Internet of Things-Based Smart Electric Energy Meter Using Telegram App

Industry oriented mini project report submitted in partial fulfilment of the requirements

for the degree of

Bachelor of Technology

in

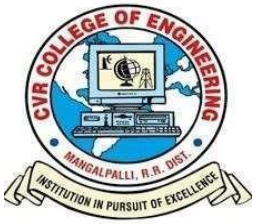
Electronics and Communication Engineering

Submitted by

Akula Rohan (22B81A04A6)

Porandla Rishikesh (2281A04A5)

Padakanti Sai Sumith (22B81A04A9)



DEPARTMENT OF ELECTRONICS & COMMUNICATION ENGINEERING

CVR COLLEGE OF ENGINEERING

(An Autonomous Institution & Affiliated to JNTUH)
Ibrahimpatnam (M), Ranga Reddy (D), Telangana

2024-25

Internet of Things-Based Smart Electric Energy Meter Using Telegram App

Industry oriented mini project report submitted in partial fulfilment of the requirements

for the degree of

Bachelor of Technology

in

Electronics and Communication Engineering

Submitted by

Akula Rohan (22B81A04A6)

Porandla Rishikesh (2281A04A5)

Padakanti Sai Sumith (22B81A04A9)

Under the Supervision of

Mr.B. Ramesh

Senior Assistant Professor

DEPARTMENT OF ELECTRONICS & COMMUNICATION ENGINEERING

CVR COLLEGE OF ENGINEERING

(An Autonomous Institution & Affiliated to JNTUH)
Ibrahimpatnam (M), Ranga Reddy (D), Telangana

2024-25

Cherabuddi Education Society's
CVR COLLEGE OF ENGINEERING
(An Autonomous Institution)
ACCREDITED BY NATIONAL BOARD OF ACCREDITATION, AICTE
(Approved by AICTE & Govt. of Telangana and Affiliated to JNT University)
Vastunagar, Mangalpalli (V), Ibrahimpatan (M), R.R. District, PIN - 501 510
Web : http://cvr.ac.in, email : info@cvr.ac.in
Ph : 08414 - 252222, 252369, Office Telefax : 252396, Principal : 252396 (O)

Certificate

This is to certify that this industry oriented mini project work titled "**Internet of Things-Based Smart Electric Energy Meter Using Telegram App**" submitted to the **CVR College of Engineering**, affiliated to **JNTUH**, by **A. Rohan (22B81A04A6), P. Rishikesh (22B8104A5), P. Sai Sumith (22B81A04A9)**, is a Bonafide record of the work done by the students towards partial fulfilment of requirements for the award of the degree of **Bachelor of Technology in Electronics & Communication Engineering** during the academic year 2024-2025.

Supervisor

Mr.B.Ramesh
Senior Assistant Professor
Dept. of ECE

Head of the Department

Dr. P. Srinivas Rao
Head of Department
ECE

Project In-charge

Dr.K.A.Jyotsna
Associate Professor
L.Manjunath
Associate Professor
Dept. of ECE

External Examiner

Place:
Date:

Acknowledgement

The satisfaction that accompanies the successful completion of any task would be incomplete without the mention of the people who made it possible and whose encouragement and guidance has been a source of inspiration throughout the course of the project work.

We would like to express our sincere gratitude to our supervisor, **Mr.B.Ramesh, Senior Assistant Professor**, ECE Dept, CVR College of Engineering, whose guidance and valuable suggestions have been indispensable to bring about the successful completion of our project.

We wish to acknowledge special thanks to the Project Coordinator, **Dr.K.A. Jyotsna, Associate Professor, L. Manjunath, Associate Professor,** Designation, of ECE Dept. for assessing seminars, inspiration, moral support and giving us valuable suggestions in our project.

We would also like to express our gratitude to all the staff members and lab faculty, department of Electronics and Communication Engineering, CVR College of Engineering for the constant help and support.

We would like to express our sincere thanks to our Head of the ECE Department, **Dr. P.Srinivas Rao** for his continuous support and valuable suggestions in completing our project.

It is great pleasure to convey our profound sense of gratitude to our principal **Dr. K**. **Ramamohan Reddy**, CVR College of Engineering for having been kind enough for arranging the necessary facilities for executing the project in the college.

We wish a deep sense of gratitude and heartfelt thanks to management for providing excellent lab facilities and tools. Finally, we thank all those whose guidance helped us in this regard.

Abstract

The increasing demand for real-time electricity monitoring has driven the need for smart metering solutions that enhance energy efficiency and user accessibility. This project presents the development of an IoT-based Smart Electricity Energy Meter utilizing the ESP32 microcontroller and the Telegram app for remote monitoring. By incorporating high-precision sensors—SCT-013 for current measurement and ZMPT101B for voltage detection—the system accurately measures voltage, current, power, and total energy consumption in kilowatt-hours (kWh). The collected data is displayed in real-time on both an LCD and the Telegram app, allowing users to monitor their electricity usage remotely. This smart metering solution aims to improve monitoring electricity consumption, promote energy conservation, and facilitate data-driven decision-making for users.

Keywords: IoT, Smart Energy Meter, ESP32, Telegram Bot, Real-time Monitoring, Voltage and Current Measurement, Energy Consumption, Remote Monitoring, Power Efficiency.

