A Project Report

On

Construction of a Microcontroller-Based Protection System for Voltage and Frequency Fluctuation

Submitted by

B. Sc. Engineering Part-IV (Batch 2) Examinee-2021 Examination

Roll No: 1810874144

Registration No: 1810874144

Session: 2017-18

A Project Report

Submitted in Partial Fulfillment of the Requirements for the Degree of Bachelor of Engineering in Electrical and Electronic Engineering

Department of Electrical and Electronic Engineering,
University of Rajshahi, Rajshahi-6205, Bangladesh

AUTHOR'S DECLARATION

I do hereby declare that the project work incorporated in this report entitled ***Construction of a Microcontroller-Based Protection System for Voltage and Frequency Fluctuation*** has been performed by me. To the best of my knowledge, the project report has not been submitted previously to any University or Institution for any other degree.

With this declaration, I affirm that the work presented in this project is entirely my own, and I have not used the work of others without proper acknowledgment.

Roll No: 1810874144
Session: 2017-18 (Batch-2)
Department of
Electrical and Electronic Engineering
University of Rajshahi,
Rajshahi, Bangladesh

Abstract

The "Construction of a Microcontroller-Based Protection System for Voltage and Frequency Fluctuation" project presents a comprehensive solution to tackle the persistent challenges posed by AC voltage fluctuations and frequency deviations in electrical systems. These fluctuations can lead to operational inefficiencies, equipment malfunction, and even irreversible damage. This project introduces a sophisticated protection system that leverages microcontroller technology to continuously monitor incoming AC voltage and frequency levels, detect anomalies in real-time, and initiate prompt responses to safeguard connected equipment. Using intelligent algorithms and predefined thresholds, the system identifies abnormal voltage and frequency conditions. Upon detection, the microcontroller activates protective relays, triggers visual or auditory indicators, and can even disconnect sensitive equipment to prevent damage. By combining the power of microcontrollers with electrical engineering principles, this project offers a versatile and adaptable solution for various applications, from domestic setups to industrial environments.

Table of Contents

Chapter One

Introduction

1.Introduction

In a world driven by technological advancements, the efficient and reliable operation of electrical equipment is paramount across various industries and daily life. Voltage fluctuations and frequency deviations are among the critical challenges that can lead to the malfunctioning or even damage of sensitive equipment. To address these issues, this project emerges as a crucial solution to ensure the seamless functioning of electrical systems.

Modern electrical grids are susceptible to voltage sags, surges, and frequency variations due to factors such as load fluctuations, grid disturbances, and faults. These irregularities can have detrimental effects on connected devices, ranging from household appliances to industrial machinery. A robust protection system is essential to monitor, detect, and mitigate voltage and frequency anomalies promptly, safeguarding equipment and maintaining system stability.

The proposed project leverages the capabilities of microcontroller technology to create a sophisticated protection system. Microcontrollers provide the processing power required for real-time monitoring, analysis, and decision-making. By interfacing with appropriate sensors, the system can continuously monitor the incoming AC voltage and frequency levels. In the event of an abnormality, such as voltage exceeding safe thresholds or frequency drifting beyond acceptable ranges, the protection system will trigger responsive actions to prevent any potential damage.

Key Objectives of the Project:

1. Real-time Monitoring: The system will employ advanced sensors to continuously monitor the AC voltage and frequency of the incoming power supply. This real-time data will serve as the foundation for accurate anomaly detection.

2. Anomaly Detection: Through intelligent algorithms and predefined thresholds, the microcontroller will be programmed to identify voltage and frequency anomalies. This could include voltage spikes, drops, frequency drifts, or rapid fluctuations.

3. Immediate Response: Once an anomaly is detected, the microcontroller will initiate

appropriate response mechanisms. This could involve activating protective relays, disconnecting sensitive equipment, and alerting relevant personnel via visual or auditory indicators.

1.1 Microcontroller Based Over-Voltage Protection

Overvoltage events can pose a significant threat to the operation and longevity of sensitive devices and appliances. To counter this challenge, microcontroller-based overvoltage protection systems have emerged as valuable solutions. These systems employ microcontrollers, which are compact and efficient computing units, to monitor and control voltage levels in real-time. Microcontroller-based overvoltage protection systems are designed to swiftly detect any voltage spikes or surges that surpass safe operating thresholds. These spikes can result from lightning strikes, switching actions, or other transient disturbances in the power grid. Upon detection, the microcontroller triggers protective measures to mitigate the risk. This can involve activating circuit breakers, isolating the affected section of the grid, or signaling to other control systems for coordinated response. [1]

1.2 Microcontroller Based Under-Voltage Protection

Under voltage events can occur due to various reasons such as load fluctuations, faults, or external factors. These events can lead to malfunction or damage of sensitive devices, disruption of critical operations, and even system-wide instability. Microcontroller-based under voltage protection systems continuously monitor the incoming voltage levels. These microcontrollers are programmed to analyze the voltage data in real-time and compare it against preset thresholds. If the voltage falls below a certain acceptable level, the protection system triggers appropriate actions to mitigate potential harm. These actions can include sending alerts to operators, initiating load shedding procedures, or activating backup power sources like generators or batteries [1].

1.3 Microcontroller Based Over-Frequency Protection

Microcontroller-based over-frequency protection is a technique used to protect power systems from the harmful effects of frequency fluctuations. This technique involves designing a microcontroller-based over frequency relay that can detect frequency fluctuations and take appropriate action to protect the power system [2].

1.4 Microcontroller Based Under-Frequency Protection

Microcontroller-based under-frequency protection is a system designed to monitor the frequency of an electrical power system and take appropriate actions if the frequency drops below a certain threshold. This type of protection is crucial in maintaining the stability and reliability of power systems, as significant deviations from the nominal frequency can indicate an imbalance between power generation and consumption.

Chapter Two

Hardware Description

2.1 Microcontroller

A microcontroller, also known as an MCU (microcontroller unit), constitutes a compact computer housed within a solitary VLSI integrated circuit (IC) chip. It houses one or multiple CPUs (central processing units) in addition to memory and configurable input/output peripherals. The chip typically incorporates program memory, which can take the form of ferroelectric RAM, NOR flash, or OTP ROM, along with a limited quantity of RAM. These microcontrollers are meticulously crafted for embedded applications, setting them apart from the microprocessors employed in personal computers or other versatile applications that rely on an assortment of separate chips [3].

Modern terminology defines a microcontroller as akin to, albeit less intricate than, a system on a chip (SoC). While an SoC can encompass external microcontroller chips as motherboard constituents, it typically consolidates advanced peripherals such as a graphics processing unit (GPU) and Wi-Fi interface controller into its internal microcontroller unit circuits [4].

