BRAC UNIVERSITY

Inspiring Excellence

PROJECT

EEE366 Microprocessor and Interfacing Lab (Sec 02)

Project Title: Home Monitoring System

Group 07

Name	ID
Sk Tahmed Salim Rafid	19121028
Asef Jamil Ajwad	19121040
Redwan Ahmed Miazee	19121033

Declaration: We certify that this project is entirely our own work except where we have fully documented references to the works of others, and that the material contained in this project has not been submitted previously for assessment in any formal course of study.

Objective: The objective of this project is to implement a "Home Monitoring System" using ATmega32 with incorporation of our gathered knowledge from EEE366.

Apparatus:

Currently we are going through a pandemic so it is not possible for us to implement the project physically. So we have used "Proteus 8.9 SP2" as our simulation software for this project. To complete the project we have used the following components:

- ATmega32
- Op Amp
- LCD Display
- Transformer (Voltage Step-Down and Current)
- Relay
- Buzzer
- L293D motor driver
- Flame sensor
- PIR sensor
- NFC/BLE sensor
- LM35 Temperature Sensor
- HIH-5031 Humidity Sensor
- MQ-135 Air Quality Sensor
- Humidifier, Dehumidifier
- Zener diode, Diode, Resistor, BJT
- Led, Button, Lamp and Inductor

Additionally in our project we wanted to implement an IOT feature. But due to computational constraints of "Proteus" we couldn't implement it. However as a proof of the concept we have implemented it on a small scale. So for that we have used the **'ESP-01'** to enable internet connectivity to ATmega32.

Softwares used:

- Proteus 8.9 SP2
- Code vision AVR
- Arduino IDE
- Blynk iot platform

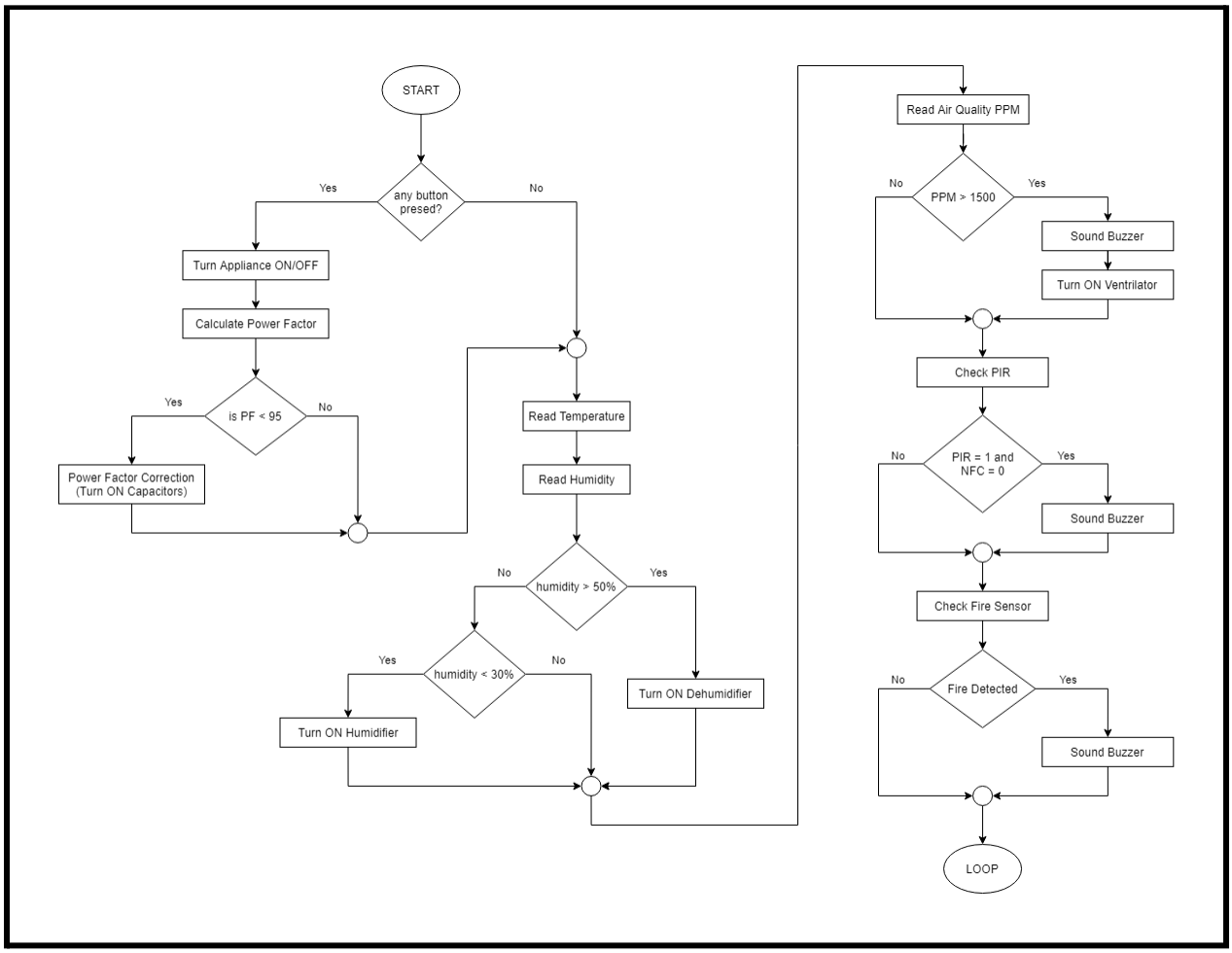
Project specification:

As the title of the project suggests, it's a home monitoring system so we used various sensors to monitor various aspects of our day to day life. The specifications are mentioned below:

- In our project we have implemented a capacitor based PFI (Power factor improvement) system. In our day to day life most of the loads we use are inductive, so we see a change in the power factor, mainly lagging. Due to a power factor less than 0.85 we might face power factor penalty or increased demand charge as demand charge might depend upon the ratio of KVA and KW.
- In the project we implemented a system to monitor the temperature of the home using a LM-35 sensor.
- In the project we have implemented a relative humidity monitoring system. According to www.lennox.com the humidity range of a home should be somewhere in between 30% to 50% for maximum comfort. So to monitor the relative humidity we implemented this feature.
- It's important to keep the humidity of a house in control to stop molds from growing. So to control the humidity we have used a humidifier and a dehumidifier to increase and reduce the relative humidity.
- Presence of a very high PPM of CO2 is harmful for the human body. So to monitor the CO2 of a house we implemented that feature in our circuit. Which will give warning to the user and if the PPM of CO2 is very high the system will start a ventilator in order to control the PPM of CO2.
- In order to make the house secured we have implemented a NFC based security and surveillance system. Whenever the authorised user will be in the house the PIR sensor will be disabled and if the house doesn't have authorised people near the PIR sensor will be activated and if it detects any unwanted motions, the system will notify the surrounding about the unauthorised access.
- To notify the user about any fire we have used a fire detector sensor.
- To notify the user about any information of the monitoring system or any warning we have used an LCD display and a buzzer in the system.

Methodology:

To develop the project we have gone through a lot of research papers and gathered knowledge about how the things work and how to implement it. During this phase based on our gathered knowledge and understanding we have developed an algorithm to complete the whole project. The algorithm is presented in a flow-chart in the following image:



Design choices and alternatives:

Power factor improvement:

The initial part of our project was to implement the power factor improvement unit. For that we have used a capacitor bank system. The logic behind the system is to calculate power factor and add capacitors in parallel until a certain target is met. To implement the system first we had to know the power factor of the system and to know the power factor we had to know the time delay between voltage and current. In the system we have assumed that the loads will always be

inductive thus we will always have a lagging power factor. To determine the time delay between two curves of voltage and current we have used a technique called zero cross detection. A zero-crossing detector or ZCD is one type of voltage comparator, used to detect a sine waveform transition from positive and negative, that coincides when the signal crosses the zero voltage condition. To design the circuit we have used an op amp as a comparator. But we couldn't directly input the 220V AC to the op amp. So we have used a transformer to step-down the voltage to 5V. The design of the transformer is given below:

Voltage transformer:

We know, $\frac{V_S}{V_P} = CP\sqrt{\frac{L_S}{L_P}} = \frac{I_P}{I_S}$

Here we assumed a coupling factor CP=0.8 and a primary inductance of 4H. We set the primary side voltage as 220V and secondary side as 5V. Putting these values in the equation we get,

$L_S = 0.0032283H$

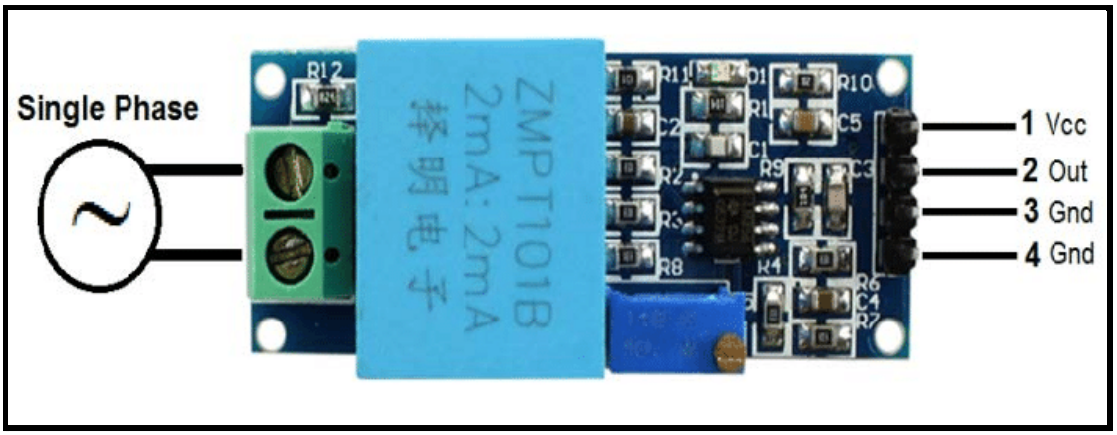
We assumed the primary side will have a very small current of 0.1A. From that as we know, $R_P = \frac{V_P}{I_P}$.

Using this equation we found out the primary side resistance 22Ω.

From the first equation we can calculate the secondary current and using that the secondary dc resistance as 0.011363640Ω.

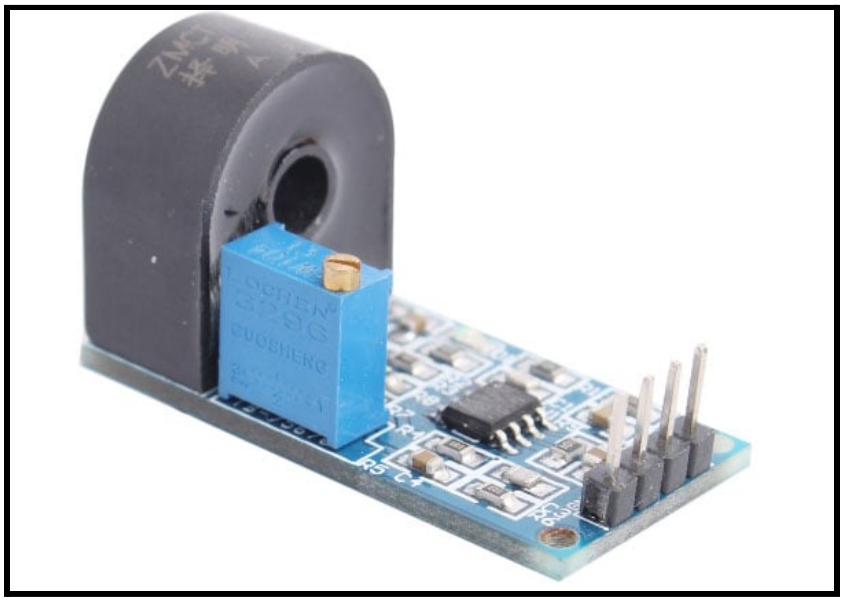
Finally to take a voltage as output we have connected a resistor and connected the other side with ground. The transformer is connected in parallel with the main AC power supply.