Table of Contents

Chapter 5 Hardware Description

List of Figures

List of Tables

List of Abbreviations

MQTT – Message Queuing Telemetry Transport

ADC – Analog to Digital Converter

LCD – Liquid Crystal Display

kWh – Kilowatt-hour

LED – Light Emitting Diode

DC – Direct Current

RMS – Root Mean Square

IRMS – Current Root Mean Square

VRMS – Voltage Root Mean Square

CT – Current Transformer

Chapter 1
OVERVIEW

1.1 Introduction

Energy consumption is escalating globally at an unprecedented pace, driven by rapid industrialization, urban expansion, and the proliferation of electronic devices. This surge in demand is not only inflating energy costs but also exerting tremendous pressure on finite natural resources, pushing the world toward an unsustainable trajectory. Traditional energy meters, while functional, suffer from significant limitations—they operate passively and provide consumption data only after a billing cycle is completed. This delay in feedback leaves consumers unaware of their real-time usage patterns, thereby reducing opportunities to adjust behavior, identify energy leaks, or optimize appliance operation.

To bridge this critical gap, this project introduces an IoT-based Smart Energy Meter leveraging the powerful and versatile ESP32 microcontroller, integrated with SCT-013 current sensors and ZMPT101B voltage sensors. Together, these components enable accurate, real-time measurement of key electrical parameters including voltage, current, power, and cumulative energy (kWh). The captured data is processed and transmitted seamlessly via a Telegram bot, offering users instant access to their energy metrics from any location. This empowers users with actionable insights, allowing them to monitor usage trends, detect anomalies, reduce unnecessary consumption, and ultimately lower electricity bills.

1.2 Aim of the System

The primary aim of the Smart Energy Meter is to enable real-time electricity monitoring, allowing users to track consumption, optimize usage, reduce wastage, and lower costs through instant updates in the Telegram app.

1.3 Methodology

The development of the Smart Energy Meter followed a systematic and hardware-focused design flow, beginning with component selection and ending with real-time performance validation. The ESP32 development board was chosen as the core processing unit due to its

built-in Wi-Fi and low-power operation, ideal for IoT applications. Accurate current and voltage measurements were enabled using the SCT-013 and ZMPT101B sensors respectively. These analog values were digitized, processed, and used to calculate real-time power and energy consumption (kWh). The processed data was simultaneously displayed on an LCD screen and transmitted via the Telegram platform using the ESP32's network capabilities. Software development was carried out using the Arduino IDE, where customized firmware was written to handle sensor interfacing, data computation, and API communication. To maintain signal stability and reduce noise interference, discrete passive components were mounted on a perfboard for compactness and reliability. The overall circuit was optimized for low cost, modularity, and ease of replication, ensuring the system could be deployed as a scalable solution for real-time, remote energy monitoring.

1.4 Significance of work

The Smart Energy Meter provides real-time monitoring of voltage, current, power, and energy consumption. It helps users reduce wastage by tracking usage instantly and avoiding unexpected high bills. The integration with Telegram enables remote access and control through IoT. The system identifies inefficient appliances and promotes energy-saving habits.

1.5 Organisation of work

This report details the development of a Smart IoT-based Energy Monitoring System designed for accurate power tracking, efficient energy use, and real-time remote monitoring. The project combines embedded systems with IoT technologies, utilizing components like the ESP32 microcontroller, current and voltage sensors, and a display unit, while enabling Telegram-based notifications. The organization of the report is as follows: Chapter 1 outlines the motivation, methodology, and structure of the work. Chapter 2 introduces energy monitoring and smart meters, discussing performance metrics, system architecture, communication protocols, and security concerns. Chapter 3 presents a literature review of existing solutions and technologies. Chapter 4 explores the fundamentals of embedded systems and IoT, including real-time operating systems and their characteristics. Chapter 5 describes the hardware used, its specifications, and operating principles. Chapter 6 explains the complete system architecture and its functional workflow. Chapter 7 showcases the experimental results from different system states and monitoring features. Chapter 8

concludes the work, while Chapter 9 discusses the future scope and potential system enhancements.

1.6 Conclusion

The Smart IoT-based Energy Monitoring System offers an efficient and practical solution for real-time power tracking and remote monitoring. By combining embedded hardware with IoT connectivity, the system ensures accuracy, low power consumption, and user-friendly access. This approach is well-suited for smart homes, industries, and future energy-aware environments, making it a valuable step toward intelligent energy management.

Chapter 2
INTRODUCTION

2.1 Introduction to Energy Monitoring

Energy monitoring refers to the systematic process of measuring, recording, and analysing the consumption of electrical energy over time. In a world where electricity powers everything from homes and industries to infrastructure and healthcare systems, efficient energy utilization has become a global priority. The rise in global energy demand is driven by population growth, urbanization, and the proliferation of electronic devices. Simultaneously, depleting natural resources and rising electricity costs pose serious challenges to sustainability and affordability.

Traditional energy management systems are reactive, not proactive. Consumers receive electricity bills only at the end of a billing cycle, often without any insight into when or how energy was consumed. This lack of visibility makes it difficult for users to make informed decisions about reducing their consumption or identifying waste. Moreover, utility providers also face challenges in managing load distribution, detecting faults, and optimizing the supply-demand balance across the grid.

Energy monitoring systems address these challenges by providing a real-time or near-real-time view of energy usage. By continuously tracking parameters such as voltage, current, power, and total energy consumed, these systems empower users to observe their usage behaviour, detect anomalies, and adopt energy-saving practices. Effective energy monitoring not only helps individuals reduce costs but also contributes to larger environmental goals by promoting more efficient use of electricity and reducing carbon emissions.

Furthermore, as energy tariffs become more dynamic with the introduction of time-of-use pricing models, the need for precise and responsive monitoring tools becomes even more essential. Households and businesses equipped with energy monitors can schedule high-energy tasks during off-peak hours, reducing their bills and easing the strain on the power grid during peak times. In this context, energy monitoring is not just a tool it is a foundational element of modern, sustainable energy systems.

