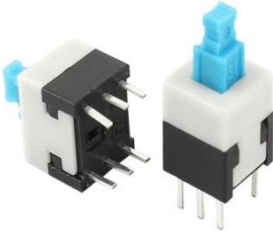


Figure 2.18: Pushbutton

There are various types of push button. If the bottom view is considered then the 6-pin push button's shorted pins are the upper left 2 pins. When it's pressed the those two pins are getting shorted and it completes the whole path of signal flow, as a result a led, buzzer or any other device can be turned on.

## 2.15 Potentiometer:



**Figure 2.19: Potentiometer**

A potentiometer, informally a pot, is a three-terminal resistor with a sliding or rotating contact that forms an adjustable voltage divider. If only two terminals are used, one end and the wiper, it acts as a variable resistor or rheostat.

A potentiometer measuring instrument is essentially a voltage divider used for measuring electric potential (voltage); the component is an implementation of the same principle, hence its name.

Potentiometers are commonly used to control electrical devices such as volume controls on audio equipment. Potentiometers operated by a mechanism can be used as position transducers, for example, in a joystick. Potentiometers are rarely used to directly control significant power (more than a watt), since the power dissipated in the potentiometer would be comparable to the power in the controlled load.

## 2.17 D400 transistor

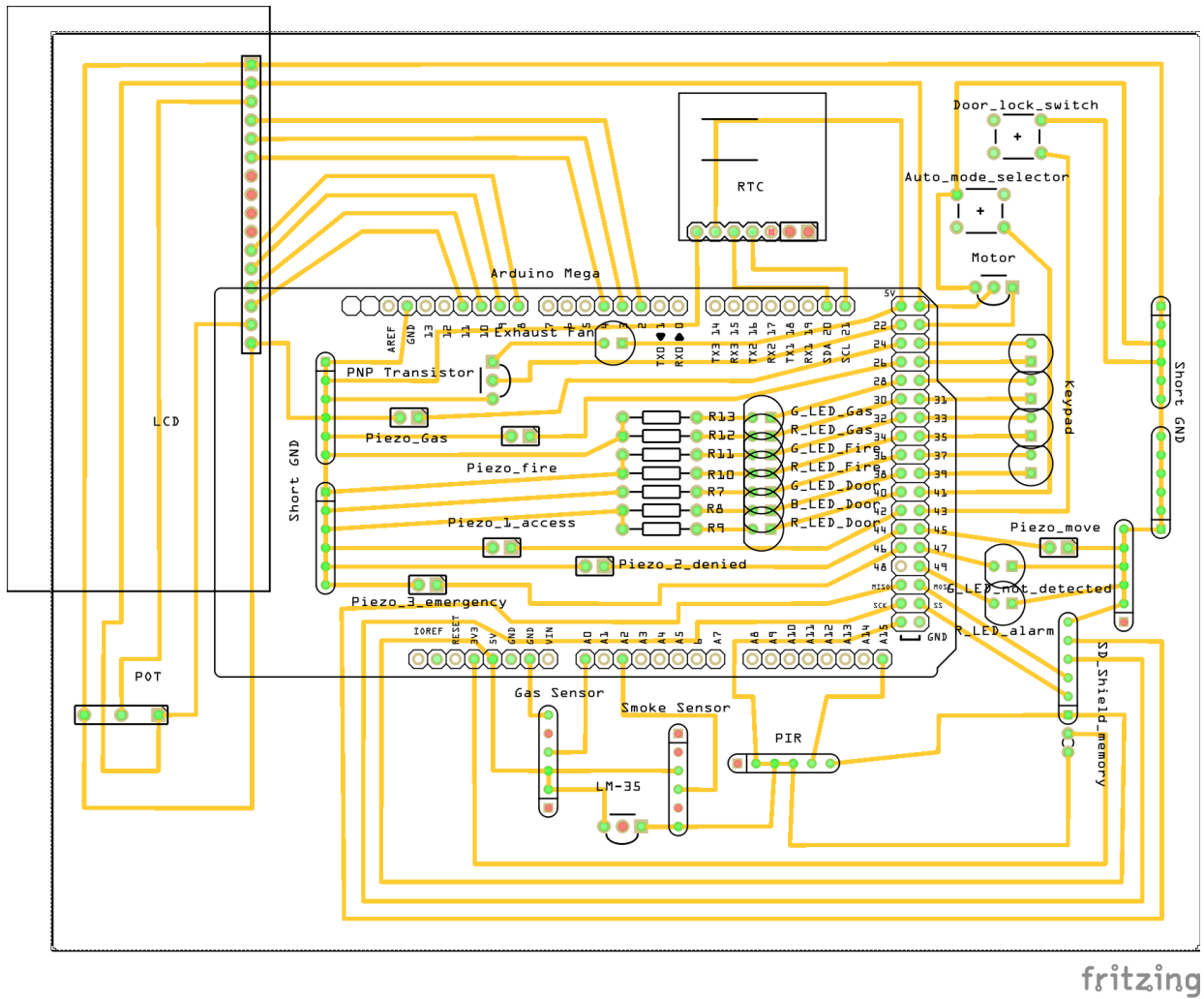


**Figure 2.20: D400 transistor**

The general electric d400d is a power transistor designed for various specific and general purpose application such as: output and drive stages of amplifiers operating at frequencies from DC to greater than 1.0 MHz series, shunt and switching regulators low and high frequency inverters/converters and many others [11].



## 2.18 Diagram showing connections of equipment with ATMEGA 2560 pins for designing the hardware of our project



We have designed the overall connection diagram by using fritzing [13] software.

### **//for door lock system**

#### ***//connect the arduino pins to these lcd pins***

// GND, VDD, pot----> 1,2,3

//LCD Pin 4 --> Arduino Pin 2

//LCD Pin 5 --> Arduino Pin 3

//LCD Pin 6 --> Arduino Pin 4

//LCD Pin 11 --> Arduino Pin 8

//LCD Pin 12 --> Arduino Pin 9

//LCD Pin 13 --> Arduino Pin 10

//LCD Pin 14 --> Arduino Pin 11

#### ***//door switch pin connections***

dsw=41; ledg = 36; ledb = 38; ledr=40; piezo1 =42; piezo2 =44; piezo3=46

#### ***//Keypad pin connections***

rowPins[ROWS] = {25, 27, 29, 31 }; colPins[COLS] = {33, 35, 37, 39 };

#### ***//The SD memory shield pin connections:***

//SPI Uno Mega

//SS 10 53

//MOSI 11 51

//MISO 12 50

//SCK 13 52

### **//For fire and gas leakage detection system**

// PIN\_22=Exhaust Fan, PIN\_24=Gas Buzzer, PIN\_26=Fire Buzzer, Pin\_28=Green\_LED\_Gas,

// Pin\_30=Red\_LED\_Gas, PIN\_32=GREEN\_LED\_Fire, PIN\_34=RED\_LED\_Fire

//PIN\_22 is connected to base of a PNP Transistor which turns on the Exhaust fan when it gets High signal

### **//For motion detection system**

//buttonPin =15; onLedPin = 49; offLedPin = 47; alertLedPin = 45;

## **Chapter 3**

### **Design of safe and secure home security system (SOFTWARE)**

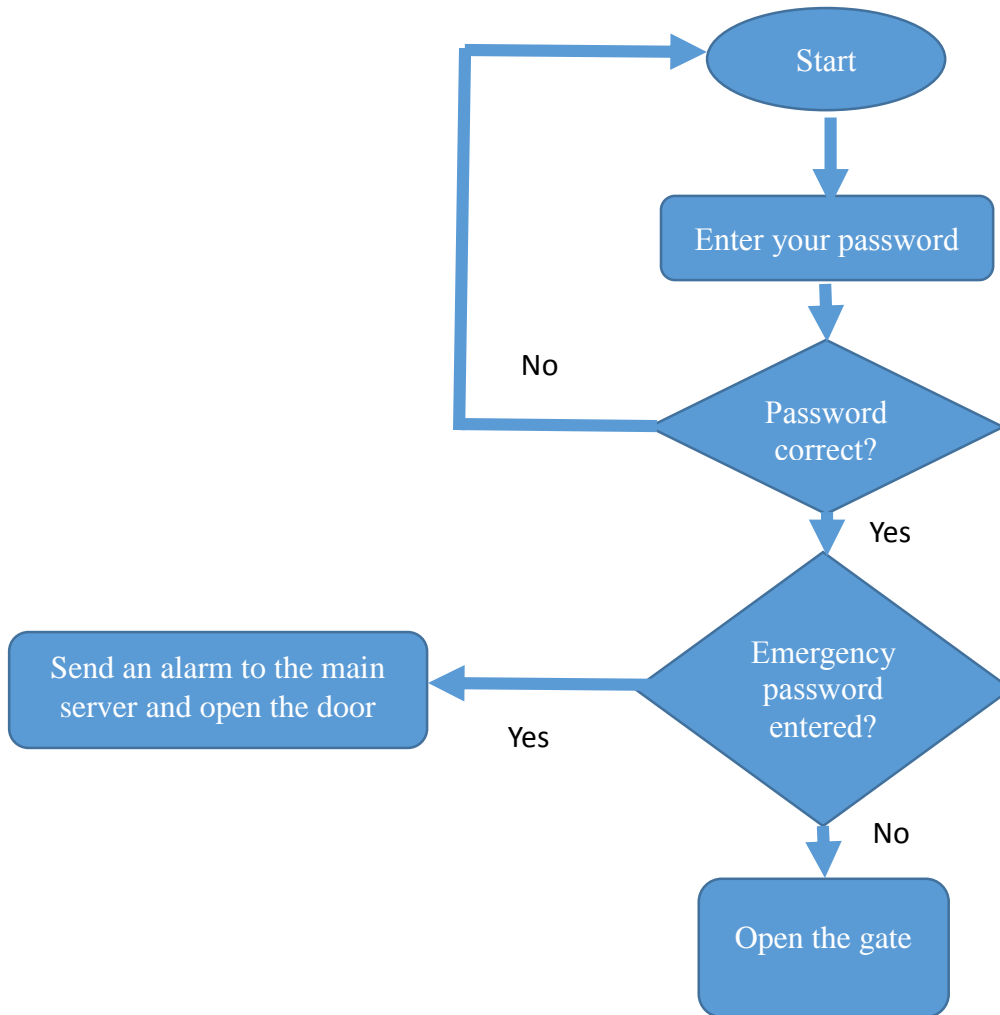
We used many equipment for making our safe and secured home device. The hardware information were given in the previous chapter. In this chapter we will discuss about the software part of our design and analysis of the embedded safe and secure home system. For designing our total system we went through some steps. The steps are shown by the following flow charts and the corresponding coding reference are given with it.

### 3.1 PRIMARY DOOR ENTRANCE FLOW CHART:

Each residents will have a password to open the door.

The primary entrance data system which will be connected to the front door has its switching option as well as it will have a storage device system that will keep the information who entered at which time through the gate.

There will be an emergency password facility provided with the system that will help the security purpose of the dwellers too. In Flow Chart 1 we can see the total procedure and respective coding is given in the appendix in section A.2

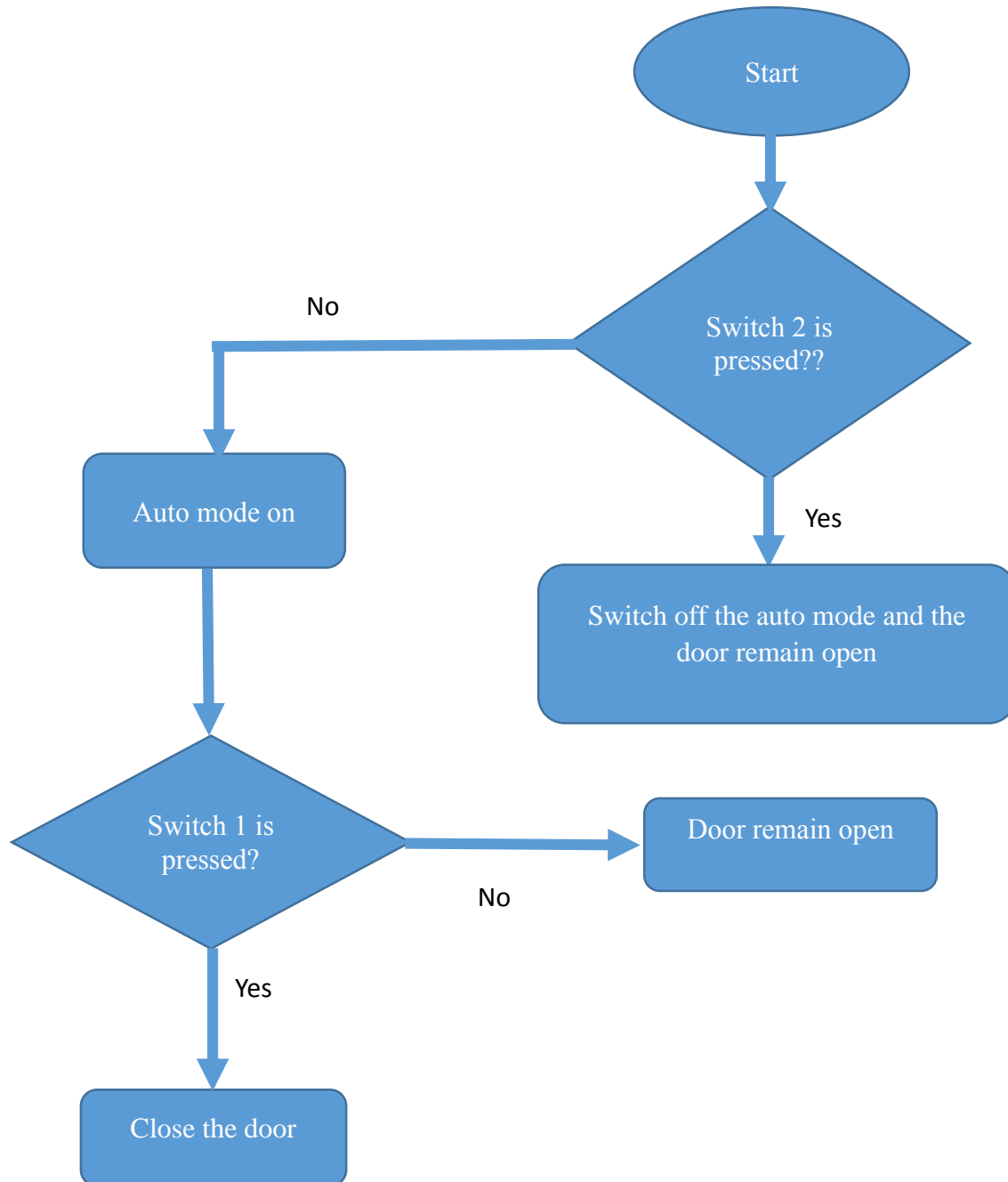


**Flow Chart 1: PRIMARY DOOR ENTRANCE FLOW CHART**

### 3.2 SECONDARY DOOR ENTRANCE FLOW CHART:

There will be two switches totally with the system-the first one will act as a lock switch and the other one will act as an auto mode selector switch.

In Flow Chart 2 we can see the total procedure and respective coding is given in the appendix in section A.1

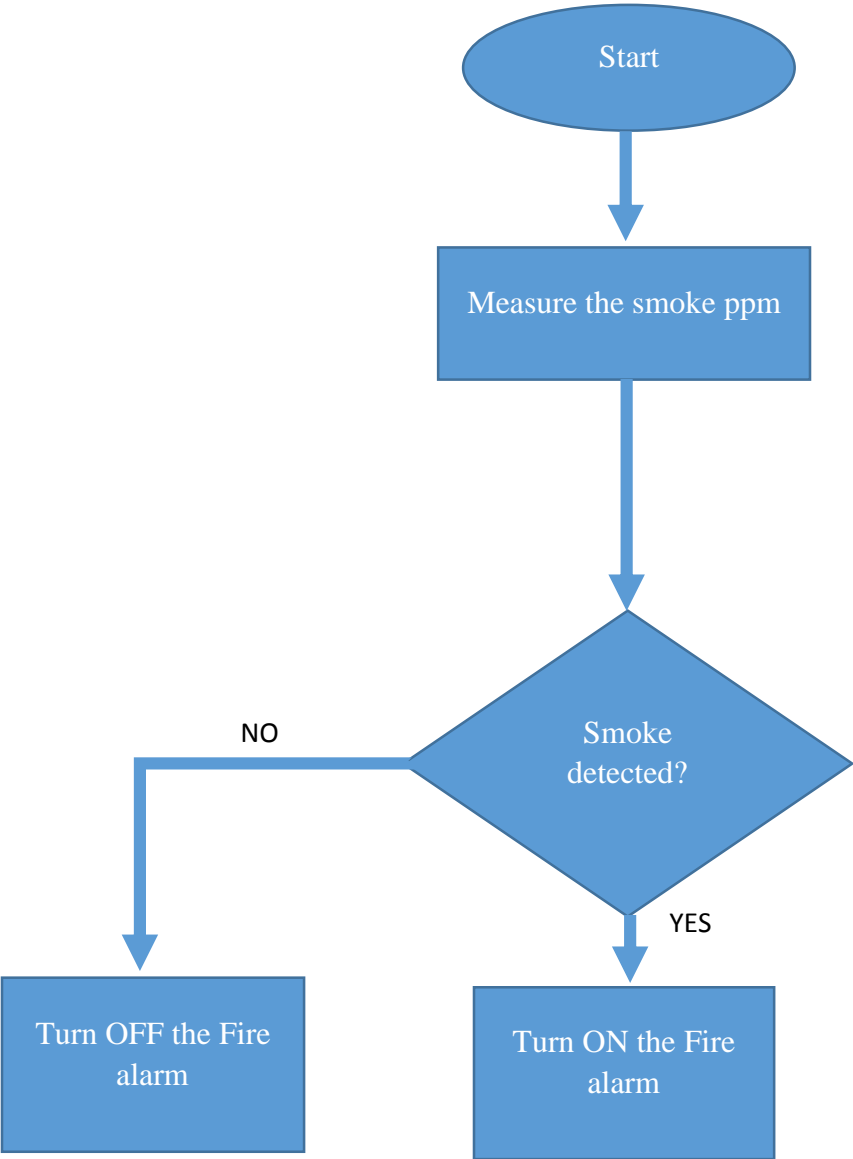


Flow Chart 2: SECONDARY DOOR ENTRANCE FLOW CHART

### 3.3 FIRE DETECTION BY SMOKE SENSOR FLOW CHART:

The smoke sensor detects the smoke ppm. If smoke is detected then it turns on the alarm and if not detected then there is no alarm.