As of 2008, there are several dozen microcontroller architectures and vendors including:

- ARM core processors (many vendors)
- ARM Cortex-M cores are specifically targeted toward microcontroller applications
- Microchip Technology Atmel AVR (8-bit), AVR32 (32-bit), and AT91SAM (32-bit)
- Cypress Semiconductor's M8C core used in their PSoC (Programmable System-on-Chip)
- Freescale ColdFire (32-bit) and S08 (8-bit)
- Freescale 68HC11 (8-bit), and others based on the Motorola 6800 family
- Intel 8051, also manufactured by NXP Semiconductors, Infineon, and many others
- Infineon: 8-bit XC800, 16-bit XE166, 32-bit XMC4000 (ARM based Cortex M4F), 32-bit TriCore and, 32-bit Aurix Tricore Bit microcontrollers [30]
- Maxim Integrated MAX32600, MAX32620, MAX32625, MAX32630, MAX32650, MAX32640
- Microchip Technology PIC, (8-bit PIC16, PIC18, 16-bit dsPIC33 / PIC24), (32-bit PIC32)

This project is done by Atmega328p.

2.1.1 Atmega328p Microcontroller

The AVR® core combines a rich instruction set with 32 general purpose working registers. All the 32 registers are directly connected to the arithmetic logic unit (ALU), allowing two independent registers to be accessed in one single instruction executed in one clock cycle. The resulting architecture is more code efficient while achieving throughputs up to ten times faster than conventional CISC microcontrollers.

The Atmel® ATmega328P provides the following features: 32K bytes of in-system programmable flash with read-while-write capabilities, 1K bytes EEPROM, 2K bytes SRAM, 23 general purpose I/O lines, 32 general purpose working registers, three flexible Timer/Counters with compare modes, internal and external interrupts, a serial programmable USART, a byte oriented 2-wire serial interface, an SPI serial port, a 6-channel 10-bit ADC (8 channels in TQFP and QFN/MLF packages), a programmable watchdog timer with internal oscillator, and five software selectable power saving modes. The idle mode stops the CPU while allowing the SRAM, Timer/Counters, USART, 2-wire serial interface, SPI port, and interrupt system to continue functioning. The power-down mode saves the register contents but freezes the oscillator, disabling all other chip functions until the next interrupt or hardware reset. In power-save mode, the asynchronous timer continues to run, allowing the user to maintain a timer base while the rest of the device is sleeping. The ADC noise reduction mode stops the CPU and all I/O modules except asynchronous timer and ADC, to minimize switching noise during ADC conversions. In standby mode, the crystal/resonator oscillator is running while the rest of the device is sleeping. This allows very fast start-up combined with low power consumption.

The ATmega328P AVR is supported with a full suite of program and system development tools including: C compilers, macro assemblers, program debugger/simulators, in-circuit emulators, and evaluation kits [5].

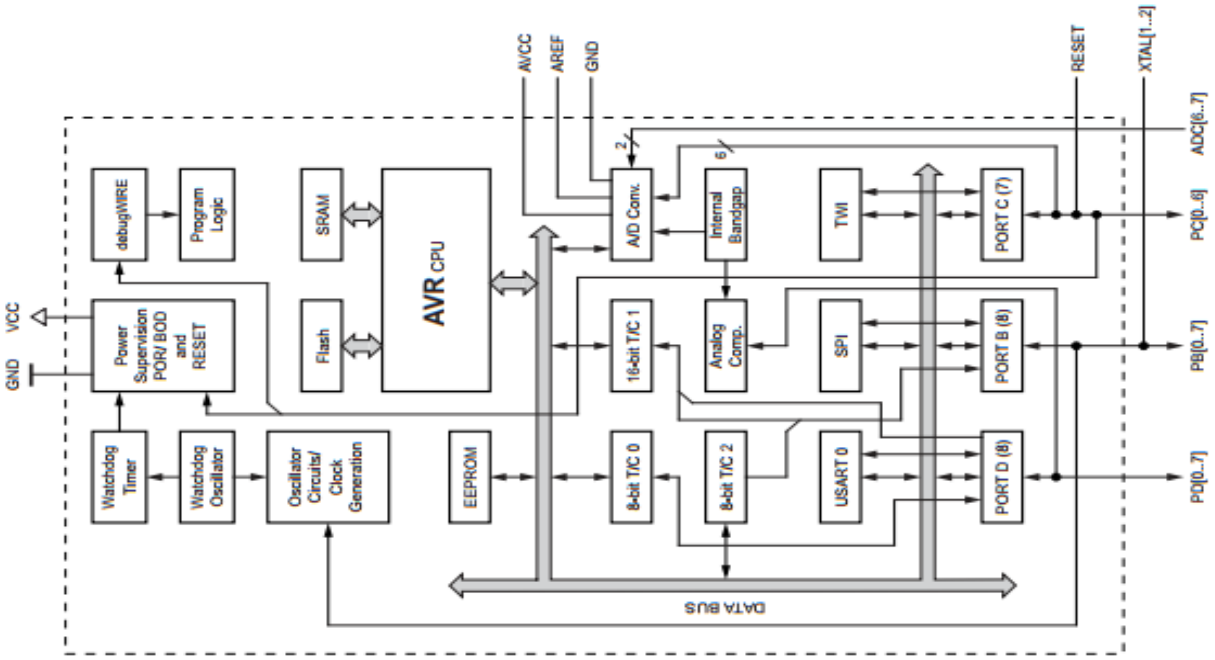


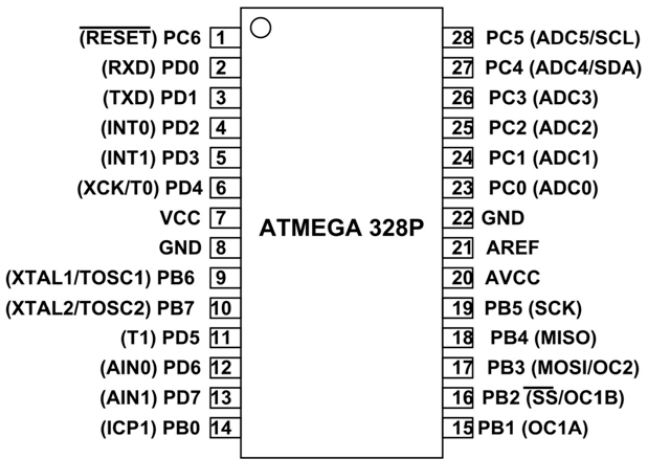
FIGURE 1: Block Diagram of Atmega328p Microcontroller [5].