2.2 Smart Energy Meter

A Smart Energy Meter is an advanced energy measurement device designed to provide continuous, accurate, and real-time monitoring of electrical energy consumption. Unlike traditional analog or digital meters that accumulate energy usage and display only cumulative values, smart meters enable bidirectional communication between the consumer and the energy provider. This functionality allows for automated readings, dynamic pricing, improved load management, and greater transparency in electricity usage.

The integration of embedded systems and Internet of Things (IoT) technologies is central to the functionality of modern smart meters. Typically, the system consists of a microcontroller current and voltage sensing modules , a display interface (LCD), and a wireless communication channel. These components work in tandem to measure real-time parameters like voltage, current, power, and energy (kWh), which can then be viewed locally or transmitted remotely to user interfaces such as mobile apps or messaging platforms.

The importance of smart meters lies in their ability to empower users with actionable insights into their energy consumption. By enabling real-time awareness, users can identify high-consumption appliances, adjust their usage patterns, and reduce energy waste. Moreover, service providers benefit from accurate billing, automated data collection, and improved demand forecasting. On a larger scale, smart meters contribute to the development of intelligent energy grids by supporting decentralized energy management and integrating renewable sources.

2.3 Performance Parameters in Smart Energy Meter

The performance of a smart energy meter system can be evaluated using parameters like response time, power consumption of the system, data accuracy, and reliability. The efficient integration of hardware components with IoT services plays a crucial role in determining system performance.

2.3.1 Data Accuracy

Data accuracy is a fundamental metric that reflects the precision of voltage, current, and power measurements. The system must minimize noise, drift, and offset errors introduced by sensors and analog front-end circuitry. Calibration techniques, such as software-based gain correction and hardware filtering, are employed to enhance measurement fidelity. The

use of precision analog-to-digital converters (ADCs) and differential input configurations can further improve accuracy. An error margin within ±1% is typically expected for reliable consumer-level applications.

.

2.3.2 Power Efficiency

Power efficiency refers to the energy consumed by the metering system itself. Since smart meters are designed for continuous operation, the internal circuitry, including microcontrollers, sensors, displays, and communication modules, must be optimized for low power consumption. Techniques such as duty-cycling, sleep modes, and dynamic frequency scaling are often implemented in the firmware to reduce energy draw during idle periods. Additionally, the use of efficient power regulation circuits, such as low-dropout regulators and switching converters, contributes to overall system efficiency.

2.3.3 Real-Time Response

Real-time response is critical for effective monitoring and alert generation. It is influenced by sensor sampling frequency, data processing latency, and communication speed. Typical real-time systems aim for a data refresh interval of 1 to 2 seconds. The responsiveness of the system is enhanced through non-blocking sensor reads, interrupt-driven data acquisition, and buffered communication stacks. Furthermore, protocols with low overhead such as MQTT or HTTP with RESTful APIs help achieve lower latency in cloud-based transmission.

2.4 System Architecture of a Smart Energy Meter

The architecture of the smart energy meter is designed to enable accurate acquisition, processing, and transmission of real-time energy data using a combination of embedded systems and wireless communication technologies. The system integrates multiple functional blocks that operate in coordination to deliver timely power monitoring and reporting.

The architecture is designed to be flexible and component-agnostic, allowing for integration with a wide range of sensors, microcontrollers, and communication platforms based on application needs and deployment environments.

The fundamental building blocks of a smart energy meter include:

- **Sensing Unit**: This block is responsible for capturing electrical parameters such as voltage and current from the connected load. A range of sensors—like Hall-effect sensors, current transformers, or resistive voltage dividers—may be employed, depending on the accuracy requirements and system voltage levels. Signal conditioning circuits such as operational amplifiers, filters, and voltage clamping diodes ensure safe and accurate analog signal delivery to the processing unit.
- **Data Acquisition and Processing Unit**: This module comprises a digital controller or microcontroller equipped with ADCs (Analog-to-Digital Converters). It digitizes the incoming analog signals and runs embedded algorithms to calculate key metrics including root-mean-square voltage (Vrms), current (Irms), active power, reactive power, and cumulative energy in kilowatt-hours (kWh). Processing is often enhanced using real-time operating systems or interrupt-driven architecture for improved response time.
- **Display Unit**: A 20x4 LCD module (with I2C interface) displays real-time energy metrics to the user. This interface provides immediate feedback without needing internet access.
- **Communication Unit**: This unit enables the transmission of computed data to external platforms or user interfaces. Wireless connectivity can be implemented through Wi-Fi, LoRa, GSM, or Zigbee based on the deployment scenario. Data can be pushed to cloud dashboards or messaging platforms (e.g., Telegram) using lightweight protocols like MQTT or HTTP. Telegram bots are commonly used to send updates to users, providing an intuitive and accessible communication channel for real-time usage data.
- **Power Supply Unit**: The entire system operates on a stable regulated DC supply. A power module with surge protection ensures consistent performance and isolates the control electronics from power-line fluctuations.

2.5 Communication Protocols in Smart Metering

Communication protocols form the backbone of smart metering systems by facilitating the transmission of data between the device and remote platforms. Depending on the application and environment, different protocols may be employed:

- Wi-Fi: Offers high data throughput and easy integration with cloud-based platforms, ideal for home and office applications with existing network infrastructure.
- Bluetooth: Suitable for short-range device configuration or local monitoring, though limited in range and bandwidth.
- Zigbee and LoRa: Low-power, long-range communication technologies ideal for wide-area monitoring applications in industrial and rural settings.
- MQTT (Message Queuing Telemetry Transport): A lightweight publish/subscribe protocol that is widely used for IoT communications due to its efficiency and low bandwidth consumption.
- HTTP/HTTPS: Standard web protocols used for data transmission between the device and web servers or REST APIs, suitable for integration with mobile apps and dashboards.

The choice of protocol depends on trade-offs between range, power consumption, data rate, and system complexity.

