**IOT BASED SMART ENERGY METER**



A Report Submitted to

**Rajiv Gandhi Proudyogiki Vishwavidyalaya, Bhopal** Towards the partial fulfillment of the Degree of

### Bachelor of Technology

in

### Electronics and Telecommunication Engineering

**Guided by:**

Prof. Ajay Parmar Prof. Deepali Kothari

### Submitted By:

Falguni Chaskar (0801EC201015) Geetika Maheshwari (0801EC201053) Anshuman Singh (0801EC201055) Uday P. S. Chauhan (0801EC201098)

Department of Electronics and Telecommunication Engineering Shri G. S. Institute of Technology and Science Indore – 452003 (M.P.)

May 2024



## RECOMMENDATION

This report entitled, **“IOT BASED SMART ENERGY METER”**, submitted to the Rajiv Gandhi Proudyogiki Vishwavidyalaya, Bhopal, by **Falguni Chaskar, Geetika Maheshwari, Anshuman Singh, Uday Pratap Singh Chauhan** during the academic year **2023-24** as a partial fulfillment for the award of the degree of **Bachelor of Technology** in **Electronics and Telecommunication Engineering**, is a record of student’s own work carried out by them under our direct supervision, in the **Department of Electronics and Telecommunication Engineering, S.G.S.I.T.S.** Indore. The work contained in the report is a satisfactory account of this project work and is recommended for the award of the degree.

**Prof. Ajay Parmar (Guide)**

**Prof. Deepali Kothari**

**(Co-Guide)**

**Prof. Anjana Jain**

**(Head of Department)**

**Electronics and Telecommunication Engineering**



## CERTIFICATE

This is to certify that the project entitled **“IOT BASED SMART ENERGY METER”**, submitted to the Rajiv Gandhi Proudyogiki Vishwavidyalaya, Bhopal, by **Falguni Chaskar, Geetika Maheshwari, Anshuman Singh, Uday Pratap Singh Chauhan** during the academic year **2023-2024**, is a record of their own work and is accepted in the partial fulfillment for the award of the degree of **Bachelor of Technology** in **Electronics and Telecommunication Engineering**.

### Internal Examiner External Examiner

Date: Date:

# ACKNOWLEDGEMENT

We would like to express our sincere gratitude to **Prof. Ajay Parmar** Department of Electronics and Telecommunication, S.G.S.I.T.S, Indore whose constant encouragement enabled us to work enthusiastically. Hir patience and technical expertise in discussion benefited us, in completing our thesis work. The trust he placed in our abilities was always a great source of motivation. We would also like to express our gratitude towards our co- guide **Prof. Deepali Kothari**, for her valuable time and suggestions given on our project work.

We are also grateful to our Head of Department **Prof. Anjana Jain** for her caring attitude and encouragement. We thank all other faculty members, staff, our seniors and our batch-mates whose consistent effort in maintaining academic excellence has resulted in NBA accreditation of our B.Tech (Electronics and Telecommunication Engineering) course. We also appreciate the support of **Prof. R. K. Saxena**, Director, SGSITS Indore for providing the academic facilities in this Institute.

Finally, we owe our special thanks to the Almighty God and our beloved Parents for supporting and encouraging us with their support, patience, blessings and understanding to pursue this work, without which we would not have finished our degree.

**Falguni Chaskar**

**Geetika Maheshwari**

**Anshuman Singh**

**Uday Pratap Singh Chauhan**

# ABSTRACT

The purpose of the project is the development and implementation of a Smart Energy Meter with Home Automation system. The project aims to monitor and manage electricity consumption of various devices in a household, providing real-time data visualization via a local website and enabling control through an MQTT server.

The energy metering system utilizes a current transformer to measure power consumption, with data processed by the ESP32 microcontroller. The microcontroller is programmed to transmit this data to a local website for display and analysis. Additionally, it is connected to an MQTT server, enabling remote control of connected devices. If the power consumption of any device exceeds a predetermined threshold, the system automatically cuts off power to that device.

This project contributes to energy conservation by providing users with insights into their energy consumption patterns and enabling them to efficiently manage their devices. The combination of energy monitoring and home automation features enhances user convenience while promoting sustainable energy practices.

**Contents**

**LIST OF FIGURES**

[**LIST OF ABBREVIATIONS**](https://docs.google.com/document/d/1Mzw8KJPDAgidEKuBTG_MiL6O4tnAzYqX/edit#heading%3Dh.tyjcwt)