2.1.2 Pin Descriptions of Atmega328p Microcontroller

- VCC: Digital supply voltage [5].
- GND: Ground [5].
- Port B (PB7:0) XTAL1/XTAL2/TOSC1/TOSC2: Port B is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port B output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, port B pins that are externally pulled low will source current if the pull-up resistors are activated. Depending on the clock selection fuse settings, PB6 can be used as input to the inverting oscillator amplifier and input to the internal clock operating circuit. Depending on the clock selection fuse settings, PB7 can be used as output from the inverting oscillator amplifier. If the internal calibrated RC oscillator is used as chip clock source, PB7-6 is used as TOSC2-1 input for the asynchronous Timer/Counter2 if the AS2 bit in ASSR is set [5].
- Port C (PC5:0): Port C is a 7-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The PC5-0 output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port C

pins that are externally pulled low will source current if the pull-up resistors are activated. The port C pins are tri-stated when a reset condition becomes active, even if the clock is not running [5].

- PC6/RESET: If the RSTDISBL fuse is programmed, PC6 is used as an input pin. If the RSTDISBL fuse is unprogrammed, PC6 is used as a reset input. A low level on this pin for longer than the minimum pulse length will generate a reset, even if the clock is not running [5].
- Port D (PD7:0): Port D is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The port D output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, port D pins that are externally pulled low will source current if the pull-up resistors are activated. The port D pins are tri-stated when a reset condition becomes active, even if the clock is not running [5].
- AVCC: AVCC is the supply voltage pin for the A/D converter, PC3:0, and ADC7:6. It should be externally connected to VCC, even if the ADC is not used. If the ADC is used, it should be connected to VCC through a low-pass filter. Note that PC6-4 use digital supply voltage, VCC. [5]
- AREF: AREF is the analog reference pin for the A/D converter. [5]
- ADC7:6 (TQFP and QFN/MLF Package Only): In the TQFP and QFN/MLF package, ADC7:6 serve as analog inputs to the A/D converter. These pins are powered from the analog supply and serve as 10-bit ADC channels. [5]



`

FIGURE 2: Pin Configuration of Atmega328p Microcontroller [6].

2.1.3 ADMUX Register Description

ADMUX Register

7	6	5	4	3	2	1	0
REFS1	REFS0	ADLAR	MUX4	MUX3	MUX2	MUX1	MUX0

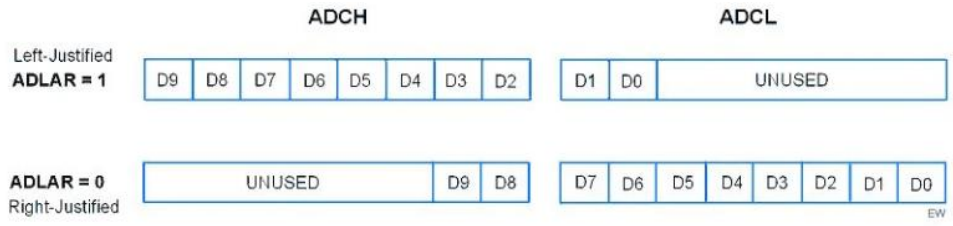
- Bit 7: 6 – REFS1 : 0: Reference Selection Bits

Table 1: Reference voltage selection for ADC [7]

REFS1	REFS0	Vref to ADC
0	0	AREF pin
0	1	AVCC pin i.e. Vcc 5 V
1	0	Reserved
1	1	Internal 2

- Bit 5 – ADLAR: ADC Left Adjust Result

  Use 10-bits output as upper bits or lower bits in ADCH & ADCL [7].



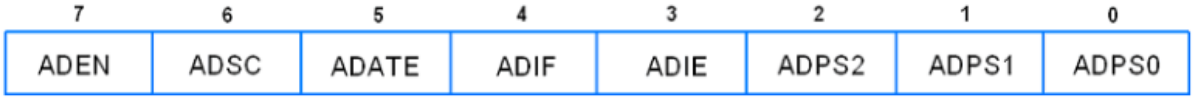
- Bits 4:0 – MUX4:0: Analog Channel and Gain Selection Bits

  We can select input channel ADC0 to ADC7 by using these bits. These bits are also used to select comparator (inbuilt in AVR) inputs with various gain. We will cover these comparator operations in another part.

  Suppose connecting the input to ADC channel 2, we need to put 00010 in MUX4:0 [7].

2.1.4 ADCSRA Register Description

ADCSRA Register:



- Bit 7 – ADEN: ADC Enable

Writing one to this bit enables the ADC. By writing it to zero, the ADC is turned off. Turning the ADC off while a conversion is in progress, will terminate this conversion [7].

- Bit 6 – ADSC: ADC Start Conversion

  Writing one to this bit starts the conversion [7].

- Bit 5 – ADATE: ADC Auto Trigger Enable

  Writing one to this bit, results in Auto Triggering of the ADC is enabled [7].

- Bit 4 – ADIF: ADC Interrupt Flag

  This bit is set when an ADC conversion completes and the Data Registers are updated [7].

- Bit 3 – ADIE: ADC Interrupt Enable

  Writing one to this bit, the ADC Conversion Complete Interrupt is activated [7].

- Bits 2:0 – ADPS2:0: ADC Pre-scaler Select Bits

  These bits determine the division factor between the XTAL frequency and the input clock to the ADC [7].

Table 2: Division Factor Selection for ADC [7].

ADPS2	ADPS1	ADPS0	Division Factor
0	0	0	2
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

We can select any divisor and set frequency F/2, F/4, etc. for ADC. But in AVR, ADC requires an input clock frequency less than 200KHz for max. accuracy. So, we must always take care of not exceeding ADC frequency more than 200KHz.

Suppose that clock frequency of AVR is 8MHz, then we must use devisor 64 or 128. Because it gives 8MHz/64 = 125KHz, which is lesser than 200KHz [7].

2.2 Arduino Uno

Arduino UNO (Figure 1) is a microcontroller board based on the ATmega328P. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz ceramic resonator (CSTCE16M0V53-R0), a USB connection, a power jack, an ICSP header and a reset button. [8]