2.6 Security in Smart Energy Metering

As smart energy meters often operate over public networks and transmit user-sensitive data, robust security mechanisms are essential to protect against cyber threats and data breaches. Key security considerations include:

- Data Encryption: Secure data transmission using AES or TLS encryption ensures confidentiality and integrity.
- Authentication and Authorization: Devices must authenticate with servers and users to prevent unauthorized access.
- Firmware Integrity: Secure bootloaders and encrypted firmware updates help prevent tampering and malware injection.
- Network Security: Firewalls, VPNs, and intrusion detection systems can be integrated into the cloud infrastructure for added protection.

By implementing multi-layered security strategies, smart metering systems can offer reliable and safe data exchange between the device and external platforms.

2.7 Motivation

The rapid increase in global energy consumption, coupled with rising electricity prices, makes it imperative to optimize energy usage. Traditional energy metering systems, which provide only cumulative data over a billing cycle, fail to give consumers real-time insight into their power consumption. This lack of real-time monitoring results in inefficient energy use, as users are unaware of how their consumption patterns affect their bills.

The Smart Energy Meter addresses this gap by providing real-time data on energy consumption, empowering users to monitor and optimize their energy usage. With real-time feedback, consumers can identify high-energy-consuming appliances and adjust their usage, leading to reduced wastage and significant cost savings. By incorporating embedded systems and IoT technology, this project offers a scalable and low-cost solution for smart energy management. It also facilitates automation and remote monitoring, making it easier for consumers to stay informed and take immediate actions to reduce their electricity bills.

The system's integration with platforms like Telegram for remote data access offers unprecedented flexibility. This project aims to make energy consumption more transparent and user-centric, not only benefiting individual consumers but also aiding power companies with better load management and more accurate billing. Furthermore, the system supports the development of a smart grid, where energy consumption data can be utilized for efficient energy distribution, helping reduce the overall carbon footprint and contributing to more sustainable energy consumption practices.

2.8 Conclusion

This chapter introduced the fundamentals of smart energy meters, with a focus on IoT-based systems for monitoring and controlling energy consumption in real-time. It outlined the benefits of smart energy meters over traditional systems, including better accuracy, remote accessibility, and energy optimization. Key components such as sensor integration, wireless communication, and cloud-based platforms were discussed, highlighting their role in the functionality of modern energy meter.The motivation for this work lies in creating an affordable, scalable solution that empowers users to track their energy usage, receive real-time notifications, and optimize their consumption.

Chapter 3

LITERATURE REVIEW

3.1 Introduction

The rapid advancements in Internet of Things (IoT) and smart grid technologies have revolutionized energy monitoring systems, enabling more efficient and real-time power tracking. Embedded systems like ESP32 and NodeMCU, coupled with non-invasive sensors such as SCT-013 (current transformer) and ZMPT101B (voltage sensor), provide an affordable solution for accurate power measurement. Recent research highlights various approaches to optimizing these systems for scalability, low-cost implementation, and user engagement.

Notable works have integrated IoT platforms like Blynk and Telegram for real-time communication, enabling users to receive instant notifications and control their energy consumption remotely. This project builds upon these advancements, proposing a smart energy meter that combines precise sensor data with cloud-based analytics, leveraging ThingSpeak for data visualization and Telegram's API for real-time alerts. The aim is to provide an intelligent, cost-effective solution for energy monitoring in both residential and industrial settings.

3.2 Literature Review

The rapid evolution of IoT and smart grid technologies has catalyzed significant advancements in energy monitoring systems, focusing on real-time data acquisition, cloud-based analytics, and user-friendly interfaces. A growing body of research has explored the development of smart energy meters using cost-effective embedded systems like the ESP32 and NodeMCU, integrated with non-invasive sensors such as the SCT-013 current transformer and ZMPT101B voltage sensor, showcasing their ability to deliver accurate power measurements while ensuring affordability and scalability. Prabu (2023) developed a robust Blynk 2.0-based monitoring system utilizing these sensors, demonstrating their reliability for real-time energy tracking [1]. In further optimization, Paul Macheso and Doreen Thotho (2022) enhanced the

ESP32's capabilities by incorporating advanced CT sensors for precise power measurements and visualizations via Blynk [2].

Building on these innovations, Hala J. El-Khozondar et al. (2024) proposed a unique integration of WhatsApp notifications and secure Wi-Fi protocols, significantly improving user engagement and system reliability [3]. In the automation domain, Santosh Gadekar et al. (2021) utilized the HLW8012 sensor with the ESP32 for seamless billing and consumption tracking, marking a significant leap in smart energy systems [4]. Meanwhile, Ramyar R. Mohassel et al. (2014) offered valuable theoretical insights in their survey on AMI systems, providing a solid foundation for smart grid integration [5].

To address the need for intuitive interfaces, S. Vivekanandan et al. (2021) and Dicky A. Nugraha & Amirullah (2023) pioneered Telegram-based monitoring solutions, excelling in low-latency communication and remote-control features [6][7]. Expanding on real-time data visualization, Vineela Reddy et al. (2021) introduced dynamic IoT dashboards, further enriching user experiences [8]. A. D. Patel & J. D. Patel (2020) leveraged Firebase's cloud infrastructure for scalable data storage and analytics, contributing to the optimization of smart energy monitoring systems [9]. Furthermore, M. Alvi et al. (2019) ensured uninterrupted operations with GSM fallback mechanisms in unstable network environments [10]. The MQTT protocol, as implemented by R. Kumar & S. Das (2018), further optimized communication efficiency, while A. Khan & S. Roy (2020) demonstrated the potential of ThingSpeak for advanced energy data processing [11][12].