In Flow Chart 3 we can see the total procedure and respective coding is given in the appendix in section A.4

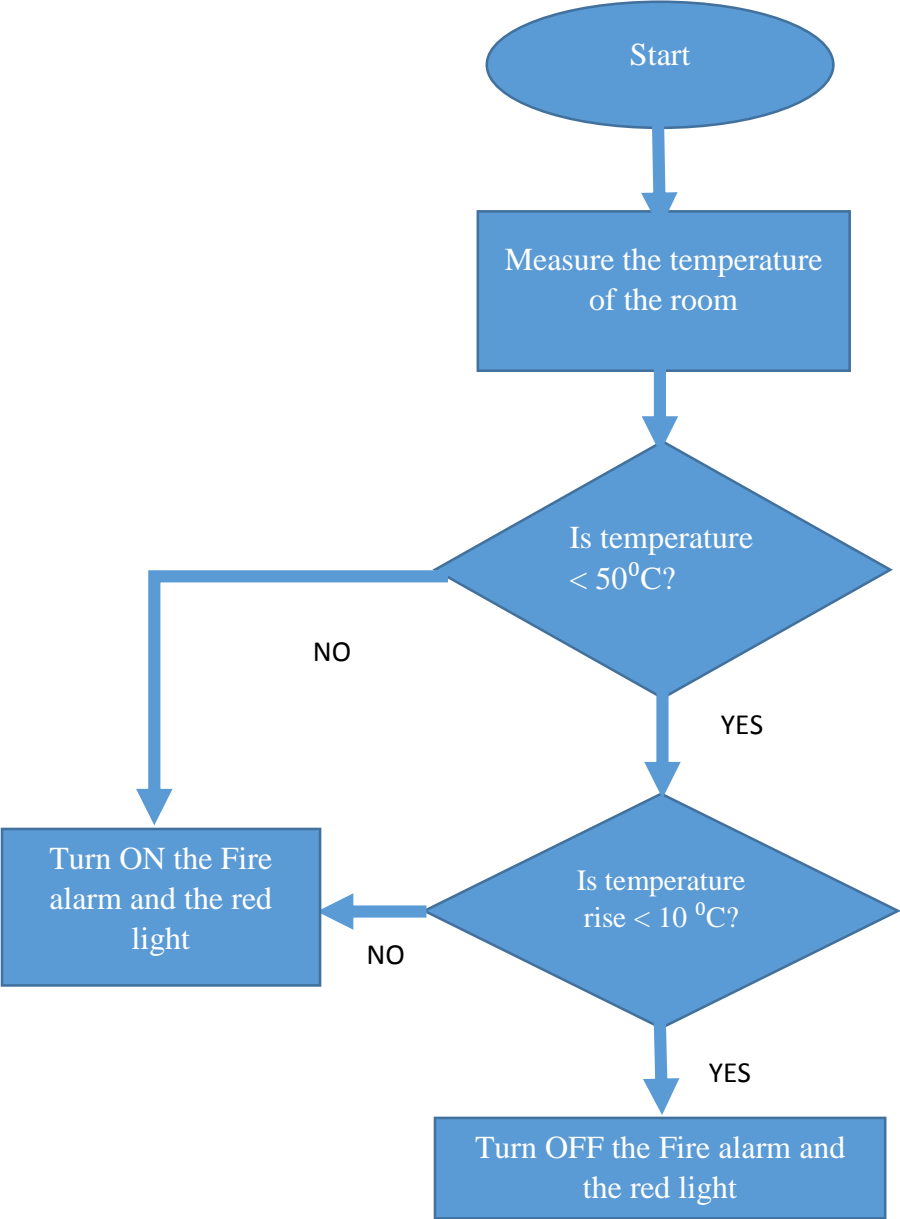


**Flow Chart 3: FIRE DETECTION BY SMOKE SENSOR FLOW CHART**

### 3.4 FIRE DETECTION BY HEAT SENSOR FLOW CHART:

Heat sensor detects the temperature of the surrounding. The alarm remains deactivated for room temperature. When there is a fire the temperature of the surroundings increases. After it exceeds certain temperature (i.e. 50 degree Celsius) the alarm is activated. Also if there is temperature rise of more than 10 degree Celsius the alarm is activated.

In Flow Chart 4 we can see the total procedure and respective coding is given in the appendix in section A.5.

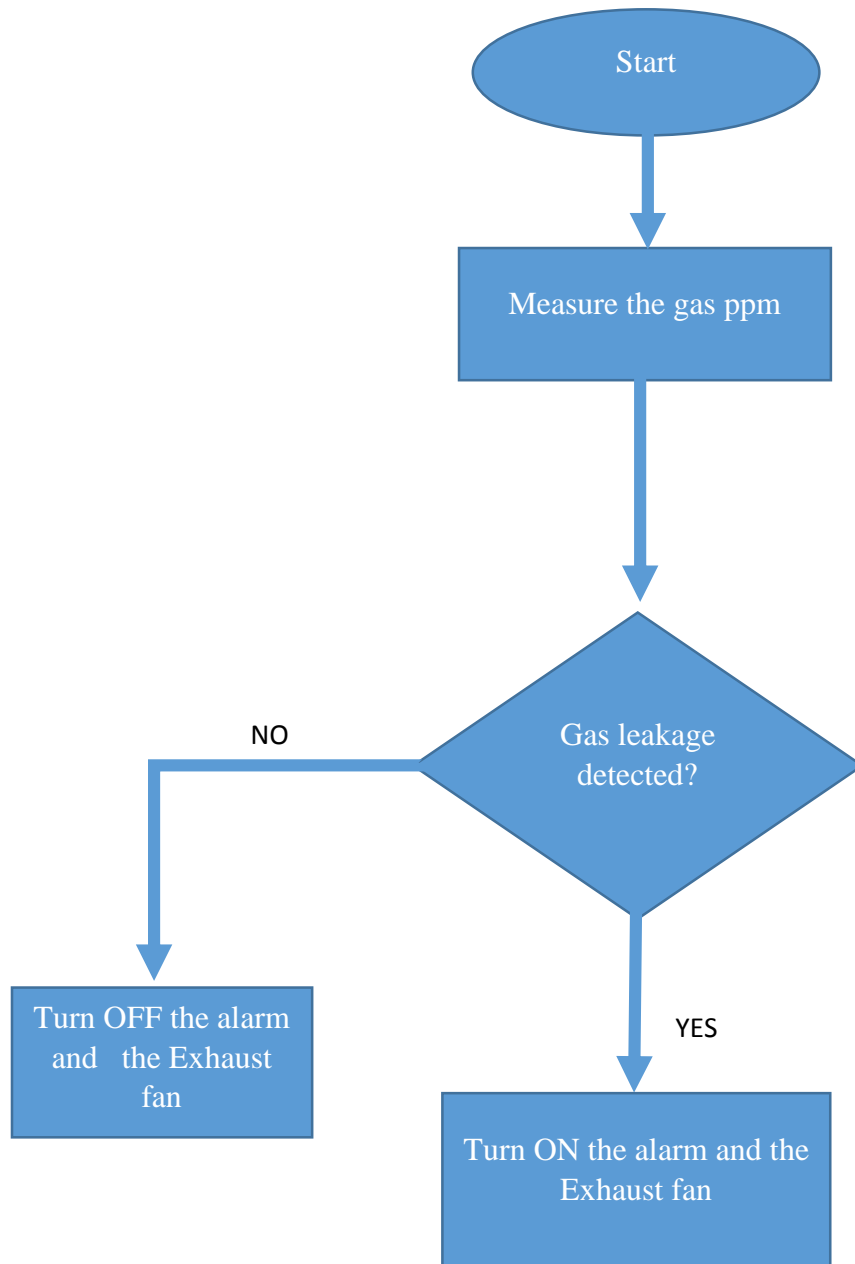


Flow Chart 4: FIRE DETECTION BY HEAT SENSOR FLOW CHART

### 3.5 GAS LEAKAGE DETECTION FLOW CHART:

The Gas sensor measures the combustible gas ppm. When there is a gas leakage the gas sensor detects it and the alarm is activated. At the same time exhaust fan is turned on. If there is no leaked gas then there is no alarm and the exhaust fan remains off.

In Flow Chart 5 we can see the total procedure and respective coding is given in the appendix in section A.3



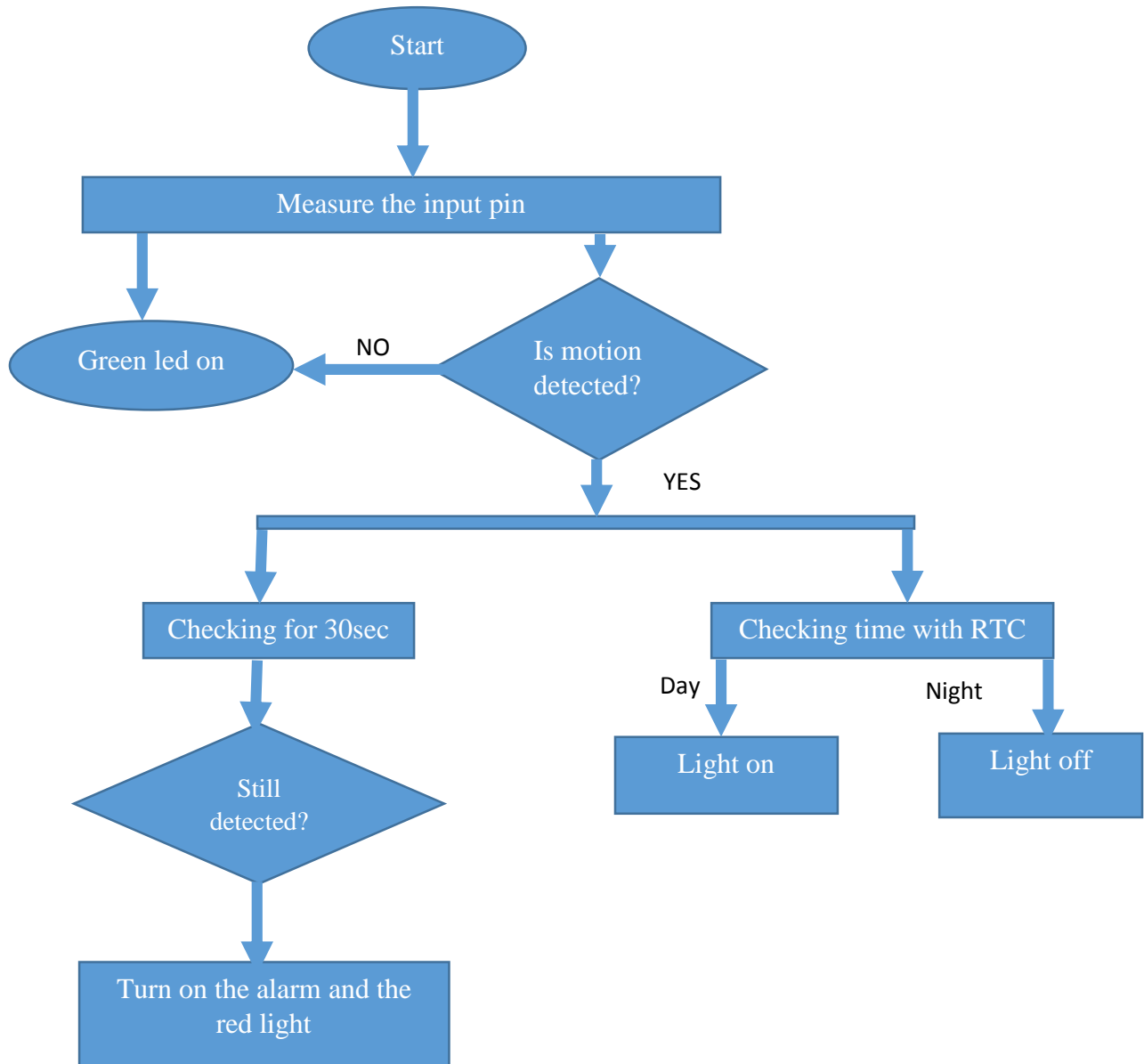
Flow Chart 5: GAS LEAKAGE DETECTION FLOW CHART



### 3.6 PIR SENSOR DETECTION FLOW CHART:

PIR detects the motion of humans. Homo sapiens radiate at wavelength of 9-10micrometers. PIR will be attached above the locked password protected door so that if anyone stays more time than usual (we set it as 30 sec. because 30 sec is enough for giving a password) then it seems having problem with the password and therefore it will create an alarm.

In Flow Chart 6 we can see the total procedure and respective coding is given in the appendix in section A.6



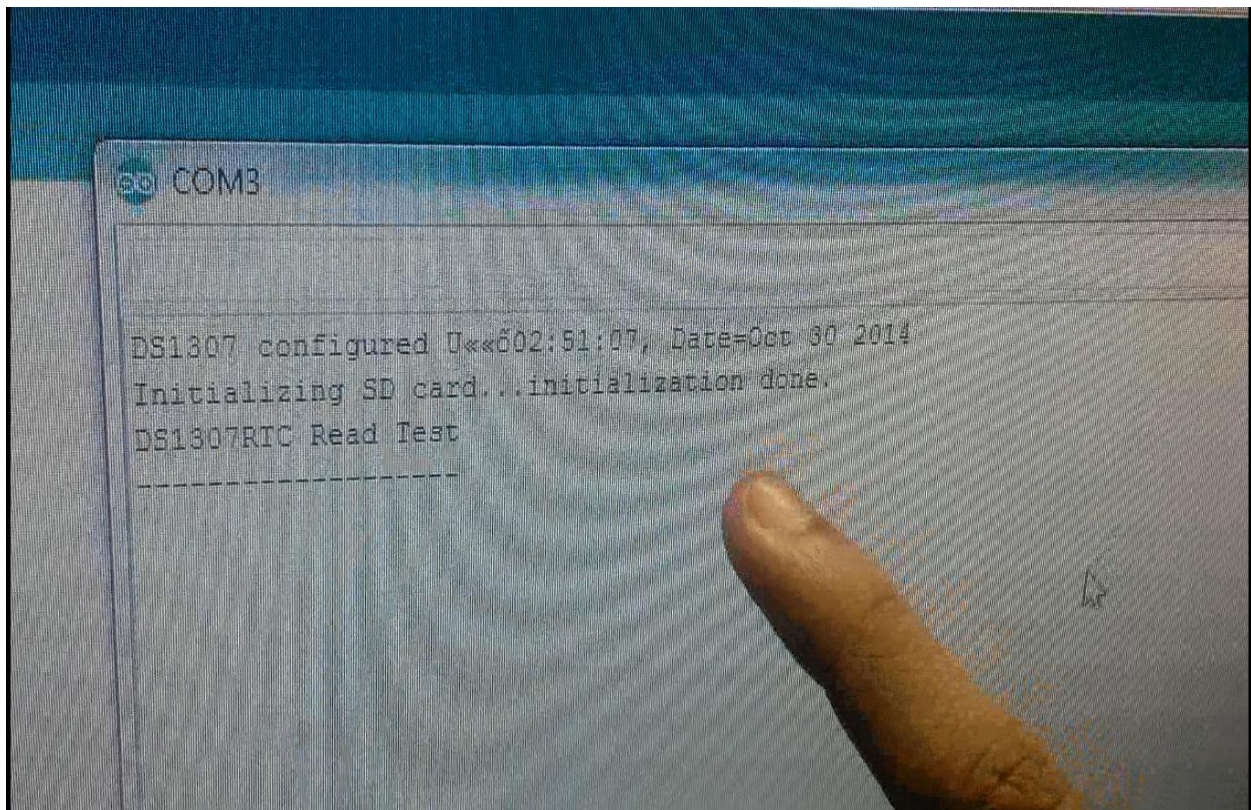
Flow Chart 6: PIR SENSOR DETECTION FLOW CHART