1. [**Introduction**](https://docs.google.com/document/d/1Mzw8KJPDAgidEKuBTG_MiL6O4tnAzYqX/edit#heading%3Dh.2s8eyo1) **10**
   1. [Necessity](https://docs.google.com/document/d/1Mzw8KJPDAgidEKuBTG_MiL6O4tnAzYqX/edit#heading%3Dh.3rdcrjn) 10
   2. [Function](https://docs.google.com/document/d/1Mzw8KJPDAgidEKuBTG_MiL6O4tnAzYqX/edit#heading%3Dh.35nkun2) 11
   3. [Advantages](https://docs.google.com/document/d/1Mzw8KJPDAgidEKuBTG_MiL6O4tnAzYqX/edit#heading%3Dh.44sinio) 11
   4. [Features](https://docs.google.com/document/d/1Mzw8KJPDAgidEKuBTG_MiL6O4tnAzYqX/edit#heading%3Dh.z337ya) 13
2. [**Literature Review**](https://docs.google.com/document/d/1Mzw8KJPDAgidEKuBTG_MiL6O4tnAzYqX/edit#heading%3Dh.1y810tw)14
3. [**Problem Formulation**](https://docs.google.com/document/d/1Mzw8KJPDAgidEKuBTG_MiL6O4tnAzYqX/edit#heading%3Dh.3fwokq0) **17**
   1. [Objective](https://docs.google.com/document/d/1Mzw8KJPDAgidEKuBTG_MiL6O4tnAzYqX/edit#heading%3Dh.4f1mdlm) 17
   2. [Problem Statement](https://docs.google.com/document/d/1Mzw8KJPDAgidEKuBTG_MiL6O4tnAzYqX/edit#heading%3Dh.19c6y18) 17
      1. [Overview](https://docs.google.com/document/d/1Mzw8KJPDAgidEKuBTG_MiL6O4tnAzYqX/edit#heading%3Dh.28h4qwu) 17
   3. [Challenges](https://docs.google.com/document/d/1Mzw8KJPDAgidEKuBTG_MiL6O4tnAzYqX/edit#heading%3Dh.1mrcu09) 17
      1. [Hardware Challenge](https://docs.google.com/document/d/1Mzw8KJPDAgidEKuBTG_MiL6O4tnAzYqX/edit#heading%3Dh.2lwamvv) 18
      2. [Programming Challenge](https://docs.google.com/document/d/1Mzw8KJPDAgidEKuBTG_MiL6O4tnAzYqX/edit#heading%3Dh.3l18frh) 19
4. [**Proposed Methodology**](https://docs.google.com/document/d/1Mzw8KJPDAgidEKuBTG_MiL6O4tnAzYqX/edit#heading%3Dh.4k668n3) **20**
   1. [Proposed Work](https://docs.google.com/document/d/1Mzw8KJPDAgidEKuBTG_MiL6O4tnAzYqX/edit#heading%3Dh.1egqt2p) 20
   2. [Applications](https://docs.google.com/document/d/1Mzw8KJPDAgidEKuBTG_MiL6O4tnAzYqX/edit#heading%3Dh.sqyw64) 21
5. [**Hardware Implementation**](https://docs.google.com/document/d/1Mzw8KJPDAgidEKuBTG_MiL6O4tnAzYqX/edit#heading%3Dh.1rvwp1q) **23**
   1. [Components Used](https://docs.google.com/document/d/1Mzw8KJPDAgidEKuBTG_MiL6O4tnAzYqX/edit#heading%3Dh.2r0uhxc) 23
      1. Transformer 23
      2. Voltage Regulator 23
      3. Capacitor 24
      4. Relay 12V… 24
      5. NPN Transistor… 24
      6. Current Transformer 25
      7. ESP 32 25
      8. LM1117 Voltage Regulator 26
6. [**Software Used**](https://docs.google.com/document/d/1Mzw8KJPDAgidEKuBTG_MiL6O4tnAzYqX/edit#heading%3Dh.2nusc19) **27**
   1. .MQTT Server 27

6.6.1 MQTT Server Open Source 27

6.1.2 Arduino 27

* 1. Using Apache Tomcat 32
     1. Apache Tomcat 32
     2. MySQL-For Database 32
     3. PHP-for code implementation 32

1. [**Results and Conclusion**](https://docs.google.com/document/d/1Mzw8KJPDAgidEKuBTG_MiL6O4tnAzYqX/edit#heading%3Dh.2fk6b3p) **37**
   1. [Results](https://docs.google.com/document/d/1Mzw8KJPDAgidEKuBTG_MiL6O4tnAzYqX/edit#heading%3Dh.3ep43zb) 37
   2. [Conclusion](https://docs.google.com/document/d/1Mzw8KJPDAgidEKuBTG_MiL6O4tnAzYqX/edit#heading%3Dh.184mhaj) 38

Bibliography 39

**List of Figures**

1.1: IOT Based Smart Energy Meter……………… 10

2.1: Smart Energy Meter……………………… 15

2.2: Flow Chart of the Process……………... 16

4.3: Bock Diagram……………………………. 22

5.1: Transformer……………………………….. 23

5.2: Voltage Regulator………………………… 23

5.3: Capacitor…………………………………… 24

5.4: Relay 12V………………………………. 24

5.5: NPN Transistor…………………………….. 25

5.6: Current Transformer………………………. 25

5.7: ESP 32 ……………………………………… 25

5.8: LM1117 Voltage Regulator………………. 26

6.1: PCB Circuit………………………………… 35

6.2: PCB Schematic…………………………….. 36

# LIST OF ABBREVIATIONS

**AC A**lternating **C**urrent

**DC D**irect **C**urrent

**HMI H**uman **M**achine **I**nterface

**CT Current Transformer**

**LED L**ight **E**mitting **D**iode

SQL Structured Query Language

# Chapter 1 Introduction

**1.1 Necessity**

The issue of power consumption is multifaceted and presents a range of challenges at both global and local levels. One of the primary concerns is the rapidly increasing demand for energy, driven by population growth, industrialization, and technological advancements. This escalating demand often leads to strain on existing power generation infrastructure, potentially resulting in energy shortages, increased costs, and environmental impacts.

**Figure 1.1:** IOT Based Smart Energy Meter.

Implementing a power-saving or preservation system alongside the Smart Energy Meter with Home Automation will aid in understanding and managing power usage effectively, thus reducing wastage. By accurately calculating power consumption and analyzing usage patterns, the system can identify areas of inefficiency and suggest optimizations.

**1.2 Function**

1. **Real-time Energy Monitoring**: Monitors energy consumption of devices.
2. **Automatic Power Management**: Cuts off power to devices exceeding thresholds.
3. **Remote Device Control**: Allows remote device control via web interface.
4. **Data Visualization**: Displays energy consumption data on a website.
5. **Fault Detection and Notification**: Alerts users of abnormal energy consumption.

**1.3 Advantages**

Intelligent energy meters integrated with home automation systems, utilizing Current Transformers (CT), Potential Transformers (PT), ESP32 (a low-cost, low-power system-on-a- chip microcontroller), and voltage regulators offer several advantages. Here are some key benefits:

1. **Real-time Monitoring:**

The intelligent energy meter allows users to monitor their energy consumption in real-time.

This helps in understanding and optimizing energy usage patterns.

1. **Energy Efficiency:**

Home automation features enable users to control and automate devices based on energy consumption data. This promotes energy efficiency by allowing users to identify and reduce power-hungry appliances.

1. **Cost Savings:**

By understanding energy consumption patterns and optimizing usage, users can potentially reduce their electricity bills. Smart automation can also enable load shedding during peak hours, leading to cost savings.

1. **Remote Monitoring and Control:**

ESP32 provides connectivity options, allowing users to monitor and control their energy usage remotely. This is especially beneficial for homeowners who want to manage their energy consumption while away from home.

1. **Integration with Smart Home Devices:**

The system can be integrated with other smart home devices, such as smart thermostats, lighting systems, and security systems. This enables a holistic approach to home automation and energy management.

1. **Data Logging and Analysis:**

Intelligent energy meters can log historical data, allowing users to analyze trends and patterns in energy consumption. This data can be valuable for making informed decisions about energy-efficient practices.