Despite these remarkable developments, existing systems often overlook the powerful integration of the SCT-013 and ZMPT101B sensors with the dual-core ESP32 processing power, along with fully utilizing Telegram's instant notification capabilities. This project addresses this gap by offering an intelligent, cost-effective solution that not only ensures precise energy measurements through a well-calibrated sensor network but also delivers superior user experiences by leveraging Telegram's robust API for real-time alerts, historical data analysis, and remote load control. This new approach sets a higher standard in smart energy metering systems.

In addition to the advances in hardware and software integration, ThingSpeak has proven to be an excellent platform for real-time data processing and analytics in smart energy systems. By providing cloud-based data storage, visualization tools, and easy integration with IoT devices,

ThingSpeak helps in monitoring and optimizing energy usage patterns efficiently. This platform's ability to handle large volumes of data and generate real-time reports plays a crucial role in enhancing energy monitoring applications, making it an essential component of modern smart grid and energy metering systems.

3.3 Conclusion

Smart Energy Meter Design and Implementation plays a critical role in real-time energy monitoring and efficient power management. Developing an effective, low-cost, and scalable solution remains a significant challenge for researchers. This project explores the design methodologies and application strategies for IoT-based smart energy meters, focusing on the integration of ESP32, NodeMCU, and non-invasive sensors like SCT-013 and ZMPT101B.

The design approaches for smart energy meters aim to optimize sensor accuracy, enhance data acquisition rates, and minimize power consumption while maintaining system scalability. Cloud-based platforms like ThingSpeak and user interfaces such as Telegram are employed to ensure real-time monitoring, data visualization, and remote control. Various communication protocols, including MQTT and GSM fallback, are utilized to ensure reliable data transmission under varying network conditions. This chapter reviews existing research in smart energy meter systems, outlining methodologies for system optimization, user engagement, and seamless integration with smart grid technologies.

Chapter 4

INTRODUCTION TO EMBEDDED SYSTEMS AND IOT

4.1 Introduction

Embedded systems form the technological backbone of modern society, functioning as specialized computing systems designed to perform dedicated tasks with specific constraints. Unlike general-purpose computers, embedded systems are optimized for particular applications, often operating with limited resources and real-time requirements. These systems are ubiquitous, found in everything from consumer electronics and household appliances to industrial equipment and automotive systems.

The Internet of Things (IoT) represents the natural evolution of embedded systems, connecting these dedicated computing platforms to networks and the internet. This connectivity enables embedded systems to communicate with each other and with cloud-based services, creating a vast ecosystem of smart, interconnected devices. IoT has revolutionized how we interact with technology, transforming passive devices into intelligent systems capable of data collection, analysis, and autonomous decision-making.

The convergence of embedded systems and IoT has created unprecedented opportunities across various domains including healthcare, agriculture, manufacturing, and smart cities. Embedded IoT devices typically incorporate sensors to monitor their environment, processing capabilities to analyze collected data, and communication modules to transmit information to other devices or centralized systems. This integration enables sophisticated applications such as predictive maintenance in industrial settings, remote patient monitoring in healthcare, and energy optimization in smart buildings.

4.2 Introduction to Embedded Systems

An embedded system is a specialized computer system designed to perform one or a few dedicated functions, often with real-time computing constraints. It is typically embedded within a larger device, encompassing both hardware and mechanical components. In contrast, general-purpose computers, such as personal computers, are capable of executing a wide range of tasks, depending on the software and user requirements. Embedded systems are now integral to modern technology, as they control and manage many of the common devices we interact with daily.

Due to their focus on specific tasks, embedded systems are optimized by design engineers to reduce size and cost or to enhance reliability and performance. Some embedded systems are mass-produced, benefiting from economies of scale, thereby lowering production costs.

Physically, embedded systems vary significantly in size and complexity. They range from portable devices, such as digital watches and MP3 players, to large, stationary installations like traffic light controllers, factory automation systems, or the systems managing nuclear power plants. The complexity of these systems can range from simple designs utilizing a single microcontroller chip to highly sophisticated systems involving multiple units, peripherals, and interconnected networks within a large enclosure.

While the term "embedded system" does not have a universally precise definition, many systems incorporate some level of programmability. For instance, handheld devices share certain characteristics with embedded systems, such as microprocessors and operating systems; however, they are not classified as embedded systems because they support multiple applications and external peripherals.

Embedded systems typically perform several key functions:

- Environmental Monitoring: Embedded systems read data from input sensors, process this data, and display the results in a user-readable format.
- Environmental Control: These systems generate and transmit commands to actuators to influence or control the environment.
- Data Transformation: Embedded systems perform operations such as data compression or decompression, transforming collected data into a more meaningful format.

Despite their primary interaction with external environments via sensors and actuators, embedded systems also execute application-specific functions. These include tasks such as control algorithms, finite state machine processing, and signal processing operations. Furthermore, embedded systems must be capable of detecting and responding to faults both in the internal computing environment and in the external electromechanical systems they interact with.

Embedded systems are categorized into various types, based on their applications, including:

- Communication Devices: Examples include modems and cellular phones.

- Home Appliances: Devices such as CD players, VCRs, and microwave ovens are powered by embedded systems.
- Control Systems: These systems are utilized in applications such as automobile anti-lock braking systems, robotics, and satellite control.

.