# Chapter 4

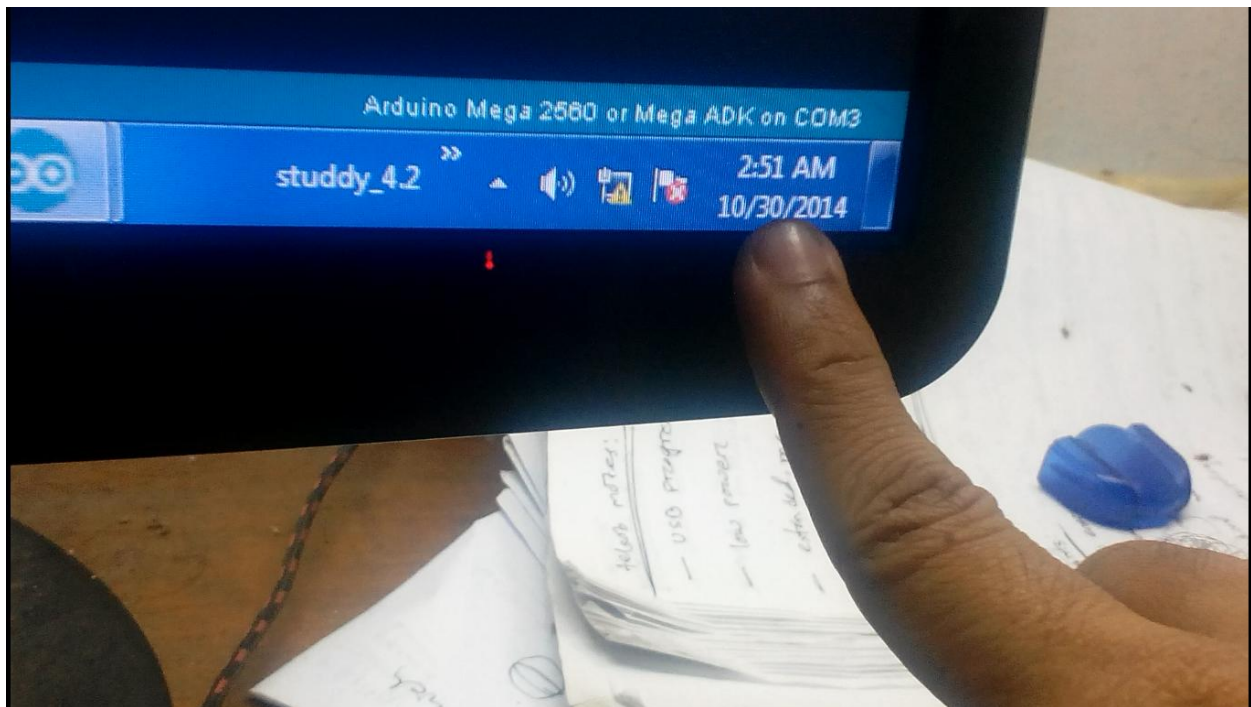
## Analysis

### 4.1 Implementation of the equipment in designing our project:

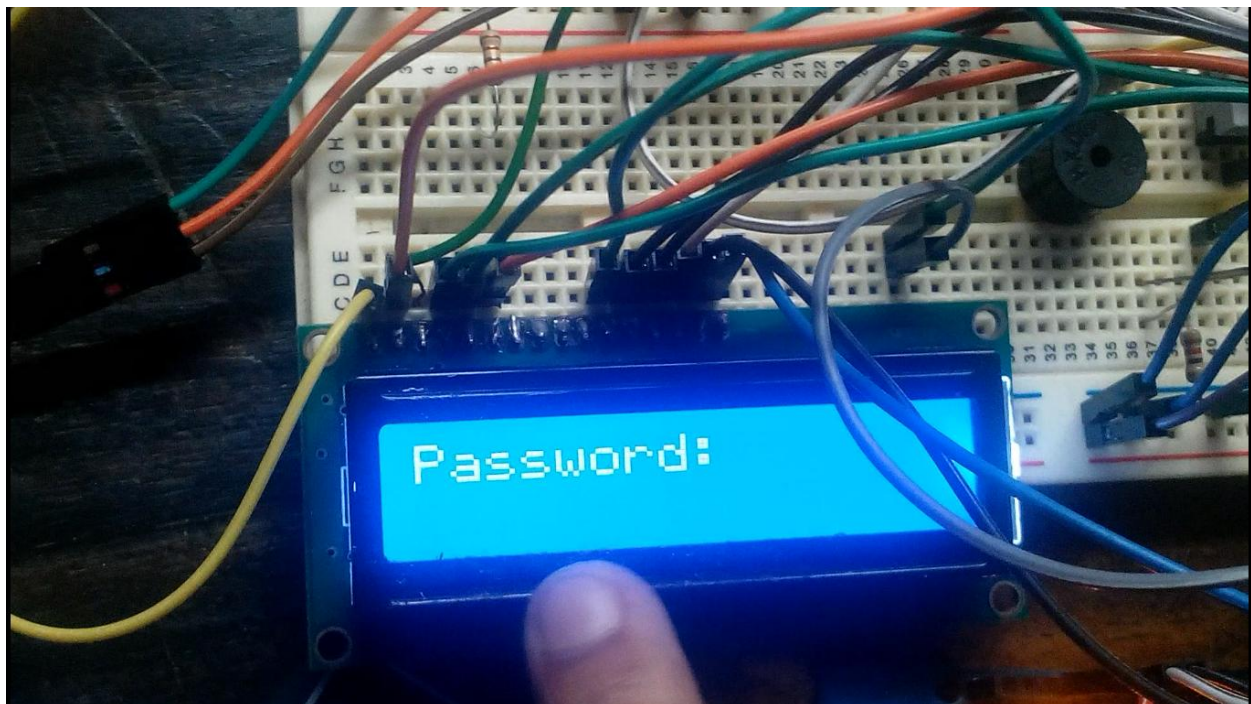
- When the system started to work then the following windows appeared and showed us that RTC had already extracted the data from the power supplying device which was our PC.



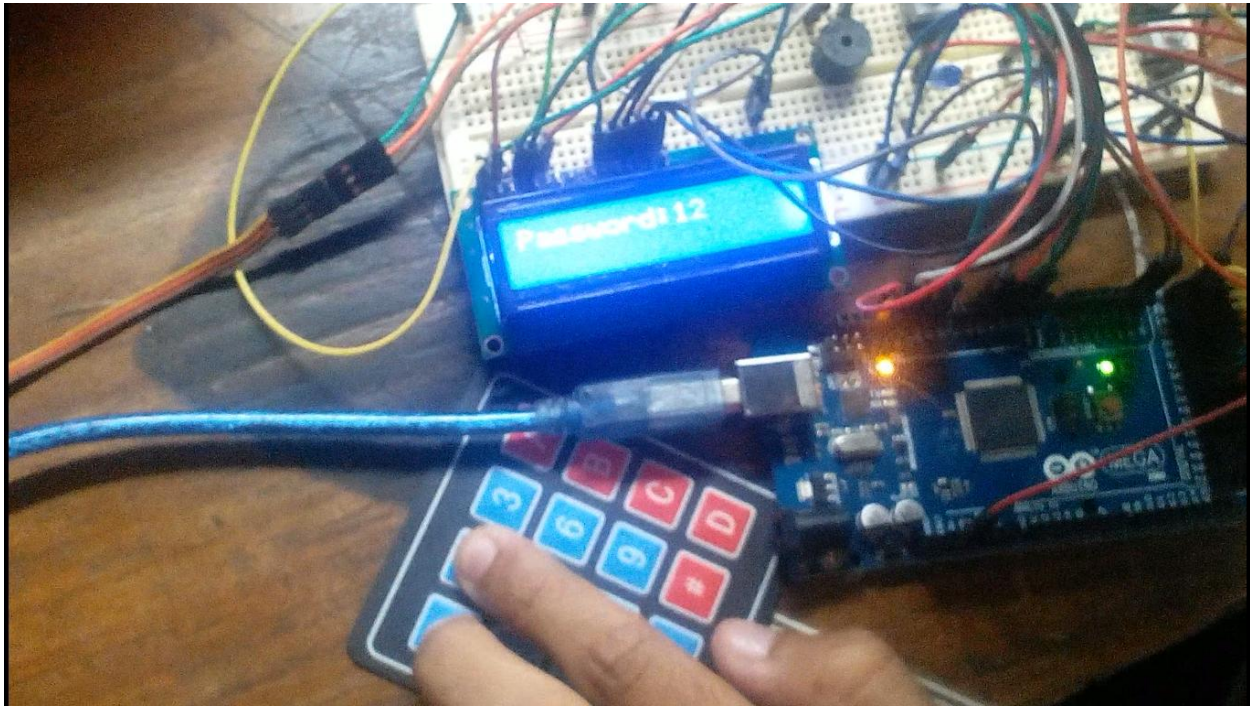
- From the following picture it can be seen perfectly.



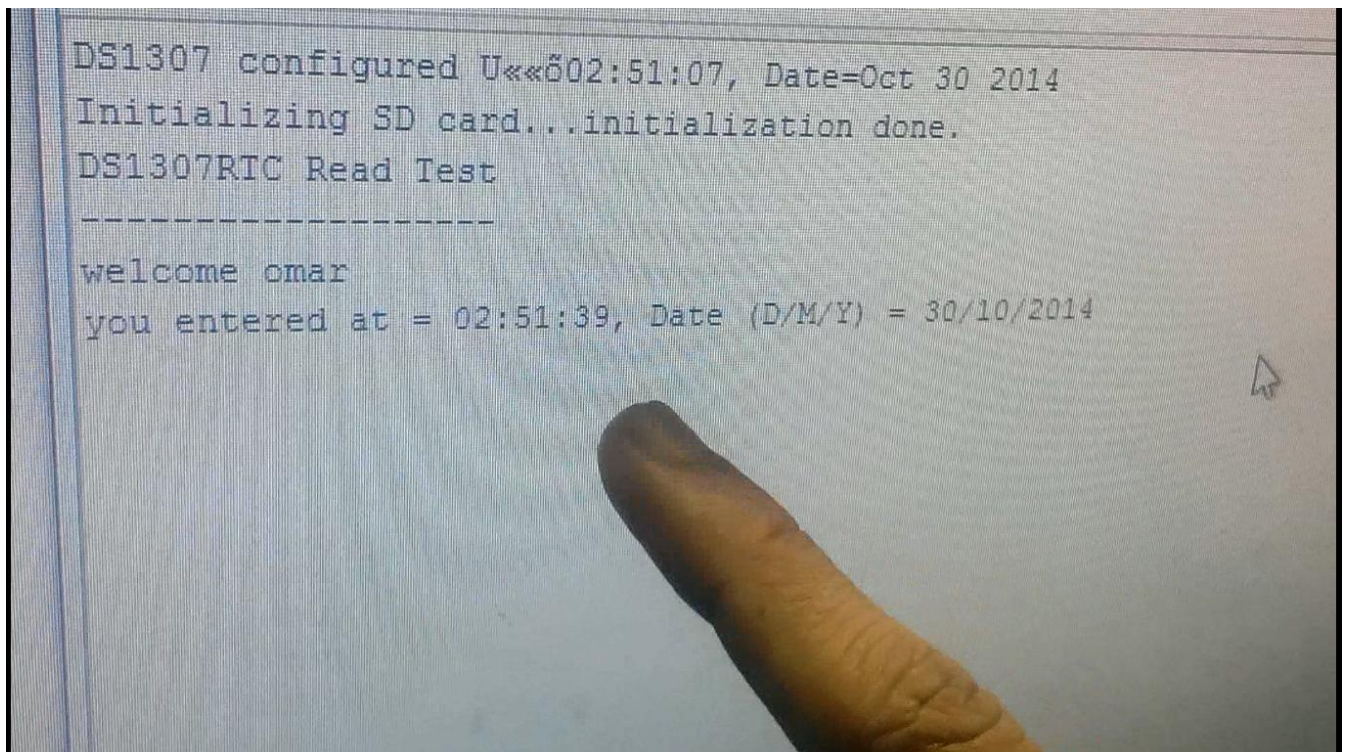
- Each time the device would want a password and from the LCD we could see whether anyone had already given the password or not.



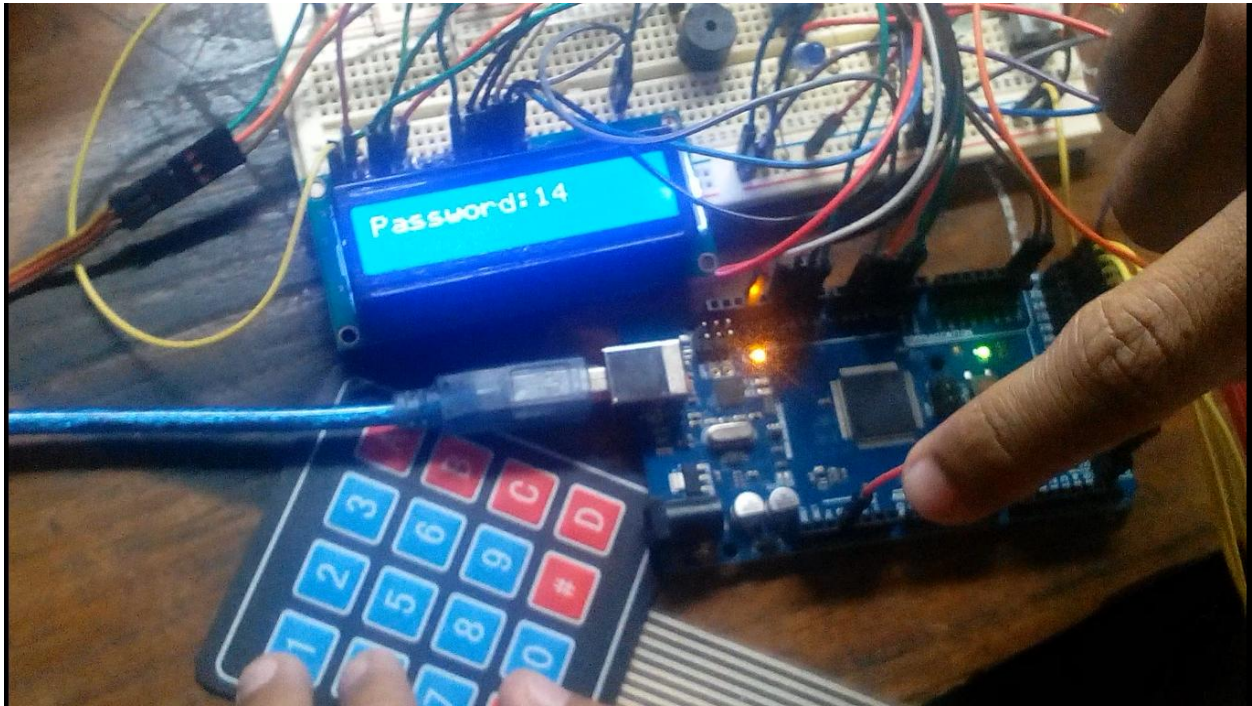
- We started to give our first password and it was 12



- It detected the owner of the 1<sup>st</sup> password and also showed the correct time which was extracted by the RTC module.



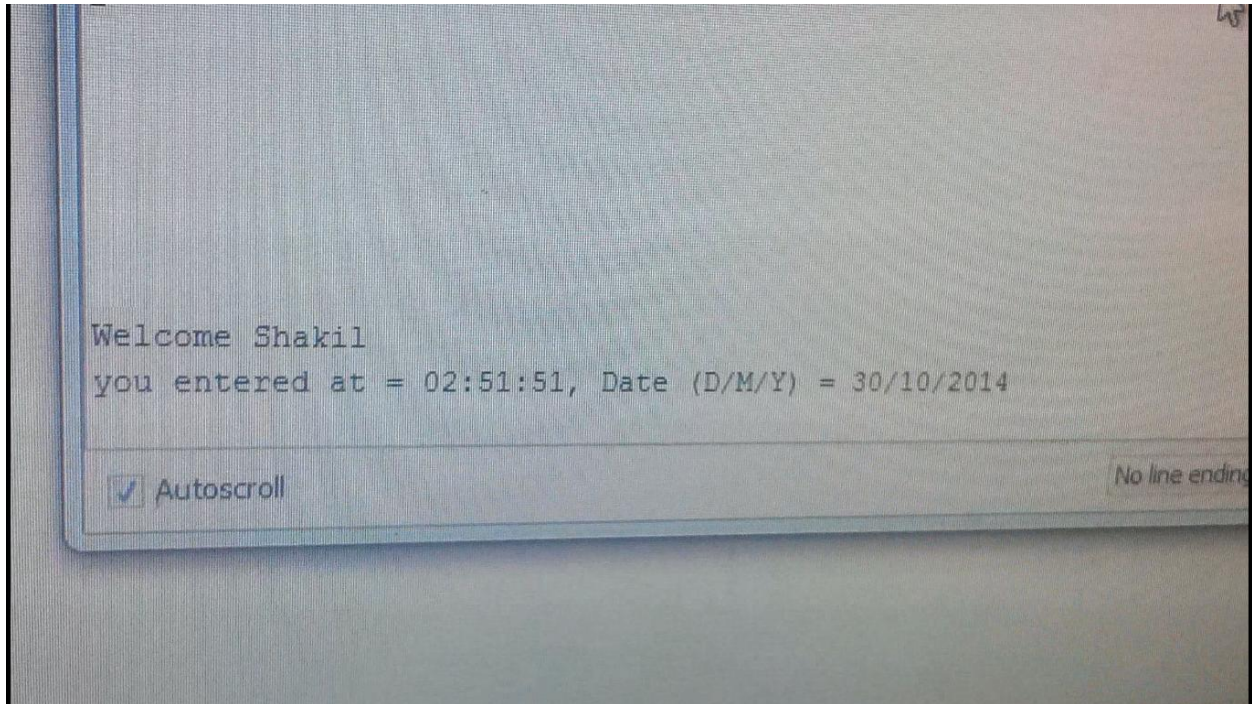
- We started to give our 2<sup>nd</sup> password and it was 14.



