FINAL YEAR PROJECT REPORT

BS Electrical Engineering

Batch 2020-2024

Development of a Power Monitoring and Control System for Smart Grid

DEPARTMENT OF ELECTRICAL ENGINEERING AND TECHNOLOGY

GOVERNMENT COLLEGE UNIVERSITY, FAISALABAD

2024

Submitted by

Mubashar Bashir	12708
Muhammad Aamir Shahzad	12715
Muhammad Zahid	12727
Muhammad Umer Farooq (Group Leader)	12766

Project Supervisor

Engr. Dr. Rehan Liaqat

Signature: ………………………………….

DEPARTMENT OF ELECTRICAL ENGINEERING AND TECHNOLOGY
GOVERNMENT COLLEGE UNIVERSITY, FAISALABAD

2024

COPYRIGHT STATEMENT

DECLARATION

We certify that this Final Year Project Titled *"Development of a Power Monitoring and Control System for Smart Grid"* is our project. The project has not been presented elsewhere for assessment. The material that has been used from other sources has been properly acknowledged /referred.

Name: MUBASHAR BASHIR Roll No: 12708 Signature: _________

Name: M. AAMIR SHAHZAD Roll No: 12715 Signature: _________

Name: MUHAMMAD ZAHID Roll No: 12727 Signature: _________

Name: M. UMER FAROOQ Roll No: 12766 Signature: _________
(Group Leader)

ACKNOWLEDGEMENTS

First, we would like to thank Allah Almighty for giving us a beautiful life, after that, we would like to thank our parents for providing us with this golden opportunity where we stand now just because of them.

We would like to express our heartfelt gratitude to our supervisor, **Engr. Dr. Rehan Liaqat**, for his exceptional guidance and unwavering support throughout this thesis. His expertise and valuable insights have been instrumental in shaping the direction of our research. We are grateful for his constant encouragement, feedback, and mentorship, which have challenged us to push the boundaries of our knowledge and skills.

We extend our appreciation to the Department of Electrical Engineering for providing a conducive academic environment and resources that have facilitated our research. The department's commitment to excellence in education and research has been a constant source of inspiration. We are grateful for the opportunities to collaborate with faculty members and fellow students, whose diverse perspectives have enriched our learning experience. We would like to thank our families and loved ones for their unwavering support and understanding throughout this journey. Their belief in our abilities and encouragement have been our source of strength. We are forever grateful for their sacrifices and the countless ways they have supported us in pursuing our academic goals.

Table of Contents

List of Figures

ABSTRACT

Electric energy systems are fundamentally important to industrialized society, responsible for supplying the electrical energy needed for things as basic as lighting and heating but also as essential in contemporary life as manufacturing and transportation. When power quality is not reliable, so hardware will be easily damaged, energy consumption will be increased, and system efficiency will be reduced. This has further increased the importance of monitoring and analyzing the power quality of electrical smart grids. A smart grid is an electrical grid which includes a variety of operational and energy measures including smart meters, smart appliances, renewable energy resources, and energy efficient resources.

The project aims to construct a monitoring and analysis system of the quality of electrical power network. The purposes of the project are to be identifying problems in connection to power quality like harmonic distortion, power factor imbalances, and voltage fluctuations and to act corrective activities like power factor correction or voltage regulation for a better power quality for decreasing energy waste. Ultimately this will result in a more resilient and efficient electrical grid resulting in cost savings, less down time and happier customers.

The project will test and evaluate the system against large changes in power demand before it is field-tested on a live electrical grid. The project aims to create a reference document that will be useful to power utilities, system operators and other stakeholders hoping to improve the power quality of electrical grids.

List of Tables

CHAPTER 1: INTRODUCTION

1.1 Background

Electric power systems are critical to modern society, providing the energy required for everything from lighting and heating to manufacturing and transportation. However, poor power quality can result in equipment damage, increased energy consumption, and reduced system efficiency. Therefore, the need for monitoring and analyzing the power quality of electrical smart grids has become increasingly important. A smart grid is an electricity network that uses digital and other advanced technologies to monitor and manage the transport of electricity from all generation sources to meet the varying electricity demands of end users.

The purpose of this project is to develop a system for monitoring and analyzing the power quality of an electrical grid. The objectives of the project are to identify issues related to power quality such as harmonic distortion, power factor imbalances, and voltage fluctuations, and to implement corrective actions such as power factor correction or voltage regulation to improve power quality and reduce energy waste. The project will improve the reliability and efficiency of the electrical grid, leading to cost savings, reduced downtime, and increased customer satisfaction.

This project will involve selecting appropriate sensors and measurement devices, collecting and processing data, identifying corrective actions, implementing those actions, and continuously monitoring the system to ensure that it is working effectively. The project will be conducted on a simulated electrical smart grid to test and evaluate the system before implementation on a real, world electrical grid. The results of the project will be useful to power utilities, system operators, and other stakeholders interested in improving the power quality of electrical grids.

1.2 Literature Review

Power quality parameters such as voltage, power factor, and harmonics are crucial for ensuring the efficient operation of electrical grids. Monitoring and analyzing these parameters are essential for identifying issues and implementing effective solutions [1]. Recent advancements in power quality monitoring include the utilization of high-resolution power quality analyzers, smart grid technologies, and sophisticated software tools. These tools enable real-time data collection and analysis, allowing for the detection of transient events and long-term trends [2].

Harmonic distortion, low power factor, and voltage fluctuations are common issues affecting power quality. Identifying these issues is essential for implementing corrective measures and improving grid reliability [3]. Harmonic distortion arises from non-linear loads, while low power factor and voltage fluctuations result in inefficiencies and operational challenges. Detection and diagnosis techniques such as harmonic analysis and power factor measurement are employed to identify and address these issues [4].

Various power quality improvement devices, including harmonic filters, power factor correction devices, and voltage regulators, play a vital role in enhancing grid reliability and efficiency [5]. Harmonic filters mitigate harmonic distortion, while power factor correction devices and voltage regulators address power factor issues and voltage fluctuations, respectively. These devices are essential for maintaining stable and efficient grid operations [6].

Monitoring and analyzing power quality parameters, including voltage, power factor, and harmonics, is vital for maintaining the efficiency and reliability of electrical grids. Advanced monitoring technologies and analytical methods facilitate the detection and mitigation of power quality issues [7].

High-resolution power quality analyzers and smart grid technologies are instrumental in real-time data collection and analysis. Software tools like MATLAB, Simulink, and ETAP are widely used for simulating and analyzing power quality data. Machine learning algorithms have also been applied for predictive analysis, enhancing the capability to foresee and address disturbances [8].

1.3 Objectives

- To monitor and analyze the power quality parameters of an electrical grid including voltage, power factor, and harmonics.
- To identify issues related to power quality such as harmonic distortion, low power factor, and voltage fluctuations.
- To improve the reliability and efficiency of the electrical grid using power quality improvement devices.

CHAPTER 2: PROJECT METHODOLOGY

2.1 Proposed Model

1. Sensors and measurement devices: To monitor the power quality of an electrical grid, sensors and measurement devices will be installed. These may include voltage and current sensors, power analyzers, and data loggers. Some examples of sensors that can be used include Rogowski coils, current transformer, and potential transformer.
2. Communication devices: To transfer data from the sensors and measurement devices to a central processing unit, communication devices such as Ethernet switches or wireless communication devices are used.
3. Central processing unit: The central processing unit is responsible for collecting and processing data from the sensors and measurement devices. Software tools such as Arduino IDE are employed to analyze the data and identify any issues related to power quality.
4. Power factor correction devices: For power factor correction, devices such as capacitors or static var compensators are used.
5. Voltage regulation devices: If voltage regulation is required, voltage regulation devices such as transformers with on-load tap changers are used.
6. Harmonic filter: To protect the system from excessive harmonic level, a harmonic filter can be used.