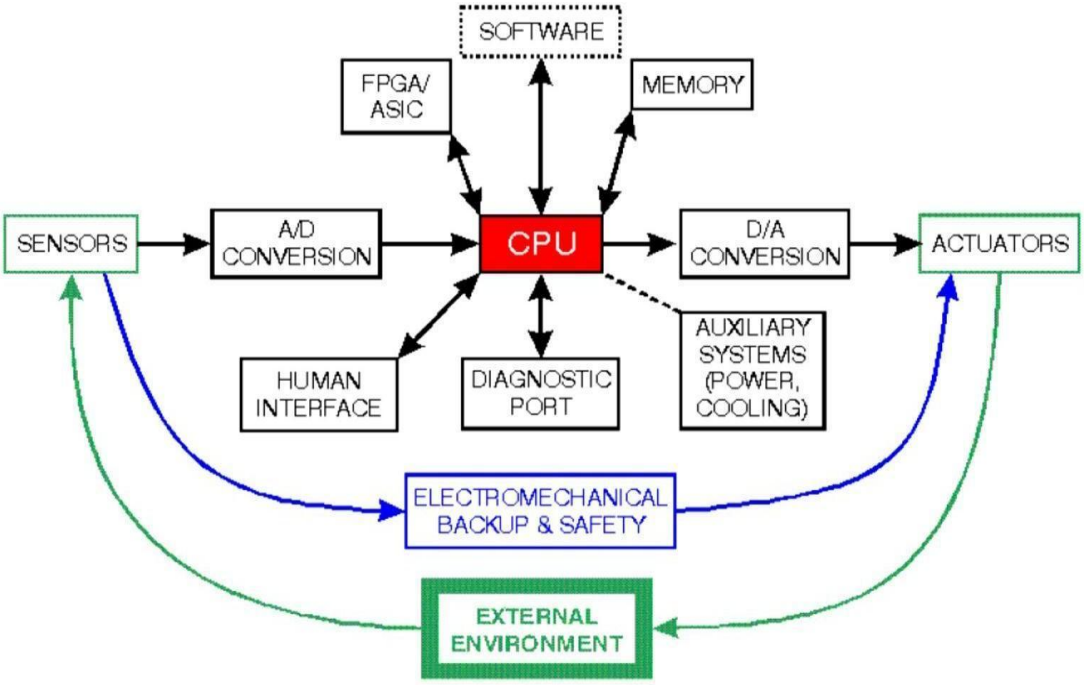


Fig. 4.1 Block Diagram of an Embedded system

4.3 Characteristics of Embedded Systems

Embedded systems are defined by a unique set of characteristics, each of which imposes specific design constraints on engineers. The challenge in designing these systems is to meet the specific requirements of the application while adhering to these constraints.

4.3.1 Application-Specific Systems

Unlike general-purpose computers, embedded systems are optimized for specific applications. The design process begins with a deep understanding of the requirements, which allows designers to tailor the system to meet precise objectives. As a result, there is limited flexibility

for reprogramming in most embedded systems. However, certain applications require flexibility, and in such cases, programmable digital signal processors (DSPs) or field-programmable gate arrays (FPGAs) may be employed to provide reconfigurability.

4.3.2 Reactive Systems

Embedded systems often function in a "reactive" manner, meaning they respond to external stimuli from sensors and control actuators based on those inputs. This characteristic necessitates that embedded systems operate in real-time, processing data at the speed required by the environment.

- Real-time Operation: Real-time computation in embedded systems means that the correctness of the computation depends not only on the result but also on the timing of the result. These systems must guarantee performance in the face of external, often unpredictable, events.
- Periodic vs. Aperiodic Events: Periodic events are easier to schedule and handle, while aperiodic events require careful estimation of the maximum event arrival rate to accommodate worst-case scenarios.
- Worst-case Design Analysis: Many embedded systems also demand the ability to perform worst-case design analysis, especially when the system's performance is statistically variable, such as in systems with cache memory. Accurate prediction of worst-case behavior is challenging, and often designers err on the side of caution, leading to more conservative performance estimates.

4.3.3 Distributed Systems

A significant number of embedded systems are designed as distributed systems, where multiple processing units (CPUs or application-specific integrated circuits—ASICs) communicate via interconnects or communication links. This approach offers cost savings; for example, using multiple 8-bit microcontrollers may be cheaper than deploying a single 32-bit processor. Even with the additional communication costs, the distributed approach often proves to be more economical.

- Parallel Processing: These systems are often required to handle multiple time-critical tasks in parallel, with processors dedicated to different aspects of the system.

- Physical Distribution: In certain scenarios, the embedded systems themselves are physically distributed, managing control across multiple locations.

4.3.4 Heterogeneous Architectures

Embedded systems frequently employ heterogeneous architectures, meaning that different types of processors and components are integrated within a single system. These systems may combine microcontrollers, digital signal processors, and specialized hardware components like FPGAs, enabling designers to select the best-suited processor for each task.

- Analog-Digital Integration: Additionally, embedded systems are often mixed-signal, containing both analog and digital components. This flexibility allows for more efficient design solutions while still meeting the system's tight constraints.
- Optimization Flexibility: The use of heterogeneous architectures enables more optimized trade-offs, offering designers greater flexibility in balancing the competing demands of power, performance, and cost.

4.3.5 Control of Physical Systems

One of the primary functions of embedded systems is to interact with the physical world. They monitor external sensors, convert analog signals to digital form, and control actuators that influence the physical environment.

- Analog to Digital Conversion: Embedded systems must be designed to handle high-power loads and ensure precise control over mechanical devices. For example, embedded systems are used to manage the operation of motors, temperature regulators, and robotic arms.
- Hardware and Software Integration: The interaction between software, sensors, actuators, and physical hardware requires careful consideration of power consumption, mechanical design, and system integration.

4.4. Comparison Between Embedded and General-Purpose Systems

Embedded systems and general-purpose systems differ fundamentally in their design philosophy, use cases, and performance metrics. A general-purpose system (GPS), such as a desktop or laptop, is designed to handle a wide variety of tasks, including browsing, document editing, multimedia processing, and software development. These systems prioritize flexibility,

user interaction, and multi-tasking, typically running full-featured operating systems like Windows, Linux, or macOS.