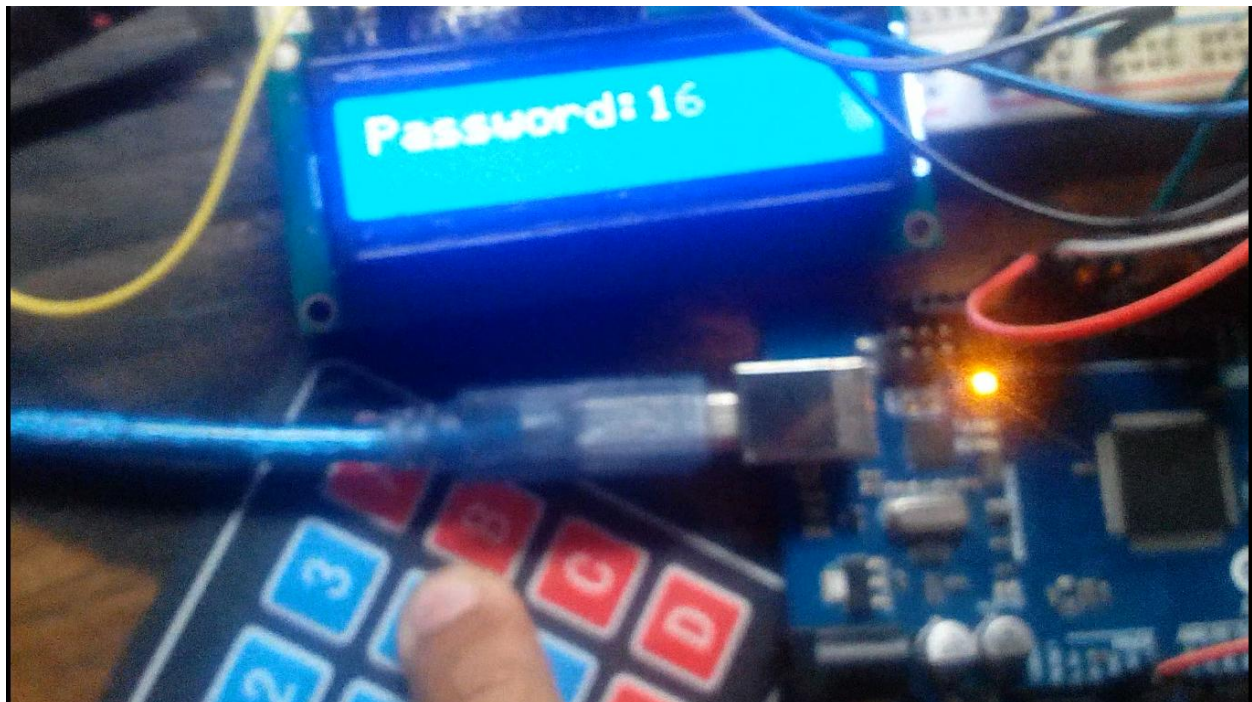
- It detected the owner of the 2<sup>nd</sup> password and from LCD we could see that.



- It detected the owner of the password and also showed the correct time which was extracted by the RTC module. This window appeared from COM port 3.



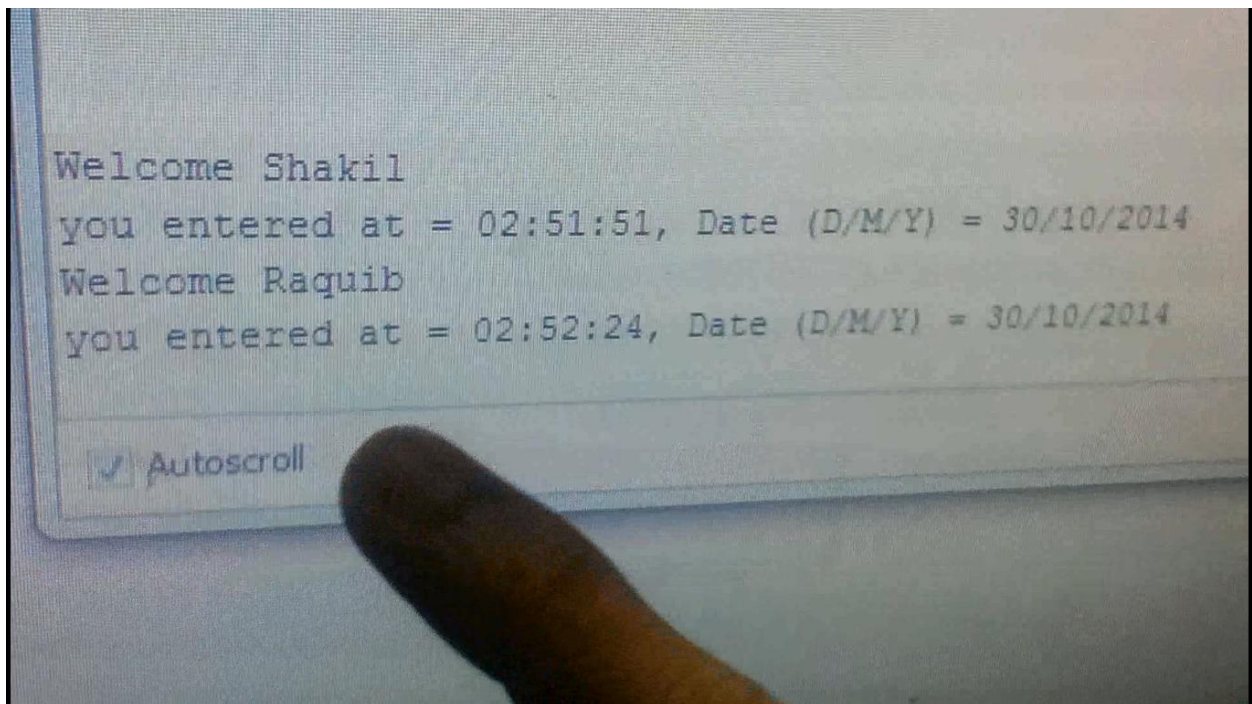
- We started to give our last password and it's 16.



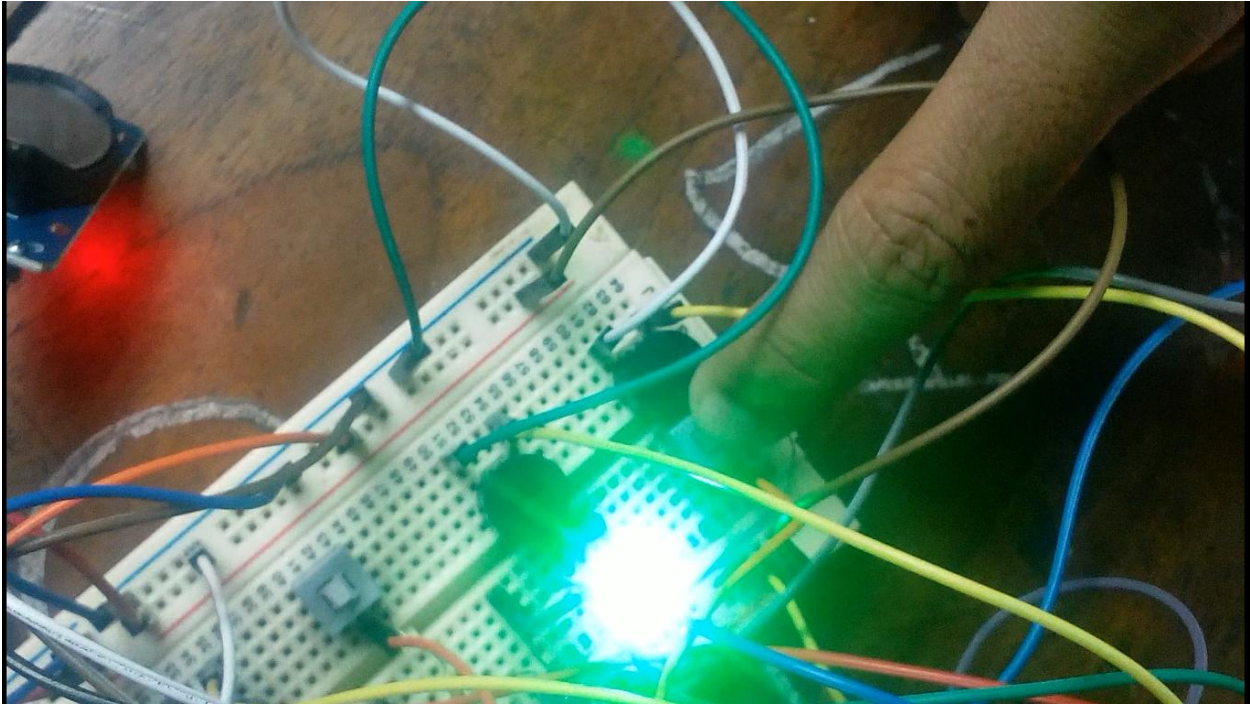
- It detected the owner of the last password and from LCD we could see that.



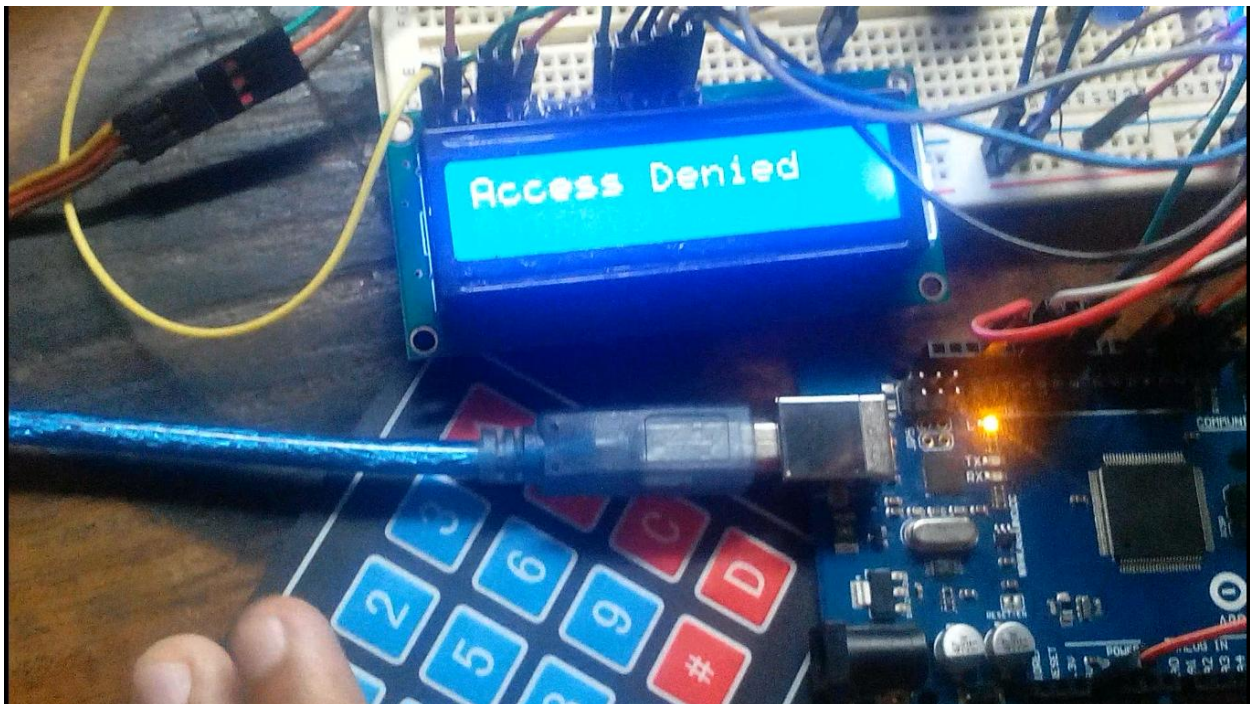
- It detected the owner of the password and also showed the correct time which was extracted by the RTC module. This window appeared from COM port 3.



- For every entry the green LED would indicate it which is shown here.

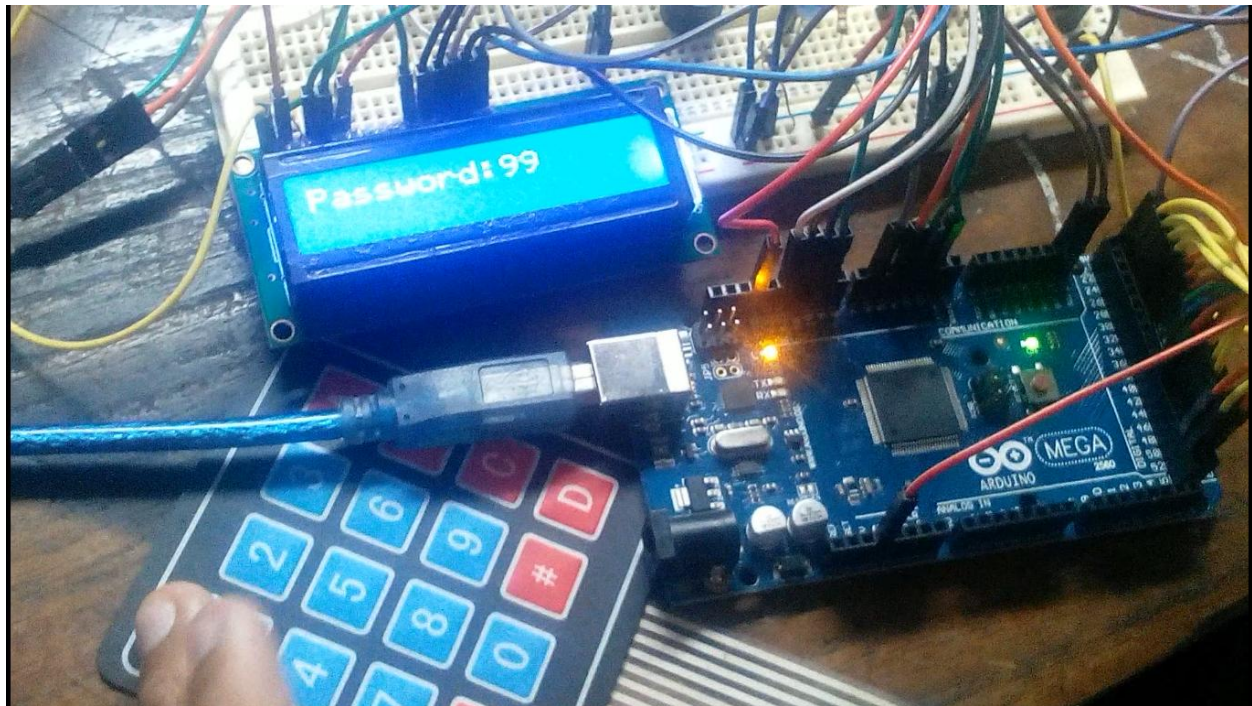


- For pressing the wrong password to enter the building, the access would be denied and it could be seen from the LCD display.