Alternatively, we could've used the ZMPT101B sensor. But due to unavailability in proteus we used this technique. We have attached an image of the alternative solution to measure the voltage.



Current transformer:

Next to measure the current's waveform we needed a current transformer. We used a proteus's simulation model of a current transformer based on hall effect. It outputs 0.001A current for every 1A current. To convert it to a voltage reading we have connected a burden resistor parallel to the output of the current transformer. It is connected in series with the main circuit.



Loads:

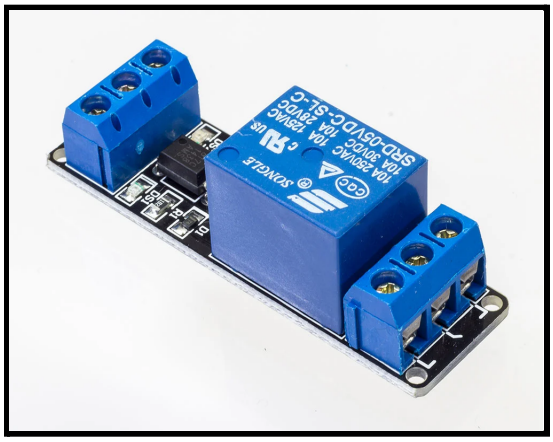
As mentioned earlier we have assumed the loads to be inductive. To simulate it we have connected a resistive lamp and an inductor in series and connected 4 of them in parallel to represent the loads of a house.

Capacitor bank:

The way this PFI system works is it adds capacitors in parallel to the circuit and by doing that it compensates for the lagging power factor. So to make that we have taken 4 capacitors and connected them parallel with the main loads.

Relays:

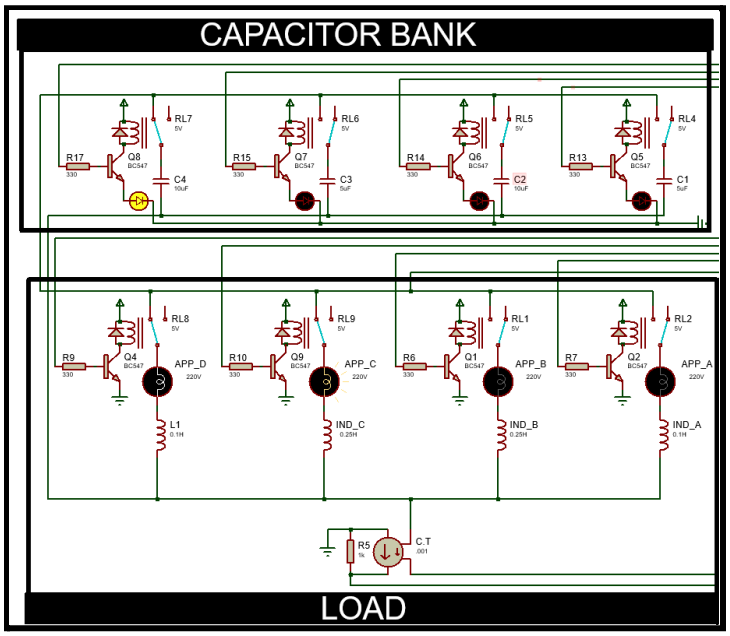
In the circuit we were working with high voltage and current. This kind of voltage and current can't be supplied by the microcontroller. So to isolate the microcontroller unit from the main AC power we have used a few 5V relays in the circuit. A flyback diode is placed with reverse polarity from the power supply and in parallel to the relay's inductance coil. The use of a diode in a relay circuit prevents huge voltage spikes from arising when the power supply is disconnected.



Relay switching by BJT:

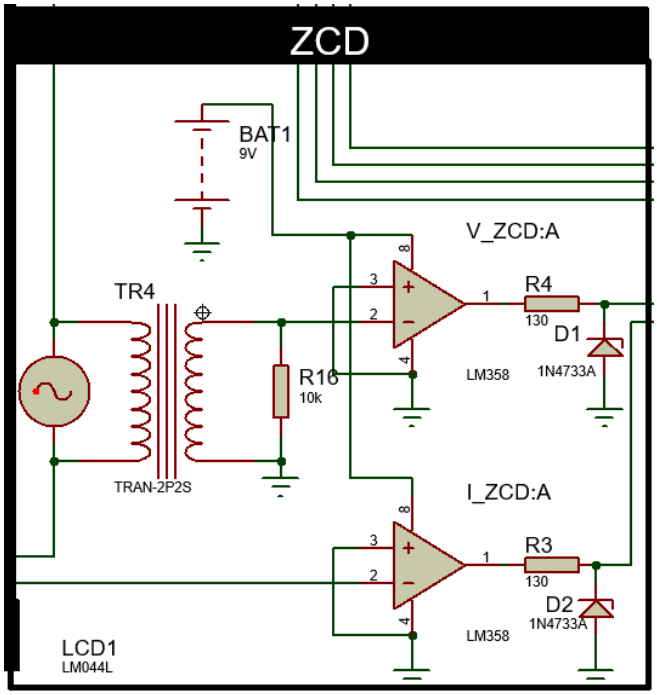
To drive the relays we had to design a relay driving circuit using BJT. The input is given to the base of the transistor and resistors are set such that the BJT operates in the saturation region and works as a switch for the relay.

Designed circuit so far:



Zero crossing detector:

The output of the transformers are then connected with the op amps. The op amp is used as a comparator. If the op amp detects the zero voltage the op amp outputs a high pulse. Then using the microcontroller we calculated the delay between the two pulses. This is called zero cross detection. The op amps output has a zener diode in parallel to limit the output voltage to 5V. The designed circuit is below:

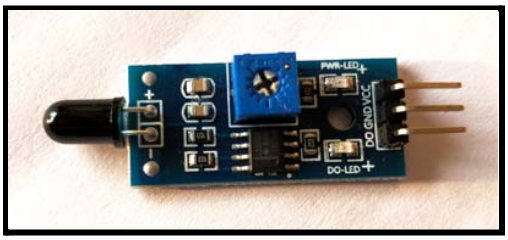


To control the loads manually we have connected 4 switches with the microcontroller.

Safety and monitoring:

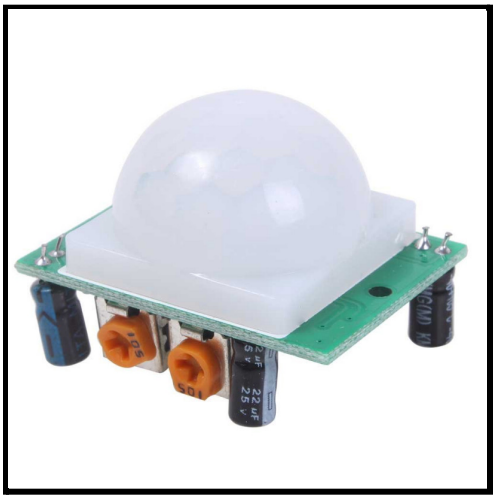
Fire sensor:

To determine whether any fire hazard is there we have used one fire detection sensor. It detects flame.



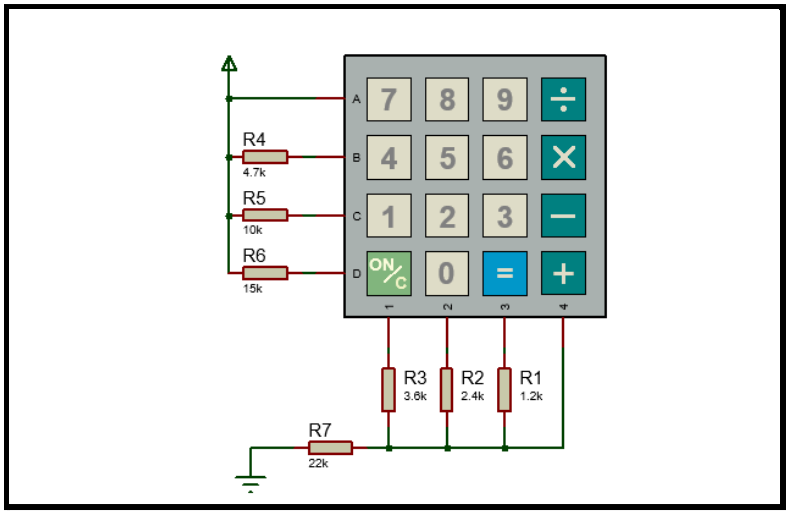
PIR sensor:

Here we have used a PIR or passive infrared sensor to detect any unwanted movement. But the problem with this sensor is it can detect a certain range and isn't as accurate as other methods of motion detector. We could've used a camera and with help of AI we could have easily implemented face recognition and motion detection. But due to our limitation of using only ATmega32 we had to keep ourselves limited to PIR sensor. Below is an image of it.



NFC/BLE sensor:

Here to determine whether an authorised person is in the house we used a NFC sensor. NFC stands for near field communication. Whenever someone authorised is near the system the NFC sensor will output high. From this the system will unlock or lock itself. Here the alternative was to use an OTP based password system but due to limitation of CodeVisionAvr we couldn't use it. However the designed keypad based on voltage divider and a single pin interfacing is given below.

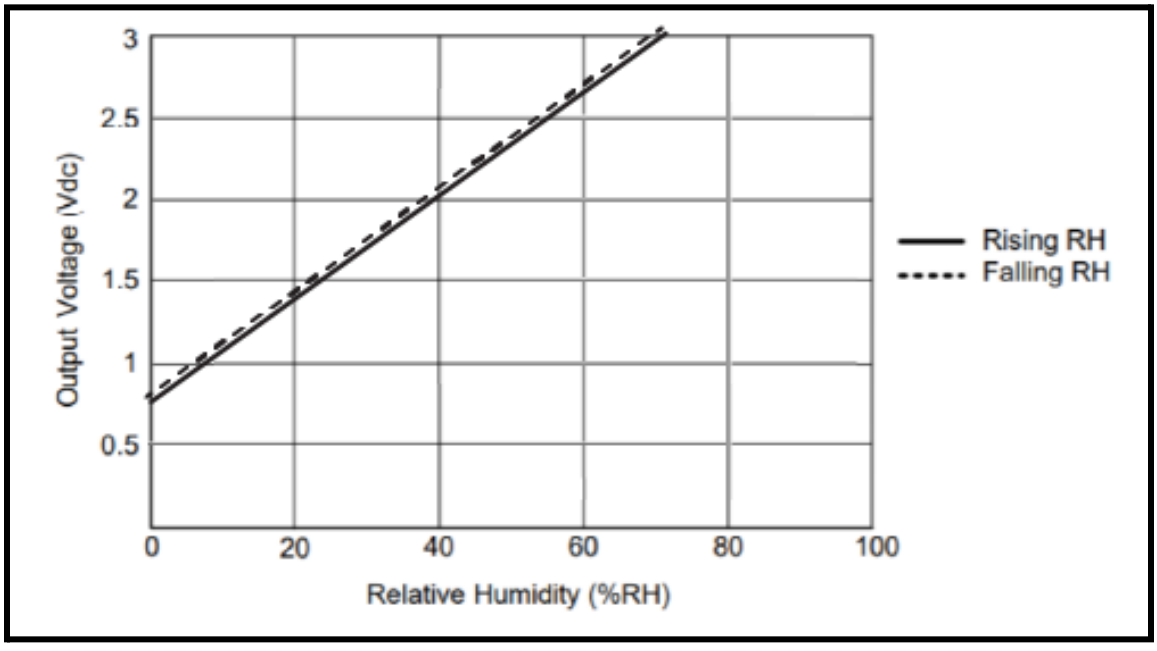


Temperature sensor:

To determine the temperature of the ambient we have used a LM-35 temperature sensor. It outputs analog data and for every 1C it outputs 10mV.