2.2 Hardware

Hardware components used in this project are given in table 1.

Final hardware view is given in the Figure 2:

Sr. No	Components
1	LCD
2	Inter-Integrated Circuit (I2C)
3	Arduino
4	Current Sensor
5	Voltage Sensor
6	Voltage Stabilizer Transformer
7	220V Resistive load
8	220V Variable load

Table 1: Hardware Components

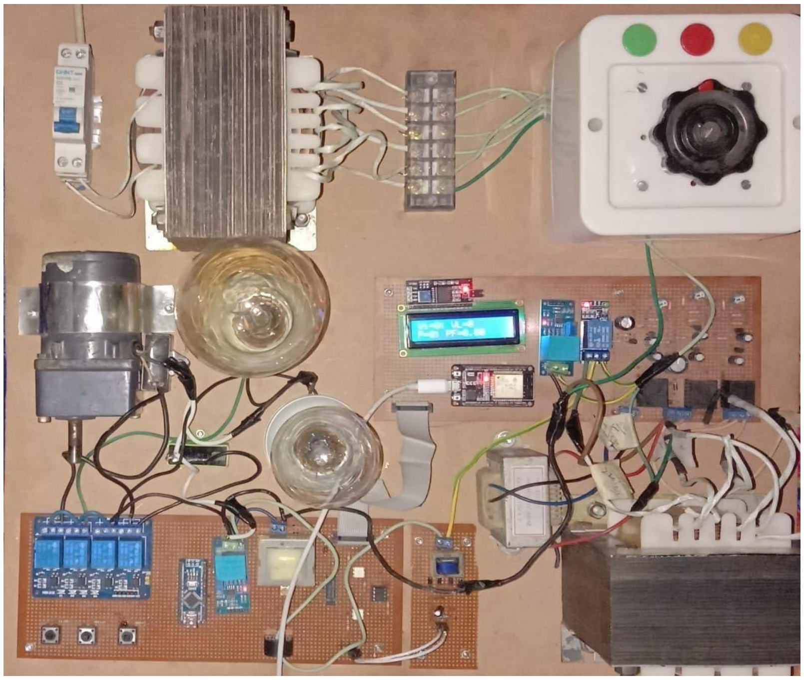


Figure 2: Hardware

2.2.1 LCD

- A Liquid Crystal Display (LCD) is a commonly used output device in electronics. It allows to display information and messages to the user in a visual format. • The physical display screen that shows characters or graphics
- Black text on a green background.
- This LCD has two registers, namely, Command and Data.
- This is a standard HD44780 controller LCD.
- LCD is shown in Figure 3.

Figure 3: LCD

2.2.2 Integrated Circuit (I2C)

I2C uses two lines to send and receive data:

- a serial clock pin (SCL) that the Arduino Controller board pulses at a regular interval
- a serial data pin (SDA) over which data is sent between the two devices.
- Serial Communication.
- Half Duplex.
- I2C is shown in Figure 4.

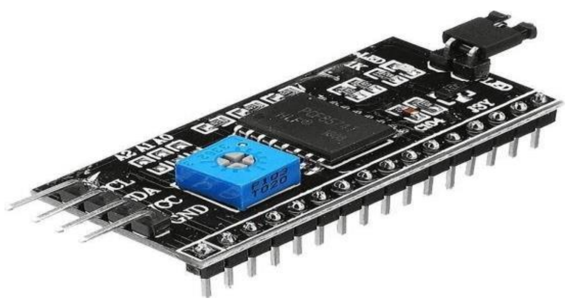


Figure 4: I2C

2.2.3 Arduino

- Microcontroller
- Open Source: Hardware and software are freely available.
- IO Pins: 14 digital, 6 analog input/output pins.
- Power Supply: USB or external power (5V).
- Programming: Arduino IDE, based on Wiring language.
- Community Support: Large and active user community.
- Versatility: Suitable for a wide range of projects.
- Shields: Compatible with expansion boards for added functionalities.
- Arduino Nano is shown in Figure 5.

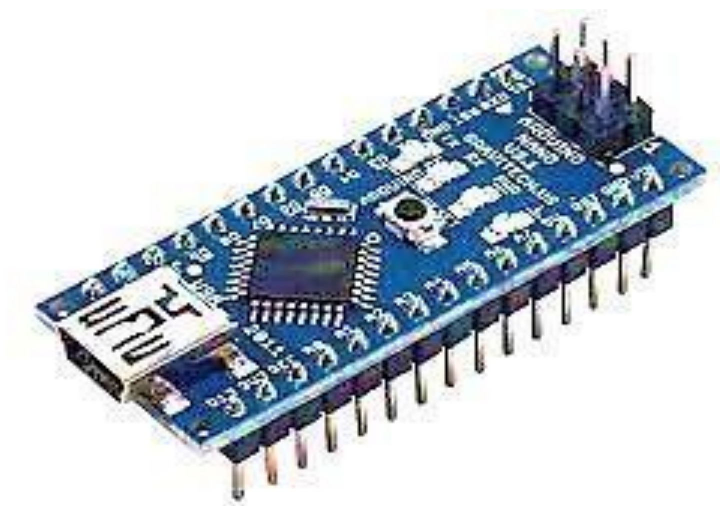


Figure 5: Arduino Nano

2.2.4 Current Sensor

- Used to measure electric current.
- Measurement of current ranges from picoamps to tens of thousands of amperes. Its VCC pin connects with the Microcontroller Volt pin. The OUT pin connects with the Analog pin. The current sensor output pin connects with Automatic change over where it monitors the consumption of load current.

  Figure 6 shows the Current Sensor.

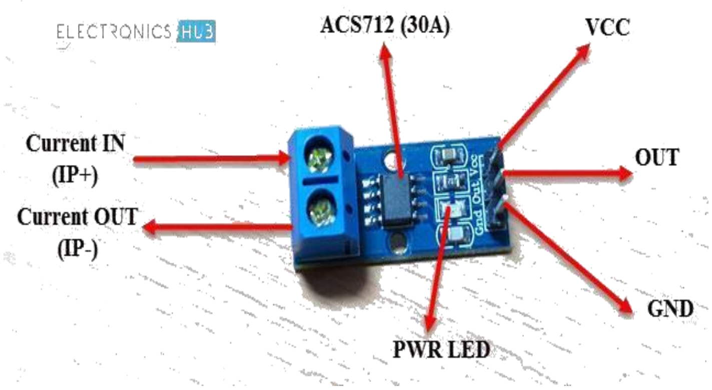


Figure 6: Current Sensor

2.2.5 Voltage Sensor

- Voltage up to 250 volts can be measured.
- Operating temperature: -40°C to 70°C.
- Supply voltage 5 volts to 30 volts.
- Very efficient and accuracy.
- Good consistency, for voltage and power measurement.
- Voltage Sensor is shown in Figure 7.