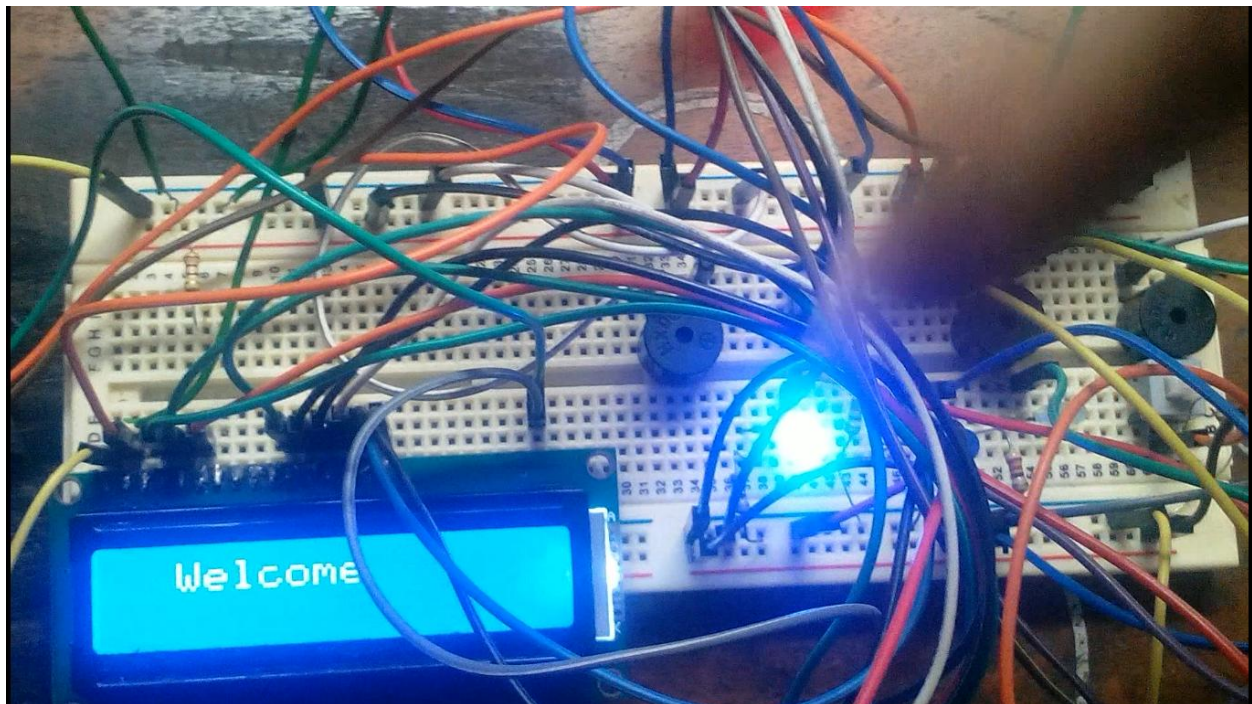




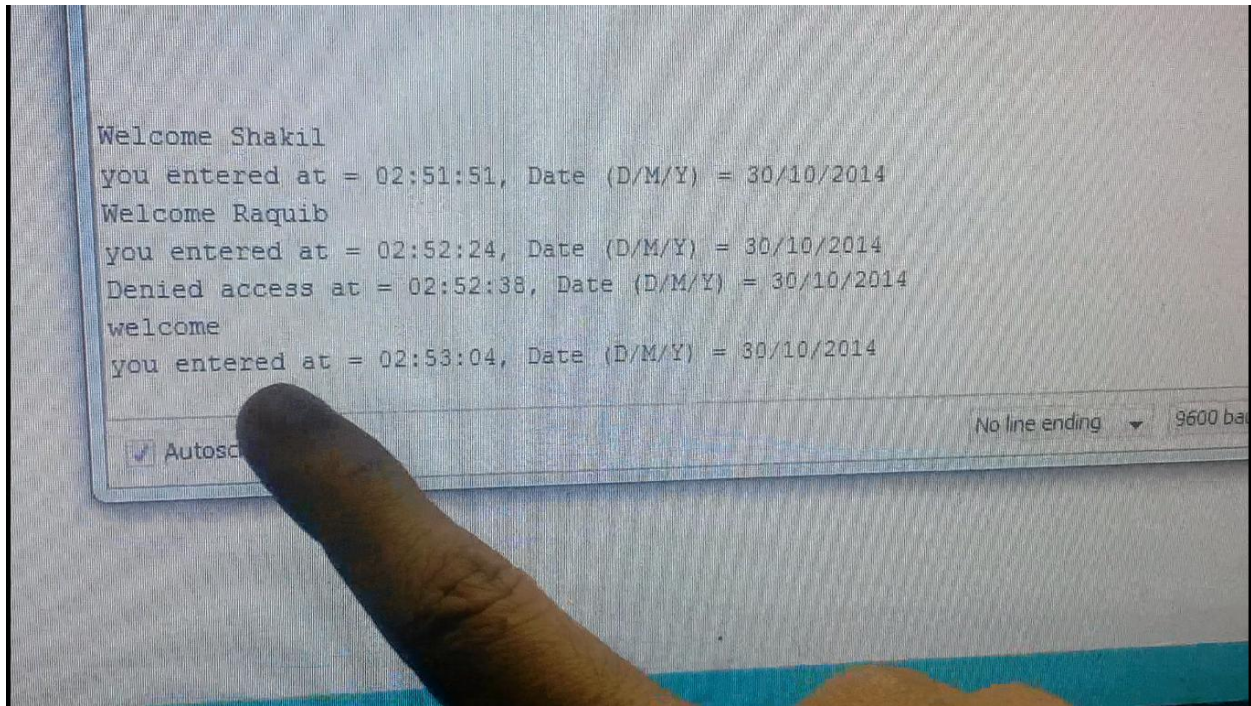
- Emergency password was pressed in this section and the password was 99.



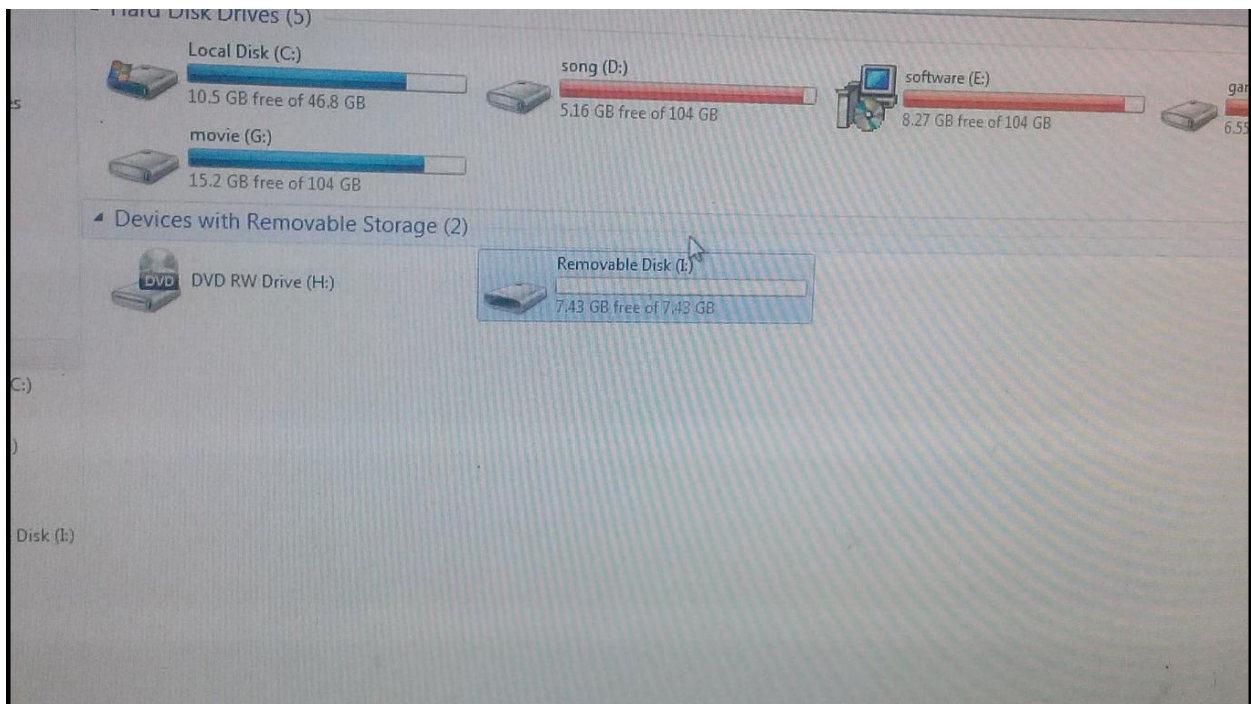
- In the LCD only "Welcome" was displayed and the blue LED turned on in the admin domain with a buzzer alarm.



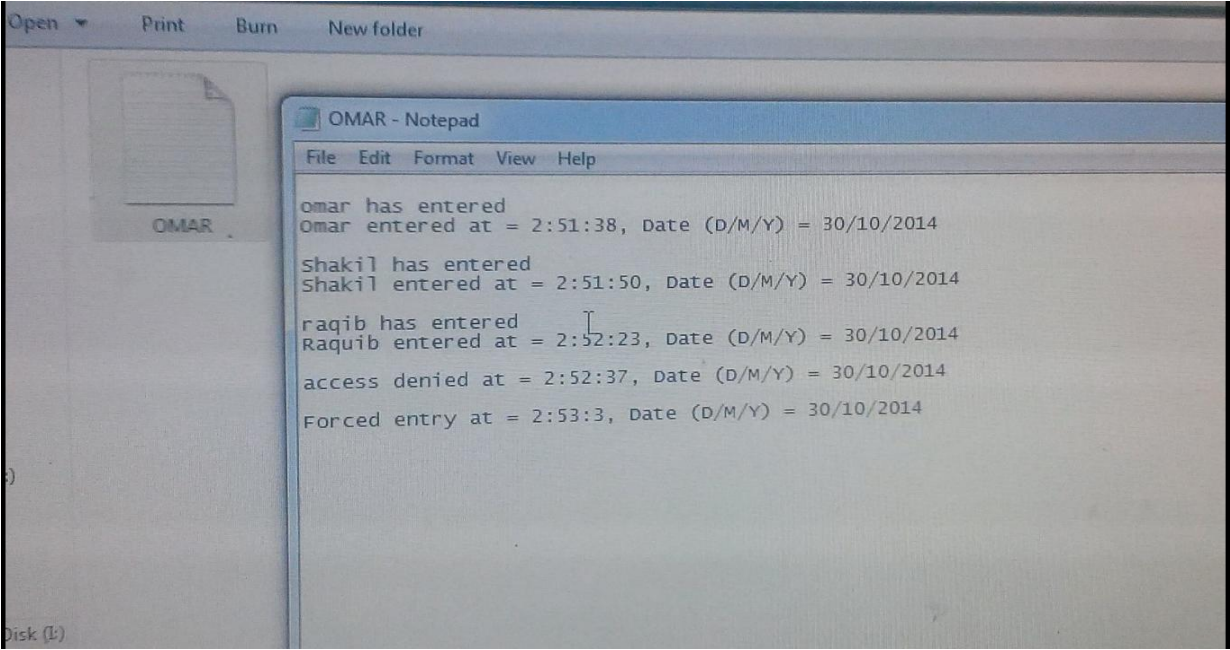
- From the COM port we could see this occurrence.



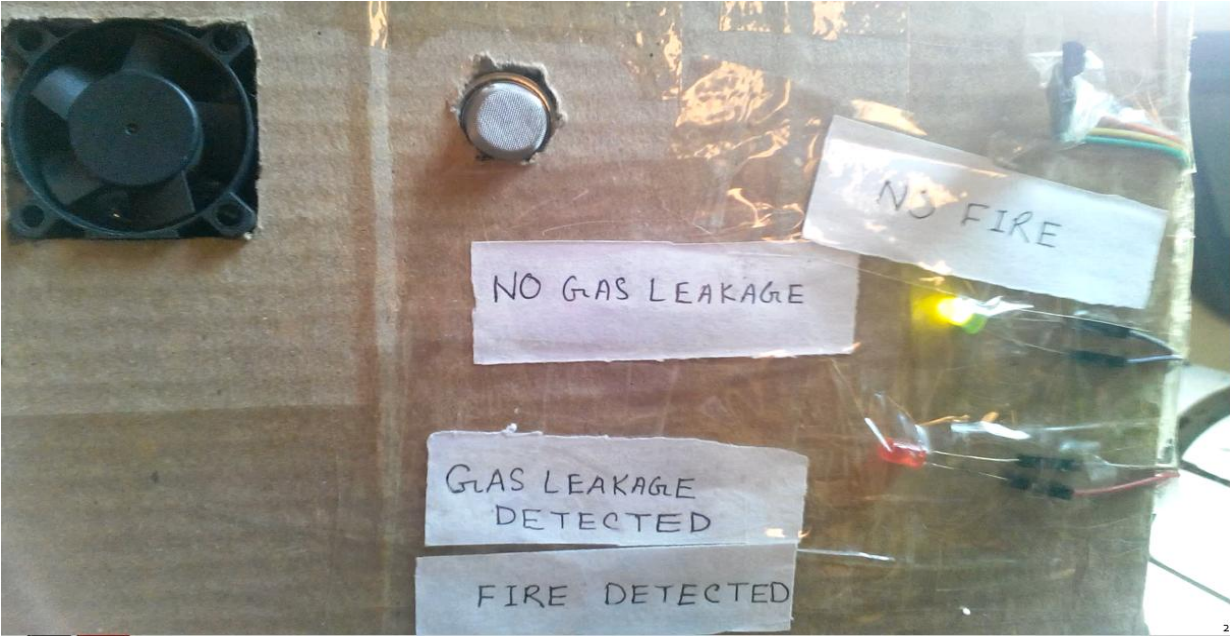
- After extracting the SD card we used a card reader to see the stored data in the memory storage device.



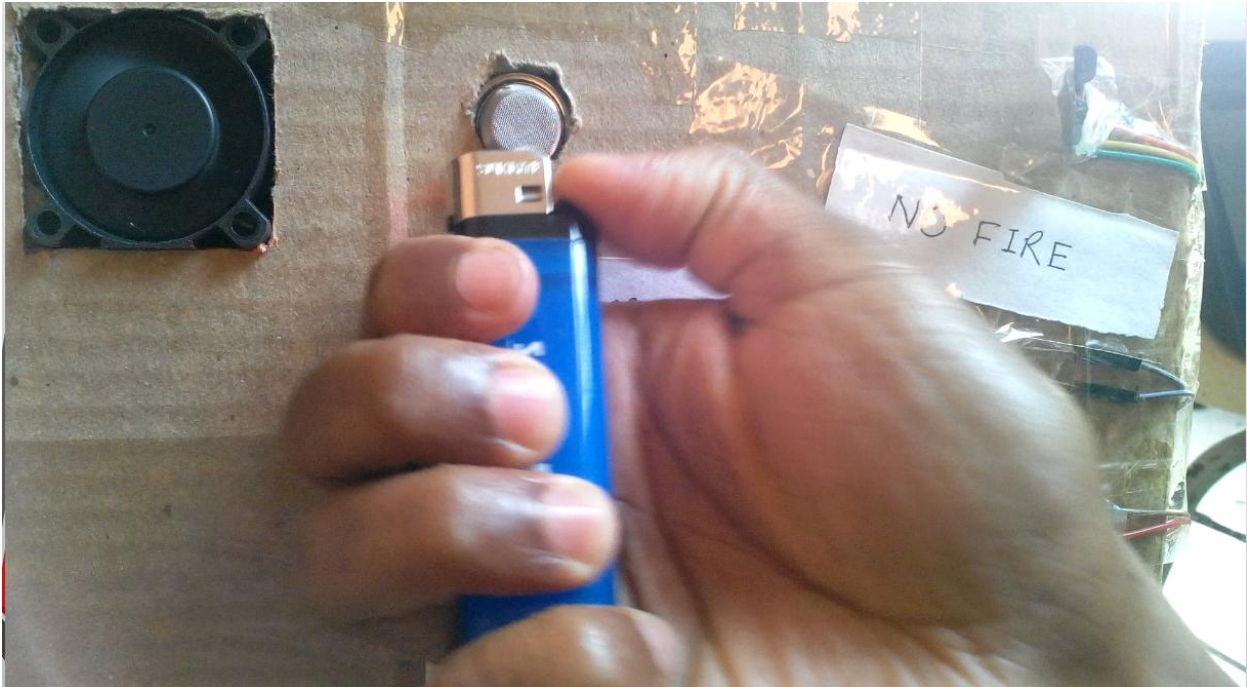
- Here we could see the stored data from the OMAR.txt file and noticeable part was that the forced entry time had been detected by the system and the data was written accordingly.



- The Fire detection and Gas leakage detection prototype is shown here. The MQ-2 gas sensor detects the presence of combustible gas. There was no leaked gas so the Green light was on and the exhaust fan was off. Also there was no fire so the fire alarm was off.



- In the following picture we leaked some gas to the Gas sensor intentionally to see the consequence. Immediately after detecting the gas the exhaust fan started to rotate.



- The presence of the leaked gas could be also seen from the red LED. At the same time a buzzer alarm was activated to show the presence of the leaked gas.



- The temperature of the LM-35 sensor was increased by using a flame. As the temperature increased above 50°C the red LED was turned on and also a buzzer alarm was activated to represent that fire was detected. Since there was no leaked gas so the green LED was on. It is shown in the following picture.



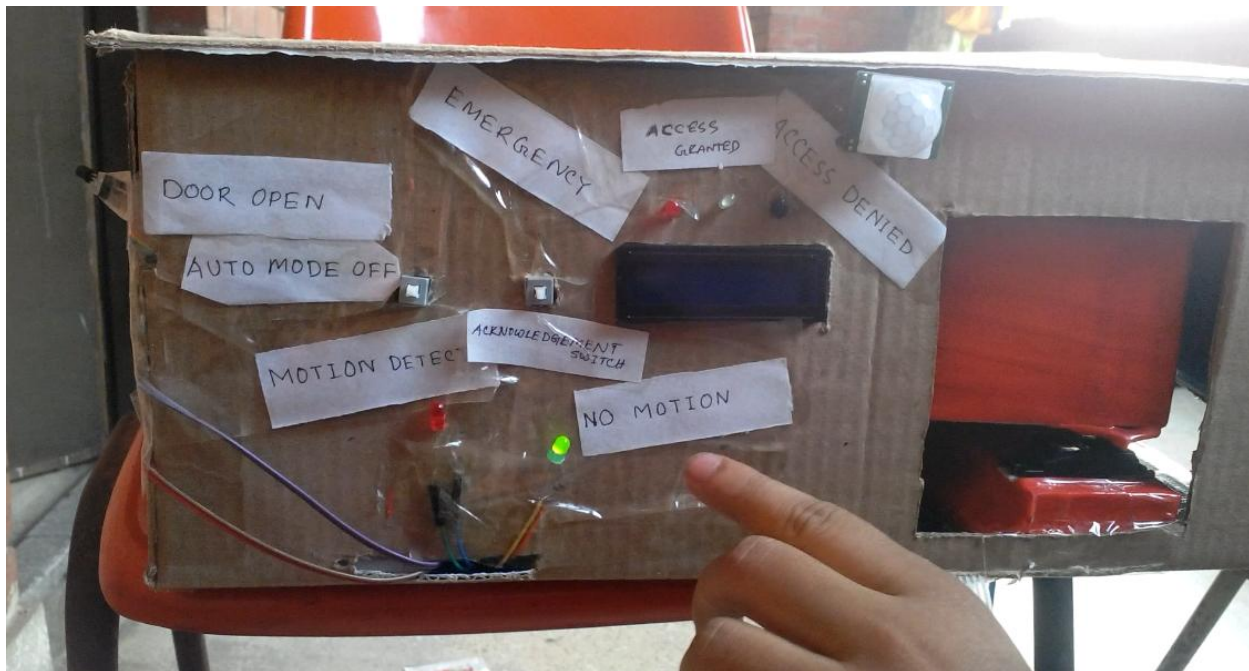
- In the following picture we used some smoke which is detected by the MQ-2 sensor. It represented the presence of fire and the fire alarm was activated and also the red LED was turned on.