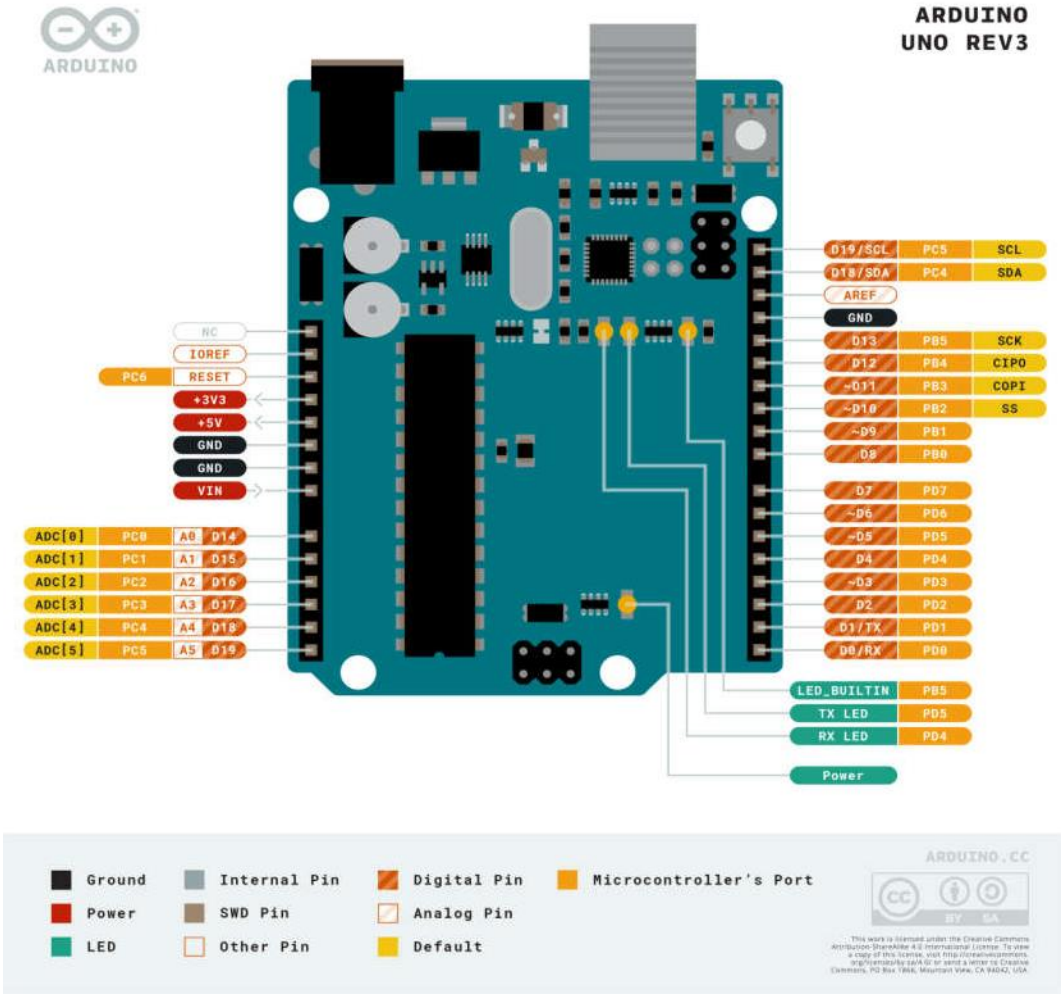


Figure 3: Pictorial view of Arduino UNO

2.2.1 Features of Arduino UNO Board

The features of Arduino Uno ATmega328 include the following:

- The operating voltage is 5V
- The recommended input voltage will range from 7v to 12V
- The input voltage ranges from 6v to 20V
- Digital input/output pins are 14
- Analog i/p pins are 6
- DC Current for each input/output pin is 40 mA
- DC Current for 3.3V Pin is 50 mA
- Flash Memory is 32 KB
- SRAM is 2 KB
- EEPROM is 1 KB
- CLK Speed is 16 MHz [8]

Chapter Three

Software Description

In this section highlighted on the desired software components with special emphasis on Arduino code, Proteus software. Primarily, Arduino UNO board is connected with the Laptop through USB port and the board gets power from the Laptop. In Laptop, Arduino IDE 2.0 (beta) version is downloaded from the Arduino software page. [9] Using Arduino IDE, a sketch code is developed and compiled to run. The sketch code used in Arduino IDE is given below.

3.1 Arduino IDE

The Arduino Integrated Development Environment (IDE) is an open-source software platform used for programming and developing projects with Arduino boards. Arduino is a popular platform for creating interactive electronic projects, prototyping, and building various electronic systems. The Arduino IDE provides an intuitive interface to write, compile, and upload code to Arduino boards, making it accessible for both beginners and experienced programmers.

3.1.1 Program

```
#include "EmonLib.h"   // Include Emon Library
EnergyMonitor emon1;   // Create an instance

// include the library code:
#include <LiquidCrystal_I2C.h>  //library for LCD including I2C
#include <math.h> //library for mathematical calculation

LiquidCrystal_I2C lcd(0x27, 16, 2); // initialize the library with the LCD I2C
display interface

int X;
int Y;
int Voltage = 0;
float Time;
float frequency;
const int input = A1; //Frequency input pin
const int test = 9;
const int Relay_PIN = 7;
const int LED_PIN = 6;
```

```
void setup()
{
  Serial.begin(9600);
  pinMode(Relay_PIN, OUTPUT);  // Set the pin as OUTPUT
  pinMode(LED_PIN, OUTPUT);  // Set the LED pin as OUTPUT
  pinMode(input,INPUT);
  pinMode(test, OUTPUT);
  analogWrite(test,127);

  emon1.voltage(A0,187, 1.7); // Voltage: input pin, calibration, phase_shift

  lcd.begin();
  lcd.backlight();


}

void loop()
{
  emon1.calcVI(20,2000); // Calculate all. No.of half wavelengths (crossings),
time-out
  Voltage   = emon1.Vrms;  //extract Vrms into Variable

  X=pulseIn(input,HIGH);
  Y=pulseIn(input,LOW);
  Time = X+Y;
  frequency=1000000/Time;

  if (Time == 0)
  {
   frequency = 0;
  }

  lcd.setCursor(0,0);
  lcd.print("V=");
  lcd.print(Voltage);
  lcd.print("V, ");
  lcd.print("F=");
  lcd.print(frequency);
  lcd.print("Hz");

//NV=Normal Voltage, NF=Normal Frequency, UV=Under Voltage, UF=Under
Frequency, OV=Over Voltage, OF=Over Frequency

if(200<=Voltage<=240 && 49<=frequency<=52)
{
    digitalWrite(Relay_PIN, HIGH);
    digitalWrite(LED_PIN, LOW);
```

```
    lcd.setCursor(0,1);
    lcd.print("NV, NF");
}

if(200<=Voltage<=240 && frequency>52)
{
    digitalWrite(Relay_PIN, LOW);
    digitalWrite(LED_PIN, HIGH);
    lcd.setCursor(0,1);
    lcd.print("NV, OF");
}

if(200<=Voltage<=240 && frequency<49)
{
    digitalWrite(Relay_PIN, LOW);
    digitalWrite(LED_PIN, HIGH);
    lcd.setCursor(0,1);
    lcd.print("NV, UF");
}

if(Voltage<200 && 49<=frequency<=52)
{
    digitalWrite(Relay_PIN, LOW);
    digitalWrite(LED_PIN, HIGH);
    lcd.setCursor(0,1);
    lcd.print("UV, NF");
}

if(Voltage>240 && 49<=frequency<=52)
{
    digitalWrite(Relay_PIN, LOW);
    digitalWrite(LED_PIN, HIGH);
    lcd.setCursor(0,1);
    lcd.print("OV, NF");
}

if(Voltage>240 && frequency>52)
{
    digitalWrite(Relay_PIN, LOW);
    digitalWrite(LED_PIN, HIGH);
    lcd.setCursor(0,1);
    lcd.print("OV, OF");
}

if(Voltage>240 && frequency<49)
{
    digitalWrite(Relay_PIN, LOW);
    digitalWrite(LED_PIN, HIGH);
```

```
    lcd.setCursor(0,1);
    lcd.print("OV, UF");
}

if(Voltage<200 && frequency<49)
{
    digitalWrite(Relay_PIN, LOW);
    digitalWrite(LED_PIN, HIGH);
    lcd.setCursor(0,1);
    lcd.print("UV, UF");
}

if(Voltage<200 && frequency>52)
{
    digitalWrite(Relay_PIN, LOW);
    digitalWrite(LED_PIN, HIGH);
    lcd.setCursor(0,1);
    lcd.print("UV, OF");
}

  delay(200);

}
```