Figure 7: Voltage Sensor

2.2.6 Voltage Stabilizer Transformer

- Input Voltage Range: 160V - 260V AC
- Output Voltage Range: 220V ± 1-5% (adjustable)
- Common ratings could be 500VA, 1kVA, 2kVA, 5kVA, etc., depending on the load requirements.
- Typically, a good voltage stabilizer should have a correction speed in the range of 20-50 milliseconds.
- High accuracy, with the stabilizer maintaining the output voltage within ±1% or ±5% of the set value.
- Stabilizers often have an efficiency of 95% or higher, indicating minimal power loss during voltage regulation.
- Rapid response time, often less than 100 milliseconds, to swiftly correct voltage fluctuations.
- Overload protection may be designed to handle short-term overloads of up to 150-200% of the rated capacity.

  Transformer shown in Figure 8.

Figure 8: Transformer

2.2.7 220V to 12VAC transformer

It is designed to reduce the voltage (220v) from the primary winding to 12v in the secondary winding. As a step-down unit, the transformer converts high-voltage, low-current power into low-voltage, high-current power. The larger-gauge wire used in the secondary winding is necessary due to the increase in current.

220V to 12VAC transformer shown in Figure 9.

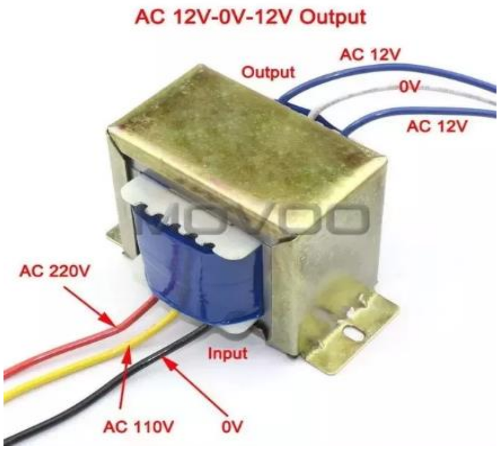


Figure 9: 12V Transformer

2.2.8 Channel Relay Module

The 4 Channel Relay Module is a convenient board which can be used to control high voltage, high current load such as motor, solenoid valves, lamps and AC load. It is designed to interface with microcontroller such as Arduino, PIC and etc. The relays terminal (COM, NO and NC) is being brought out with screw terminal.

4 Channel Relay is shown in Figure 10.

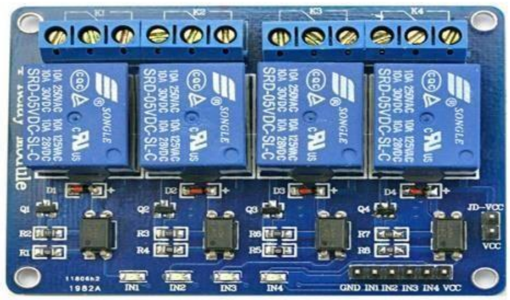


Figure 10: Relay

2.2.9 ESP32

ESP32 is a chip that provides Wi-Fi and (in some models) Bluetooth connectivity for embedded devices – in other words, for IoT devices. While ESP32 is technically just the chip, the modules and development boards that contain this chip are often also referred to as "ESP32" by the manufacturer.

ESP32 is shown in Figure 11.

Figure 11: ESP32

2.2.10 Capacitors

Capacitors contained in most power factor correction equipment draw current that leads the voltage, thus producing a leading power factor. If capacitors are connected to a circuit that operates at a nominally lagging power factor, the extent that the circuit lags is reduced proportionately.

Capacitors are shown in Figure 12.

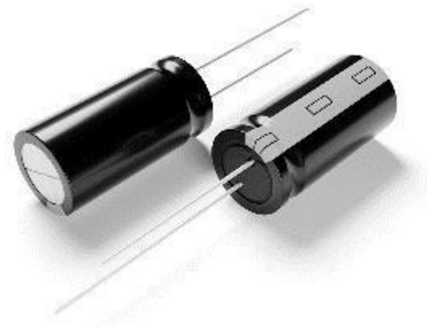


Figure 12: Capacitor

2.2.11 Motor Load

When AC motors handle variable loads, their power factor changes, typically being lower at light loads and higher at full loads. Variable Frequency Drives (VFDs) help maintain a better power factor by adjusting the motor's speed and torque. This improves energy efficiency and reduces operational costs in applications with fluctuating load demands. Efficient power factor management is crucial for optimizing motor performance and longevity.

Motor is shown in Figure 13.

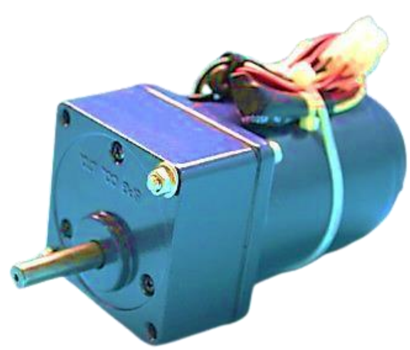


Figure 13: Load Motor

2.3 Software

Following software are used in the project:

- Proteus
- Arduino IDE
- Thing Speak

2.3.1 Proteus

Proteus Design Suite is a proprietary software tool suite used primarily for electronic design automation. Virtual Terminal is used to control the process if we send 1,2,3 and 4 then the lights are on and resend it lights off. By pressing 0 it goes into a place where we give the tariff price of two companies it compares which load tariff is low whole system sun on it.

Proteus Simulation is shown in Figure 14.

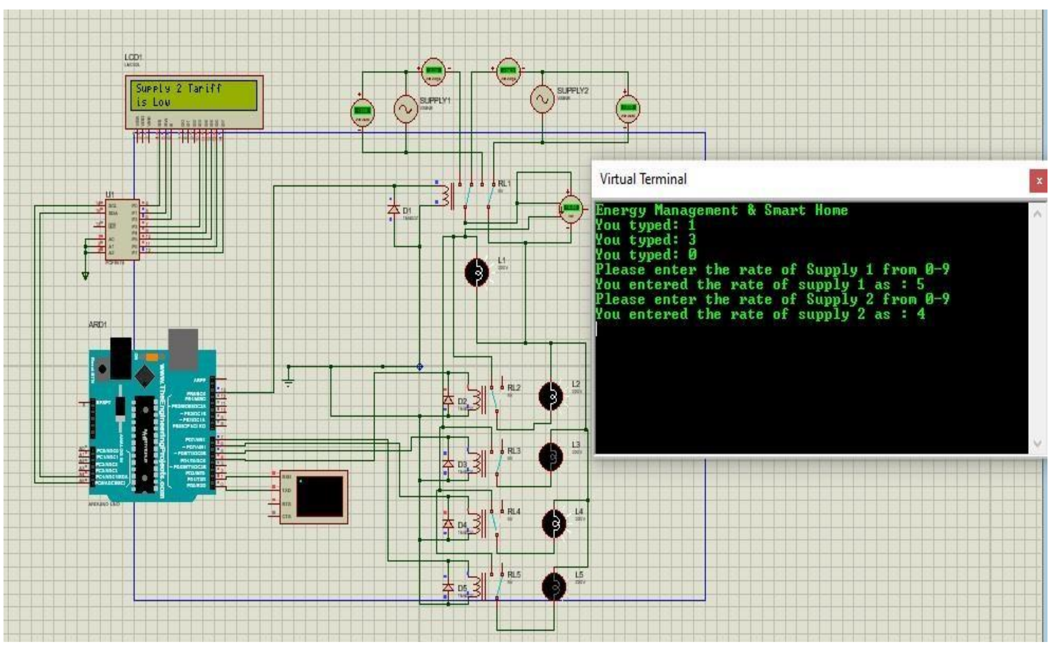


Figure 14: Proteus Simulation

2.3.2 Arduino IDE

The Arduino IDE is a software platform used for programming Arduino microcontroller boards. It provides a user-friendly interface for writing, compiling, and uploading code to Arduino boards. The IDE has a built-in library manager that helps us easily search for and install libraries. The Arduino IDE provides a built-in serial monitor tool that allows you to communicate with your Arduino board and view the data being sent or received through the serial port. This is useful for debugging and monitoring the behavior of your Arduino programs. Figure 17 shows a dashboard of Arduino IDE. The Arduino IDE itself is open source, which means the source code is freely available for modification and customization. We put our whole code in this software if any error comes it shows at the bottom bar where we can see at which line the error comes. Figure shows different boards which we need during compiling.