1. **Voltage Regulation:**

The inclusion of a voltage regulator ensures a stable and consistent power supply, which is crucial for the proper functioning of electronic devices and appliances. It helps prevent damage caused by voltage fluctuations.

1. **User-Friendly Interface:**

ESP32 can be used to create a user-friendly interface for monitoring and controlling the energy system. This can be accessed through a smartphone app or a web portal, making it easy for users to interact with the system.

1. **Scalability:**

The system can be easily expanded and scaled to accommodate additional sensors or smart devices. This makes it adaptable to future needs and advancements in home automation technology.

**10.** **Environmental Impact:**

By optimizing energy consumption, users contribute to a reduction in overall energy demand, which can have positive environmental implications, including a decrease in greenhouse gas emissions.

**11.** **Security:**

- With the integration of the ESP32, security features can be implemented to safeguard the system from unauthorized access, ensuring the privacy and integrity of the data collected.

In summary, the combination of an intelligent energy meter, CT, PT, ESP32, and voltage regulator provides a comprehensive solution for efficient energy management and home automation, offering benefits in terms of cost savings, convenience, and environmental impact.

**1.4 Features**

**ESP32 Integration**: The IoT-based home automation system utilizes the ESP32 microcontroller for seamless connectivity and communication between devices. The ESP32's versatility and built-in Wi-Fi capabilities make it an ideal choice for IoT applications.

**Power Monitoring Capability**: One of the key features is the ability to monitor power consumption in real-time.

**Remote Control and Monitoring**: Users can remotely control and monitor their home devices through a dedicated mobile app or web interface.

**Smart Scheduling**: The home automation system supports intelligent scheduling of devices based on power consumption patterns and user preferences.

**Data Storage**: The ESP-32 module send data to server where data get stored and can be reviewed for future reference.

**Secure Communication**: The ESP32-based system prioritizes security by employing encryption protocols for communication between devices.

**Chapter 2 Literature Review**

**2.1 Review**

The electrical system begins with a transformer stepping down 200V AC mains power to 12V AC. Subsequently, a bridge rectifier converts the 12V AC to DC. Capacitors are employed for filtering and stabilizing the DC voltage. To ensure a stable 5V output, a voltage regulator (7805) is utilized. Since the ESP-32 operates on 3.3V, another voltage regulator (LM1117) is employed to stabilize the voltage at this level. The ESP32 microcontroller orchestrates device control using transistors and relays.

In the CT-circuit, individual component currents are calculated through current transformers (CTs). This data is then transmitted to the ESP-32 module. The ESP module relays this information to a server for further processing.

The server, in turn, forwards the data to a database for storage and analysis. An HTML page is integrated with the database, enabling users to access and interact with the data. The HTML page displays current, voltage, and power data, allowing users to monitor the system's performance. Additionally, users have the option to delete records and visualize data through pie charts, enhancing data analysis capabilities.

Overall, this integrated system facilitates efficient monitoring and management of electrical parameters, ensuring optimal performance and enabling informed decision-making.

# 

**Figure 2.1**: Smart Energy Meter

# 2.2 Flow Chart

**Figure 2.2**: Flow Chart of the process.

# Chapter 3

**Problem Formulation**

### Objective

Objective: Develop an IoT-based Electricity Energy Meter utilizing SCT-013 Non-Invasive AC Current Sensor and ZMPT101B AC Voltage Sensor integrated with an ESP32 WiFi Module and MQTT Server Application. This automated system aims to monitor and track electricity consumption in real-time, providing accurate data on voltage, current, power, and total energy consumption to streamline energy management processes.

### Problem Statement

#### Overview

The manual process of monitoring electricity consumption is cumbersome, necessitating physical access to meter reading rooms. Leveraging the Internet of Things (IoT) for automated data collection can streamline this process, saving time and costs. To achieve this, the project aims to construct an IoT-based Electricity Energy Meter utilizing the SCT-013 Non-Invasive AC Current Sensor Split Core Type Clamp Meter Sensor and the ZMPT101B AC Voltage Sensor Module. Through interfacing these sensors with the ESP32 Wifi Module and transmitting data to the MQTT Server Application Dashboard, the system can display real- time information on voltage, current, power, and total energy consumption in kilowatt-hours (kWh).

### Challenges

Building an IoT-based Electricity Energy Meter using the SCT-013 Current Sensor, the ZMPT101B Voltage Sensor, and the ESP32 WiFi Module, while integrating it with the MQTT Server Application, comes with its own set of hardware challenges. Some potential hardware challenges you might encounter include:

* + 1. Hardware Challenge

**Sensor Calibration**: Calibrating the SCT-013 Current Sensor and ZMPT101B Voltage Sensor to ensure accurate readings can be challenging. Accuracy is crucial for precise energy measurements, and any inconsistencies between the sensors and the actual power consumption can lead to erroneous data and misinformed decisions.

**Interference and Noise**: Electrical noise and interference can significantly affect the accuracy of the readings from the sensors. This interference can come from various sources, such as electromagnetic fields from nearby electronic devices, other power lines, or appliances. Designing a robust circuit to minimize interference and noise is crucial for reliable data collection.

**Power Supply Stability**: Ensuring a stable power supply for the sensors and the microcontroller is essential for consistent and accurate readings. Fluctuations in the power supply can affect the performance of the sensors and the overall functionality of the IoT device. Implementing proper power management and regulation techniques to maintain a stable power supply is essential.

**Data Transmission and Connectivity Issues**: Relaying the data from the sensors to the MQTT Server Application via the ESP32 WiFi Module requires a stable internet connection. Issues such as network latency, packet loss, and intermittent connectivity can impact the real-time monitoring and data transmission process. Implementing robust data transmission protocols and error-handling mechanisms is essential to ensure seamless communication between the IoT device and the application.

**Heat Dissipation and Thermal Management**: Operating the sensors and the microcontroller for extended periods can generate heat, potentially leading to overheating issues. Adequate heat dissipation and thermal management measures need to be implemented to prevent damage to the sensors and the microcontroller. Proper ventilation, heat sinks, and thermal insulation techniques may be necessary to maintain optimal operating temperatures.

**Enclosure and Environmental Protection**: Designing a suitable enclosure to house the IoT device that protects it from environmental factors such as dust, moisture, and physical damage is crucial. The enclosure should be durable and robust enough to withstand various environmental conditions and ensure the longevity of the device.