3.2 Proteus Software

Proteus software is a specialized software tool suite used primarily for electronic design automation. It is used mainly by electronic design engineers and technicians to create schematics and electronic prints for manufacturing printed circuit boards.

Before hardware implementation, I did the simulation of this project using Proteus Software to ensure the proper working of the system for safety purposes.

Proteus Professional 8 version is used in this simulation. [10]

Chapter Four

Methodology

4.1 Overview of the Project

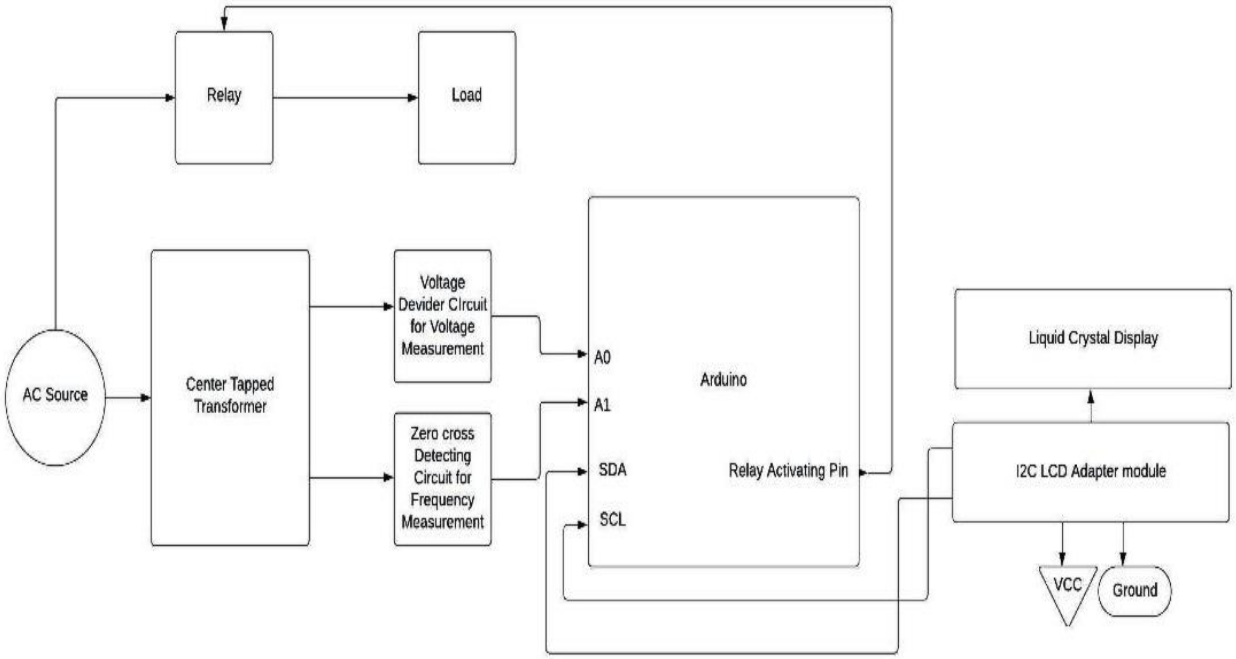


FIGURE 4: Schematic Diagram of the Project.

The AC supply voltage is passed through the center-tapped step-down transformer for lowering the line AC voltage into a 12V supply. The one terminal output of the transformer is connected to a voltage divider circuit for converting the voltage within 5V for connecting to the Arduino ADC terminal. The other output terminal of the transformer is passed through the zero cross-detecting circuit for frequency measurement also converting the voltage within 5V for the ADC A1 input pin of the Arduino. A Liquid Crystal Display (a I2C converter to interface LCD) connected to Arduino is used for showing the result of the output. The protective relay is connected via load to the system. So that whenever the voltage or frequency exceeds the threshold limit relay will operate and open the load from the supply.

4.2 Analog to Digital conversion by Atmega328

Arduino Uno is a microcontroller board having atmega328p 8-bit microcontroller. Atmega328p has a 10-bit ADC (analog-to-digital converter) having 2^{10}=1024 steps ranging from 0 to 1023. In this project I have used maximum 5V as the input voltage of the ADC. So, the resolution of the ADC is 5V/1024 = 4.88 mV. [7]

First the ADC value is to be stored in a variable. Secondly this digital which is stored into a variable is multiplied with resolution 4.88mV. By doing this multiplication the resultant voltage will be in milli-volt unit. In order to get the output voltage in volt unit the resolution 4.88 of unit milli-volt must be divided by the 1000 [7].

4.3 Protective Relay

A relay functions as an electrically driven switch, comprising input terminals for single or multiple control signals, alongside a collection of operational contact terminals. The switch can incorporate numerous contacts in diverse forms including make contacts, break contacts, or a blend of these configurations. Relays find application in scenarios necessitating circuit regulation through an autonomous low-power signal or the centralized control of multiple circuits via a solitary signal. [12]

FIGURE 5: Single Channel 5V Relay Module

In this project I have used a simple 5V DC relay in order to provide over/under voltage or over/under frequency protection with minimum delay or tripping time. Here, delay or tripping time is a very important factor. If the tripping time is much larger then there is a risk of damaging the electrical appliances or in come this may lead to fire hazards.

The phase of the AC source is connected through a normally close (NC) of the Relay and the relay coil excitation is given from a digital pin of Arduino Uno. In this case I have used digital pin number 7 as relay coil excitation.

ADC of the atmega328p is continuously monitoring the load current. Here, I have taken a threshold under-voltage of 200V and over-voltage of 240V. This means out of the range 200-240V of the voltage, the digital pin no. 7 of the Arduino Uno will be high which connected to the relay coil. Thus, energizing the relay coil and make the relay operate. Same as the threshold limit of the under-frequency of 49Hz and over-frequency of 52Hz exceeds the limit the relay will operate.