Arduino IDE interface are shown in Figure 15 &16.

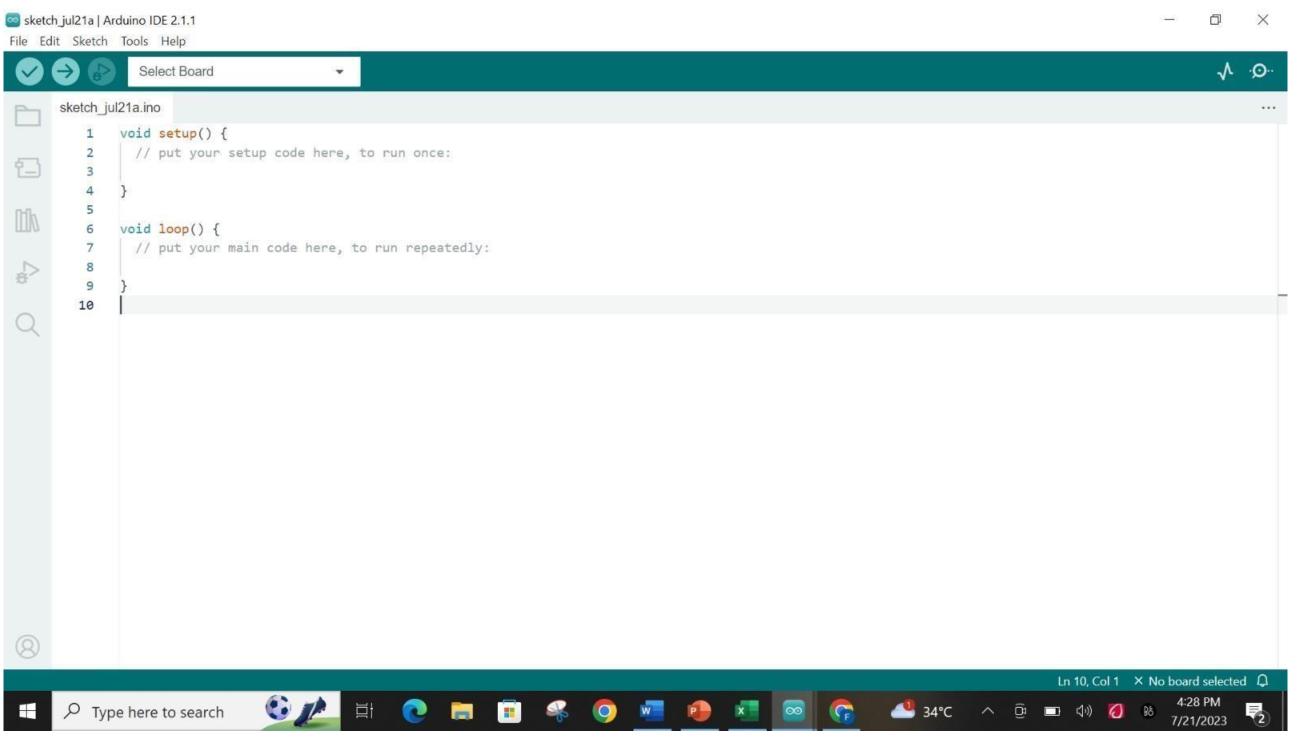


Figure 15: Arduino IDE Dashboard

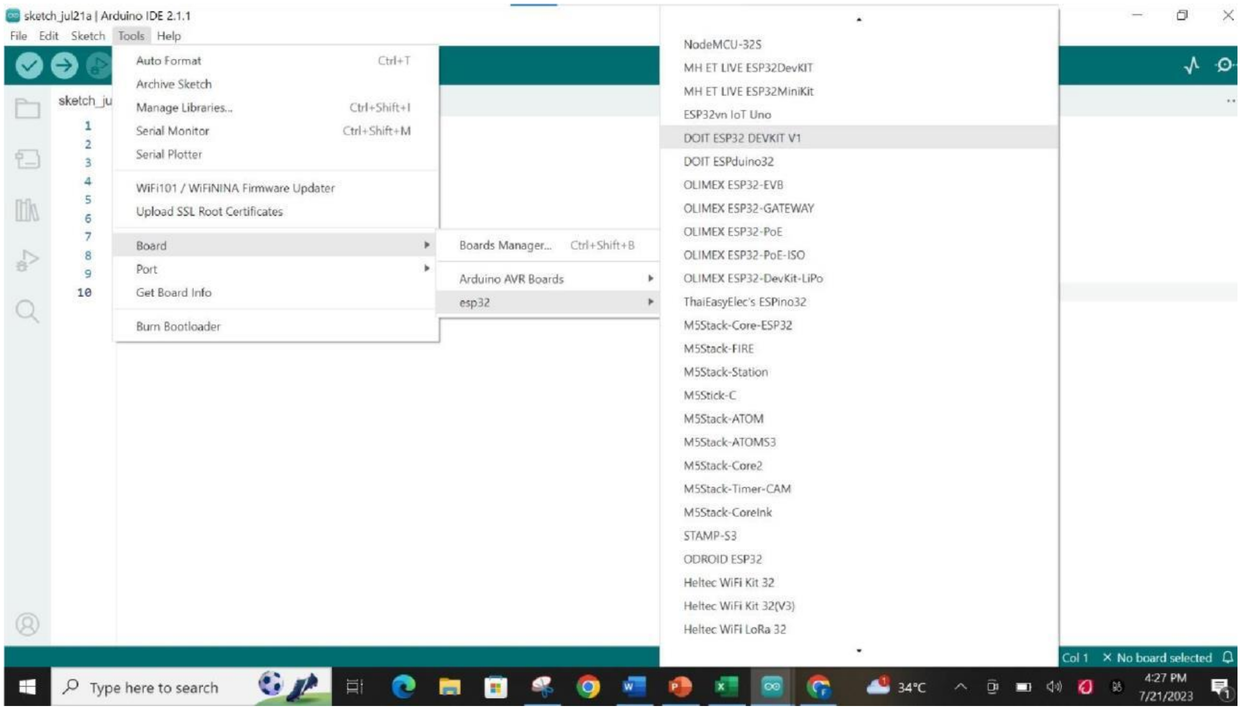


Figure 16: Arduino IDE Board Selection

2.3.3 ThingSpeak

Thing Speak is an IoT analytics platform service that allows you to aggregate, visualize, and analyze live data streams. Once you send data to Thing Speak from your devices, you can create instant visualizations of live data without having to write any code.

ThingSpeak interface is shown in Figure 17.