* + 1. Programming Challenge
       1. **Sensor Data Reading and Calibration:**

Reading data from the SCT-013 current sensor and the ZMPT101B voltage sensor accurately. Calibrate the sensors to ensure precise readings for power calculations.

Elaboration: You'll need to write code to read analog signals from the sensors and convert them into meaningful values for current and voltage. Additionally, you'll have to calibrate the sensors to ensure accurate readings by considering factors like noise, offset, and sensitivity.

* + - 1. **Power Calculation and Energy Consumption Tracking:**

Compute real-time power consumption by integrating the readings from the current and voltage sensors. Implement algorithms to track and accumulate energy consumption over time.

Elaboration: Develop algorithms that take the current and voltage values as inputs to calculate the real-time power consumption. Also, create mechanisms to track the cumulative energy consumption in kilowatt-hours (kWh) over a specific time period, ensuring accurate data representation.

* + - 1. **ESP32 and MQTT Server Integration:**

Establish a stable connection between the ESP32 Wifi Module and the MQTT Server Application. Implement data transfer protocols to send sensor data to the MQTT Server Application Dashboard in a secure and efficient manner.

Elaboration: Configure the ESP32 to communicate with the MQTT Server Application through Wi-Fi and develop a data transfer protocol that securely sends sensor data to the MQTT Server Application Dashboard. Ensure that the connection is reliable and can handle potential network disruptions.

* + - 1. **Real-time Data Visualization:**

Design an interactive and user-friendly dashboard in the MQTT Server Application for real-time visualization of voltage, current, power, and total energy consumption. Enable users to monitor and analyze their electricity consumption patterns effectively.

Elaboration: Create an intuitive dashboard using MQTT Server's interface builder that displays the real-time voltage, current, power, and cumulative energy consumption. Implement visualizations such as graphs, charts, and numerical displays to present data in a user-friendly and comprehensible format.

# Chapter 4

**Proposed Methodology**

### 4.1 Proposed Work

The proposed methodology for the project involves the following steps:

1. **Power Supply Section:**

**Transformer:** The project begins with a 200V AC input from the mains, which is stepped down to 12V AC using a transformer. This transformer ensures the voltage is at a suitable level for further processing.

**Rectification:** The 12V AC is then passed through a bridge rectifier composed of four diodes (1N4007), converting it into pulsating DC. This rectification process ensures that the output remains positive with respect to ground.

**Filtering:** A capacitor (1k microfarad) is added across the rectified output to smoothen the pulsating DC, reducing ripples and providing a relatively stable DC voltage.

**Voltage Regulation:** The filtered DC voltage is then regulated to 5V using a 7805 voltage regulator. This ensures a constant and stable 5V output regardless of variations in input voltage or load.

**Additional Capacitor**: Another capacitor is placed across the output of the 7805 regulator to further stabilize the voltage and reduce noise.

**LM1117 Voltage Regulator:** The 5V output from the 7805 regulator is further regulated to 3.3V using an LM1117 voltage regulator. This lower voltage level is suitable for powering the ESP32 microcontroller.

1. **Device Control Section:**

**Transistor as a Switch:** The regulated 3.3V output from the LM1117 regulator is used to control a transistor acting as a switch. When the microcontroller sends a signal to turn on a device, it activates the transistor, allowing current to flow through it.

**Relay Control:** The transistor's activation also triggers a relay, which acts as a switch for turning on/off larger loads such as bulbs. The relay is connected to the household electrical system, controlling the power supply to the bulbs.

1. **Current Transformer (CT) Circuit:**

**AC Output to DC Conversion**: The CT circuit consists of three different CTs (current transformers) which are connected to monitor the current flowing through the electrical system of the house.

**Rectification**: The AC output from the CT is rectified into DC using a rectifier.

**Voltage Division**: The rectified DC voltage is then divided using resistors, producing a voltage proportional to the current passing through the CT.

**Analog Input to ESP32**: The voltage proportional to the current is fed into the ESP32 microcontroller as an analog input. The ESP32 reads this voltage and converts it into a digital value using its built-in ADC (Analog-to-Digital Converter).

**Power Consumption Calculation**: Using the digital value obtained from the CT circuit, the ESP32 calculates the power consumption of the devices connected to the system. This calculation is based on the relationship between current, voltage, and power (P = VI).

### Applications

**1.Residential Energy Monitoring:** Residents can monitor real-time power consumption to make informed decisions and reduce electricity bills.

**2. Commercial Energy Management:** Businesses optimize energy usage, reduce costs, and enhance efficiency by monitoring consumption patterns.

**3. Industrial Power Monitoring:** Industries maintain efficiency and prevent failures by tracking energy usage and identifying improvement areas.

**6. Energy Conservation Initiatives:** Gather data on consumption patterns, identify high-use areas, and implement energy-saving initiatives.

**7. Data-Driven Decision Making:** Businesses use real-time data to optimize energy usage, reduce waste, and enhance efficiency.

### Block Diagram

### 

### Figure 4.1: Bock Diagram of IOT based energy smart meter.

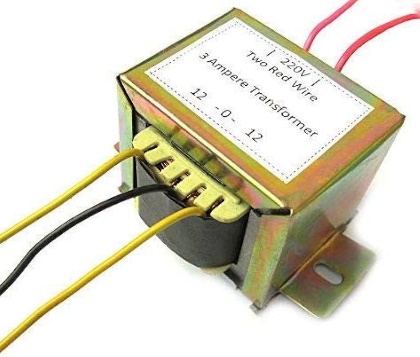
# Chapter 5

**Hardware Implementation**

### Components Used

**1.** **Transformer that converts 200 volts to 12 volts**

 A transformer is an electrical device that transfers electrical energy between two or more circuits through electromagnetic induction. In this case, it converts high voltage (200 volts) to a lower voltage (12 volts). It typically consists of two coils of wire wound around a core, where the input voltage is applied to one coil (primary) and the output voltage is induced in the other coil (secondary) through magnetic coupling.