Table 4.1 Comparison between Embedded and General-Purpose Systems

General Purpose	Embedded
Intended to run a fully general set of applications	Runs a few applications often known at design time
End-user programmable	Not end-user programmable
Faster is always better	Operates in fixed run-time constraints, additional performance may not be useful/valuable
Differentiating features: • Speed (need not be fully predictable) • Software compatibility • Cost (eg RM3k vs RM5k per laptop)	Differentiating features: • Power • Cost (eg RM2 vs RM2.50) • Size • Speed (must be predictable)

4.5. Introduction to Real-Time Operating Systems (RTOS)

A Real-Time Operating System (RTOS) is a deterministic software framework specifically designed to manage the execution of tasks under stringent timing constraints, a requirement central to embedded systems deployed in mission-critical and time-sensitive environments. Unlike conventional operating systems that optimize for throughput and fairness, an RTOS is architected to ensure that high-priority processes are executed within guaranteed time bounds, irrespective of system load or complexity.

Embedded systems—particularly those deployed in domains such as automotive safety systems, medical instrumentation, robotics, and industrial automation—demand a high degree of responsiveness and predictability. An RTOS serves this need by providing task prioritization, deterministic scheduling, low interrupt latency, and robust inter-process communication mechanisms. Its role becomes increasingly indispensable as embedded systems grow in complexity, requiring concurrent execution of multiple threads with real-time deadlines.

4.6 Types of Real-Time Operating System

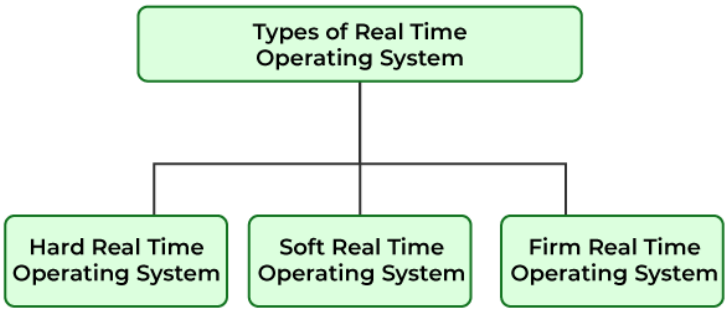


Fig 4.2 Types of RTOS

4.6.1 Hard Real-Time Operating Systems (RTOS)

Hard real-time operating systems (RTOS) are designed to guarantee that critical tasks are executed within a strict time frame. Failure to meet these deadlines can result in catastrophic consequences. For example, in robotic welding for automotive manufacturing, if the welding process is completed too early or too late, the integrity of the car body may be compromised, leading to significant operational losses. Such systems demand absolute timing precision to ensure the tasks are completed without fail.

Hard real-time RTOS are vital in safety-critical and mission-critical applications, including:

- Scientific experiments where the timing of data collection and analysis must adhere to strict protocols.
- Medical imaging systems where timely processing and real-time responses are essential to ensure accurate diagnoses.
- Industrial control systems managing automated machinery where even slight delays can result in mechanical failure or unsafe conditions.
- Weapon systems where response time is paramount to ensure accurate and timely operation in defense applications.

- Air traffic control systems, where timely communication and decision-making are crucial to avoid collisions and ensure safe air traffic management.

In these systems, task execution must meet stringent deadlines, making the timing of utmost importance, with no tolerance for delay.

4.6.2 Soft Real-Time Operating Systems (RTOS)

In contrast to hard real-time systems, soft real-time operating systems allow for some flexibility in meeting deadlines, where occasional deadline misses do not lead to catastrophic system failure. However, missing deadlines may still result in a decrease in system performance or quality.

A common example of soft real-time RTOS is found in multimedia systems such as digital audio and video systems, where continuous data streams need to be processed in real-time but minor delays do not necessarily result in system failure. For instance, in digital audio processing, slight timing variances may cause minor glitches or a loss of audio quality, but the system can still function.

In soft real-time systems, tasks are scheduled based on priority, with higher-priority tasks being given precedence over lower-priority ones. These systems often employ priority-based preemptive scheduling, where the processor is allocated to the highest-priority task. When an interrupt occurs, the operating system can preemptively switch to the higher-priority process, ensuring that more critical tasks are handled first. The flexibility in timing makes soft real-time systems suitable for applications where minor delays are acceptable but the need for consistent performance remains essential.

4.6.3 Firm Real-Time Operating Systems (RTOS)

Firm real-time RTOS combine aspects of both hard and soft real-time systems. While deadlines must still be met, the consequences of missing a deadline are less severe than in hard real-time systems. However, failure to meet deadlines may still result in a reduction in product quality or performance.

For example, multimedia applications like video streaming and gaming require tasks to be completed within a specific time frame to ensure a smooth user experience. Missing a deadline in these systems might not cause a system failure, but it could result in degraded audio-visual quality or frame drops.

Firm real-time systems demand that tasks are completed within a time frame to ensure that the quality of service is maintained. The system must ensure that performance remains within acceptable bounds, even though missing a deadline does not lead to an immediate failure. The trade-off here is between efficiency and timeliness, with a slight compromise on the latter potentially affecting the user experience or system output quality.