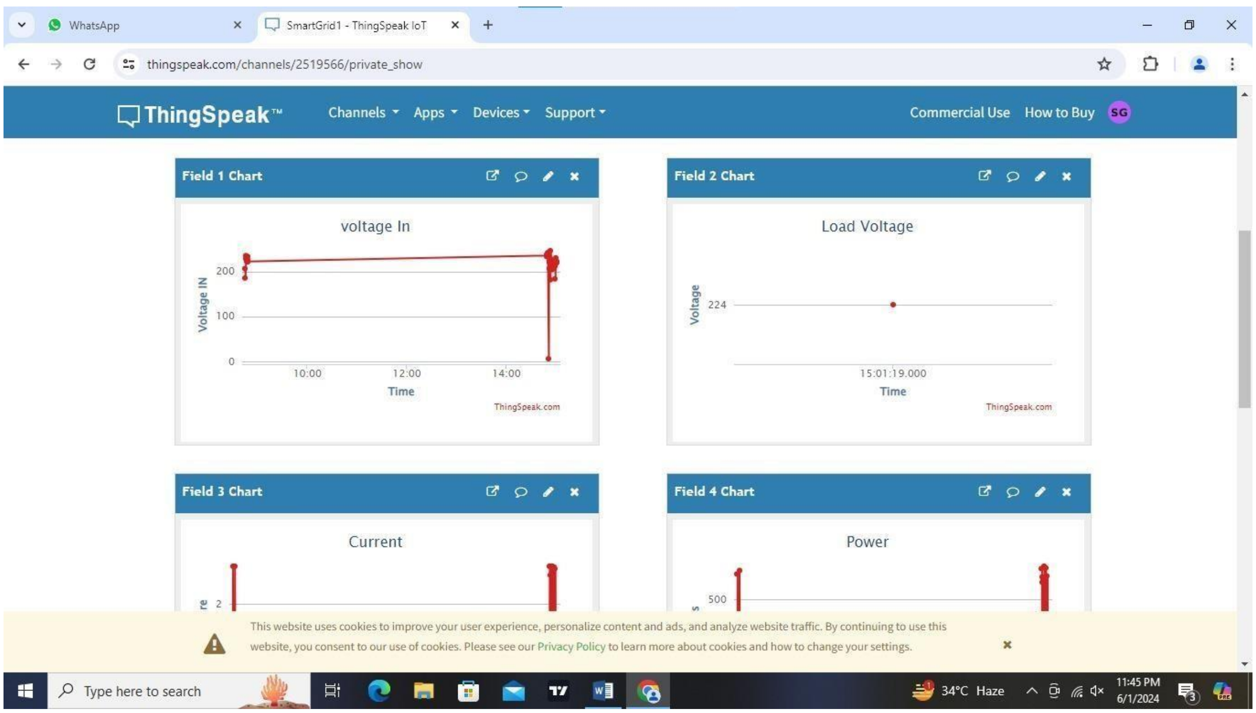


Figure 17: Thing Speak

2.4 Results

Input

Input voltages and Input current shows on cloud are given in the Figure 18:

Figure 18: Input

Output

Output voltages power and power factor after correction are given in Figure 19:

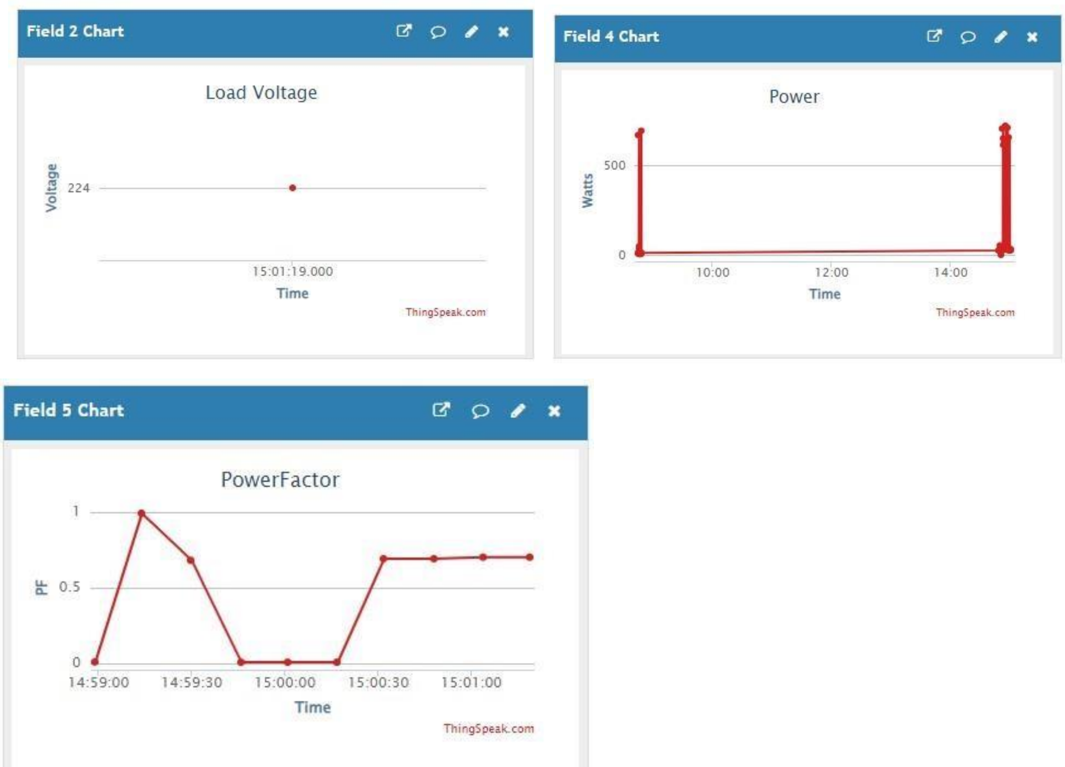


Figure 19: Output

CHAPTER 3: COMMERCIALIZATION

3.1 End Product

This project has demonstrated the critical importance of monitoring and analyzing power quality within electrical smart grids. By addressing issues such as harmonic distortion, power factor imbalances, and voltage fluctuations, we can significantly improve the reliability and efficiency of power systems. The development and implementation of a comprehensive monitoring system on a simulated electrical smart grid have provided valuable insights and practical solutions for enhancing power quality.

Through the careful selection of sensors and measurement devices, data collection and processing, and the identification and implementation of corrective actions, this project has paved the way for more resilient and efficient power grids. The continuous monitoring framework ensures that the system remains effective, adapting to changing conditions and maintaining optimal performance.

The findings and methodologies established in this project are not only beneficial for immediate application but also serve as a foundation for future advancements in power quality management. The results will be instrumental for power utilities, system operators, and other stakeholders committed to improving the performance and sustainability of electrical grids.

In conclusion, enhancing power quality through advanced monitoring and corrective measures is essential for reducing energy waste, minimizing downtime, and increasing customer satisfaction. This project underscores the transformative potential of smart grid technologies in achieving these goals and highlights the ongoing need for innovation and diligence in the field of electrical power systems.