**Figure 5.1**: Transformer

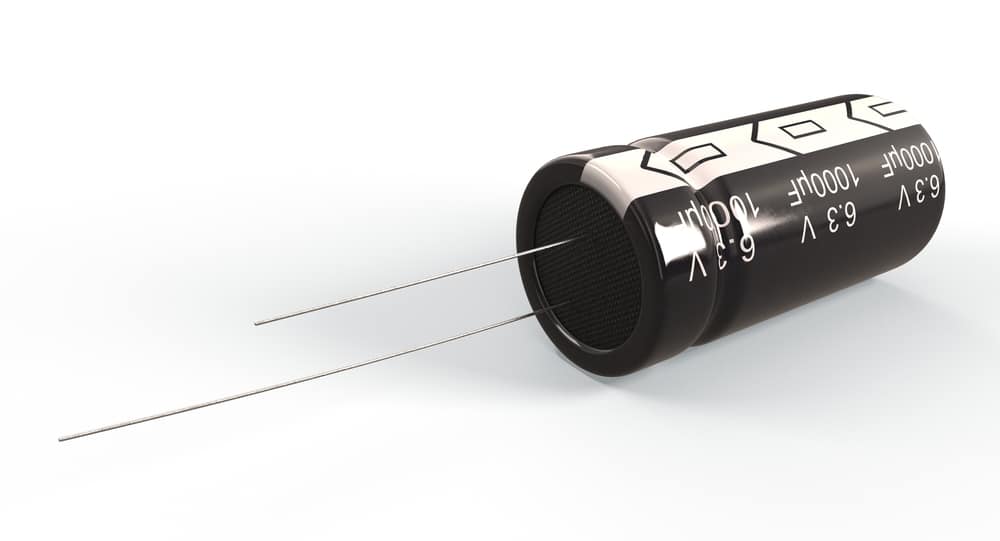
**2.** **7805 voltage regulator**

 The 7805 is a popular linear voltage regulator integrated circuit (IC) that provides a stable output voltage of +5 volts. It is widely used in electronic circuits to regulate voltage and maintain a constant output voltage despite fluctuations in input voltage or load.

**Figure 5.2**: Voltage Regulator

**3.** **Capacitor 470uF**

 Capacitors are passive electronic components that store and release electrical energy in the form of an electric field. The "470uF" designation indicates the



**Figure 5.3:** Capacitor

capacitance of the capacitor, which is 470 microfarads (uF). Capacitors are commonly used in circuits for filtering, decoupling, timing, and energy storage.

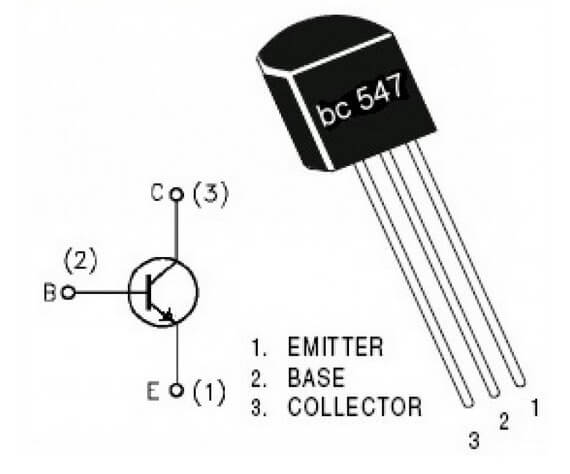
**4.** **Relay 12 volt handling current 7.5 A**

A relay is an electromechanical switch operated by an electrical current. It consists of a coil and one or more sets of contacts. When the coil is energized, it creates a magnetic field that pulls the contacts to make or break connections. This specific relay operates at a coil voltage of 12 volts and can handle a maximum current of 7.5 amps. Relays are used in circuits to control high-power devices with low-power signals.

**Figure 5.4:** Relay 12V

**5.** **NPN transistor BC547**

The BC547 is a common NPN bipolar junction transistor (BJT) used in amplification and switching applications. It consists of three layers of semiconductor materials and acts as a current amplifier. As an NPN transistor, it conducts current when a small current is applied to its base terminal. It is often used in low-power amplification circuits and as a switch to control higher currents.



**Figure 5.5:** NPN Transistor

**6.** **CT (Current Transformer)**

A current transformer is a type of transformer that is used to measure alternating current (AC) by producing a current in its secondary winding that is proportional



**Figure 5.6:** Current Transformer

to the current flowing in its primary winding. It is commonly used in electricity metering and protection systems to monitor and measure current flow in high-voltage circuits.

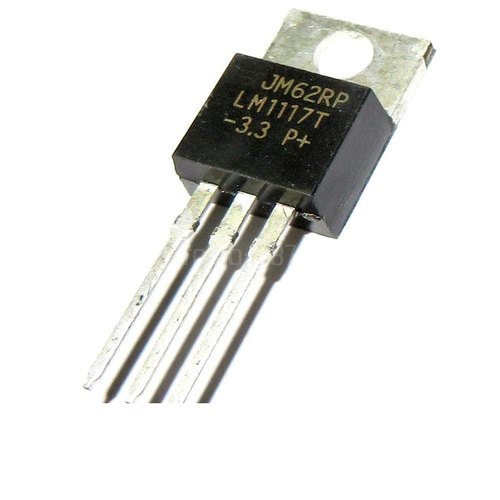
**7.** **ESP32**

 The ESP32 is a highly integrated Wi-Fi and Bluetooth-enabled system-on-chip (SoC) designed for Internet of Things (Io Transformer T) applications.

**Figure 5.6:** ESP 32

It features a dual-core processor, built-in Wi-Fi and Bluetooth connectivity, various digital and analog I/O interfaces, and support for a wide range of peripheral devices. The ESP32 is widely used for developing IoT projects, smart devices, and wireless sensor networks.

**8**. **LM1117 Voltage Regulator**

 The LM1117 is a series of low dropout voltage regulators that provide a stable output voltage with a minimal voltage dropout.

**Figure 5.8:** LM1117 Voltage Regulator

It is commonly used to regulate voltage in electronic circuits, especially in applications where a stable voltage supply is required despite variations in input voltage or load conditions. The LM1117 comes in various voltage output options and is available in different package types for versatility in circuit design.

# 

# Chapter 6

# Software Used

# 6.1 MQTT Server

6.1.1MQTT Server Open Source

MQTT (Message Queuing Telemetry Transport) is an open-source messaging protocol designed for efficient communication between devices in the Internet of Things (IoT) and other resource- constrained environments. MQTT facilitates lightweight, real-time data exchange between connected devices by utilizing a publish-subscribe model. In this model, devices can publish information to specific "topics," and other devices interested in that information can subscribe to those topics.