- In the following figure the motion detection system is shown.



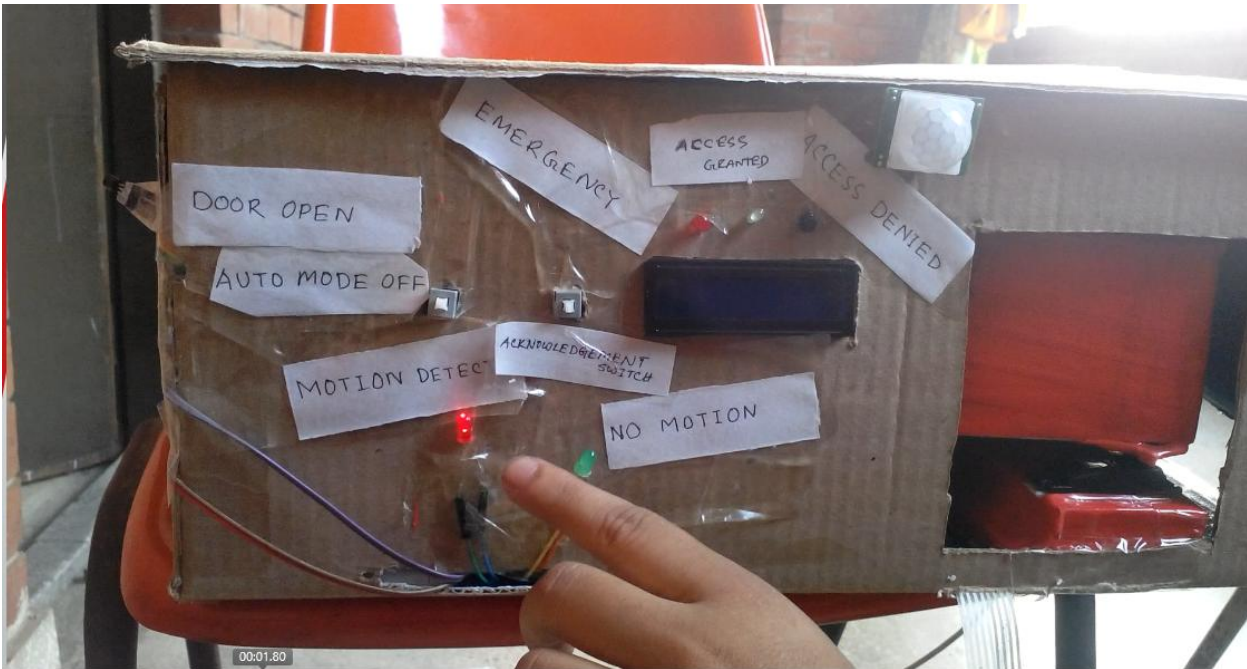
- The PIR sensor detects the motion of the human body. Since no motion was detected the Green LED was turned on which is shown here.



- The hand was moving in front of the PIR sensor. As PIR detects only motion of human body so in this case it could detect the hand movement.

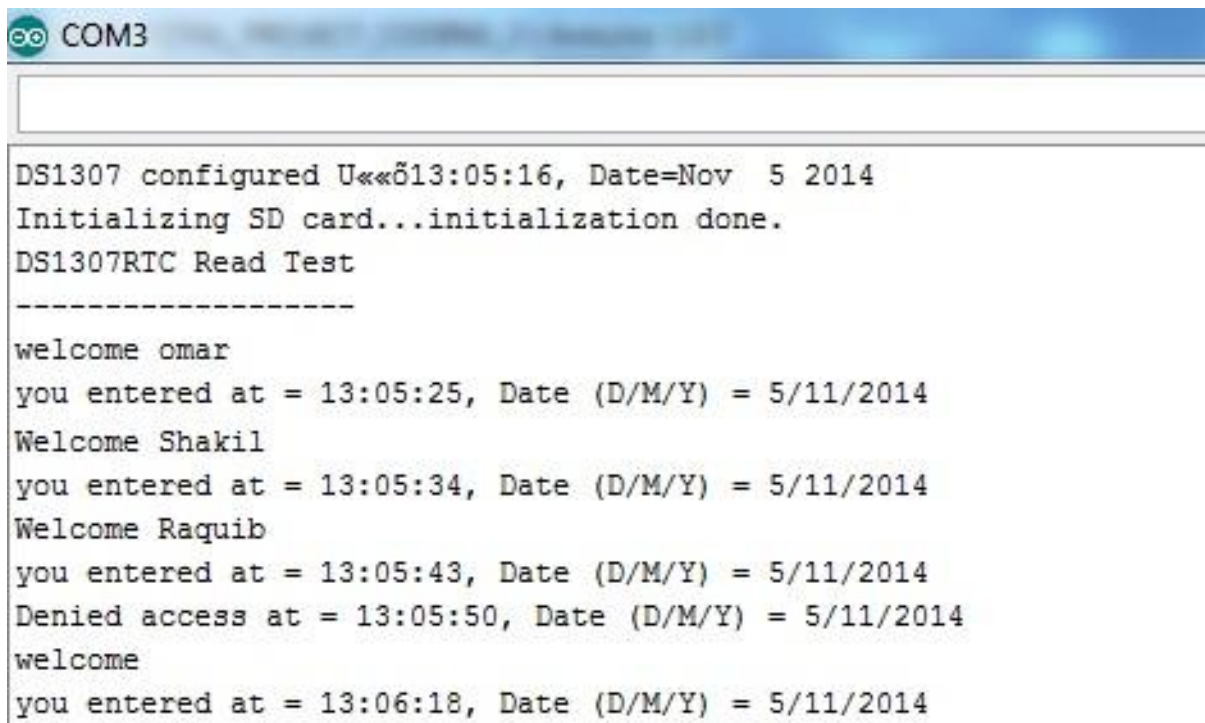


- As the hand stayed in front of the sensor more than 30 seconds the red LED was turned on.



## 4.2 Data Analysis

For the password protected locked door, the display device will show the information about the entered person and the exact time of entrance. From figure 4.1 it can be seen that the first person entered at 13:05:25 at 5<sup>th</sup> Nov, 2014 & the user name is Omar. The second person entered at 13:05:34 at 5<sup>th</sup> Nov, 2014 & the user name is Shakil. The third person entered at 13:05:43 at 5<sup>th</sup> Nov, 2014 & the user name is Raquib. Next time a person is force fully entered at 13:06:18 at 5<sup>th</sup> Nov, 2014 & the user name is not available this time.



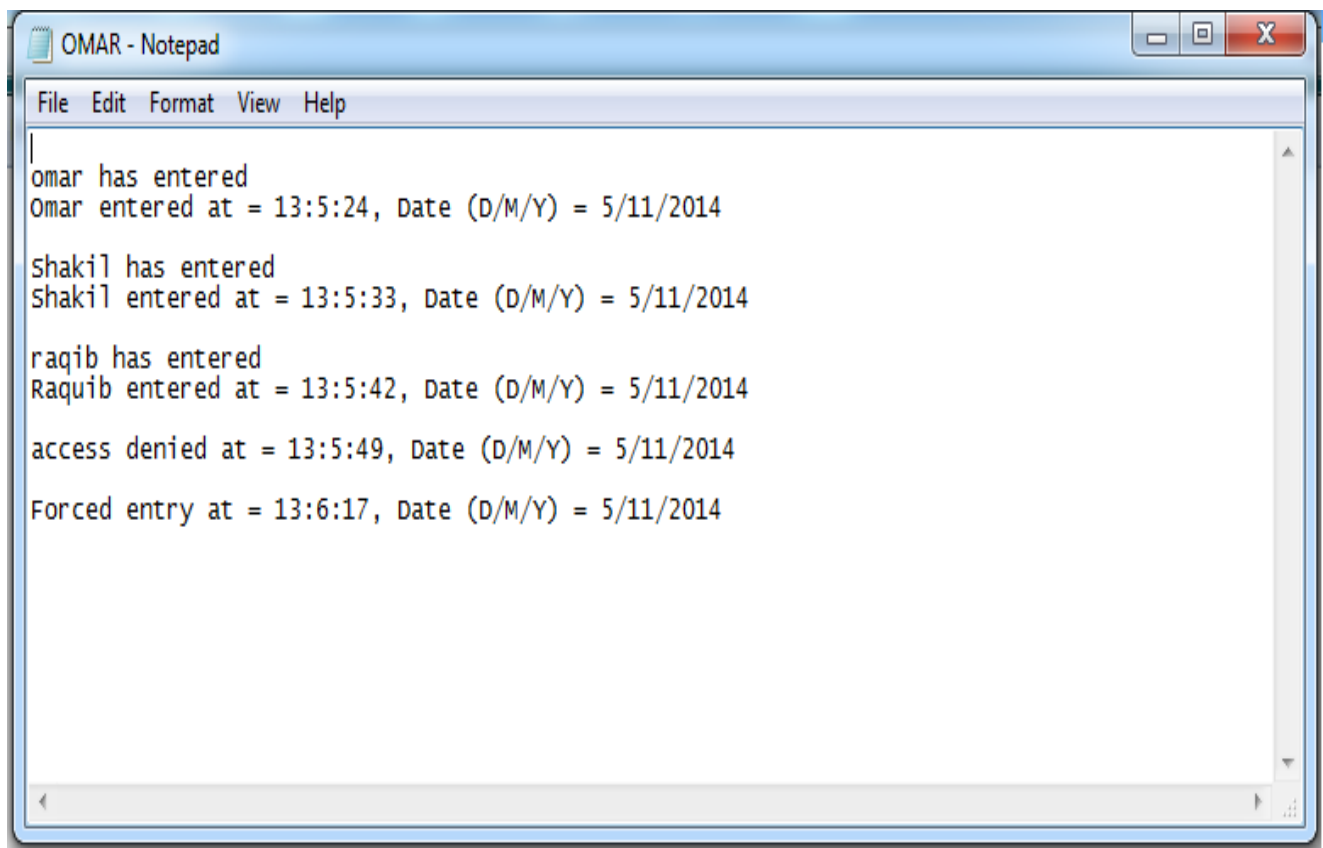
```
COM3
DS1307 configured U=13:05:16, Date=Nov 5 2014
Initializing SD card...initialization done.
DS1307RTC Read Test
-----
welcome omar
you entered at = 13:05:25, Date (D/M/Y) = 5/11/2014
Welcome Shakil
you entered at = 13:05:34, Date (D/M/Y) = 5/11/2014
Welcome Raquib
you entered at = 13:05:43, Date (D/M/Y) = 5/11/2014
Denied access at = 13:05:50, Date (D/M/Y) = 5/11/2014
welcome
you entered at = 13:06:18, Date (D/M/Y) = 5/11/2014
```

Figure 4.1: Displayed data at COM port



The total information will be saved to a memory storage device and can be extracted later if required. If the owner of the device wants to find out the time of entrance of each individual and the time of denied entrance, he would be able to see it from memory storage device.

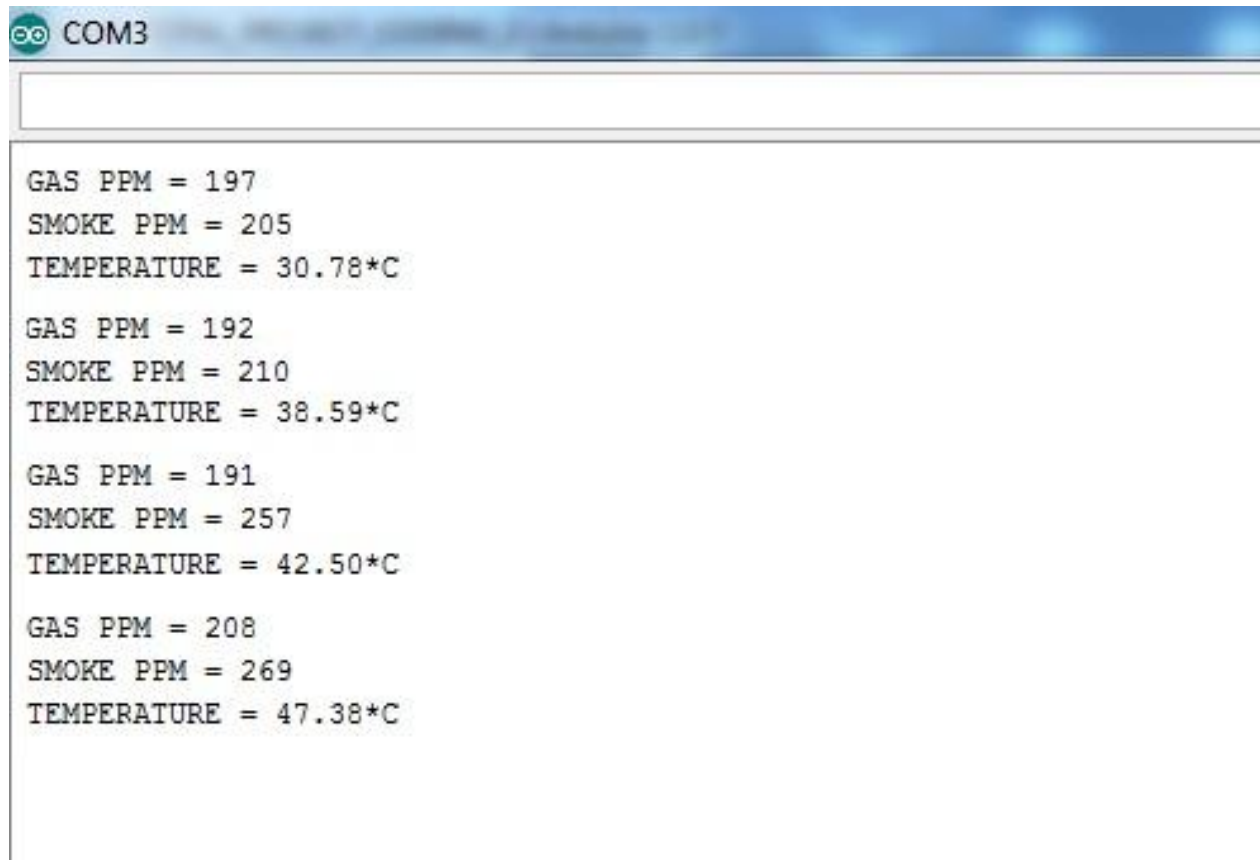
Also whether any forced entry occurred or not he would be able to see it from memory storage device. In Figure 4.2 there is slight delay from the actual data due to looping delay of the programme for very few seconds. So this delay is within the tolerance limit.



**Figure 4.2 extracted data from memory card**

The fire detection system detects the occurrence of fire based on the smoke ppm level and heat level. The leakage gas detector system detects the leaked gas. In figure 4.3 the simulating results are for normal condition i.e when the gas or smoke ppm is less than 400 ppm and the temperature is below 50°C. Normal environment ppm is 200-250 ppm and the temperature in the summer in bangladesh is max about 40-45°C.

But if the situation deviates from the normal operating condition such as if the ppm changes or increases more than 400ppm or temperature rises above 50°C then our gas detector and temperature sensor will react respectively and the assigned buzzer, LED will also react accordingly [12].



```
COM3  
  
GAS PPM = 197  
SMOKE PPM = 205  
TEMPERATURE = 30.78*C  
  
GAS PPM = 192  
SMOKE PPM = 210  
TEMPERATURE = 38.59*C  
  
GAS PPM = 191  
SMOKE PPM = 257  
TEMPERATURE = 42.50*C  
  
GAS PPM = 208  
SMOKE PPM = 269  
TEMPERATURE = 47.38*C
```

**Figure 4.3 Simulating results for fire and leakage gas detection**

### 4.3 Cost analysis

Our security device was designed for supplying the security service to the middle class people of our country. So the device was build considering the scenario around the household area of Bangladesh. But the security system of ADT was designed for the scenario of western countries. There system provides more facilities than our system. But we provide only the basic safety and security purpose facility.

In Table 4 we showed the cost comparison of our security system and ADT security system. Here we only considered the facilities that we have provided.

<b>Total cost of ADT security device = 450\$ (36000 BDT)</b>	<b>Total cost of our embedded security system</b>	
	Arduino mega	1300
	Arduino uno	1200
	PIR sensor module	600
	MQ-2 sensor	400
	RTC module	250
	Others	6750
	<b>Total cost</b>	<b>10500</b>

**Table 4: Cost comparison between ADT and our security system.**

Table 4 shows that our system price is lower than ADT (we took it as a sample).

# Chapter 5

## Conclusion & Future work

### 5.1 Conclusion

We wanted to design and analyze a wireless embedded safe and secure home system for Bangladeshi middle class people. We gave emphasize on the cost of the system. Our target was to set a home security system in a way such that it will be easily affordable for the middle class people of Bangladesh.

- **Advantages**

We can get some advantages from our device. The password protected door lock system is so well equipped that even if anyone is forced to enter into the room or forced to give the password can be easily modified. With the help of the storage device the total system information can be recaptured. The PIR sensor is specially tuned to detect human motion so that it can differentiate among other objects and human body and its coverage area is sufficient enough to do so. Fire detection system detects the fire by detecting the presence of smoke and with the temperature rise in the room and the leakage gas system efficiently detects the leaked gas on the condition of local environment of Bangladesh.

- **Limitations**

We have some drawbacks for the system as the PIR motion detector detects only the motion of a human body. But if a human is protected by a suit which is heat resistive so that it doesn't allow the infrared to penetrate. For the fire and leakage gas detection the range of sensors are limited so the number of the total sensor may be higher. We have estimated a sample area so the number of sensor depends on the location area. So the civil calculation will be needed there. For the password protected door locked system a continuous power supply will be needed. So if the power supply is not available, then the automated password system will not work properly and as well as the motion detector and the fire alarm and gas leakage sensors.