3.2 Business Model Canvas

1. Customer Segments:
    - Homeowners with smart homes
    - Individuals interested in energy efficiency and cost savings
    - Individuals with disabilities seeking accessible home automation solutions
2. Value Proposition:
    - Energy optimization and cost savings through intelligent monitoring and control
    - Enhanced comfort
3. Channels:
    - Online platforms and websites
    - Smart home automation retailers and distributors
    - Partnerships with home appliances sellers
    - Direct sales and installation services
4. Customer Relationships:
    - Personalized customer support for installation, setup, and troubleshooting
    - Ongoing customer engagement through newsletters, blogs, and educational content
    - Feedback mechanisms to gather customer insights and improve the system
5. Revenue Streams:
    - Sale of devices and sensors for energy monitoring and control
    - Subscription fees for access to advanced energy management features and analytics
      • Service fees for installation, setup, and maintenance
6. Key Activities:
    - Research and development of devices, sensors, and software
    - Continuous improvement of algorithms and energy optimization strategies
    - Marketing and sales activities to promote the solution
7. Key Resources:
    - Devices, sensors, and connectivity infrastructure
    - Software development expertise
    - Partnerships with energy suppliers and utility companies
    - Skilled installation and support staff
8. Key Partnerships:

    - Smart home automation system providers for interoperability
    - Accessibility organizations for guidance on meeting the needs of individuals with online monitoring
9. Cost Structure:
    - Research and development costs for hardware and software
    - Manufacturing and production expenses
    - Marketing and sales expenditures
    - Ongoing maintenance and support costs
    - Partnerships and licensing fees

3.3 Marketability

The marketability of IoT-based Energy Management and Control systems in a Smart Home with Multiple Supply Connections is quite high. Here are some reasons why:

1. Increasing Demand for Energy Efficiency: With growing concerns about energy consumption, environmental sustainability, and rising energy costs, there is a significant demand for solutions that help individuals and businesses optimize their energy usage. IoT based energy management systems offer real-time monitoring, control, and optimization capabilities, making them highly attractive to homeowners seeking energy efficiency.
2. Convenience and Comfort: Smart homes equipped with IoT-based energy management systems provide convenience and comfort by allowing homeowners to control and automate various devices and appliances remotely. The ability to manage multiple supply connections and balance energy sources enhances the overall efficiency and reliability of the home's energy infrastructure, creating a seamless and comfortable living environment.
3. Cost Savings: Energy management and control systems help homeowners reduce their energy consumption, resulting in cost savings on utility bills. By optimizing energy usage, identifying energy wastage, and providing recommendations for efficiency improvements, these systems offer tangible financial benefits to consumers.
4. Integration with Renewable Energy Sources: Multiple supply connections enable integration with renewable energy sources such as solar panels, wind turbines, or batteries. This allows homeowners to harness clean energy and reduce their dependence on the main electrical grid. With the growing popularity of renewable energy solutions, the ability to

manage and control these sources through IoT-based systems further enhances their market appeal.

5. Government Incentives and Regulations: Many governments and regulatory bodies worldwide are promoting energy efficiency and renewable energy adoption. They offer incentives, subsidies, or tax benefits to individuals and businesses implementing energy management and control systems. These favorable policies and regulations further drive the marketability of IoT-based energy management solutions.
6. Accessibility and Inclusivity:

   IoT-based energy management systems can be designed with accessibility features to accommodate individuals with disabilities. Voice control interfaces, intuitive user interfaces, and integration with assistive technologies make it easier for people with disabilities to control and manage their home's energy usage. This inclusivity aspect enhances the market potential of such systems.
7. Scalability and Flexibility:

   IoT-based energy management systems are highly scalable and adaptable to different types and sizes of homes. They can be easily integrated into existing infrastructure or implemented in new construction. This scalability and flexibility make them applicable to a wide range of residential properties, catering to a larger market segment.

REFERENCES

[1] Bollen, M. H. J., & Hassan, F. (2011). *Integration of Distributed Generation in the Power System*. Wiley-IEEE Press.

[2] Ferrero, A., & Sangiovanni, S. (2018). *Power Quality Analysis Using Wavelet Transform: A Review*. IEEE Transactions on Instrumentation and Measurement, 67(1), 1-15.

[3] Dugan, R. C., McGranaghan, M. F., Santoso, S., & Beaty, H. W. (2012). *Electrical Power Systems Quality*. McGraw-Hill.

[4] El-Saadany, E. F., & Salama, M. M. A. (2001). *A New Voltage Stabilization and Harmonic Mitigation Technique*. IEEE Transactions on Power Delivery, 16(2), 129-134.

[5] Blaabjerg, F., Teodorescu, R., Liserre, M., & Timbus, A. V. (2006). *Overview of Control and Grid Synchronization for Distributed Power Generation Systems*. IEEE Transactions on Industrial Electronics, 53(5), 1398-1409.

[6] Lopes, J. A. P., Hatziargyriou, N., Mutale, J., Djapic, P., & Jenkins, N. (2007). *Integrating distributed generation into electric power systems: A review of drivers, challenges and opportunities*. Electric Power Systems Research, 77(9), 1189-1203.

[7] Bollen, M. H. J., & Hassan, F. (2011). *Integration of Distributed Generation in the Power System*. Wiley-IEEE Press.

[8] Ferrero, A., & Sangiovanni, S. (2018). *Power Quality Analysis Using Wavelet Transform: A Review*. IEEE Transactions on Instrumentation and Measurement, 67(1), 1-15.

APPENDIX A

Datasheets

Arduino:

Analog:

Pin	Function	Type	Description
1	NC	NC	Not connected
2	IOREF	IOREF	Reference for digital logic V - connected to 5V
3	Reset	Reset	Reset
4	+3V3	Power	+3V3 Power Rail
5	+5V	Power	+5V Power Rail
6	GND	Power	Ground
7	GND	Power	Ground
8	VIN	Power	Voltage Input
9	A0	Analog/GPIO	Analog input 0 /GPIO
10	A1	Analog/GPIO	Analog input 1 /GPIO
11	A2	Analog/GPIO	Analog input 2 /GPIO
12	A3	Analog/GPIO	Analog input 3 /GPIO
13	A4/SDA	Analog input/I2C	Analog input 4/I2C Data line
14	A5/SCL	Analog input/I2C	Analog input 5/I2C Clock line

Digital:

Pin	Function	Type	Description
1	D0	Digital/GPIO	Digital pin 0/GPIO
2	D1	Digital/GPIO	Digital pin 1/GPIO
3	D2	Digital/GPIO	Digital pin 2/GPIO
4	D3	Digital/GPIO	Digital pin 3/GPIO
5	D4	Digital/GPIO	Digital pin 4/GPIO
6	D5	Digital/GPIO	Digital pin 5/GPIO
7	D6	Digital/GPIO	Digital pin 6/GPIO
8	D7	Digital/GPIO	Digital pin 7/GPIO
9	D8	Digital/GPIO	Digital pin 8/GPIO
10	D9	Digital/GPIO	Digital pin 9/GPIO
11	SS	Digital	SPI Chip Select
12	MOSI	Digital	SPI1 Main Out Secondary In
13	MISO	Digital	SPI Main In Secondary Out
14	SCK	Digital	SPI serial clock output
15	GND	Power	Ground
16	AREF	Digital	Analog reference voltage
17	A4/SD4	Digital	Analog input 4/I2C Data line (duplicated)
18	A5/SD5	Digital	Analog input 5/I2C Clock line (duplicated)

Current Sensor:

Selection Guide

Part Number	Packing*	T_A (°C)	Optimized Range, I_P (A)	Sensitivity, Sens (Typ) (mV/A)
ACS712ELCTR-05B-T	Tape and reel, 3000 pieces/reel	–40 to 85	±5	185
ACS712ELCTR-20A-T	Tape and reel, 3000 pieces/reel	–40 to 85	±20	100
ACS712ELCTR-30A-T	Tape and reel, 3000 pieces/reel	–40 to 85	±30	66

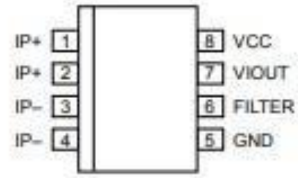
*Contact Allegro for additional packing options.