The key characteristics of MQTT include its minimal overhead, making it well-suited for low- bandwidth and high-latency networks. It operates on top of the TCP/IP protocol, ensuring reliability and scalability. MQTT's open-source nature means that its specifications and implementations are publicly available, allowing developers to integrate it into their applications and systems freely.

**6.1.2 Arduino**

Arduino is an open-source electronics platform that allows for easy creation of interactive and programmable electronic projects. It consists of a microcontroller board and a development environment for writing and uploading code to the board.

The Arduino board contains a microcontroller that can be programmed to perform various functions, such as reading inputs from sensors, controlling outputs to motors or LEDs, and communicating with other devices using serial communication, Bluetooth, or Wi-Fi.

The Arduino platform is widely used in hobbyist and educational settings, as well as in professional applications such as robotics, automation, and data logging. Its open- source nature and large community of developers make it easy to find resources and support for building a wide variety of electronic projects.

Source CODE

#include "EspMQTTClient.h"

#define CT1 35

#define CT2 32

#define CT3 33

#define Relay1 2

#define Relay2 15

#define Relay3 13 WiFiClient APIClient;

EspMQTTClient client( "moniter", "moniter123",

"broker.hivemq.com", // MQTT Broker server ip "", // Can be omitted if not needed

"", // Can be omitted if not needed

"374sd6MMSDFG54fTestClient", // Client name that uniquely identify your device 1883 // The MQTT port, default to 1883. this line can be omitted

);

void setup() { pinMode(Relay1, OUTPUT); pinMode(Relay2, OUTPUT); pinMode(Relay3, OUTPUT); digitalWrite(Relay1, 1);

digitalWrite(Relay2, 1);

digitalWrite(Relay3, ); Serial.begin(115200);

client.enableDebuggingMessages(); // Enable debugging messages sent to serial output client.enableHTTPWebUpdater(); // Enable the web updater. User and password default to values of MQTTUsername and MQTTPassword. These can be overridded with enableHTTPWebUpdater("user", "password").

client.enableOTA(); // Enable OTA (Over The Air) updates. Password defaults to MQTTPassword. Port is the default OTA port. Can be overridden with enableOTA("password", port).

}

long t1 = 0; long t2 = 0; long t3 = 0;

byte CtCut1 = 0; byte CtCut2 = 0; byte CtCut3 = 0; long t = 0;

void loop() {

float c1 = ReadCurrent1(CT1); float c2 = ReadCurrent2(CT2); float c3 = ReadCurrent3(CT3);

Serial.print(c1); Serial.print(", \t"); Serial.print(c2); Serial.print(", \t"); Serial.print(c3); Serial.println(", \t");

//return; if(c1 > 150)

{

int i = 0;

for(i = 0 ; i < 5 ;i++)

{

c1 = ReadCurrent1(CT1); if(c1 > 150)

{

}

if(i > 4)

{

CtCut1 = 1;

}

}

if(millis() > t+1000)

{

t1--;

t2--;

t3--;

t = millis();

}

if(t1 > 0 & CtCut1 == 0)

{

client.publish("indore/gsits/indicator1", String(1)); digitalWrite(Relay1, 1);

}

else

{

client.publish("indore/gsits/indicator1", String(0)); digitalWrite(Relay1, 0);

}

if(t2 > 0 & CtCut2 == 0)

{

client.publish("indore/gsits/indicator2", String(1)); digitalWrite(Relay2, 1);

}

else

{

client.publish("indore/gsits/indicator2", String(0)); digitalWrite(Relay2, 0);

}

if(t3 > 0 & CtCut3 == 0)

{

client.publish("indore/gsits/indicator3", String(1)); digitalWrite(Relay3, 1);

}

else

{

client.publish("indore/gsits/indicator3", String(0)); digitalWrite(Relay3, 0);

delay(50);

client.publish("indore/gsits/ct1", String(c1)); client.publish("indore/gsits/ct2", String(c2)); client.publish("indore/gsits/ct3", String(c3)); client.loop();

// delay(50);

}

void onConnectionEstablished()

{

// Subscribe to "mytopic/t est" and display received message to Serial client.subscribe("indore/gsits/r1", [](const String & payload) { Serial.println(payload.toInt());

t1 = payload.toInt(); CtCut1 = 0;

});

client.subscribe("indore/gsits/r2", [](const String & payload) { Serial.println(payload.toInt());

t2 = payload.toInt(); CtCut2 = 0;

});

client.subscribe("indore/gsits/r3", [](const String & payload) { Serial.println(payload.toInt());

t3 = payload.toInt(); CtCut3 = 0;

});

client.executeDelayed(5 \* 1000, []() { Serial.println("send");

//client.publish("mytopic/wildcardtest/test123", "This is a message sent 5 seconds later");

});

}

float ReadCurrent1(int pin)

{

int maxVal = 0 ;

for(int i = 0 ; i < 300 ; i++)

{

int x = analogRead(pin);

// Serial.println(x);

if(x > maxVal)

{

maxVal = x;

}

delay(1); client.loop();

}

// Convert the analog reading (which goes from 0 - 1023) to a voltage (0 - 5V): float voltage = maxVal \* (3300.0 / 4096.0);

// print out the value you read:

// Serial.println(voltage);

return voltage \* 0.0934579439252336;

}

float ReadCurrent2(int pin)

{

int maxVal = 0 ;

for(int i = 0 ; i < 300 ; i++)

{

int x = analogRead(pin);

// Serial.println(x); if(x > maxVal)

{

maxVal = x;

}

delay(1); client.loop();

}

// Convert the analog reading (which goes from 0 - 1023) to a voltage (0 - 5V): float voltage = maxVal \* (3300.0 / 4096.0);

// print out the value you read:

// Serial.println(voltage);

return voltage \* 0.0982318271119843;

}

float ReadCurrent3(int pin)

{

int maxVal = 0 ;

for(int i = 0 ; i < 300 ; i++)

{

int x = analogRead(pin);

// Serial.println(x); if(x > maxVal)

{

maxVal = x;

}

delay(1); client.loop();

}

// Convert the analog reading (which goes from 0 - 1023) to a voltage (0 - 5V): float voltage = maxVal \* (3300.0 / 4096.0);

// print out the value you read:

// Serial.println(voltage);

return voltage \* 0.0757575757575758;

}

**6.2  Using Apache Tomcat**

**6.2.1 Apache Tomcat**

Apache Tomcat (called "Tomcat" for short) is a free and open-source implementation of the Jakarta Servlet, Jakarta Expression Language, and WebSocket technologies. It provides a "pure Java" HTTP web server environment in which Java code can also run. Thus it is a Java web application server, although not a full JEE application server.