After the operation of the Relay normally close (NC) contact will be opened and normally open (NO) contact will be closed. Thus, disconnecting the load from the supply. Ideally this incident should be happened instantly. But, in practical there will some delay for delay operation which is known as tripping time. However, this delay should be as small as possible in order to achieve a reliable electricity supply. [12]

4.4 Liquid Crystal Display

A liquid-crystal display (LCD) is a flat-panel display or other electronically modulated optical device that uses the light-modulating properties of liquid crystals combined with polarizers. Liquid crystals do not emit light directly but instead use a backlight or reflector to produce images in color or monochrome. [13]

After analog-to-digital conversion by the ADC of atmega328p it is necessary to visually see the voltage and frequency condition. It is also seen that when the relay trip the circuit the reason is displayed.

In order to display different data on a screen I have used a 16x2 Liquid Crystal Display (LCD) in my project. VDD should be supplied by 5V DC voltage, VSS should be grounded and VEE connected with a potentiometer to adjust the contrast of the LCD.

FIGURE 6: Liquid Crystal Display (LCD). [13]

4.5 I2C LCD Adapter Module

Interfacing an 16x2 LCD with any microcontroller is a difficult task. In order to reduce the difficulties of interfacing an LCD with a microcontroller I2C LCD Adapter module is used.

FIGURE 7: I2C LCD Adapter module.

In this project I have also used a used a I2C LCD Adapter module. I2C LCD Adapter module has only 4 pins where in an LCD there are 16 different pins available. So, using a I2C LCD adapter module reduces the number of pins significantly. [14]

I2C LCD Adapter module has a VCC pin, a GND pin, an SDA pin and an SCL pin. VCC pin is supplied by a 5V DC voltage and GND pin is grounded. Whereas SDA and SCL pin are connected to the SDA and SCL pin of the Arduino Uno boar.

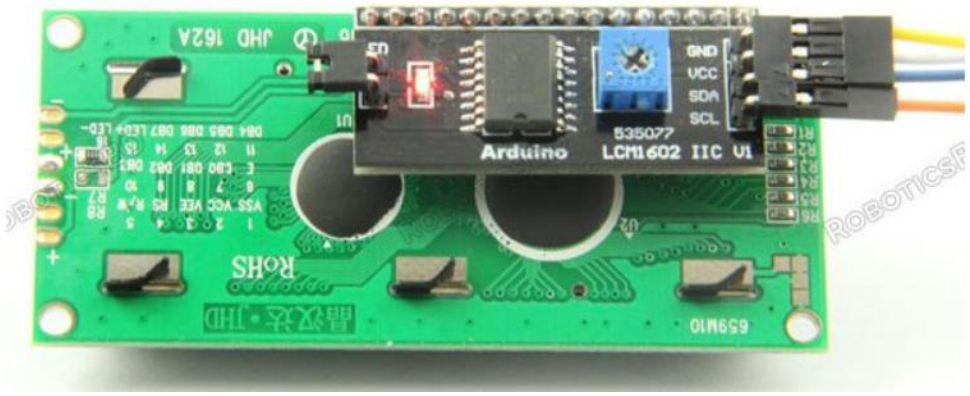


FIGURE 8: Connecting the I2C LCD Adapter module with a 16x2 LCD.

In order to display the data on LCD screen I have used a library called LiquidCrystal_I2C which available on Github and Arduino library manager.