Absolute Maximum Ratings

Characteristic	Symbol	Notes	Rating	Units
Supply Voltage	V_{CC}		8	V
Reverse Supply Voltage	V_{RCC}		–0.1	V
Output Voltage	V_{IOUT}		8	V
Reverse Output Voltage	V_{RIOUT}		–0.1	V
Reinforced Isolation Voltage	V_{ISO}	Pins 1-4 and 5-8; 60 Hz, 1 minute, T_A=25°C	2100	V
		Voltage applied to leadframe (Ip+ pins), based on IEC 60950	184	V_{peak}
Basic Isolation Voltage	$V_{ISO(bsc)}$	Pins 1-4 and 5-8; 60 Hz, 1 minute, T_A=25°C	1500	V
		Voltage applied to leadframe (Ip+ pins), based on IEC 60950	354	V_{peak}
Output Current Source	$I_{IOUT(Source)}$		3	mA
Output Current Sink	$I_{IOUT(Sink)}$		10	mA
Overcurrent Transient Tolerance	I_P	1 pulse, 100 ms	100	A
Nominal Operating Ambient Temperature	T_A	Range E	–40 to 85	°C
Maximum Junction Temperature	$T_J(max)$		165	°C
Storage Temperature	T_{stg}		–65 to 170	°C

Parameter	Specification
Fire and Electric Shock	CAN/CSA-C22.2 No. 60950-1-03 UL 60950-1:2003 EN 60950-1:2001

Pin-out Diagram



Terminal List Table

Number	Name	Description
1 and 2	IP+	Terminals for current being sensed; fused internally
3 and 4	IP–	Terminals for current being sensed; fused internally
5	GND	Signal ground terminal
6	FILTER	Terminal for external capacitor that sets bandwidth
7	VIOUT	Analog output signal
8	VCC	Device power supply terminal

COMMON OPERATING CHARACTERISTICS[1] over full range of T_A, C_F = 1 nF, and V_{CC} = 5 V, unless otherwise specified

Characteristic	Symbol	Test Conditions	Min.	Typ.	Max.	Units
ELECTRICAL CHARACTERISTICS						
Supply Voltage	V_{CC}		4.5	5.0	5.5	V
Supply Current	I_{CC}	V_{CC} = 5.0 V, output open	–	10	13	mA
Output Capacitance Load	C_{LOAD}	VIOUT to GND	–	–	10	nF
Output Resistive Load	R_{LOAD}	VIOUT to GND	4.7	–	–	kΩ
Primary Conductor Resistance	$R_{PRIMARY}$	T_A = 25°C	–	1.2	–	mΩ
Rise Time	t_r	$I_P = I_P$(max), T_A = 25°C, C_{OUT} = open	–	5	–	μs
Frequency Bandwidth	f	–3 dB, T_A = 25°C; I_P is 10 A peak-to-peak	–	80	–	kHz
Nonlinearity	E_{LIN}	Over full range of I_P	–	1.5	–	%
Symmetry	E_{SYM}	Over full range of I_P	98	100	102	%
Zero Current Output Voltage	$V_{IOUT(Q)}$	Bidirectional; I_P = 0 A, T_A = 25°C	–	$V_{CC} \times 0.5$	–	V
Power-On Time	t_{PO}	Output reaches 90% of steady-state level, T_J = 25°C, 20 A present on leadframe	–	35	–	μs
Magnetic Coupling[2]			–	12	–	G/A
Internal Filter Resistance[3]	$R_{F(INT)}$			1.7		kΩ

Voltage Sensor

Front view **Bottom view**

The main technical parameters:

Model	ZMPT101B
Rated input current	2mA
Rated output current	2mA
turns ratio	1000:1000
phase angle error	≤20′ （input 2mA，sampling resistor 100Ω）
operating range	**0~1000V 0~10mA（sampling resistor 100Ω）**
linearity	≤0.2%(20%dot~120%dot)
Permissible error	-0.3%≤ f ≤+0.2%（input 2mA，sampling resistor 100Ω）
isolation voltage	4000V
application	voltage and power measurement
Encapsulation	Epoxy
installation	PCB mounting（Pin Length>3mm）
Operating temperature	-40℃~+60℃
Case Material	**ABS (Note: ABS CASE is NOT available for wave-soldering)**

Recommended Operating Conditions

Symbol	Parameter	Min.	Max.	Units
V_{CC}	Supply Voltage	2	18	V
T_A	Operating Temperature	0	70	°C
f	Operating Frequency	0	4	kHz

APPENDIX B

Programming Code CODE: Arduino

```
#include "ZMPT101B.h"
#include <SoftwareSerial.h>
#include <math.h>


#define pushButton_pin1 10
#define pushButton_pin2 11
#define pushButton_pin3 12

#define Load1 2
#define Load2 3
#define Load3 4
#define PF_Correction 4

#define MainRelay 8

ZMPT101B voltageSensor(A0);
ZMPT101B voltageSensor1(A1);

float Voltage, Voltage1;

unsigned int  Amp = 0, abc = 0;

float pulsewidth = 0; float
powerfactor = 0; float
```

```
phase = 0; float avg_pf =
0; float avg_phase = 0;

SoftwareSerial Connection (7, 13); //RxTx

float A = 0.0; float
Amp1 = 0.0; int
intAmp1 = 0; String
stringAmp1 = "";  String
stringU = ""; String
stringU3 = ""; int
avg_pf1 = 1; String
stringavg_pf1 = "";


void setup()

{
  Serial.begin(19200);
  Connection.begin(9600);   pinMode(Load1,
OUTPUT);   pinMode(Load2, OUTPUT);
pinMode(Load3, OUTPUT);
pinMode(MainRelay, OUTPUT);
digitalWrite(Load1, HIGH);
digitalWrite(Load2, HIGH);   digitalWrite(Load3,
HIGH);   digitalWrite(MainRelay, HIGH);
```

```
  pinMode(pushButton_pin1, INPUT_PULLUP);
pinMode(pushButton_pin2, INPUT_PULLUP);   pinMode(pushButton_pin3,
INPUT_PULLUP);

  delay(100);

  Serial.println("Calibrating... Ensure that no current flows through the sensor at
this moment");   delay(100);   voltageSensor.calibrate();   delay(100);
voltageSensor1.calibrate();   delay(100);
  Serial.println("Done!");

}

void loop() {
checkpin(); int L1 =
digitalRead(Load1); int L2
=
digitalRead(Load2); int
L3 = digitalRead(Load3);
Amp = analogRead(A2);
Serial.println(Amp);   for
(int i = 0; i < 150; i++)
  {
    Amp = analogRead(A2);     if
(Amp > abc)
    {
      abc = Amp;
```

```
    }

  }
  Serial.println(abc);
  Serial.print("I: ");   A = float
((abc - 3) * 0.003);   abc = 0;
Serial.println(A);   delay(500);



  float U1 = voltageSensor.getVoltageAC(); float
  U = U1 * 80;
  //    if (U<100){
  //      U = 0;
  //    }
  Serial.println(String("VLoad = ") + U + " V");
delay(1000);

  float U2 =
voltageSensor1.getVoltageAC();   float U3
= U2 * 80;    //  if (U3 < 100) {
  //    U3 = 0;
  //  }
  Serial.println(String("Vinput = ") + U3 + " V");

  for (int i = 0; i < 100; i++)
  {
```

```
    pulsewidth = pulseIn(6, HIGH);     phase = 2
* 180 * 50 * pulsewidth / 1000000;
powerfactor = cos(phase * 3.1415 / 180);
avg_pf += powerfactor;     avg_phase += phase;
  }
  avg_pf /= 100;   avg_phase
/= 100;   Serial.print("PW:
");
Serial.print(avg_phase);
  Serial.print("\t");
  Serial.print("PF: ");
  Serial.println(avg_pf);

  delay(100);


  stringU = U;   stringU3 = U3;
//  Amp1 = A * 1000;
stringAmp1 = A;   avg_pf1 =
avg_pf * 100;   stringavg_pf1
= avg_pf1;
  String Data = "Data:"+stringU3+ "," +stringU+ ";"+stringAmp1+
"?"+stringavg_pf1+ "@";
  Serial.println(Data);
Connection.println(Data);   delay(1000);

  if(U3<205 || U3>235){
```

```
digitalWrite(MainRelay, LOW);
  }   else{
digitalWrite(MainRelay, HIGH);
  }
  avg_pf = 0; avg_phase
  = 0;
}

void checkpin()  //checks board input status  {   if
(digitalRead(pushButton_pin1) == LOW) {
digitalWrite(Load1, !digitalRead(Load1));
while
(digitalRead(pushButton_pin1) == LOW) {}
  }

  if (digitalRead(pushButton_pin2) == LOW) {
digitalWrite(Load2, !digitalRead(Load2));     while
(digitalRead(pushButton_pin2) == LOW) {}
  }

  if (digitalRead(pushButton_pin3) == LOW) {
digitalWrite(Load3, !digitalRead(Load3));     while
(digitalRead(pushButton_pin3) == LOW) {}
  }
}
```