Tomcat is developed and maintained by an open community of developers under the auspices of the Apache Software Foundation, released under the Apache License 2.0 license.

**6.2.2 MySQL - for Database**

MySQL is the world’s most popular open-source database. The “SQL” part of “MySQL” stands for “Structured Query Language.” SQL is the most common standardized language used to access databases. Depending on your programming environment, you might enter SQL directly (for example, to generate reports), embed SQL statements into code written in another language, or use a language-specific API that hides the SQL syntax.

**6.2.3 PHP - For code implementation**

PHP is a [general-purpose](https://en.wikipedia.org/wiki/General-purpose_programming_language) [scripting language](https://en.wikipedia.org/wiki/Scripting_language) geared towards [web development](https://en.wikipedia.org/wiki/Web_development). On a web server, the result of the [interpreted](https://en.wikipedia.org/wiki/Interpreter_(computing)) and executed PHP code—which may be any type of data, such as generated [HTML](https://en.wikipedia.org/wiki/HTML) or [binary](https://en.wikipedia.org/wiki/Binary_number) image data—would form the whole or part of an [HTTP](https://en.wikipedia.org/wiki/HTTP) response. Various [web template systems](https://en.wikipedia.org/wiki/Web_template_system), web [content management systems](https://en.wikipedia.org/wiki/Content_management_system), and [web frameworks](https://en.wikipedia.org/wiki/Web_framework) exist that can be employed to orchestrate or facilitate the generation of that response.

**Code - PHP**

import time

import argparse

import paho.mqtt.client as mqtt

import json

import requests

def on\_connect(client, userdata, flags, rc):

print("Conn ected with result code "+str(rc))

client.subscribe("reco",1)

count = 0

ct1 = ""

ct2 = ""

ct3 = ""

in1 = ""

in2 = ""

in3 = ""

def on\_message(client, userdata, msg):

global ct1

global ct2

global ct3

global in1

global in2

global in3

global count

#print(msg.topic)

#print(msg.topic[13:])

topic = msg.topic[13:]

value = msg.payload.decode()

print("value = " + str(float(value)))

if(topic == "ct1"):

ct1 = str(float(value))

count = count + 1

print("ct1 Recive" + str(value))

if(topic == "ct2"):

ct2 = str(float(value))

count = count + 1

print("ct2 Recive" + str(value))

if(topic == "ct3"):

ct3 = str(float(value))

count = count + 1

print("ct3 Recive" + str(value))

if(topic == "indicator1"):

in1 = str(float(value))

count = count + 1

print("Indicator 1 Recive" + str(value))

if(topic == "indicator2"):

in2 = str(float(value))

count = count + 1

print("Indicator 2 Recive" + str(value))

if(topic == "indicator3"):

in3 = str(float(value))

count = count + 1

print("Indicator 3 Recive" + str(value))

if(count == 6):

count = 0;

print("Call API")

#print("msg.payload" + str(msg.payload))

api\_url = "http://localhost/smartmeter/api.php?in1=" + in1 + "&in2=" + in2 + "&in3=" + in3 + "&ct1=" + ct1 + "&ct2=" + ct2 + "&ct3=" + ct3

print(api\_url)

res = requests.get(api\_url)

print(str(res.content))

print(res.status\_code)

time.sleep(1.0)

broker\_address="broker.hivemq.com"

port=1883

#broker\_address="iot.eclipse.org" #use external broker

client = mqtt.Client("356sflsdgxddfuhafoasp123") #create new instance

client.on\_connect = on\_connect

client.on\_message = on\_message

client.connect(broker\_address,port) #connect to broker

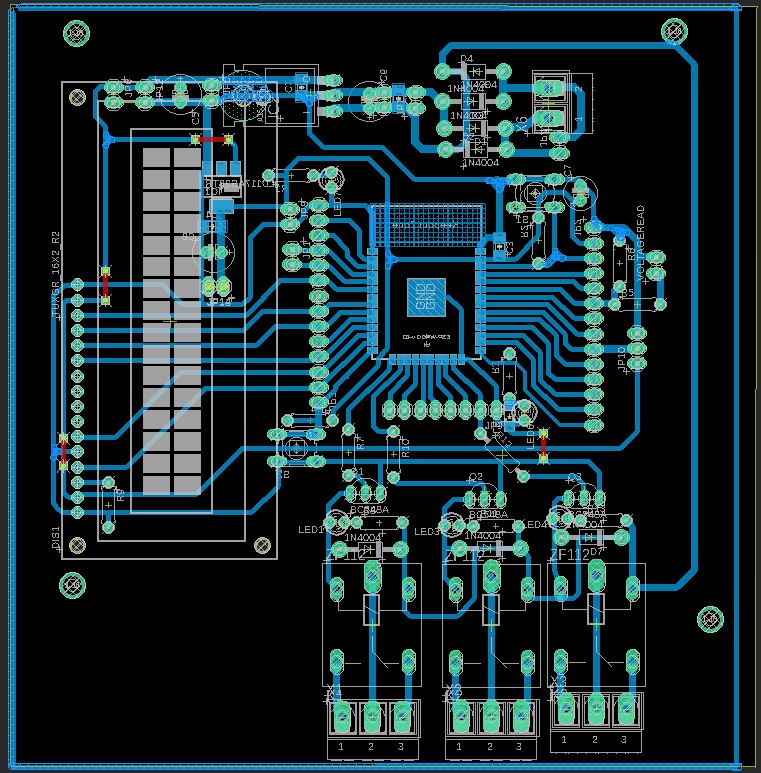
client.subscribe("indore/gsits/#")