Humidity sensor:

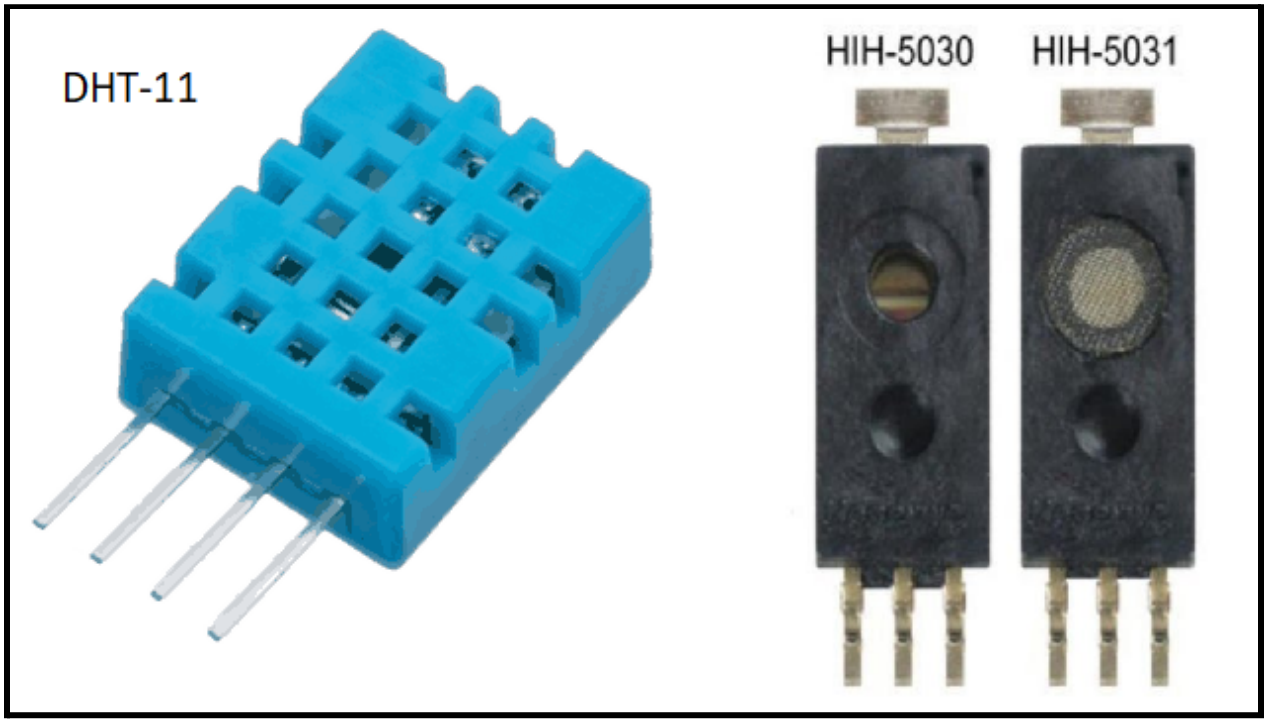
To read the relative humidity we have used the HIH-5031 sensor here. It outputs analog data. From the data sheet we have found the following graph of the sensor:



From this graph we found the following linear equation, RH= (Voltage-0.75)/0.031

Relative humidity should be in between 30%-50%. So based on the sensor reading we have implemented one humidifier and one dehumidifier. If the relative humidity is more than 50% the dehumidifier turns on and it turns off when the RH is below 40% and if RH goes below 30% the humidifier turns on and turns off when humidity is greater than 40%.

Here for temperature and humidity measurement we have used two different sensors. These are analog sensors. In spite of these we could have used one DHT-11 temperature and humidity sensor. But it depends on pulse width modulation. Because of proteus's limitation the sensor was outputting wrong data. So we decided to use these two sensors.



Air quality sensor:

For the measurement of CO2 we have used one MQ-135 sensor. Based on data sheet in ideal condition the sensor's parameters are as follows:

RLOAD=10

PARA=116.6020682

PARB=2.769034

RZERO=76.63

The sensor outputs analog voltage. To get the PPM reading from the sensor we used the following equation,

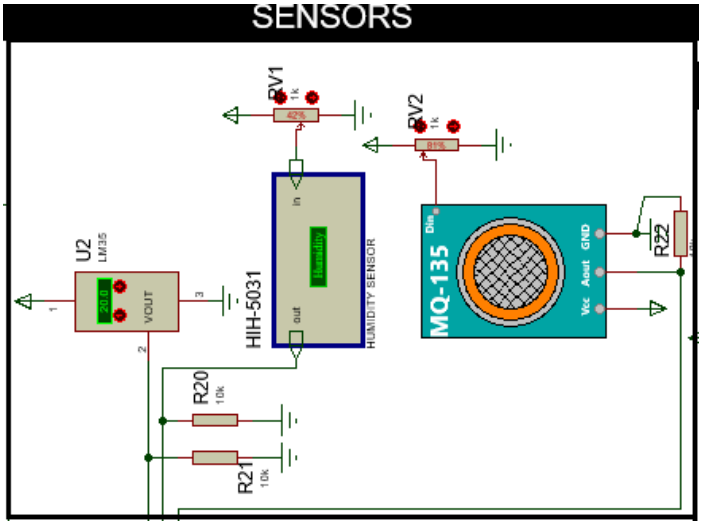
PARA*(Voltage*RLOAD)/RZERO)^-PARB

Using the sensor's reading the air quality is measured. Here we have used one ventilator. If the CO2 PPM is greater than 1500 we have used a buzzer and that's turned on and the ventilator is also turned on. Once the PPM again reaches less than 1000 the ventilator is turned off. Air quality chart based on PPM of CO2.

CO_2 [ppm]	Air Quality
2100	BAD Heavily contaminated indoor air Ventilation required
2000	
1900	
1800	
1700	
1600	
1500	MEDIOCRE Contaminated indoor air Ventilation recommended
1400	
1300	
1200	
1100	
1000	FAIR
900	
800	GOOD
700	
600	EXCELLENT
500	
400	

The alternative here was to use spec digital sensors as they are factory calibrated and digital so output data is more accurate.

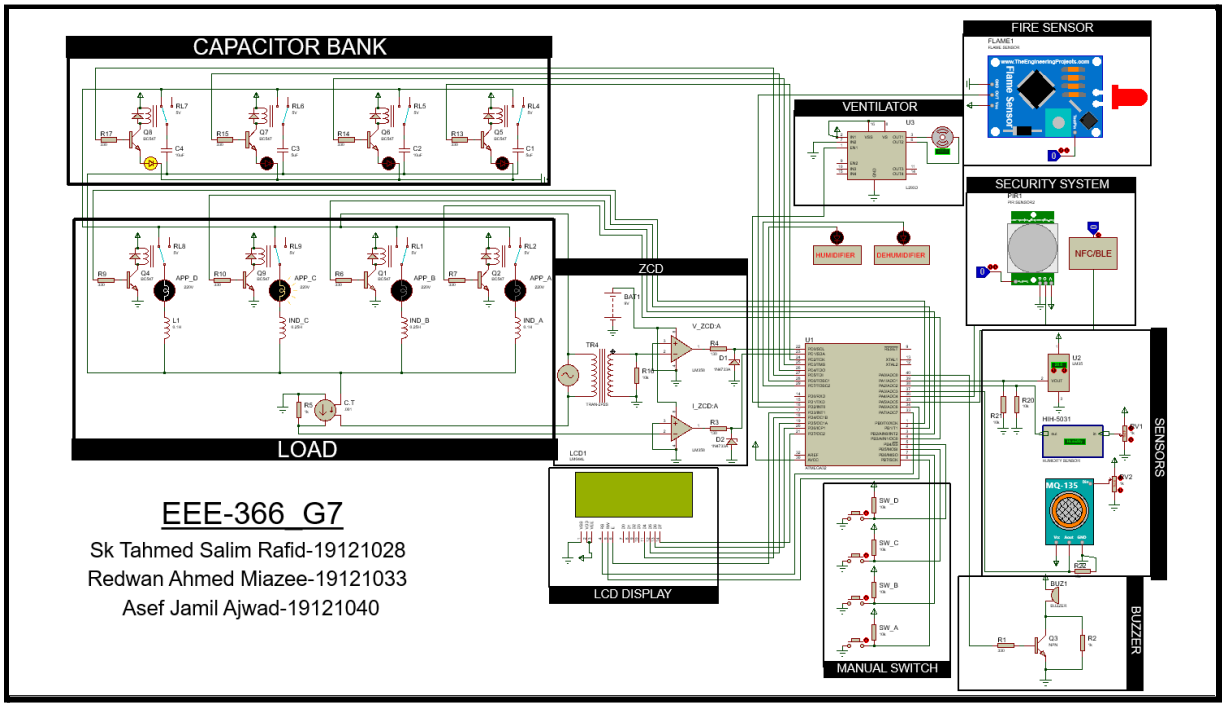
Designed sensor portion is given below:



Here we were limited to using ATmega32. Alternatively we could use a Nodemcu to implement the project. Using Nodemcu would enable us to implement the IOT features more easily. Another possible approach is to use a rasp-berry pi.

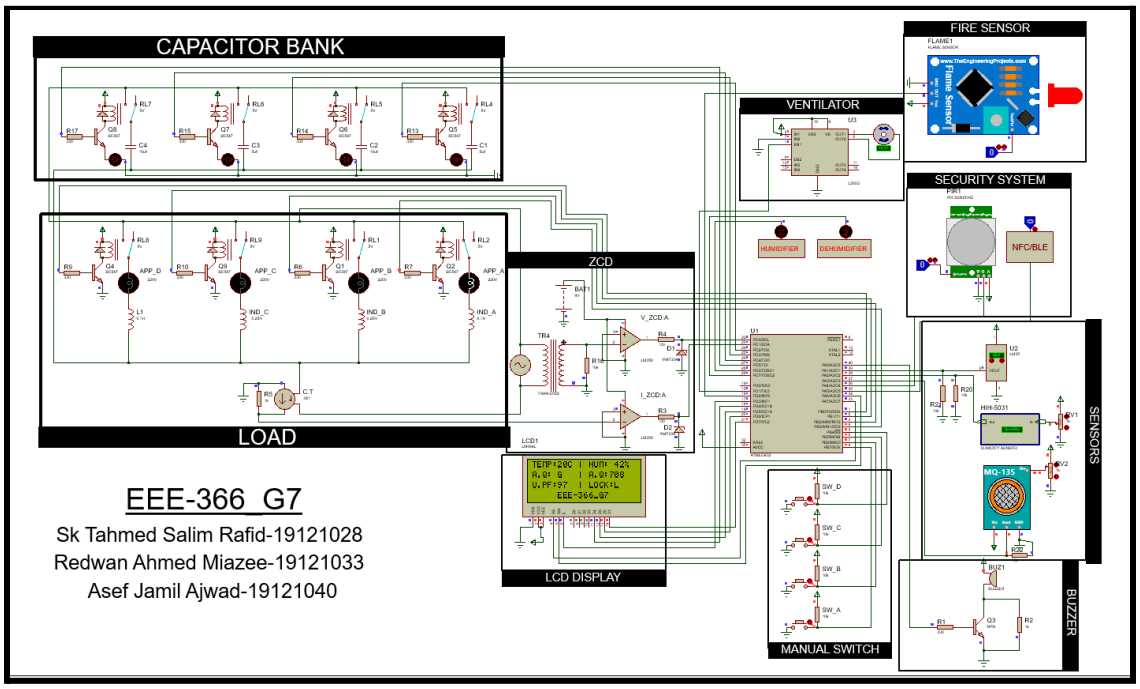
Circuit diagram:

Designed final circuit:



Proteus simulation results:

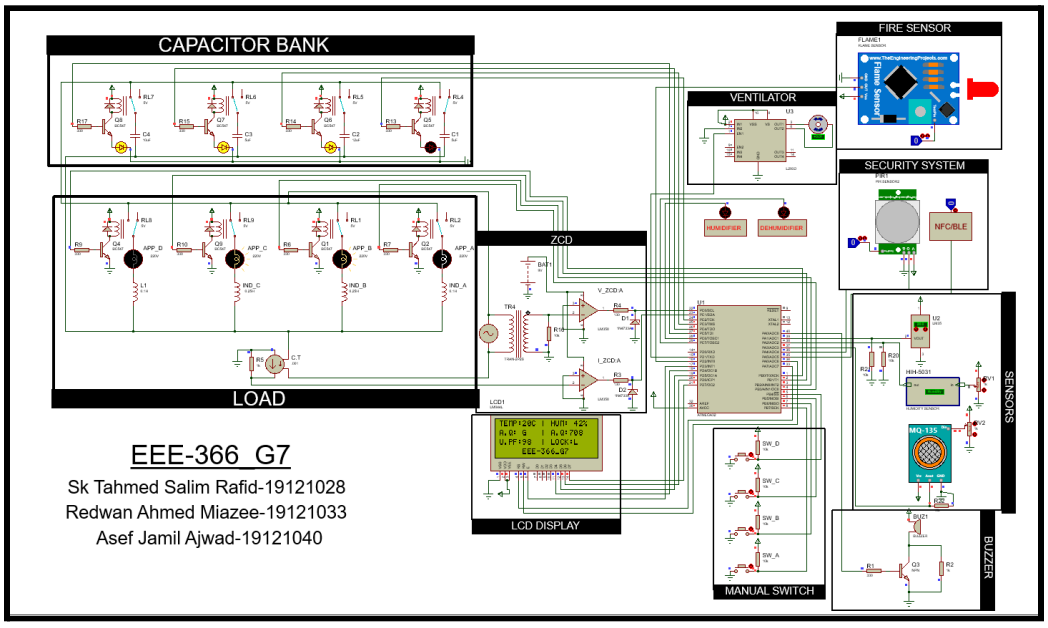
Power factor:



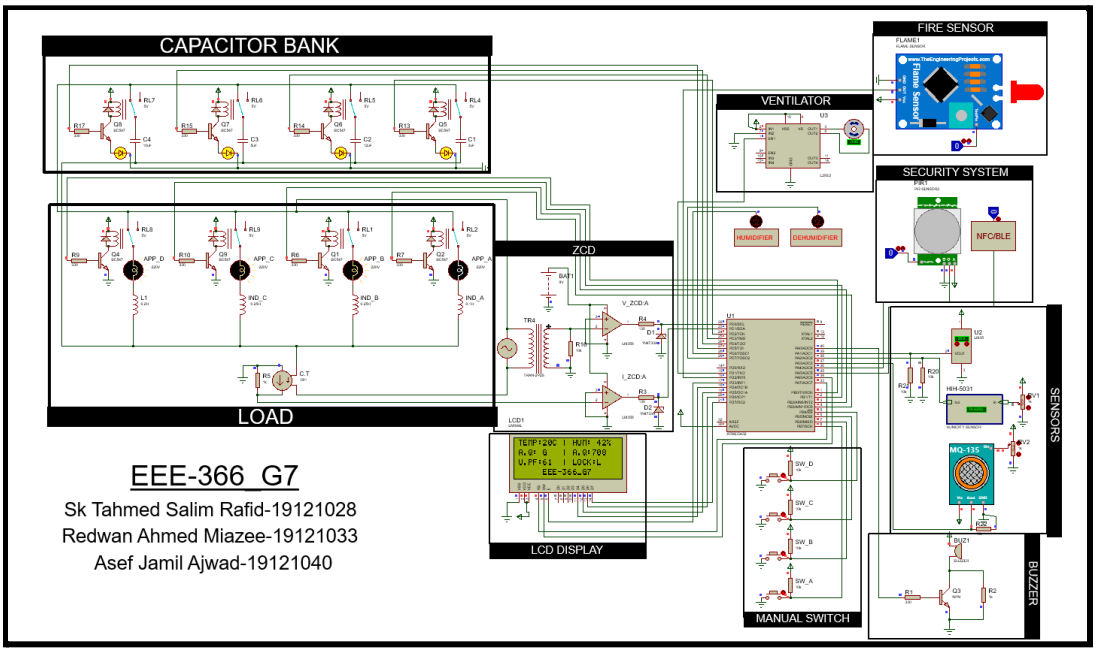
In the simulation we can see, the right most load is turned on. For this the power factor would be,

$cos(tan^{-1}(\frac{2*50*0.1*\pi}{100})) = 0.964 = 96.4\%$

In our designed system the power factor is showing 97%. So the system is quite accurate. As the power factor is not less than 95%, so none of the capacitors are turned on.

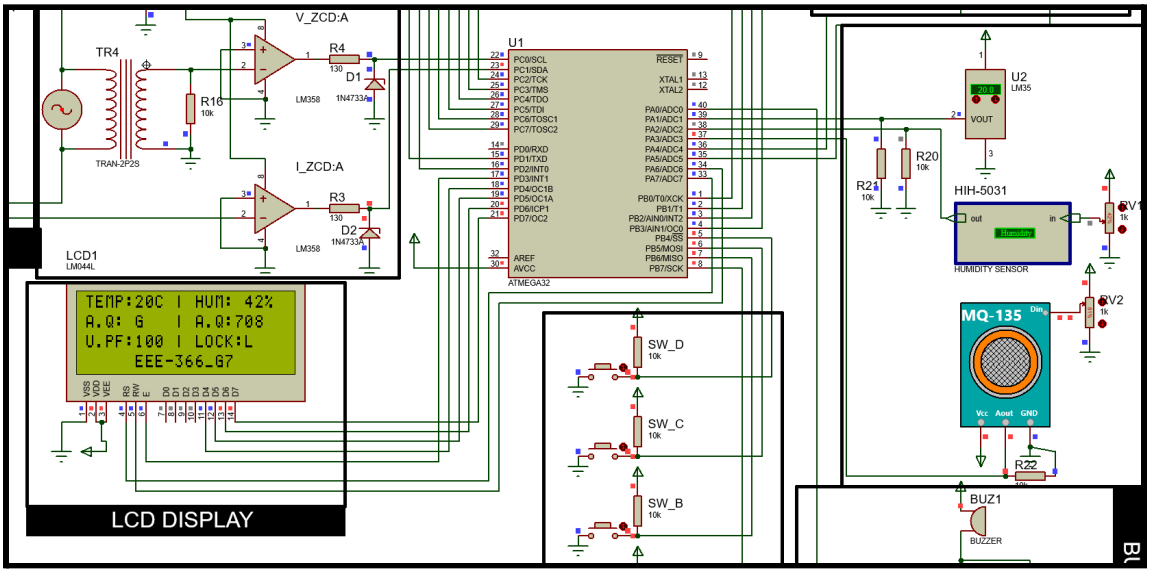


From the image we can see as the right three loads are turned on so the power factor changed and to compensate for that three capacitors are turned on, which can be seen in the capacitor bank portion. Now the corrected power factor is 98.



In the above image we can see as all the loads are on so to compensate for the power factor all the capacitors are turned on. Despite the system being unable to make the power factor 95%. So it remained in this state.

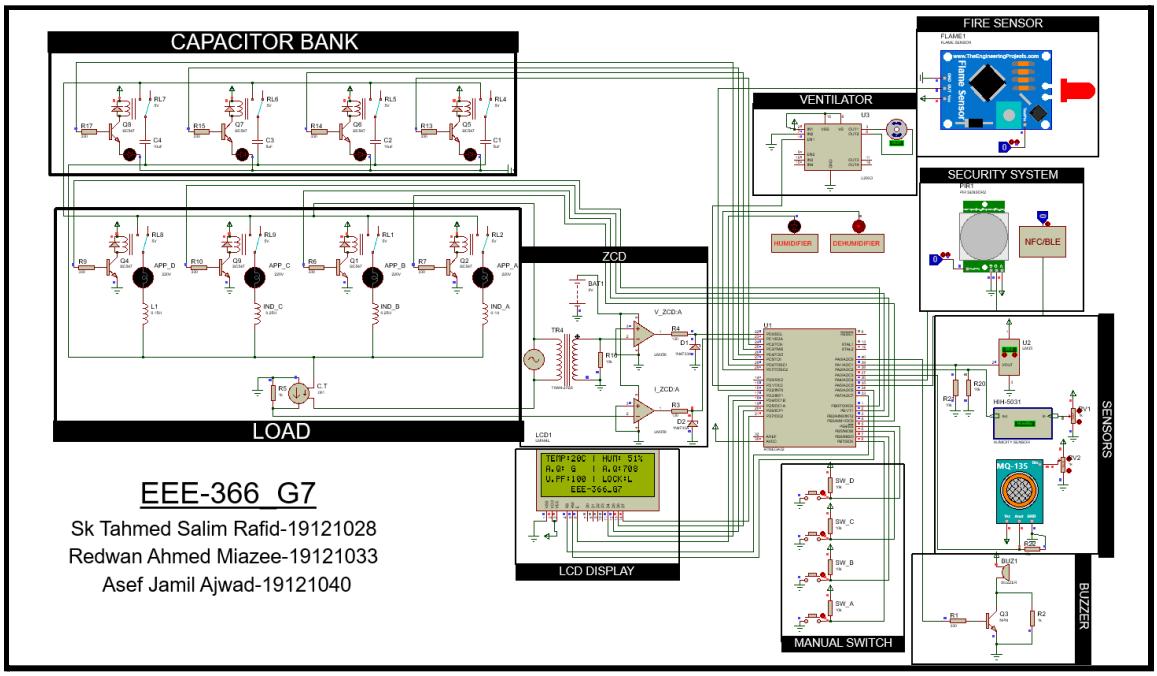
Temperature sensor:



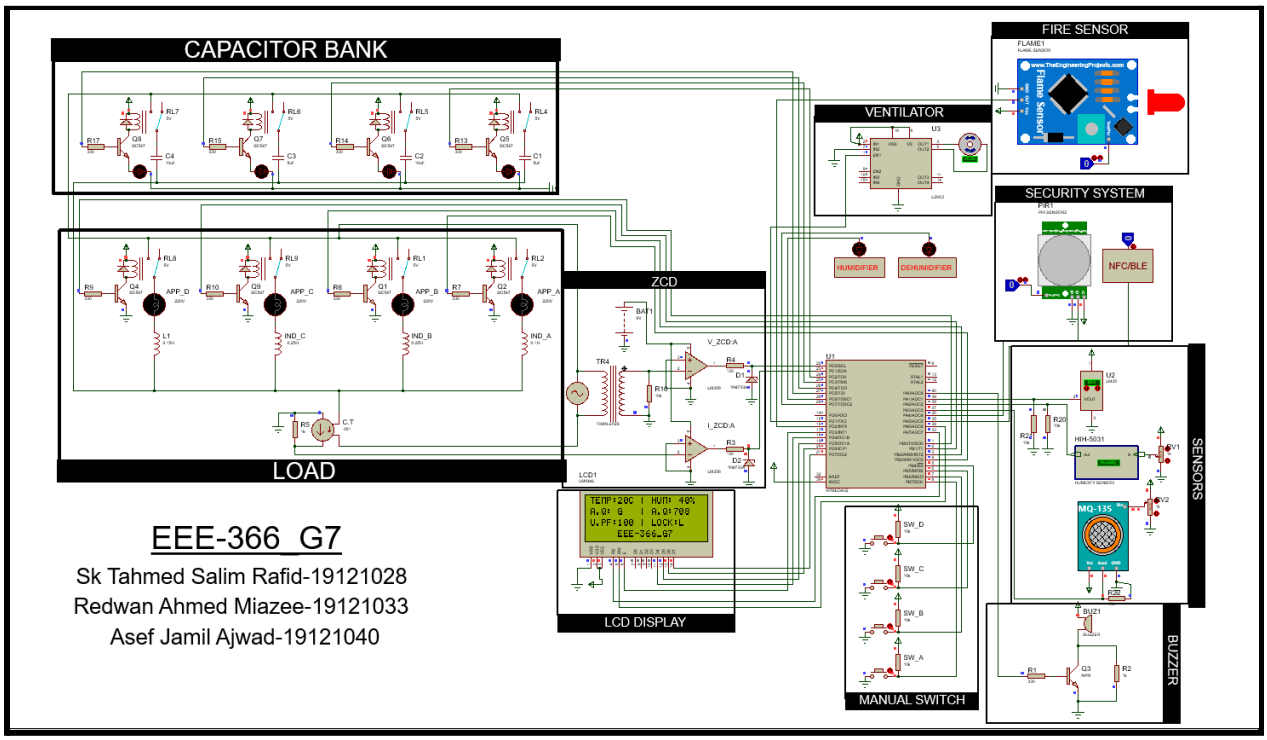
From this image we can see the temperature sensor is showing a value of 20C and so does the system.