4.6 Final Circuit Diagram

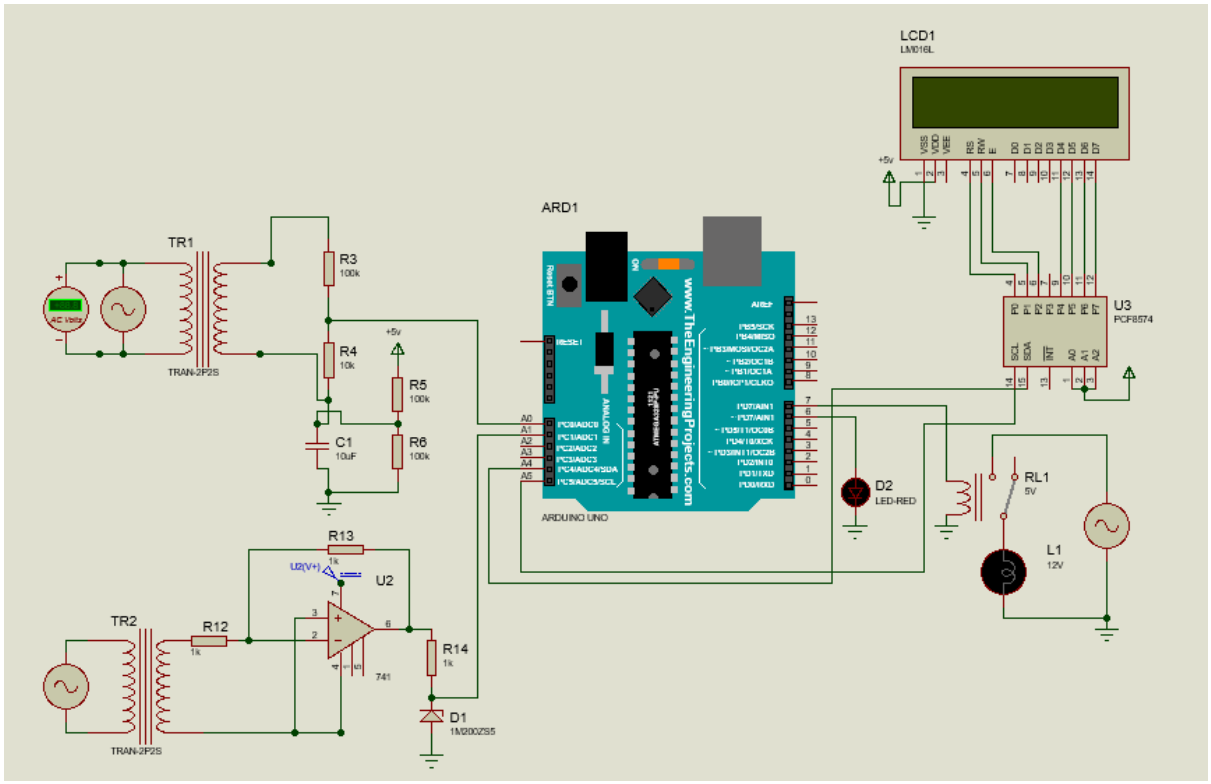


FIGURE 9: Circuit Diagram of Over/Under Voltage/Frequency Protection

4.7 Simulation When Normal Voltage and Frequency Supplied

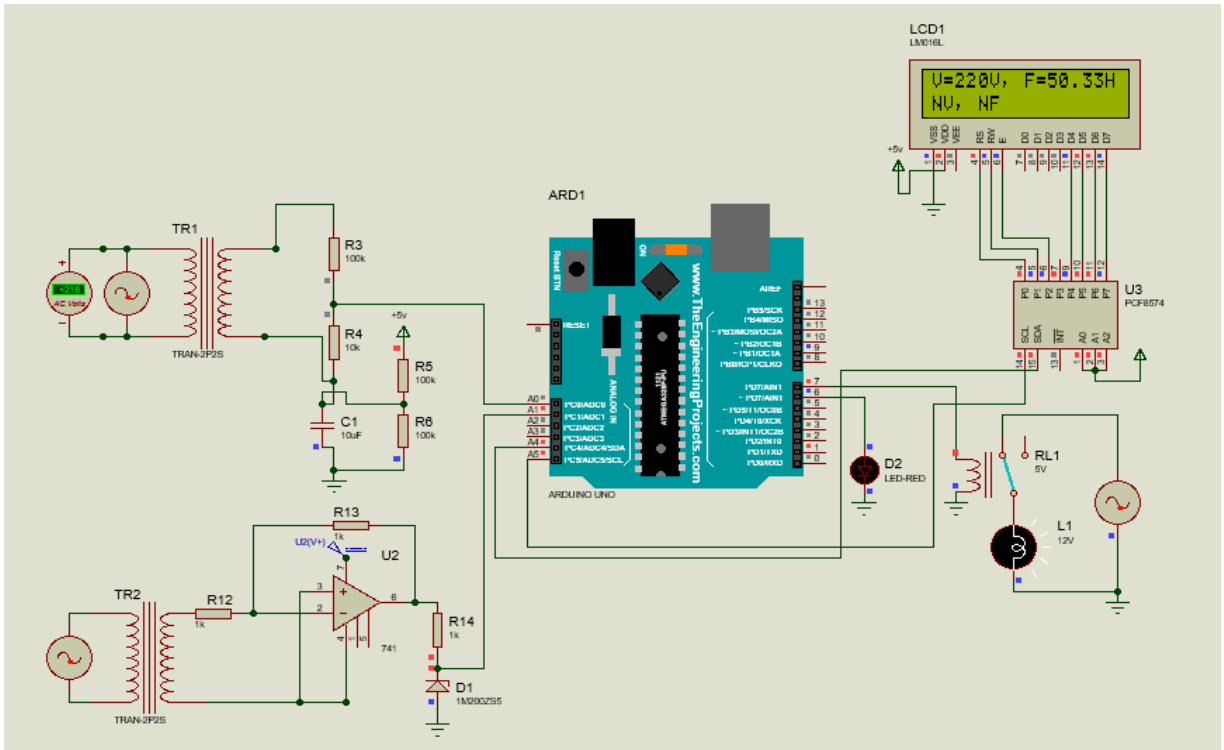


FIGURE 10: Simulation When Normal Voltage and Frequency Supplied

4.8 Simulation When Under Voltage and Normal Frequency Supplied

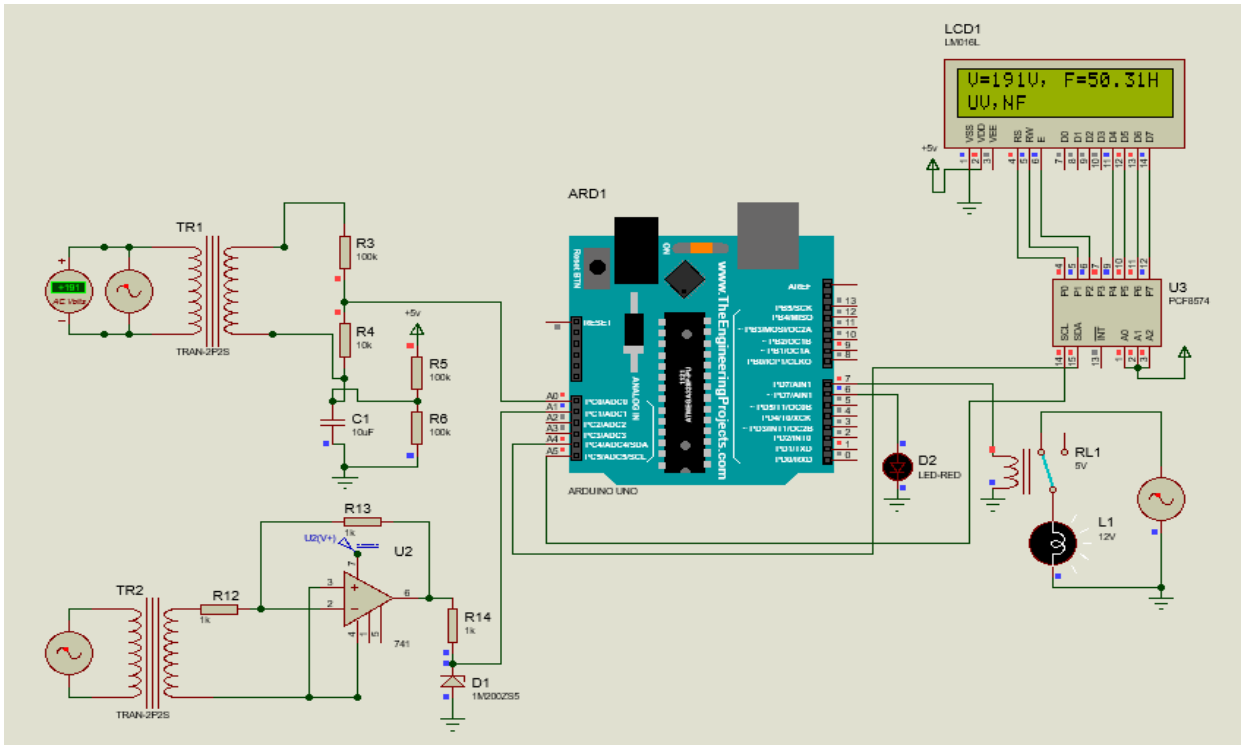


FIGURE 11: Simulation When Under Voltage and Normal Frequency Supplied

4.9 Simulation When Normal Voltage and Over Frequency Supplied

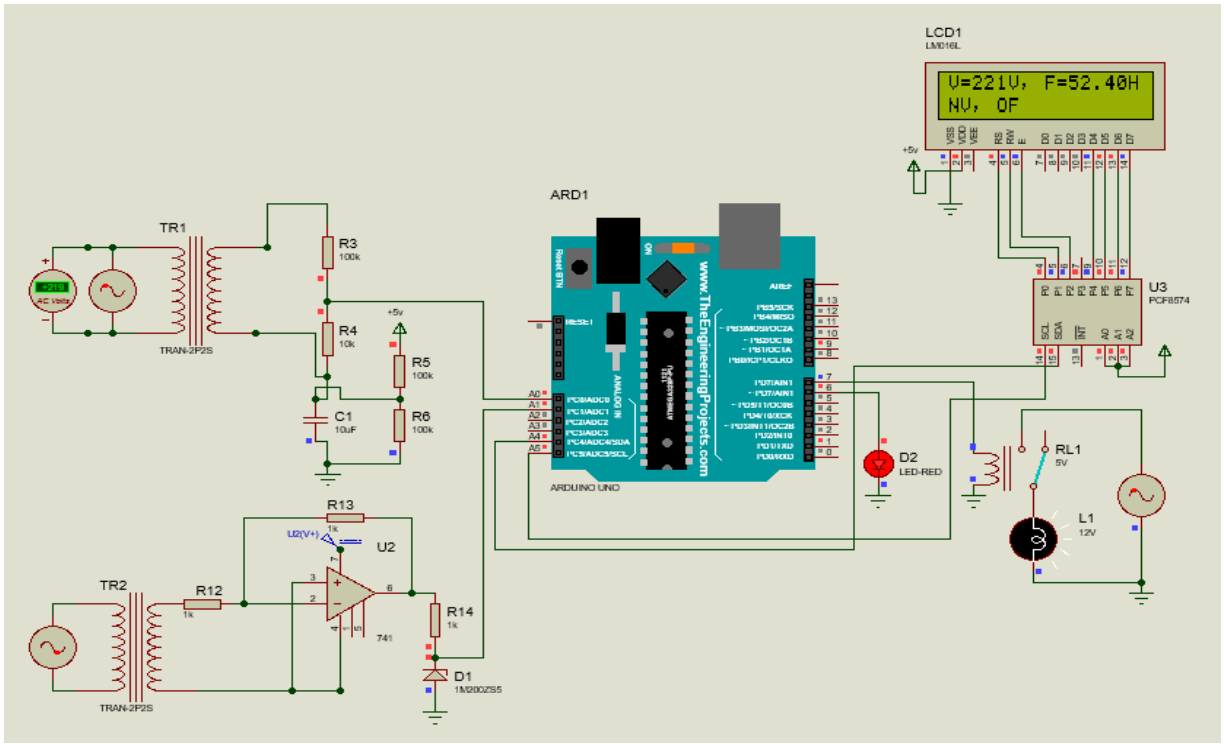


FIGURE 12: Simulation When Normal Voltage and Over Frequency Supplied

Chapter Five

Results and Discussions

5.1 Results and Discussions

In my project work significant emphasis has been given to the development of a microcontroller-based AC voltage and frequency protection system. The protective relay is used to make the system safe when the voltage or frequency exceeds the operating range. I defined the under-voltage condition when it is less than 200V and the over-voltage condition is greater than 240V, when the ac supply voltage exceeds this range of 200-240V the load becomes disconnected by a protective relay. Again, the under-frequency condition is defined at less than 49Hz and the over-frequency is greater than 52Hz, exceeding this range and making the system isolated from the supply. The microcontroller continuously monitors the data of AC supply voltage and frequency and when the data exceeds the range, the relay will be operated and interrupt the supply. In such cases, the fault duration will be a little longer. Because a delay is introduced for energizing the relay coil. After the relay coil is energized, the relay isolates the power circuit. I adjusted protection thresholds based on the specific requirements of the connected equipment. This feature enhances the system's versatility, making it suitable for a wide range of applications across different industries.

During this project, I have seen a little bit of voltage and frequency data fluctuations on the LCD, due to the system's components are not full of identical properties.

5.2 Future Enhancement

While the microcontroller-based AC voltage and frequency protection system achieved its core objectives, there are areas for potential enhancement. Further development could involve integrating communication capabilities, allowing the system to send alerts to remote monitoring stations or trigger automated shutdown procedures. Additionally, the system could benefit from predictive algorithms that anticipate voltage and frequency anomalies based on historical data, enabling even more proactive protection strategies.