## **5.2 Future work**

- **Wireless Network implementation**

Wireless networking can be implemented so that the password can be given from the mobile phone anywhere within the range. So there is no need to stand in front of the door at the time of giving the password. So the device can be controlled from a distant place.

- **Implement fire extinguishing system**

Fire extinguishing system gives the fire alarm system a safer mode. In the fire extinguishing system whenever fire is detected, fire extinguisher will exert and will extinguish the fire.

- **Use of CC camera for monitoring**

A cc camera can be attached to the motion detection system so that if there is alarm for the security problem then it will automatically capture images in front of it and will save to the storage device.

# Appendix

## //Codes For The Home Security System

```
#include <SD.h> // library for SD card
#include <Password.h> // library for password
#include <DS1307RTC.h> // library for RTC module
#include <Time.h> // library for RTC module
#include <Wire.h> // library for communicating with I2C / TWI devices
#include <Keypad.h> // library for interfacing keypad
#include <LiquidCrystal.h> // library for interfacing LCD
#include <Servo.h> // library for interfacing servo motor
Servo myservo; // defining servo motor as myservo
/*****
 * Public Constants
 *****/
#define NOTE_B0 31
#define NOTE_C1 33
#define NOTE_CS1 35
#define NOTE_D1 37
#define NOTE_DS1 39
#define NOTE_E1 41
#define NOTE_F1 44
#define NOTE_FS1 46
#define NOTE_G1 49
#define NOTE_GS1 52
#define NOTE_A1 55
#define NOTE_AS1 58
#define NOTE_B1 62
#define NOTE_C2 65
#define NOTE_CS2 69
#define NOTE_D2 73
#define NOTE_DS2 78
#define NOTE_E2 82
#define NOTE_F2 87
#define NOTE_FS2 93
#define NOTE_G2 98
#define NOTE_GS2 104
#define NOTE_A2 110
#define NOTE_AS2 117
```

```
#define NOTE_B2 123
#define NOTE_C3 131
#define NOTE_CS3 139
#define NOTE_D3 147
#define NOTE_DS3 156
#define NOTE_E3 165
#define NOTE_F3 175
#define NOTE_FS3 185
#define NOTE_G3 196
#define NOTE_GS3 208
#define NOTE_A3 220
#define NOTE_AS3 233
#define NOTE_B3 247
#define NOTE_C4 262
#define NOTE_CS4 277
#define NOTE_D4 294
#define NOTE_DS4 311
#define NOTE_E4 330
#define NOTE_F4 349
#define NOTE_FS4 370
#define NOTE_G4 392
#define NOTE_GS4 415
#define NOTE_A4 440
#define NOTE_AS4 466
#define NOTE_B4 494
#define NOTE_C5 523
#define NOTE_CS5 554
#define NOTE_D5 587
#define NOTE_DS5 622
#define NOTE_E5 659
#define NOTE_F5 698
#define NOTE_FS5 740
#define NOTE_G5 784
#define NOTE_GS5 831
#define NOTE_A5 880
#define NOTE_AS5 932
#define NOTE_B5 988
#define NOTE_C6 1047
#define NOTE_CS6 1109
#define NOTE_D6 1175
#define NOTE_DS6 1245
#define NOTE_E6 1319
#define NOTE_F6 1397
#define NOTE_FS6 1480
#define NOTE_G6 1568
```

```

#define NOTE_GS6 1661
#define NOTE_A6 1760
#define NOTE_AS6 1865
#define NOTE_B6 1976
#define NOTE_C7 2093
#define NOTE_CS7 2217
#define NOTE_D7 2349
#define NOTE_DS7 2489
#define NOTE_E7 2637
#define NOTE_F7 2794
#define NOTE_FS7 2960
#define NOTE_G7 3136
#define NOTE_GS7 3322
#define NOTE_A7 3520
#define NOTE_AS7 3729
#define NOTE_B7 3951
#define NOTE_C8 4186
#define NOTE_CS8 4435
#define NOTE_D8 4699
#define NOTE_DS8 4978

//connect the arduino pins to this lcd pins

// GND,VDD, pot----> 1,2,3
//LCD Pin 4 --> Arduino Pin 2
//LCD Pin 5 --> Arduino Pin 3
//LCD Pin 6 --> Arduino Pin 4
//LCD Pin 11 --> Arduino Pin 8
//LCD Pin 12 --> Arduino Pin 9
//LCD Pin 13 --> Arduino Pin 10
//LCD Pin 14 --> Arduino Pin 11

const char *monthName[12] = //defining the month name as char
{
  "Jan", "Feb", "Mar", "Apr", "May", "Jun",
  "Jul", "Aug", "Sep", "Oct", "Nov", "Dec"
};
tmElements_t tm; //for RTC purpose
unsigned long int starttime=0;
int melody_1[] =
{
  NOTE_DS8, NOTE_CS8,NOTE_AS7,NOTE_GS7, NOTE_FS7,NOTE_DS7,NOTE_CS7,0
}; // this section is for
creating the entry // sound

int noteDurations_1[] =
{
  4,8,8,4,4,4,4,4

```



```

}; // note durations: 4 = quarter note, 8 = eighth note, etc.:
int melody_2[] =
{
NOTE_DS7,NOTE_DS7,NOTE_DS7,NOTE_DS7, NOTE_DS7,NOTE_DS7, NOTE_DS7,NOTE_DS7
}; //this section is for creating the
// access denied sound

// note durations: 4 = quarter note, 8 = eighth note, etc.:
int noteDurations_2[] =
{
4,4,4,4,4,4,4,4
};
int interval = 8500;

/* in the following two lines several pins are defined as several purpose*/

int door = 0; int ds=43; int a=1;
int doorsw = 0; int dsw=41; int ledg = 36; int ledb = 38; int ledr=40; int piezo1 =42; int piezo2 =44; int piezo3=46;

LiquidCrystal lcd (2,3,4,9,10,11,12) ; // defined the connection sequence of Arduino pins with LCD
//chronologically

/*defining several passwords*/

Password password1 = Password( "12" );
Password password2 = Password( "14" );
Password password3 = Password( "16" );
Password password4 = Password( "99" );

const byte ROWS = 4; // Four rows
const byte COLS = 4; // Four columns

// Define the Keypad

char keys[ROWS][COLS] =
{
{'1','2','3','A'},
{'4','5','6','B'},
{'7','8','9','C'},
{'*','0','#','D'}
};

byte rowPins[ROWS] = {25, 27, 29, 31 }; // Connect keypad ROW0, ROW1, ROW2 and ROW3 to these
Arduino pins.
byte colPins[COLS] = {33, 35, 37, 39 }; // Connect keypad COL0, COL1,COL2 and COL3 to these
Arduino pins.

// Create the Keypad
Keypad keypad = Keypad( makeKeymap(keys), rowPins, colPins, ROWS, COLS );
//Creat file
File myFile;

```

```

void setup()
{
  {
    bool parse=false;
    bool config=false;
    // get the date and time the compiler was run
    if (getDate(__DATE__) && getTime(__TIME__))
      {
        parse = true;
        // and configure the RTC with this info
        if (RTC.write(tm))
          {
            config = true;
          }
      }
    Serial.begin(9600);
    while (!Serial) ; // wait for Arduino Serial Monitor
    delay(200);
    if (parse && config)
      {
        Serial.print("DS1307 configured Time=");
        Serial.print(__TIME__);
        Serial.print(", Date=");
        Serial.println(__DATE__);
      }
    else if (parse)
      {
        Serial.println("DS1307 Communication Error ");
        Serial.println("Please check your circuitry");
      }
    else
      {
        Serial.print("Could not parse info from the compiler, Time=\"");
        Serial.print(__TIME__);
        Serial.print("\", Date=\"");
        Serial.print(__DATE__);
        Serial.println("\");
      }

    //define the pin operation

    pinMode(piezo1,OUTPUT);
    pinMode(piezo2,OUTPUT);
    pinMode(piezo3,OUTPUT);
    digitalWrite(piezo1,LOW);
    digitalWrite(piezo2,LOW);
    pinMode(piezo3,OUTPUT);
    pinMode(ds,INPUT);
    pinMode(dsw,INPUT);
    pinMode(ledg,OUTPUT);
    pinMode(ledb,OUTPUT);
    pinMode(ledr,OUTPUT);

```

```

myservo.attach(23);
lcd.begin(16, 2);
lcd.print("Password:");
Serial.begin(9600);
myservo.write(0);
digitalWrite(ledg,LOW);
digitalWrite(ledb,LOW);
digitalWrite(ledr,LOW);

//add an event listener for this keypad
keypad.addEventListener(keypadEvent);

//for SD card purpose

// Open serial communications and wait for port to open:

Serial.begin(9600);
  while (!Serial)
  {
    ; // wait for serial port to connect. Needed for Leonardo only
  }
Serial.print("Initializing SD card...");

// On the Ethernet Shield, CS is pin 4. It's set as an output by default.
// Note that even if it's not used as the CS pin, the hardware SS pin
// (10 on most Arduino boards, 53 on the Mega) must be left as an output
// or the SD library functions will not work.

pinMode(53, OUTPUT);

///The communication between the microcontroller and the SD card uses SPI,
//which takes place on digital pins 11, 12, and 13 (on most Arduino boards) or 50, 51, and 52 (Arduino Mega).
//Additionally, another pin must be used to select the SD card.
//This can be the hardware SS pin - pin 10 (on most Arduino boards) or pin 53 (on the Mega)
//- or another pin specified in the call to SD.begin().
//Note that even if you don't use the hardware SS pin, it must be left as an output or the SD library won't work.
//Different boards use different pins for this functionality, so be sure you've selected the correct pin in SD.begin().

//The Arduino SPI pins are:

//SPI   Uno   Mega
//SS    10    53
//MOSI  11    51
//MISO  12    50
//SCK   13    52

if (!SD.begin(53))
  {
  Serial.println("initialization failed!");
  return;
  }
Serial.println("initialization done.");

```

```

delay(1500);
Serial.println("DS1307RTC Read Test");
Serial.println("-----");
// open the file. note that only one file can be open at a time,
// so you have to close this one before opening another.
    }
    }
void loop()
{
//section A.1

tmElements_t tm; //RTC purpose
//for the door lock purpose
doorsw=digitalRead(dsw); //checking whether the auto mode is on,it's high means auto mode is off
and //
//low means auto mode is on so door will now operate

if (doorsw==HIGH)
    {
    lcd.clear();
    door_open();
    lcd.print("Door Open");
        while (doorsw==HIGH)
            {
                doorsw=digitalRead(dsw);
                door=digitalRead(ds);
            }

        while (door!=HIGH)
            {
                door=digitalRead(ds);
            }

    delay(1000);
    door_close();
    pass_reset();
    }

else //when the auto mode is on and the door is closed then
//the device will demand the correct password
    {
        keypad.getKey();
        door=digitalRead(ds);
    }
}

//take care of some special events
void keypadEvent(KeypadEvent eKey)
{
    switch (keypad.getState())
    {
        case PRESSED:
            lcd.print(eKey);
    }
}

```

```

switch (eKey)
{
    case '*': guessPassword(); break;
    case '#': pass_reset(); break;

    default:
        password1.append(eKey);
        password2.append (eKey);
        password3.append (eKey);
        password4.append (eKey);
}
}
}

```

## //section A.2

```

void guessPassword()

{

    if (password1.evaluate())

    {
        lcd.clear();
        lcd.print("Welcome Omar");
        door_open();
        door=digitalRead(ds);
        starttime=millis();
        while (door==HIGH && a==0)
        {
            Serial.println(door);
            If(starttime+interval<=millis())
            {
                a=1;
            }
            door=digitalRead(ds);
        }
        while (door!=HIGH && a==0)
        {
            door=digitalRead(ds);
        }
        starttime=millis();
        while(starttime+1500>millis())
        {
        }
        door_close();
        pass_reset();

//sd card datalogging

myFile = SD.open("omar.txt", FILE_WRITE);

```

```

    if (myFile)
    {
        Serial.print("welcome omar ");
        Serial.println();
        myFile.println(" ");
        myFile.println("omar has entered ");

//For RTC purpose
tmElements_t tm;

//writing the entrance time into the sd card

        if (RTC.read(tm))
        {
            myFile.print("Omar entered at = ");
            myFile.print(tm.Hour);
            myFile.write(':');
            myFile.print(tm.Minute);
            myFile.write(':');
            myFile.print(tm.Second);
            myFile.print(", Date (D/M/Y) = ");
            myFile.print(tm.Day);
            myFile.write('/');
            myFile.print(tm.Month);
            myFile.write('/');
            myFile.print(tmYearToCalendar(tm.Year));
            myFile.println();
        }
else
    {
        if (RTC.chipPresent())
        {
            myFile.println("The DS1307 is stopped. Please run the SetTime");
            myFile.println("example to initialize the time and begin running.");
            myFile.println();
        }
else

        {
            myFile.println("DS1307 read error! Please check the circuitry.");
            myFile.println();
        }

        delay(9000);
    }
        delay(1000);
// RTC segment; display the data into the serial port(com port)
        if (RTC.read(tm))
        {
            Serial.print("you entered at = ");
            print2digits(tm.Hour);

```

```

        Serial.write(':');
        print2digits(tm.Minute);
        Serial.write(':');
        print2digits(tm.Second);
        Serial.print(", Date (D/M/Y) = ");
        Serial.print(tm.Day);
        Serial.write('/');
        Serial.print(tm.Month);
        Serial.write('/');
        Serial.print(tm.YearToCalendar(tm.Year));
        Serial.println();
    }
else
    {
        if (RTC.chipPresent())
            {
                Serial.println("The DS1307 is stopped. Please run the SetTime");
                Serial.println("example to initialize the time and begin running.");
                Serial.println();
            }
        else
            {
                Serial.println("DS1307 read error! Please check the circuitry.");
                Serial.println();
            }
        delay(9000);
    }
delay(1000);

// close the file:
myFile.close();
}
else
    {
        // if the file didn't open, print an error:
        Serial.println("error opening omar.txt");
    }
}

else if(password2.evaluate())
    {
        lcd.clear();
        lcd.print("Welcome Shakil");
        door_open();
        door=digitalRead(ds);
        starttime=millis();

        while (door==HIGH && a==0)
            {
                Serial.println(door);
                if(starttime+interval<=millis())
                    {
                        a=1;

```

```

    }
door=digitalRead(ds);
    }
    while (door!=HIGH && a==0)
        {
            door=digitalRead(ds);
        }
starttime=millis();
while(starttime+1500>millis())
    {
    }
door_close();
pass_reset();
//sd card datalogging
myFile = SD.open("omar.txt", FILE_WRITE);
    if (myFile)
        {
            Serial.print("Welcome Shakil ");
            Serial.println();
            myFile.println();
            myFile.println("Shakil has entered ");

//RTC purpose
tmElements_t tm;

//writing the entrance time into the sd card
if (RTC.read(tm))
    {
        myFile.print("Shakil entered at = ");
        myFile.print(tm.Hour);
        myFile.write(':');
        myFile.print(tm.Minute);
        myFile.write(':');
        myFile.print(tm.Second);
        myFile.print(", Date (D/M/Y) = ");
        myFile.print(tm.Day);
        myFile.write('/');
        myFile.print(tm.Month);
        myFile.write('/');
        myFile.print(tm.YearToCalendar(tm.Year));
        myFile.println();
    }

else
    {
        if (RTC.chipPresent())
            {
                myFile.println("The DS1307 is stopped. Please run the SetTime");
                myFile.println("example to initialize the time and begin running.");
                myFile.println();
            }
        else

```



```

        {
            myFile.println("DS1307 read error! Please check the circuitry.");
            myFile.println();
        }
    delay(9000);
    }
delay(1000);

// RTC segment; display the data into the serial port (com port)

if (RTC.read(tm))
    {
        Serial.print("you entered at = ");
        print2digits(tm.Hour);
        Serial.write(':');
        print2digits(tm.Minute);
        Serial.write(':');
        print2digits(tm.Second);
        Serial.print(", Date (D/M/Y) = ");
        Serial.print(tm.Day);
        Serial.write('/');
        Serial.print(tm.Month);
        Serial.write('/');
        Serial.print(tmYearToCalendar(tm.Year));
        Serial.println();
    }
else
    {
        if (RTC.chipPresent())
            {
                Serial.println("The DS1307 is stopped. Please run the SetTime");
                Serial.println("example to initialize the time and begin running.");
                Serial.println();
            }
        else
            {
                Serial.println("DS1307 read error! Please check the circuitry.");
                Serial.println();
            }
        delay(9000);
    }
delay(1000);

// close the file:
myFile.close();
}
else
    {
        // if the file didn't open, print an error:
        Serial.println("error opening omar.txt");
    }
}

```

```

else if(password3.evaluate())
    {
        lcd.clear();
        lcd.print("Welcome Raquib ");
        door_open();
        door=digitalRead(ds);
        starttime=millis();
        while (door==HIGH && a==0)
            {
                Serial.println(door);
                if(starttime+interval<=millis())
                    {
                        a=1;
                    }
                door=digitalRead(ds);
            }
        while (door!=HIGH && a==0)
            {
                door=digitalRead(ds);
            }
        starttime=millis();
        while(starttime+1500>millis())
            {
            }
        door_close();
        pass_reset();
        //sd card datalogging
        myFile = SD.open("omar.txt", FILE_WRITE);
        if (myFile)
            {
                Serial.print("Welcome Raquib ");
                Serial.println();
                myFile.println();
                myFile.println("raqib has entered ");
            }
    }