Humidity sensor:

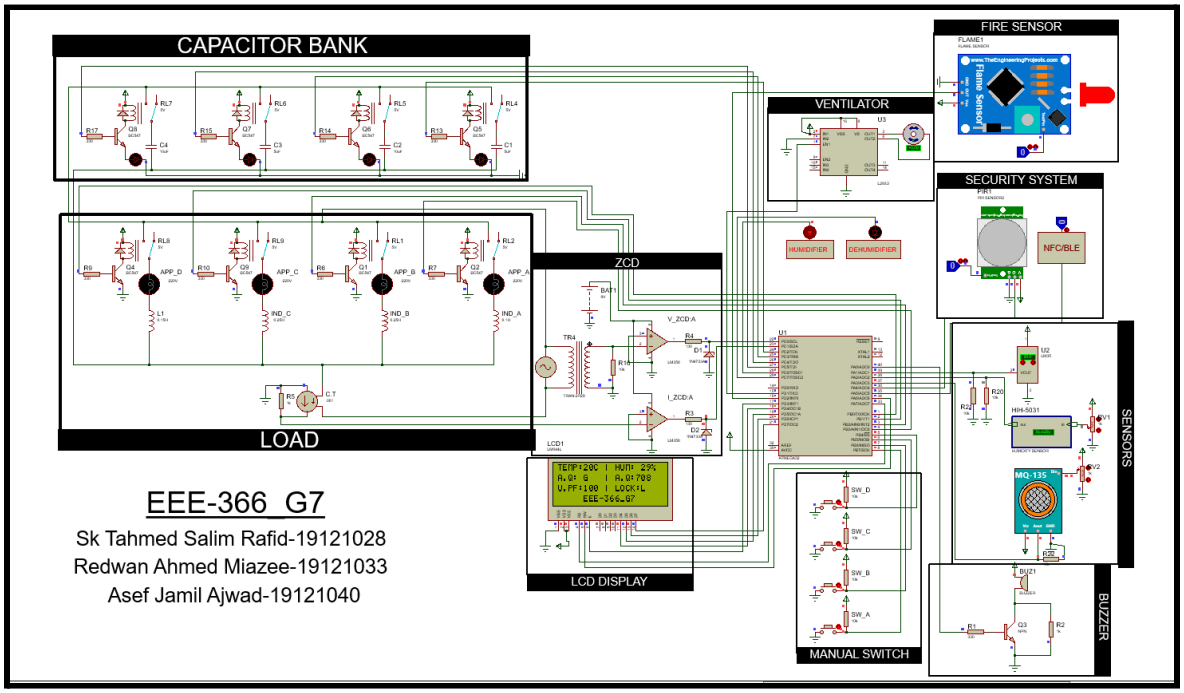
We can also observe from the image that the humidity is 42%. Now if we increase it and make it more than 50% the following will happen.



Here we can see as the humidity is more than 50% the dehumidifier is turned on. If we decrease it and make less than 40% the dehumidifier will turn off.

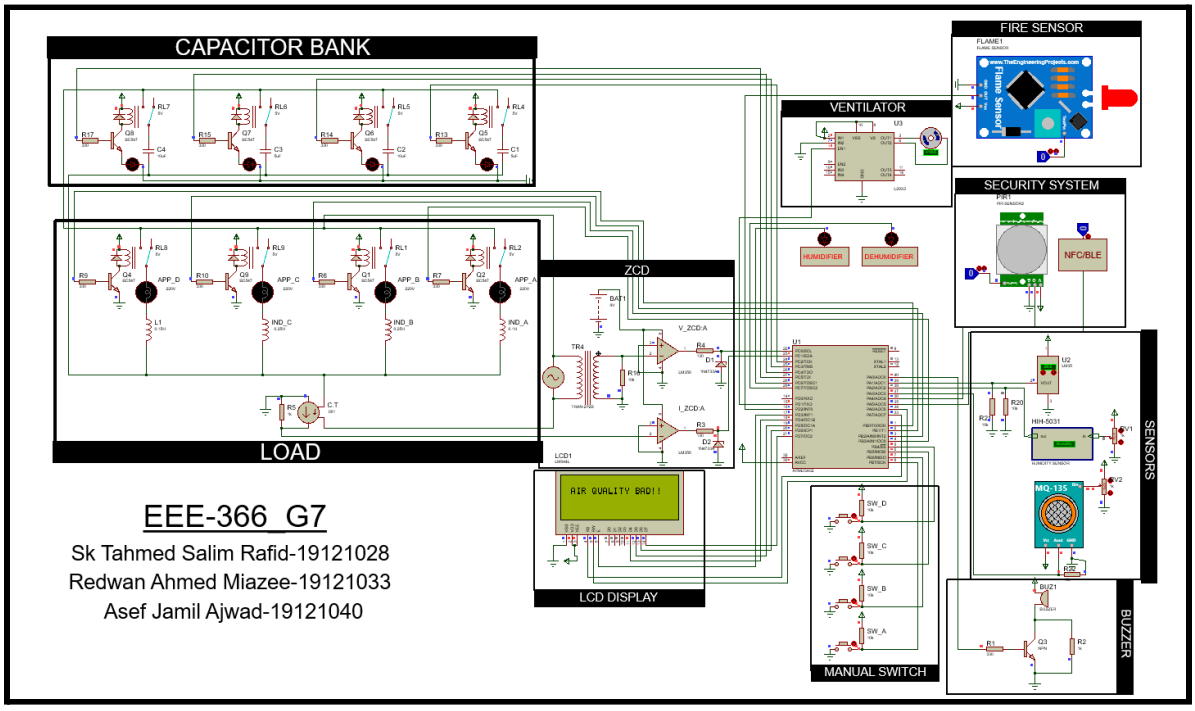


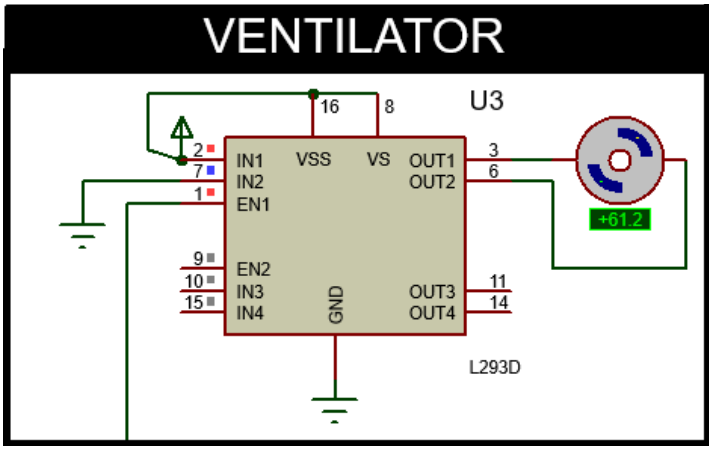
If we decrease it further and make it less than 30% the humidifier will turn on and will turn off only when humidity will be more than 39%.



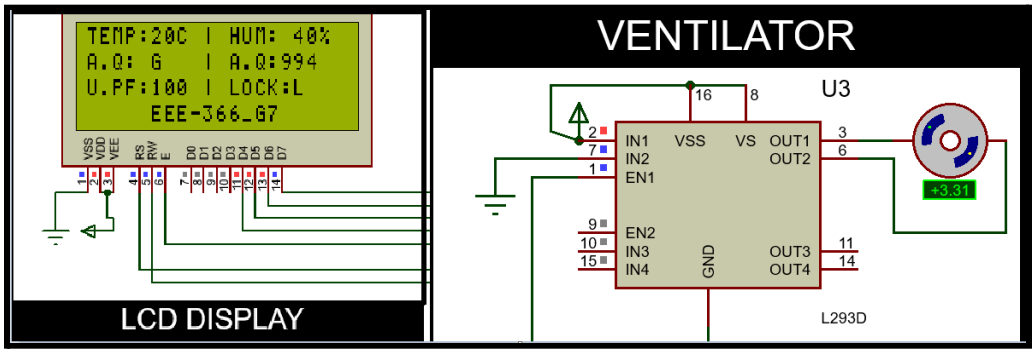
MQ-135 sensor:

From the above images we can see the PPM reading in less than 1000. So, it's showing 'G' referencing good air quality. Now if we increase it and make it more than 1500 the system will turn on the buzzer and turn on the ventilator.



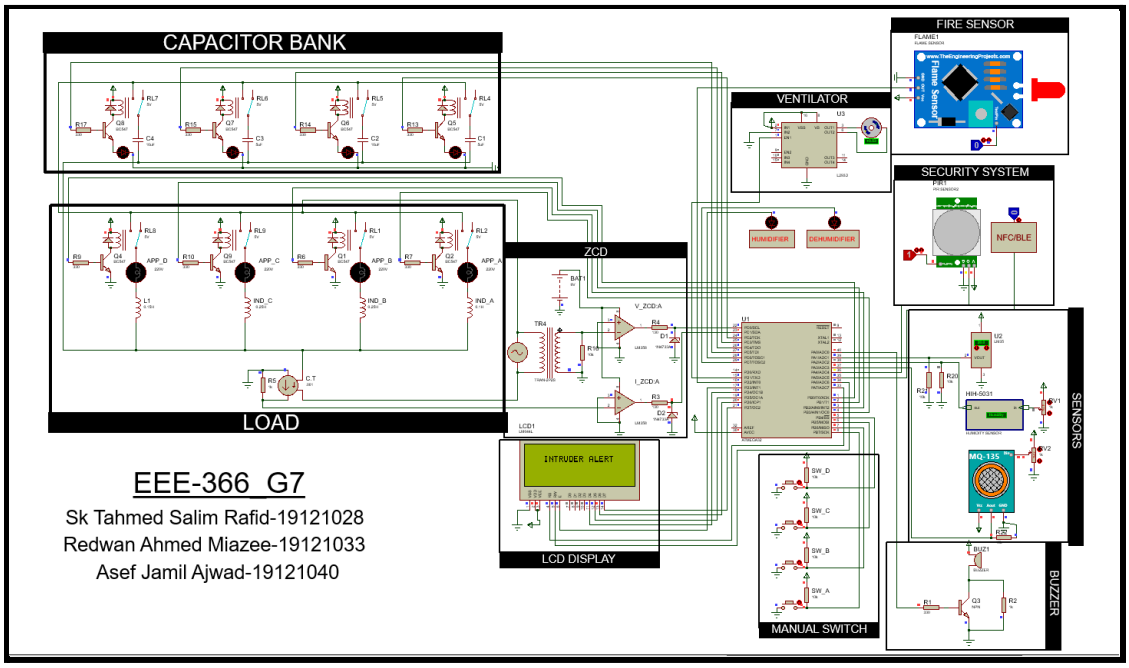


Here we can see the ventilator is also turned on.