Code: ESP32

```
#include <WiFi.h>
#include "ThingSpeak.h"
#include <LiquidCrystal_I2C.h>
#include <math.h>


int wait30 = 30000;  // time to reconnect when connection is lost.

//##############################        WIFI        CREDENTIALS
###################################

const char* password = "ProjectIoT";
const char* ssid = "IoT_Project";

WiFiClient client;

//   ##############################        Cloud        CREDENTIALS
###################################

unsigned long myChannelNumber = 2519566; const
char* myWriteAPIKey = "B8J74N7J3EO9Z7CX";


// Timer variables unsigned long
lastTime = 0;
WiFiServer server(80);

char host[] = "api.thingspeak.com";      // ThingSpeak address
String APIkey = "2519566";               // Thingspeak Read Key, works only if a PUBLIC
viewable channel
```

```
String APIreadkey = "JQNWS8M7HRKL5G1G";  // Thingspeak Read Key, works only
if a PUBLIC viewable channel const int httpPort = 80; const unsigned long
HTTP_TIMEOUT = 10000;  // max respone time from server  int a, b, c, d, h = 0;
char f; String dataIn;

String Voltage1 = "";
String Voltage2 = "";
String Current = "";
String PowerFactor = "";


int intVoltage1 = 0; int
intVoltage2 = 0; float intCurrent
= 0.0; int intPowerFactor = 0;

float PF = 0.99;

int Power = 0;

int lcdColumns = 16; int lcdRows
= 2;

LiquidCrystal_I2C lcd(0x27, lcdColumns, lcdRows);


void setup()

{
  Serial.begin(115200);
  //  Serial.begin(19200);
  Serial2.begin(9600);
```

```
  lcd.init();
lcd.backlight();



 delay(100);   Serial.println("Done!");



 WiFi.mode(WIFI_STA);
 ThingSpeak.begin(client);  // Initialize ThingSpeak
// Start Web Server.   server.begin();
 Serial.println("Web Server started.");
 Serial.println("Done!");

 lcd.setCursor(0, 0);   lcd.print("
Smart Grid");   lcd.setCursor(0,
1);   lcd.print("     Project ");
delay(5000);   lcd.clear();
}

void     loop()  {   lcd.clear();
 Serial.println("Printing Data on LCD");
lcd.setCursor(0,  0);      lcd.print("Vi=");
lcd.print(intVoltage1);           lcd.print(";
VL=");                 lcd.print(intVoltage2);
lcd.setCursor(0,   1);       lcd.print("P=");
lcd.print(Power);       lcd.print(";   PF=");
lcd.print(PF);   delay(1000);

 Serial.println("Connected and in void loop");   while
(Serial2.available()) {     f =
```

```
Serial2.read();     if (f != '\n') {       dataIn += f;
} else {       break;
   }   }   if (f
== '\n') {
   Serial.println(dataIn);     if
(dataIn.indexOf("Data") != -1) {
      Voltage1 = dataIn.substring(dataIn.indexOf(":") + 1, dataIn.indexOf(","));
      Serial.print("Voltage1 is: ");
      Serial.print(Voltage1);
      Voltage2 = dataIn.substring(dataIn.indexOf(",") + 1, dataIn.indexOf(";"));
      Serial.print("Voltage2 is: ");
      Serial.print(Voltage2);
      Current = dataIn.substring(dataIn.indexOf(";") + 1, dataIn.indexOf("?"));
      Serial.print("Current is: ");
      Serial.print(Current);
      PowerFactor = dataIn.substring(dataIn.indexOf("?") + 1, dataIn.lastIndexOf("@"));
      Serial.print("Power Factor is: ");
Serial.println(PowerFactor);
intVoltage1 = Voltage1.toInt();
intVoltage2 = Voltage2.toInt();
intCurrent = Current.toFloat();
      PF = PowerFactor.toFloat();
   }
//     h = dataIn.toInt();       f
= 0;       dataIn = "";
   }
   PF = PF/100.0;
   Serial.print("PowerFactor= ");
   Serial.println(PF);
   Power = intVoltage2*intCurrent;
```

```
  if ((WiFi.status() != WL_CONNECTED) && (millis() > wait30)) {
    Serial.println("Trying to reconnect WiFi...");
    WiFi.disconnect();
WiFi.begin(ssid, password);       wait30
= millis() + 30000;
  }
  Serial.println("Connected with WiFi.");
  Serial.print("This is IP to connect to the WebServer: ");
  Serial.print("http://");
  Serial.println(WiFi.localIP());



  ThingSpeak.setField(1, intVoltage1);
  ThingSpeak.setField(2, intVoltage2);
  ThingSpeak.setField(3, intCurrent);
  ThingSpeak.setField(4, Power);
  ThingSpeak.setField(5, PF);
  //  ThingSpeak.setField(6, status);

  int x = ThingSpeak.writeFields(myChannelNumber, myWriteAPIKey);

  if (x == 200) {
    Serial.println("Channel updated successful.");
  } else {
    Serial.println("Problem updating channel. HTTP error code " + String(x));
  }

  // Check if a client has connected..
WiFiClient client = server.available();     if
(!client) {       return;
  }
```

```
Serial.print("New client: ");
Serial.println(client.remoteIP());   }
```