client.on\_message = on\_message

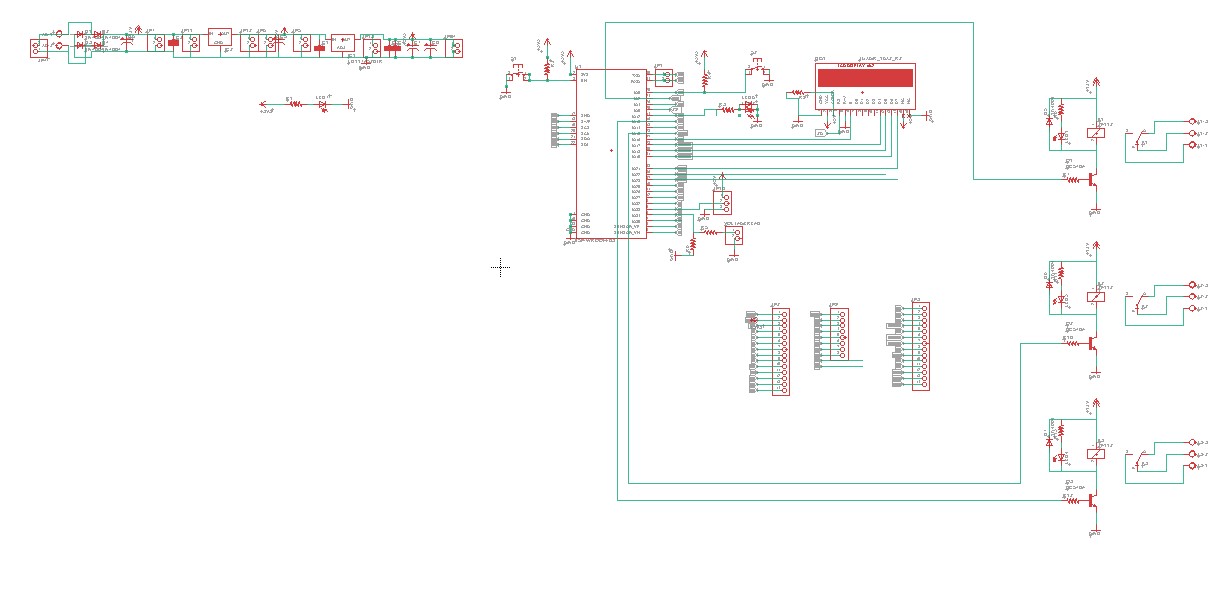
while True:

client.loop\_start()

**PCB Circuit Diagram**



**Figure 6.1:** PCB Circuit layout of the energy smart meter



**Figure 6.2:** PCB Schematic of energy smart meter

**Chapter 7**

**Results and Conclusion**

### 7.1 Results

Based on the project description provided, the following results can be achieved:

1. **Automation of Meter Reading**: With the implementation of the IoT-based Electricity Energy Meter, the need for manual reading of meters is eliminated, thereby saving time and effort.
2. **Cost and Time Efficiency**: Implementing Internet of Things technology reduces the need for physical labor, resulting in cost savings and improved efficiency in monitoring electricity consumption.
3. **Selection of Sensors**: The SCT-013 Non-Invasive AC Current Sensor Split Core Type Clamp Meter Sensor and the ZMPT101B AC Voltage Sensor are identified as the best options for measuring current and voltage, respectively, ensuring accurate data collection for the energy meter.
4. **Data Collection and Transmission**: By interfacing the selected sensors with the ESP32 WiFi Module, data can be collected and transmitted to the MQTT Server Application, facilitating remote monitoring of electricity usage.
5. **Real-Time Data Display**: The MQTT Server Application Dashboard will display essential parameters such as Voltage, Current, Power, and the total unit consumed in kilowatt-hours (kWh), providing users with real-time insights into their electricity consumption.
6. **Enhanced Control and Monitoring**: Users can conveniently monitor their energy consumption patterns, enabling them to make informed decisions regarding energy usage and potentially adopt more energy-efficient practices.

By integrating the selected sensors and implementing the proposed IoT-based Electricity Energy Meter, users can benefit from streamlined and efficient electricity monitoring and management.

### 7.2 Conclusion & future scope

Conclusion:

The implementation of the IoT-based Smart Energy Meter using the Current Transformer integrated with ESP32 WiFi Module, and MQTT Server/Apache Tomcat Server. Application Dashboard offers an efficient and automated solution for monitoring and managing electricity consumption. By leveraging IoT technology, the project has successfully simplified the process of data collection and transmission, providing users with real-time insights into their power usage. The accuracy and reliability of the selected sensors contribute to the effectiveness of the system, ensuring precise measurement of current, voltage, power, and total energy consumption.

Future Scope:

1. **Integration of Smart Grid Technology**: Further integration of the IoT-based Electricity Energy Meter with smart grid technology can enable advanced functionalities such as demand-side management, real-time pricing, and load balancing, contributing to more efficient and sustainable energy management.
2. **Expansion of Monitoring Capabilities**: The system can be expanded to monitor and manage other utility services, such as water and gas consumption, providing users with a comprehensive solution for tracking their overall resource usage and promoting sustainable practices.
3. **Enhanced User Interface and Interactivity**: Improving the user interface of the MQTT Server Application Dashboard by incorporating additional features such as energy usage alerts, customizable data visualization, and user-friendly controls can enhance the overall user experience and encourage active engagement in energy conservation efforts.

By exploring these future avenues, the IoT-based Electricity Energy Meter can evolve into a comprehensive and versatile solution for efficient energy management, promoting sustainability and empowering users to make informed decisions regarding their energy consumption.

# Bibliography

###### **[1]** "Building Smart Drones with ESP8266 and Arduino" by *Syed Omar Faruk Towaha*. This book provides insights into the integration of IoT capabilities with the ESP32 microcontroller, offering practical guidance on building smart drones using these technologies.

**[2]** "IoT Projects with ESP32" by *Agus Kurniawan*. This book explores various IoT projects and applications using the ESP32 platform, offering a comprehensive understanding of its capabilities and how it can be leveraged for diverse IoT applications.

###### **[3]** "Internet of Things: Principles and Paradigms" by *Rajkumar Buyya, Amir Vahid Dastjerdi, and Srirama Krishn*. This comprehensive guide covers the fundamental principles and paradigms of the Internet of Things, offering insights into various aspects of IoT technology, including sensors, networks, and applications.

**[4]** "Sensors for Mechatronics" by *Paul P.L. Regtien*. This book provides an in-depth understanding of sensor technologies used in mechatronic systems, including voltage and current sensors, and offers practical insights into their applications in various engineering fields.

###### **[5]** "Fundamentals of Electric Circuits" by *Charles K. Alexander and Matthew N. O. Sadiku*. This textbook covers the fundamental concepts of electric circuits, including voltage, current, and power, providing a solid foundation for understanding the principles underlying the operation of electrical systems and sensors.