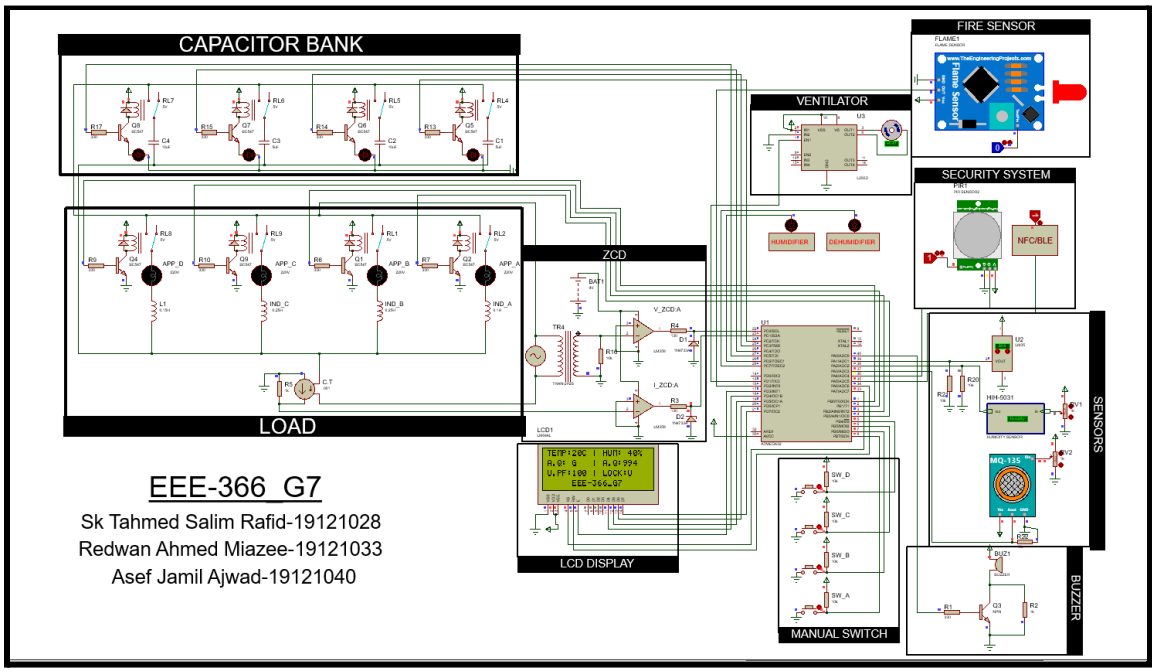


Here we can see as the air quality is less than 1000 PPM so the ventilator is turned off.

Security system:

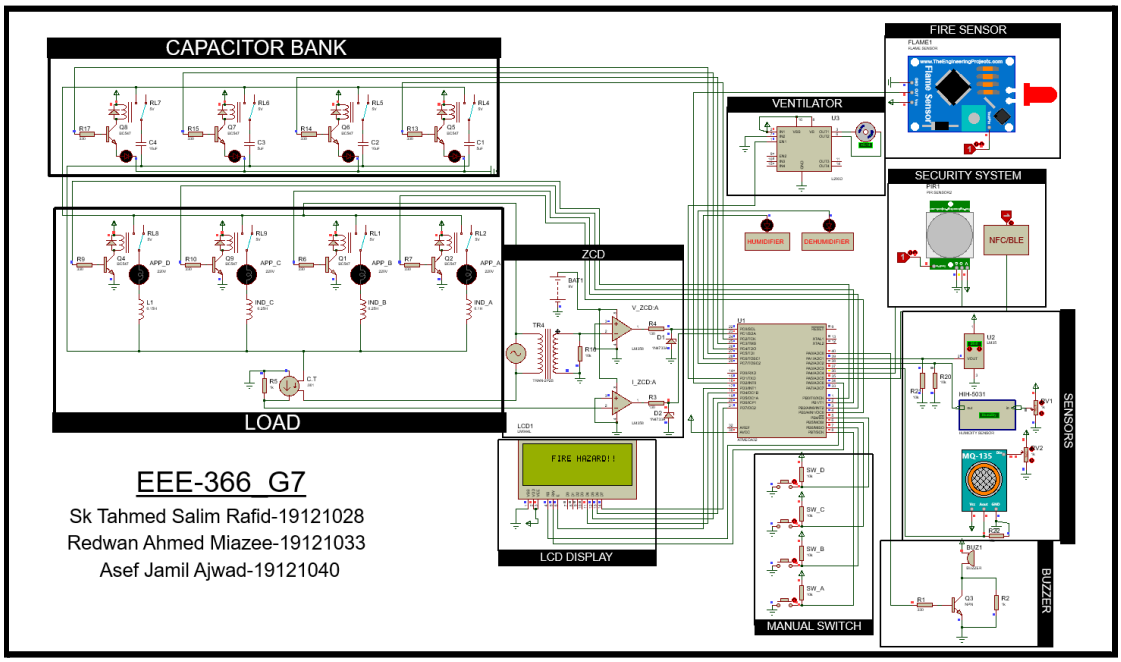


Here we can see the PIR sensor is detecting some motion when the NFC sensor isn't detecting anything. So the display is showing "INTRUDER ALERT".



Here we can see as the NFC is detecting authorized NFC around and PIR is also detecting something. But as the NFC sensor is high so "INTRUDER ALERT" is not showing rather the display is showing 'U' in the lock section.

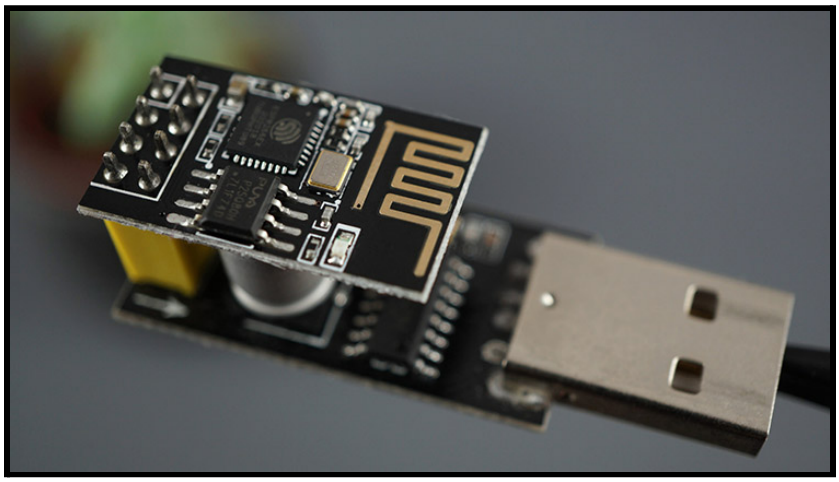
Flame sensor:



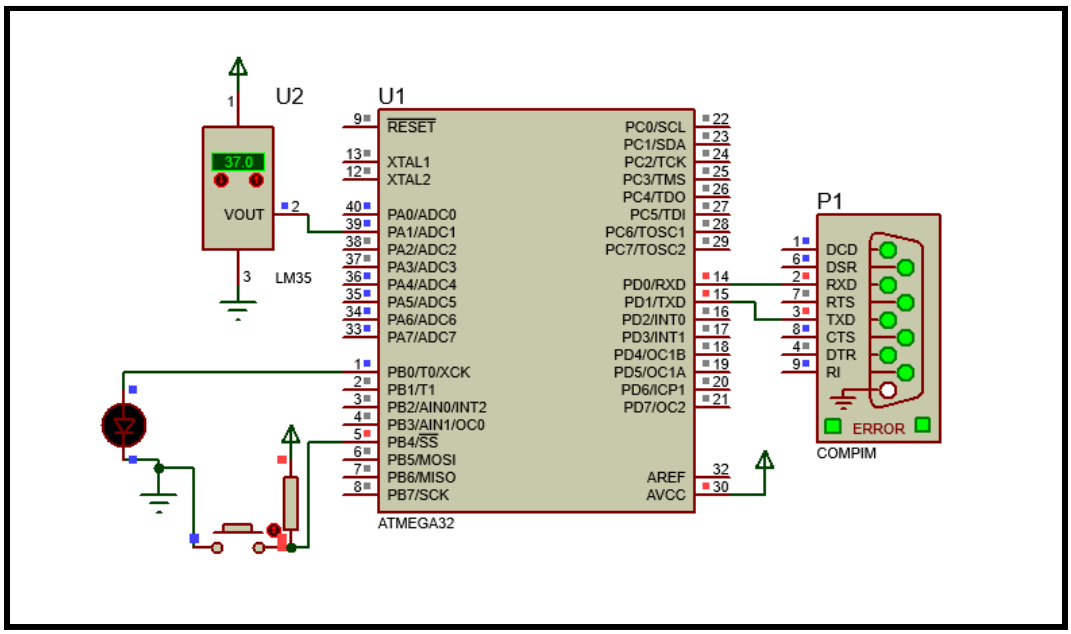
Here we can see the flame sensor is high so the display is showing “FIRE HAZARD”. During this time the buzzer will also be on.

Proof of IOT possibilities:

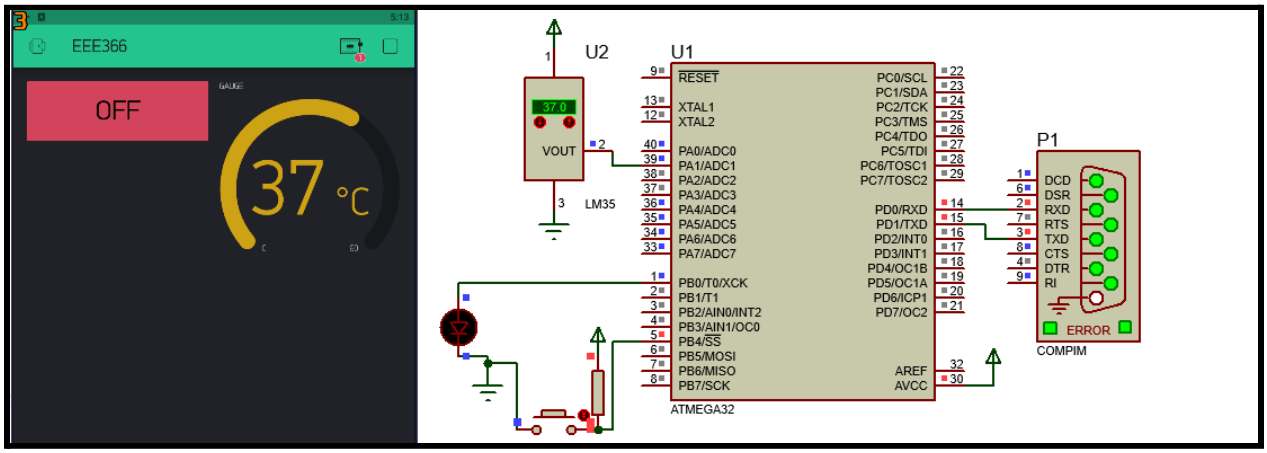
In ATmega32 we have pins for USART communication. Using those pins we can transmit and receive data from another module such as ‘ESP-01’. Due to the use of many components we couldn’t implement this feature in our main project. However as a proof of concept we have made another small design where we used an ‘ESP-01’ to communicate with ATmega32 via USART. The ESP-01 is physically connected to the comport and accessed in Proteus using the COMPIM component. Used ESP-01 module:



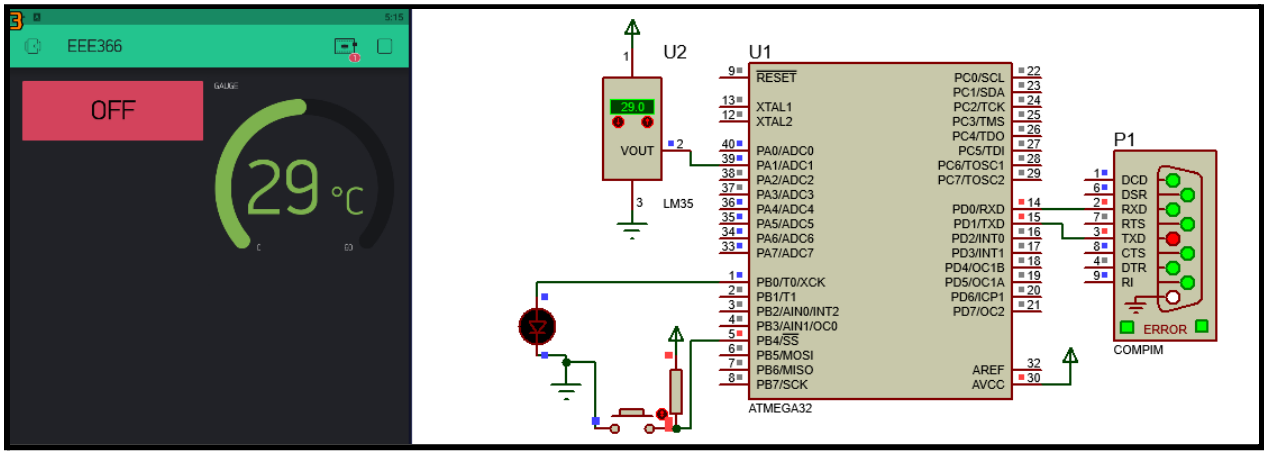
Circuit diagram:



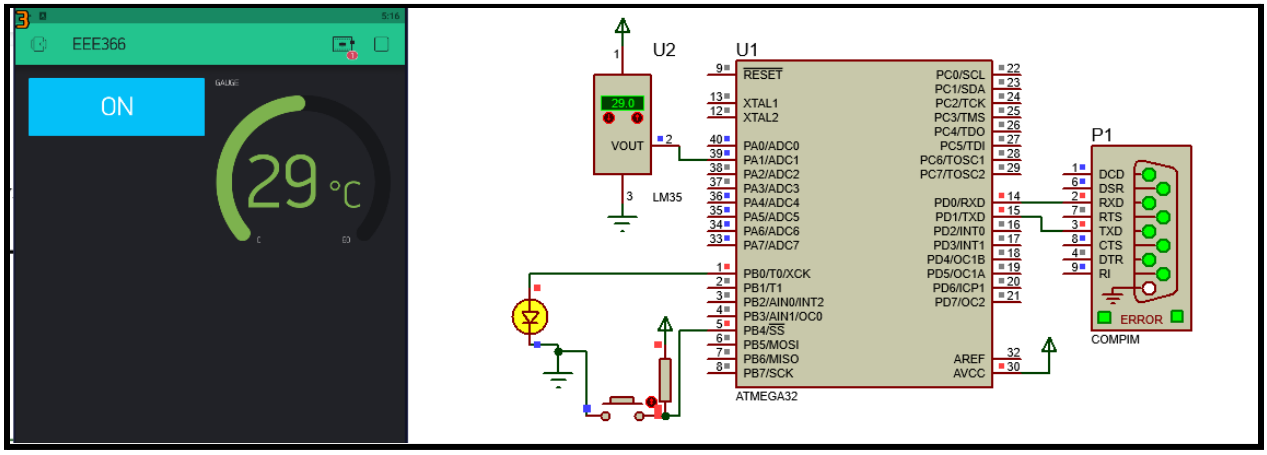
Proteus simulation:



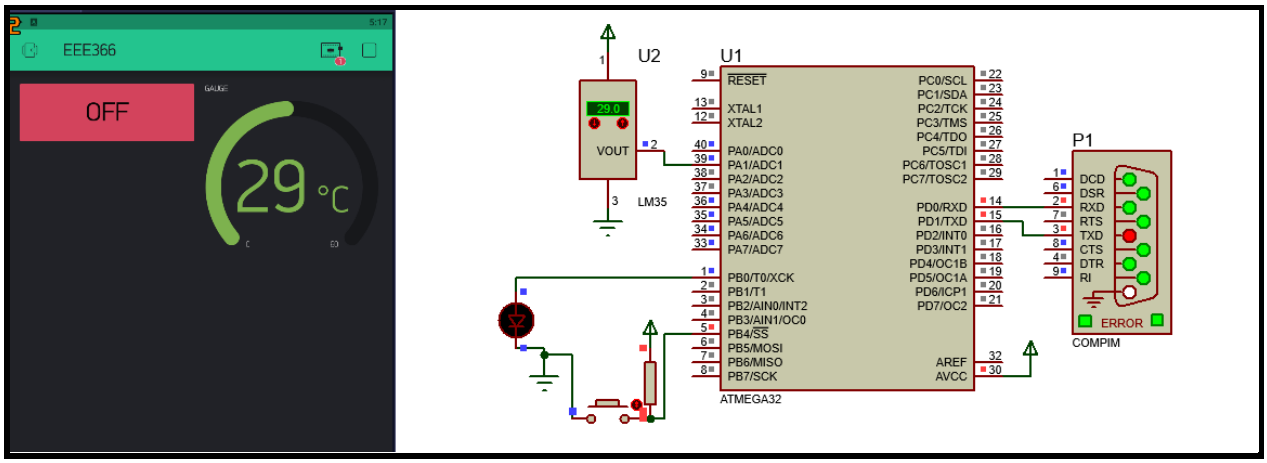
Here we can see in proteus the value is showing 37 C and in our internet connected mobile the same value is displayed.



Here we can see as LM-35 is showing 29 C the same is displayed in the mobile dashboard.



Here we can see as we have pressed the ‘ON’ button in the app the led connected to ATmega32 is also turned on.



Here we have pressed the button connected to ATmega32. By pressing it we have turned off the led and the same status is also updated in the app. So the led can be controlled by app and also physical button and the current status of the led is also updated in the app. This proves the possibilities of IOT implementation in this project.

Application:

Here we have implemented a “Home Monitoring System”. This system is a necessity in our day to day life. With the modernisation of society in the upcoming days, it is essential that such a system exists in every home. We want this project to be implemented everywhere as this is an advanced home monitoring system that modernises existing homes. People who want to modernise their existing homes with the new advancements in technology to automate and monitor their home would be highly invested in such a product. The demand for this product is very high and not many companies are selling this type of system in Bangladesh. This system could be sold to any household as a kit. Or, the design and implementation of the project can be sold to a private company to be sold as their product. Another option is that we could start a new company, dedicated to selling this product. Outside investors would also be willing to invest in such a company as the growth prospect of this system is high.

Social impact:

In our society, there is always a risk of thieves breaking into our homes and stealing our properties. This insecurity leads a lot of homeowners to have constant stress at the back of their minds when they are outside of their homes. This issue is solved by implementing a small but

very effective security feature in our system that would sound the alarm in case of theft. This advancement in security would act as a theft deterrent and thus reduce the amount of theft in our surroundings.

Economical impact:

In this system, we implemented a power factor correction system. In our electrical grid, if our power factor drops too low, we would be subject to surcharge from the electrical suppliers such as DPDC or DESCO. Thus, we would face financial loss due to it. Also, homes that use inverters, such as while using renewable energy, would lose a certain amount of the generated power of the renewable energy system due to this low power factor. Therefore, in a sense it is also a financial loss. But in our system, as we have implemented an automatic power factor correction system, these economical losses can be minimised.

Health impact:

Due to the increase of cars and industries, the air pollution, especially in cities like Dhaka, is increasing every year. This poor air quality is bad for health. Our system constantly monitors the air quality PPM. And, if the air quality becomes harmful to humans, the system automatically turns on the ventilator to bring in fresh air. In addition, our system also monitors the humidity of the environment. The ideal humidity for a human being is between 30% to 50%. Our system will automatically turn on the humidifier/dehumidifier to maintain this ideal humidity. This range of humidity would also prevent the growth of mould leading to a healthier household.

Discussion and future work:

While working for this project, we were presented with many challenges, solving which helped us learn new things. We researched for sensors which are required for this project and went through the datasheets of these sensors to work out the output relationship to make the project as accurate as possible. We learned about ZCD (Zero Crossing Detector) to calculate the phase angle for the power factor correction.

However, we failed to implement the IOT feature in our project due to excessive load on Proteus. While testing the IOT, it was working correctly. However, when we added more and more devices and sensors in our project, the CPU usage became very high. This started to cause many issues in parts of our projects. So, we ultimately decided to remove this feature from the main circuit but showcase it as conceptual proof on a small scale. We left the UART port empty on the main circuit so that in future we have the option of connecting the IOT to the project.

Another feature we removed from the final project was an "OTP based password system" for unlocking the door. There would be an app in the owners smartphone, which would generate an OTP or One-Time Password, that we would input into the system using the keypad. The system would then verify the password and unlock the door.

We tried to implement the project to the best of our abilities and looking back at it, we consider the project as a success. We were limited by the use of ATmega32 as the microcontroller in the system. Using modern microcontrollers such as a NodeMCU or Raspberry Pi in the project would allow us to implement even more advanced features such as AI assisted home security and facial recognition systems. For now, these are out of the scope of the project and we left it as future improvements.

Reference:

- https://petrowiki.spe.org/Power_factor_and_capacitors
- https://mechatrofice.com/electrical/capacitor-bank-power-factor-improvement#:~:text=Capacitor%20Bank%20is%20an%20assembly,of%20current%20from%20the%20voltage.
- https://embedschool.com/2015/05/31/power-factor-improvement-plant-using-avr-microcontroller-part02/
- https://atmega32-avr.com/power-factor-measurement-using-atmel-avr-micro-controllers/
- https://www.interstates.com/power-factor-correction/#:~:text=A%20lower%20power%20factor%20causes,the%20voltage%20at%20the%20equipment.
- https://www.theengineeringprojects.com/2019/01/introduction-to-lm35.html
- https://sensing.honeywell.com/honeywell-sensing-hih5030-5031-series-product-sheet-009050-2-en.pdf
- https://components101.com/sensors/mq135-gas-sensor-for-air-quality

- https://github.com/GeorgK/MQ135
- https://learn.adafruit.com/pir-passive-infrared-proximity-motion-sensor/how-pirs-work
- https://www.theengineeringprojects.com/2016/07/flame-sensor-library-proteus.html
- http://www.ventilationcontrolproducts.net/co2-detector-rc24
- https://www.maplesoft.com/products/maple/app_gallery/pdf/Condition_Air_into_the_Human_Comfort_Zone.pdf
- https://www.hackster.io/vladeusebiu/control-relays-in-proteus-with-nodemcu-3d95c5
- https://docs.blynk.cc/
- https://www.electronicwings.com/avr-atmega/atmega1632-adc
- https://www.electronicwings.com/avr-atmega/atmega1632-usart

Proteus simulation files & codes:

https://drive.google.com/file/d/1BpFWQzG4clmoREZqlUwA32R8LlolA0Ut/view?usp=sharing

Appendix:

ATmega32 code(No IOT):

```
#include <string.h>
#include <mega32.h>
#include <alcd.h>
#include <delay.h>
#include <math.h>
#include <stdlib.h>

// Few function declaration
void correct_pf(void);
void reinit_display(void);

// Varaible to put data in LCD
char temp_lcd[5];

// Humidity and temparature variable
long int humidity,temparature;

// CO2 PPM (Ideal) variable
long int PPM;
int prev_ppm;
float RLOAD=10,PARA=116.6020682,PARB=2.769034,RZERO=76.63;

// PF
int i=0;
float temp_pf=0;
float target_pf=0.95*100;
int pf_caps_i=0;
char pf_caps[5]={0x00,0x20,0x30,0x38,0x3C};
int del_t=0;
float prev_pf=0;

// Functions for PF
void cal_pf(void){
del_t=0;
for(i=0;i<10;i++){
 while(PINC.0!=1);
 TCNT0=0;
 while(PINC.1!=1);
  del_t=del_t+TCNT0;
}
del_t=del_t/10;
temp_pf=(cos((del_t*0.000064)*360*50*(3.14/180))*100);
}
```

```
void correct_pf(void){
cal_pf();
 PORTC.2=0;
 PORTC.3=0;
 PORTC.4=0;
 PORTC.5=0;
  while((temp_pf<target_pf && pf_caps_i<4)|| pf_caps_i==-1 ){
  pf_caps_i++;
   PORTC |=pf_caps[pf_caps_i];
   delay_ms(50);
   cal_pf();
  }
  delay_ms(50);
}

// ADC
long int ADC_Read(int channel){
    long int Ain,AinLow;
    // ADMUX change upon channel
    if(channel==0){
     ADMUX= 0x40;
    }
    if(channel==1){
     ADMUX= 0x41;
    }
    if(channel==2){
     ADMUX=0x42;
    }
    if(channel==3){
     ADMUX=0x43;
    }
    ADMUX=ADMUX|(channel & 0x0f);    /* Set channel */
    ADCSRA |= (1<<ADSC);        /* Start conv */
    while((ADCSRA&(1<<ADIF))==0);/*End of conv*/
    delay_us(10);
    AinLow = (int)ADCL; /* Read lower 8 bit */
    Ain = (int)ADCH*256; /* Read higher 2 bits */
    Ain = (Ain + AinLow);
    return(Ain); /* Return digital value*/
}


// PPM cal
void cal_ppm(void){
PPM=ADC_Read(3);
PPM=PARA*pow(((PPM/204.6)*RLOAD)/RZERO,-PARB);
if(PPM<600){
```

```
 lcd_gotoxy(5,1);
 lcd_putsf("E");
}
else{
if(PPM<1000){
 lcd_gotoxy(5,1);
 lcd_putsf("G");
}
else{
if(PPM<1500){
 lcd_gotoxy(5,1);
 lcd_putsf("M");
}
 else{
 lcd_gotoxy(5,1);
 lcd_putsf("B");
 PORTA.0=1;
 PORTD.1=1;
 lcd_clear();
 lcd_gotoxy(1,1);
 lcd_putsf("AIR QUALITY BAD!!");
 while(PPM>1000){
  PPM=PARA*pow(((ADC_Read(3)/204.6)*RLOAD)/RZERO,-PARB);
 }
 PORTA.0=0;
 prev_ppm=0;
 reinit_display();
 }
}
}
if(PPM<1000){
 PORTD.1=0; /* ventilator */
}
}


// HUMIDITY cal
void cal_humidity(void){
 humidity=ADC_Read(2);
    humidity= ((humidity/204.6)-0.75)/0.031;
    if(humidity<=30){
      PORTC.6=1; /* Humidifier */
    }
    if(humidity>=50){
      PORTC.7=1; /* Dehumidifier */
    }
}
```

```
// PIR and NFC/BLE detector
void check_pir(void){
if(PINA.5!=1 && PINA.4==1){
 //unwanted
 PORTA.0=1;
 lcd_clear();
 lcd_gotoxy(3,1);
 lcd_putsf("INTRUDER ALERT");
 while(PINA.5!=1);
 prev_ppm=0;
 reinit_display();
}
else{
   PORTA.0=0;
}
}

// Cal temp
void cal_temp(void){
 temparature=ADC_Read(1);
 temparature= ((temparature/2.046));
}

// Update appliance
void update_appliance(void){
 if (PINB.4==0){
    PORTB.0 = ~PORTB.0;
    lcd_clear();
    lcd_gotoxy(2,1);
    if(PORTB.0==1){
    lcd_putsf("LOAD-D TURNED ON");
    }
    else{
     lcd_putsf("LOAD-D TURNED OFF");
    }
    delay_ms(300);
    prev_ppm=0;
    reinit_display();
    pf_caps_i=-1;
    correct_pf();
    }
    if (PINB.5==0){
    PORTB.1 = ~PORTB.1;
    lcd_clear();
    lcd_gotoxy(2,1);
    if(PORTB.1==1){
```

```
    lcd_putsf("LOAD-C TURNED ON");
    }
    else{
     lcd_putsf("LOAD-C TURNED OFF");
    }
    delay_ms(300);
    prev_ppm=0;
    reinit_display();
    pf_caps_i=-1;
    correct_pf();
    }
    if (PINB.6==0){
    PORTB.2 = ~PORTB.2;
    lcd_clear();
    lcd_gotoxy(2,1);
    if(PORTB.2==1){
    lcd_putsf("LOAD-B TURNED ON");
    }
    else{
     lcd_putsf("LOAD-B TURNED OFF");
    }
    delay_ms(300);
    prev_ppm=0;
    reinit_display();
    pf_caps_i=-1;
    correct_pf();
    }
    if (PINB.7==0){
    PORTB.3 = ~PORTB.3;
    lcd_clear();
    lcd_gotoxy(2,1);
    if(PORTB.3==1){
    lcd_putsf("LOAD-A TURNED ON");
    }
    else{
     lcd_putsf("LOAD-A TURNED OFF");
    }
    delay_ms(300);
    prev_ppm=0;
    reinit_display();
    pf_caps_i=-1;
    correct_pf();
    }

}

// check hum dehum needs to be turned off or not
```

```
void check_hum_dehum(void){
 if(humidity>=40){
      PORTC.6=0; // Humidifier
    }
    if(humidity<=40){
      PORTC.7=0; // DEHUMIDIFIER
    }
}

// Display initial info
void display_info(void){
 lcd_clear();
 lcd_gotoxy(2,0);
 lcd_putsf("BRAC UNIVERSITY.");
 lcd_gotoxy(2,1);
 lcd_putsf("Home Monitoring");
 lcd_gotoxy(7,2);
 lcd_putsf("System");
 lcd_gotoxy(5,3);
 lcd_putsf("EEE-366_G7");
 delay_ms(2000);
}

//Create display sections
void display_table(void){
 lcd_clear();
 lcd_gotoxy(0,0);
 lcd_putsf("TEMP:  C | HUM:   %");
 lcd_gotoxy(0,1);
 lcd_putsf("A.Q:     | A.Q:    ");
 lcd_gotoxy(0,2);
 lcd_putsf("U.PF:    | LOCK:   ");
 lcd_gotoxy(5,2);
itoa(temp_pf,temp_lcd);
lcd_puts(temp_lcd);
lcd_gotoxy(3,3);
lcd_gotoxy(5,3);
 lcd_putsf("EEE-366_G7");
}

// update display
void update_display(void){
itoa(temparature,temp_lcd);
lcd_gotoxy(5,0);
lcd_puts(temp_lcd);
lcd_gotoxy(16,0);
itoa(humidity,temp_lcd);
```

```
lcd_puts(temp_lcd);
 if(prev_ppm!=PPM){
 lcd_gotoxy(18,1);
lcd_putsf("  ");
 lcd_gotoxy(15,1);
itoa(PPM,temp_lcd);
lcd_puts(temp_lcd);
prev_ppm=PPM;
 }
 if(prev_pf!=temp_pf){
 lcd_gotoxy(7,2);
 lcd_putsf(" ");
 lcd_gotoxy(5,2);
 itoa(temp_pf,temp_lcd);
 lcd_puts(temp_lcd);
 prev_pf=temp_pf;
 }
 if(PINA.5==0){
 lcd_gotoxy(16,2);
 lcd_putsf("L");
 }
 else{
   lcd_gotoxy(16,2);
 lcd_putsf("U");
 }
}

// Restrore display previous state
void reinit_display(void){
 lcd_clear();
 display_table();
 update_display();
}

// Flame check
void check_flame(void){
  if(PIND.2==1){
   PORTA.0=1;
 lcd_clear();
 lcd_gotoxy(5,1);
 lcd_putsf("FIRE HAZARD!!");
 while(PIND.2!=0);
 PORTA.0=0;
 prev_ppm=0;
 reinit_display();
  }
}
```

```
// Main code
void main(void)
{
DDRC=0b11111100;
DDRB=0x0F;
DDRD=0b11111010;
DDRA=0xF1;
ADCSRA = 0x87;
TCCR0=(0<<WGM00) | (0<<COM01) | (0<<COM00) | (0<<WGM01) | (1<<CS02) | (0<<CS01) |
(1<<CS00);
lcd_init(20);
display_info();
cal_pf();
display_table();

while (1)
    {
    update_appliance();
    cal_temp();
    cal_humidity();
    cal_ppm();
    check_pir();
    check_hum_dehum();
    check_flame();
    update_display();
    }
}
```

ATmega32 code(IOT):

```
#include <mega32.h>
#include <delay.h>
#include <string.h>
#include <stdlib.h>
#include <alcd.h>
#include <math.h>

int led1;
long int temparature;
char temp;
char  data[20];
int i=0,c=0,uart_count=0;
```

```
unsigned char temp_ar[3];
// ADC
void ADC_Init()
{
    DDRA=0xF1;            /* Make ADC port as input */
    ADCSRA = 0x87;            /* Enable ADC, fr/128  */
                /* Vref: Avcc, ADC channel: 0 */
}


long int ADC_Read(int channel)
{
    long int Ain,AinLow;
    if(channel==1){
     ADMUX= 0x41;
    }
    if(channel==2){
     ADMUX=0x42;
    }
    if(channel==3){
     ADMUX=0x43;
    }
    ADMUX=ADMUX|(channel & 0x0f);    /* Set input channel to read */

    ADCSRA |= (1<<ADSC);        /* Start conversion */
    while((ADCSRA&(1<<ADIF))==0);    /* Monitor end of conversion interrupt */

    delay_us(10);
    AinLow = (int)ADCL;        /* Read lower byte*/
    Ain = (int)ADCH*256;         /* Read higher 2 bits and
                    Multiply with weight */
    Ain = (Ain + AinLow);
    return(Ain);            /* Return digital value*/
}


 void uart_send(char* sendString){
     for (i = 0; i < strlen(sendString); i++){
         while (( UCSRA & (1<<UDRE))  == 0){
      uart_count++;
      if (uart_count>500){
      uart_count=0;
       break;
      }
     }
         UDR = sendString[i];
        delay_ms(10);
```

```
     }
}

unsigned char uart_receive ()
 {
     while(!(UCSRA&(1<<RXC))){
      uart_count++;
      delay_ms(1);
      if (uart_count>100){
      uart_count=0;
      c=5;
       return '\n';
      }
     }
 // wait until 8-bit of a character is received
     return UDR;
 }
 void uart_receive_data(void){
 temp=uart_receive();
    while (temp !='\n')
    {
        data[c++]=temp;
        temp=uart_receive();
    }
    data[c]='\0';
    c=0;
    }
 void cal_temp(void){
    temparature=ADC_Read(1);
    temparature= ((temparature/2.046));
}

void update_esp(){
    temparature = temparature*10+led1;
    itoa((int)temparature,temp_ar);
    uart_send(temp_ar);
}

void main() {

    UBRRH=0x00;
    UBRRL=0x67;
    UCSRB= (1<<RXEN) | (1<<TXEN);
    UCSRC=(1<<URSEL) | (1<<UCSZ1) | (1<<UCSZ0) ;
    ADC_Init();
    DDRB=0x01;
```

```
    while(1){

        uart_receive_data();

        if(data[0] == '0') {
            PORTB.0 = 0;
            led1=0;
        }
        else if(data[0] == '1') {
            PORTB.0 = 1;
            led1=1;
        }

        if(PINB.4==0){
         PORTB.0=~PORTB.0;
         while(PINB.4!=1);
         data[0]=PINB.0;
         led1=PINB.0;
        }

        cal_temp();
        update_esp();
        delay_ms(1000);
    }
}
```

ESP-01 code:

```
#include <string.h>
#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>

char auth[] = "sxRsk_QVDT4XZbqTc9lOX-XKlzJAhtDa";
char ssid[] = "R2";
char pass[] = "123456789";
String i = "ok";
int led1;
int temp=5;
String val  = "";
```

```
BLYNK_WRITE(V0){
  val = param.asStr(); // V0 to a variable
  led1=val[0]-48;
  Serial.println(led1); // Send back to atmega
}

BLYNK_READ(V1){
  temp=(i[0]-48)*10+(i[1]-48);
  Blynk.virtualWrite(V1,temp); // Update temp
  Blynk.virtualWrite(V0,led1); // Update button
}

void setup(){
  Serial.begin(9600);
  Blynk.begin(auth, ssid, pass);
}

void loop(){
  Blynk.run();
  if(Serial.available()>0){
    i=Serial.readString();
    led1=i[2]-48;
  }
  delay(300);
}
```