5.3 Conclusion

The successful construction and testing of the microcontroller-based AC voltage and frequency protection system validate its efficacy in safeguarding electrical systems against voltage and frequency anomalies. This project underscores the significance of integrating microcontroller technology with electrical engineering principles to address critical challenges in maintaining system stability and equipment integrity. The system's real-time monitoring, anomaly detection, and rapid response mechanisms contribute to its value in various applications, from industrial settings to residential environments.

5.4 Appendix

Price list of components used:

Components	**Price(Taka)**
Arduino Uno Board	850.00
Protective Relay	30.00
Liquid Crystal Display	200.00
I2C LCD Adapter module	120.00
Others (Jumper wires, electric wires, 2 pin plugs, transformer, breadboard etc.)	250.00
Total cost in Bangladeshi Taka	1450.00

References

[1] Savita, Shrivastava, S., Arora, A., & Varshney, V. (2018). Overvoltage and Undervoltage Protection of Load using GSM modem SMS Alert. 2018 2nd IEEE International Conference on Power Electronics, Intelligent Control and Energy Systems (ICPEICES).

[2] Omowumi Grace Olasunkanmi, et al. "Design and Development of Microcontroller
Based AC Mains Monitor Circuit." *ResearchGate*, unknown, 2019.

[3] wikipedia, "Microcontroller," August 2023. [Online]. Available: https://en.wikipedia.org/wiki/Microcontroller.

[4] Kwon, H.; Kim, Y.B.; Seo, S.C.; Seo, H. High-Speed Implementation of PRESENT on AVR Microcontroller. Mathematics 2021, 9, 374.

[5] "Datasheet" of "Atmega328p" available on: https://www.microchip.com/en-us/product/atmega328p#document-table

[6] "Pin Diagram" of "Atmega328p" available on: https://components101.com/microcontrollers/atmega328p-pinout-features-datasheet

[7] "ADC" of "ATmega328p" available on: https://www.electronicwings.com/avr-atmega/atmega328p-adc

[8] Arduino UNO available on: https://store-usa.arduino.cc/products/arduino-uno-rev3

[9] Arduino IDE available on: https://www.arduino.cc/en/Tutorial/getting-started-with-ide-v2

[10] Proteus Software available on: https://www.labcenter.com/downloads/

[11] Sam, R., Mohd Nor Md Tan, & Mohamad Safari Ismail. (2013). Quad-copter using ATmega328 microcontroller. 2013 International Conference on Electrical Machines and Systems (ICEMS).

[12] wikipedia "Relay", 2023, Available on: https://en.wikipedia.org/wiki/Relay

[13] wikipedia "Liquid Crystal Display", 2023 Available on: https://en.wikipedia.org/wiki/Liquid-crystal_display

[14] Gay, W. (2017). I2C LCD Displays. Custom Raspberry Pi Interfaces, 35–54