```

//RTC Purpose

tmElements\_t tm;

//writing the entrance time into the sd card

if (RTC.read(tm))

```

    {
        myFile.print("Raquib entered at = ");
        myFile.print(tm.Hour);
        myFile.write(':');
        myFile.print(tm.Minute);
        myFile.write(':');
        myFile.print(tm.Second);
        myFile.print(", Date (D/M/Y) = ");
        myFile.print(tm.Day);
        myFile.write('/');
        myFile.print(tm.Month);
    }

```

```

        myFile.write('/');
        myFile.print(tmYearToCalendar(tm.Year));
        myFile.println();
    }
else
    {
        if (RTC.chipPresent())
            {
                myFile.println("The DS1307 is stopped. Please run the SetTime");
                myFile.println("example to initialize the time and begin running.");
                myFile.println();
            }
        else
            {
                myFile.println("DS1307 read error! Please check the circuitry.");
                myFile.println();
            }
        delay(9000);
    }

delay(1000);

// RTC segment;display the data into the serial port(com port)

if (RTC.read(tm))
    {
        Serial.print("you entered at = ");
        print2digits(tm.Hour);
        Serial.write(':');
        print2digits(tm.Minute);
        Serial.write(':');
        print2digits(tm.Second);
        Serial.print(", Date (D/M/Y) = ");
        Serial.print(tm.Day);
        Serial.write('/');
        Serial.print(tm.Month);
        Serial.write('/');
        Serial.print(tmYearToCalendar(tm.Year));
        Serial.println();
    }
else
    {
        if (RTC.chipPresent())
            {
                Serial.println("The DS1307 is stopped. Please run the SetTime");
                Serial.println("example to initialize the time and begin running.");
                Serial.println();
            }
        else
            {
                Serial.println("DS1307 read error! Please check the circuitry.");
            }
    }

```

```

        Serial.println();
        }
        delay(9000);
    }
    delay(1000);

    // close the file:
    myFile.close();

}

else
{
    // if the file didn't open, print an error:
    Serial.println("error opening omar.txt");
}
}

else if (password4.evaluate())
{
    lcd.clear();
    lcd.print(" Welcome ");
    door_open_2();
    door=digitalRead(ds);
    starttime=millis();
    while (door==HIGH && a==0)
    {
        Serial.println(door);
        if(starttime+interval<=millis())
        {
            a=1;
        }
        door=digitalRead(ds);
    }
    while (door!=HIGH && a==0)
    {
        door=digitalRead(ds);
    }
    starttime=millis();
    while(starttime+1500>millis())
    {
    }

    door_close();
    pass_reset();

}

else
{
    lcd.clear();
    lcd.print("Access Denied");
    digitalWrite(ledb,HIGH);
}

```

```

    for (int thisNote = 0; thisNote < 8; thisNote++)
    {
        int noteDuration_2 = 1000/noteDurations_2[thisNote];
        tone(piezo2, melody_2[thisNote],noteDuration_2);

// to distinguish the notes, set a minimum time between them.
// the note's duration + 30% seems to work well:
        int pauseBetweenNotes = noteDuration_2* 1;
        delay(pauseBetweenNotes);
    }
starttime=millis();
    while(starttime+1000>millis())
    {
    }
    pass_reset();
    digitalWrite(ledb,LOW);
    myFile = SD.open("omar.txt", FILE_WRITE);
    if (myFile)
    {
myFile.println();

//For RTC Purpose
tmElements_t tm;

//writing the entrance time into the sd card

        if (RTC.read(tm))
        {
            myFile.print("access denied at = ");
            myFile.print(tm.Hour);
            myFile.write(':');
            myFile.print(tm.Minute);
            myFile.write(':');
            myFile.print(tm.Second);
            myFile.print(", Date (D/M/Y) = ");
            myFile.print(tm.Day);
            myFile.write('/');
            myFile.print(tm.Month);
            myFile.write('/');
            myFile.print(tm.YearToCalendar(tm.Year));
            myFile.println();
        }

else
    {
        if (RTC.chipPresent())
        {
            myFile.println("The DS1307 is stopped. Please run the SetTime");
            myFile.println("example to initialize the time and begin running.");
            myFile.println();
        }
        else

```

```

        {
            myFile.println("DS1307 read error! Please check the circuitry.");
            myFile.println();
        }
        delay(9000);
    }
    delay(1000);
    // RTC segment; display the data into the serial port(com port)

    if (RTC.read(tm))
    {
        Serial.print("Denied access at = ");
        print2digits(tm.Hour);
        Serial.write(':');
        print2digits(tm.Minute);
        Serial.write(':');
        print2digits(tm.Second);
        Serial.print(" , Date (D/M/Y) = ");
        Serial.print(tm.Day);
        Serial.write('/');
        Serial.print(tm.Month);
        Serial.write('/');
        Serial.print(tmYearToCalendar(tm.Year));
        Serial.println();
    }
    else
    {
        if (RTC.chipPresent())
        {
            Serial.println("The DS1307 is stopped. Please run the SetTime");
            Serial.println("example to initialize the time and begin running.");
            Serial.println();
        }
        else
        {
            Serial.println("DS1307 read error! Please check the circuitry.");
            Serial.println();
        }
        delay(9000);
    }
    delay(1000);

    myFile.close();
}
else
{
    // if the file didn't open, print an error:
    Serial.println("error opening omar.txt");
}
}
}

```

```

void pass_reset()
{
  lcd.clear();
  lcd.print("Password:");
  password1.reset();
  password2.reset();
  password3.reset();
  password4.reset();

  a=0;
}

void door_open()
{
  digitalWrite(ledg,HIGH);
  myservo.write(90);
  for (int thisNote = 0; thisNote < 8; thisNote++)
  {
    int noteDuration_1 = 1000/noteDurations_1[thisNote];
    tone(piezo1, melody_1[thisNote],noteDuration_1);
    // to distinguish the notes, set a minimum time between them.
    // the note's duration + 30% seems to work well:
    int pauseBetweenNotes = noteDuration_1* 1.30;
    delay(pauseBetweenNotes);
    noTone(piezo1);
  }
}

void door_open_2()
{
  digitalWrite(ledg,HIGH);
  myservo.write(90);
  delay(900);
  digitalWrite(ledg,LOW);
  //sd card datalogging

  myFile = SD.open("omar.txt", FILE_WRITE);
  if (myFile)
  {
    Serial.print("welcome ");
    Serial.println();
    myFile.println(" ");

//For RTC purpose

    tmElements_t tm;

//writing the entrance time into the sd card

    if (RTC.read(tm))

```

```

    {
    myFile.print("Forced entry at = ");
    myFile.print(tm.Hour);
    myFile.write(':');
    myFile.print(tm.Minute);
    myFile.write(':');
    myFile.print(tm.Second);
    myFile.print(", Date (D/M/Y) = ");
    myFile.print(tm.Day);
    myFile.write('/');
    myFile.print(tm.Month);
    myFile.write('/');
    myFile.print(tmYearToCalendar(tm.Year));
    myFile.println();
    }

else

    {
        if (RTC.chipPresent())
        {
            myFile.println("The DS1307 is stopped. Please run the SetTime");
            myFile.println("example to initialize the time and begin running.");
            myFile.println();
        }
        else
        {
            myFile.println("DS1307 read error! Please check the circuitry.");
            myFile.println();
        }
        delay(9000);
    }
    delay(1000);

// RTC segment; display the data into the serial port (com port)
if (RTC.read(tm))
{
    Serial.print("you entered at = ");
    print2digits(tm.Hour);
    Serial.write(':');
    print2digits(tm.Minute);
    Serial.write(':');
    print2digits(tm.Second);
    Serial.print(", Date (D/M/Y) = ");
    Serial.print(tm.Day);
    Serial.write('/');
    Serial.print(tm.Month);
    Serial.write('/');
    Serial.print(tmYearToCalendar(tm.Year));
    Serial.println();
}

```



```

else
{
    if (RTC.chipPresent())
    {
        Serial.println("The DS1307 is stopped. Please run the SetTime");
        Serial.println("example to initialize the time and begin running.");
        Serial.println();
    }
    else
    {
        Serial.println("DS1307 read error! Please check the circuitry.");
        Serial.println();
    }
    delay(9000);
}
delay(1000);

// close the file:
myFile.close();
}
else
{
    // if the file didn't open, print an error:
    Serial.println("error opening omar.txt");
}
digitalWrite(ledr,HIGH);
for (int thisNote = 0; thisNote < 8; thisNote++)
{
    int noteDuration_1 = 1000/noteDurations_1[thisNote];
    tone(piezo1, melody_1[thisNote],noteDuration_1);
    // to distinguish the notes, set a minimum time between them.
    // the note's duration + 30% seems to work well:
    int pauseBetweenNotes = noteDuration_1 * 1.30;
    delay(pauseBetweenNotes);
    noTone(piezo1);
    //meanwhile in the admin domain
}
digitalWrite(ledr,HIGH);

for (int thisNote = 0; thisNote < 8; thisNote++)
{
    // int noteDuration_2 = 1000/noteDurations_2[thisNote];
    // tone(piezo3, melody_2[thisNote],noteDuration_2);
    int noteDuration_2 = 10000;
    tone(piezo3, melody_2[thisNote],noteDuration_2);
    // to distinguish the notes, set a minimum time between them.
    // the note's duration + 30% seems to work well:
    int pauseBetweenNotes = noteDuration_2 * 1;
    delay(pauseBetweenNotes);
// noTone(piezo3);
}
}

```

```

void door_close()
{
    digitalWrite(ledg,LOW);
    myservo.write(0);
}
void print2digits(int number)
{
    if (number >= 0 && number < 10)
        {
            Serial.write('0');
        }
    Serial.print(number);
}
bool getTime(const char *str)
{
    int Hour, Min, Sec;

    if (sscanf(str, "%d:%d:%d", &Hour, &Min, &Sec) != 3) return false;
    tm.Hour = Hour;
    tm.Minute = Min;
    tm.Second = Sec;
    return true;
}
bool getDate(const char *str)
{
    char Month[12];
    int Day, Year;
    uint8_t monthIndex;
    if (sscanf(str, "%s %d %d", Month, &Day, &Year) != 3)
        return false;
    for (monthIndex = 0; monthIndex < 12; monthIndex++)
        {
            if (strcmp(Month, monthName[monthIndex]) == 0)
                break;
        }
    if (monthIndex >= 12)
        return false;
    tm.Day = Day;
    tm.Month = monthIndex + 1;
    tm.Year = CalendarYrToTm(Year);
    return true;
}

```

## //For Fire And Gas Leakage Detection

```
int sensorValue;
int sensorValue1;
float temp;
float temp1;
// PIN_22=Exhaust_Fan, PIN_24=Gas_Buzzer, PIN_26=Fire_Buzzer,Pin_28=Green_LED_Gas,
Pin_30=Red_LED_Gas, PIN_32=GREEN_LED_Fire, PIN_34=RED_LED_Fire
//PIN_22 is connected to base of a PNP Transistor Which turns on the Exhaust fan when it gets High signal

void setup()
//FOR FIRE AND GAS LEAKAGE DETECTION
{
  pinMode(22,OUTPUT);
  pinMode(24,OUTPUT);
  pinMode(26,OUTPUT);
  pinMode(28,OUTPUT);    //Assigns Pins as output pin
  pinMode(30,OUTPUT);
  pinMode(32,OUTPUT);
  pinMode(34,OUTPUT);

  Serial.begin(9600);    // sets the serial port to 9600
}
void loop()
{
  {
```

### //Section A.3

```
    //FOR_LEAKAGE_GAS_DETECTION:
    sensorValue = analogRead(0);    // read analog input pin 0
    Serial.println(sensorValue, DEC); // prints the value read

    if (sensorValue < 400)    // if there is little or no gas detected display green lights
    {
      digitalWrite(22, LOW); //Exhaust Fan is turned off
      digitalWrite(30, LOW); //Red light is turned off
      digitalWrite(28, HIGH); //Green light is turned on
    }
    else    // if there is medium or high ppm gas detected display red lights
    {
      digitalWrite(28, LOW); //Green light is turned off
      digitalWrite(30, HIGH); //Red light is turned on
      digitalWrite(22, HIGH); // Turns on the Exhaust fan
      tone(24,250,100);    // Turns On the Alarm
    }
  }
  {
    //FOR_FIRE_DETECTION:
    {
```

## //Section A.4

```
        //FOR_SMOKE_DETECTOR:
        sensorValue1 = analogRead(2);    // read analog input pin 2
        Serial.println(sensorValue1, DEC); // prints the value read
        if (sensorValue1 < 400)         // if there is little or no smoke detected display green lights
        {
            digitalWrite(34, LOW); //Red light is turned off
            digitalWrite(32, HIGH); //Green light is turned on
        }

    else
        // if there is medium or high ppm smoke detected display red lights
        {
            digitalWrite(32, LOW); //Green light is turned off
            digitalWrite(34, HIGH); //Red light is turned on
            tone(26,250,100); //Turns On the Alarm
        }
}
```

## //Section A.5

```
    //FOR_HEAT_DETECTOR:
    temp = analogRead(1); // read analog input pin 1
    temp = temp * 0.48828125 ; //Converts into degree celcius
    Serial.print("TEMPERATURE = ");
    Serial.print(temp);
    Serial.print("°C");
    Serial.println();

    if (temp < 49)
    {
        delay (500);
        temp1 = analogRead(1); // read analog input pin 1
        temp1 = temp1 * 0.48828125 ; // Converts into degree celcius
        Serial.print("TEMPERATURE = ");
        Serial.print(temp1);
        Serial.print("°C");
        Serial.println();

        if ((temp1-temp) < 9)
        {
            digitalWrite(34, LOW); //Red light is turned off
            digitalWrite(32, HIGH); //Green light is turned on
        }
    else
        {
            digitalWrite(32, LOW); //Green light is turned off
            digitalWrite(34, HIGH); //Red light is turned on
            tone(26,240,100); //Turns On the Alarm
        }
    }
else
    {
        digitalWrite(32, LOW); //Green light is turned off
```

```

    digitalWrite(34, HIGH); //Red light is turned on
    tone(26,240,100); //Turns On the Alarm
  }
}
}

```

## ***// For Motion Detection Sensor***

```

const int buttonPin =15; // the number of the pushbutton pin
const int onLedPin = 49; // These could be any pin number - I picked them due to spacing
const int offLedPin = 47; //
const int alertLedPin = 45; //This is the dreaded yellow pin that signals past tragedy
unsigned long int starttime=0; //When the app starts - we want to give it some time
unsigned long int stime=0;
unsigned long int time=0;
unsigned long int interval=2000;
//to let us escape the trap
// variables will change:
int buttonState = 0; // variable for reading the pushbutton status
void setup()
{

```

### **//Section A.6**

```

// initialize the LED pin as an output:

pinMode(onLedPin, OUTPUT);
pinMode(offLedPin, OUTPUT);
pinMode(alertLedPin, OUTPUT);
// initialize the pushbutton pin as an input:
pinMode(buttonPin, INPUT);
digitalWrite(offLedPin, HIGH);
digitalWrite(onLedPin, LOW);
digitalWrite(alertLedPin, LOW);
Serial.begin(9600);
}
void loop()

{
  buttonState = analogRead(buttonPin); //read analog button
  stime=millis(); // counting times
  if(buttonState == HIGH) // getting a signal
  {
    digitalWrite(onLedPin, HIGH);
    digitalWrite(offLedPin, LOW);
    while(buttonState == HIGH)
    {
      buttonState = analogRead(buttonPin);
      time=millis()-stime; //calculating time
    }
  }
  Serial.println(time);

```

```

while(time>30000) //is estimated time 30 s is over
{
  starttime=millis();
  while(starttime+2000>millis())
  {
    analogWrite(alertLedPin,200);
    time=millis()-stime;
    Serial.println(time);
  }
time=0; // if no time, then signal is not detected
  buttonState = analogRead(buttonPin); // read input signal
  analogWrite(alertLedPin,0); // making buzzer off
}
}
}

digitalWrite(offLedPin, HIGH); // making green led on
digitalWrite(onLedPin, LOW); // // making red led off
}

```

## References

- [1]. <http://arduino.cc/en/Main/arduinoBoardMega2560>
- [2]. <http://arduino.cc/en/Main/Products>
- [3]. <http://datasheets.maximintegrated.com/en/ds/DS3231.pdf>
- [4]. <http://www.aliexpress.com/item/2pcs-lot-DS3231-AT24C32-IIC-Module-Precision-Clock-Module-DS3231SN-for-Arduino-FZ0319-Memory-module-Free/1499675220.html>
- [5]. Seattle Robotics Society
- [6]. Jameco Workshop
- [7]. ELECTRONICS GURUKULAM
- [8]. National Semiconductor datasheet for LM35/LM35A/LM35C/LM35CA/LM35D Precision Centigrade Temperature Sensors
- [9]. Arduino-MQ2-Gas-Sensor-Brick
- [10]. [www.bananarobotics.com](http://www.bananarobotics.com)
- [11]. <http://www.classiccmp.org/rtellason/transdata/d40d1.pdf>
- [12]. <http://www.instructables.com/id/DIY-Temperature-Humidity-Smoke-Detector-Alarm-Syst/?ALLSTEPS>
- [13]. [www.fritzing.org](http://www.fritzing.org)
- [14]. Wikipedia