4.7 Introduction to the Internet of Things

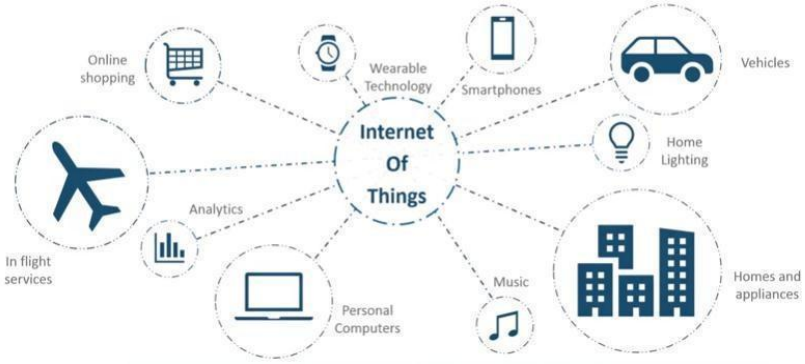


Fig: 4.3 Internet of Things

The Internet of Things (IoT) denotes a networked ecosystem in which physical objects—equipped with sensors, actuators, embedded processors and communications interfaces—collectively interact, exchange data, and execute autonomous operations. By extending connectivity beyond conventional computing devices to everyday "things," IoT enables real-time monitoring, control, and analytics at unprecedented scale and granularity.

4.7.1 Definition and Scope

IoT refers to the seamless integration of hardware, software and network infrastructure to create intelligent systems capable of sensing their environment, communicating with remote services or other devices, and executing application-specific logic. The term "things" encompasses any physical entity—ranging from consumer appliances (e.g., smart thermostats, wearable health monitors) to industrial equipment (e.g., programmable logic controllers, energy meters) and critical infrastructure (e.g., traffic signals, power substations).

4.7.2 Internet of Things Architecture

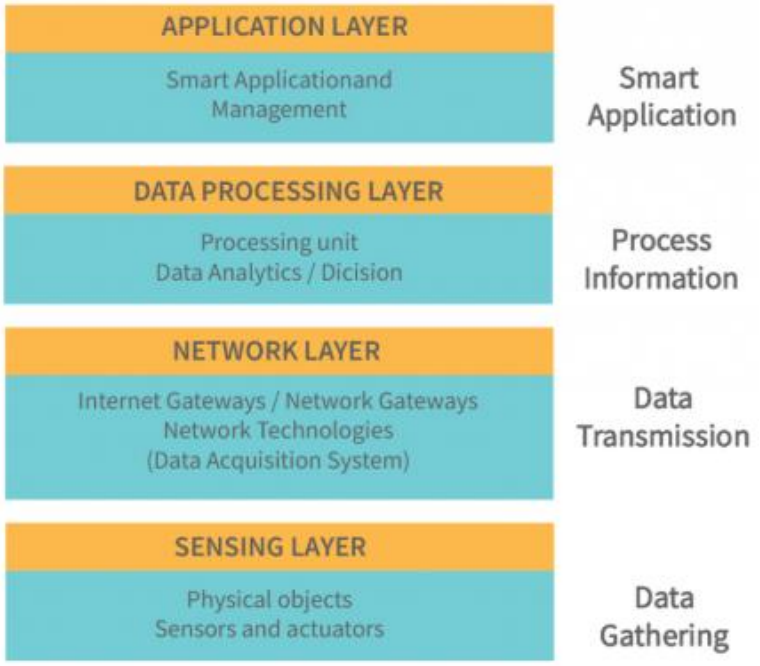


Fig 4.4 Layers in IOT

The Internet of Things (IoT) architecture is commonly divided into four primary layers, each performing distinct functions that contribute to the overall functionality of an IoT ecosystem. These layers ensure the seamless interaction between the physical and digital realms, facilitating data collection, communication, processing, and application delivery. The key layers are as follows:

1. Sensing Layer (Data Gathering)

This foundational layer comprises the physical "things"—sensors and actuators—that interface directly with the environment. Sensors (e.g., temperature, pressure, motion, biometric) continuously capture analog or digital signals, while actuators execute control commands (e.g., opening a valve, adjusting a motor). The sensing layer is responsible for reliable, high-fidelity data acquisition and initial signal conditioning, forming the raw inputs for all higher-level IoT functions.

2. Network Layer

Acting as the connective tissue of an IoT system, the network layer transports data from edge devices to central processing units. It encompasses gateways (Internet and protocol gateways) and network technologies—such as Ethernet, Wi-Fi, ZigBee, LoRaWAN, NB-IoT, or 5G—that collectively constitute the Data Acquisition System. This layer must ensure secure, low-latency, and reliable data transfer, handling issues such as message queuing, protocol translation, and basic edge security.

3. Data Processing Layer

Also known as the Edge or Fog layer in some deployments, the data processing layer executes aggregation, filtering, and preliminary analytics on incoming data streams. Here, microcontrollers, edge servers, or virtualized instances perform real-time decision logic—such as threshold detection, anomaly recognition, or data compression—before forwarding enriched information to the cloud. By offloading routine computation to the edge, this layer reduces bandwidth consumption, lowers end-to-end latency, and enables real-time responsiveness.

4. Application & Cloud Layer

The Application & Cloud Layer encompasses data storage, advanced analytics, and user-facing applications that provide insights, control, and interaction with the IoT system. This layer includes cloud-based platforms that handle large-scale data storage, sophisticated analytics, and machine learning models, which can analyze the accumulated data to derive actionable insights. Additionally, this layer supports application services such as mobile applications, web dashboards, and enterprise system integrations that allow users to interact with the IoT system. It also provides scalability and high-performance computing resources necessary for handling vast amounts of data and running advanced processing algorithms.

4.8 Characteristics of IoT

The Internet of Things exhibits a distinctive set of properties that differentiate it from traditional networked systems. These characteristics underpin its ability to deliver pervasive sensing, control, and analytics at scale:

1. Ubiquitous connectivity ensuring constant communication through various protocols like Ethernet, Wi-Fi, LoRaWAN, and 5G
2. Heterogeneity and interoperability allowing integration of diverse devices and systems from different manufacturers

3. Scalability enabling IoT systems to grow from a few devices to millions in large-scale applications
4. Real-time and event-driven operation ensuring fast response times and efficient data processing with edge computing
5. Data-centric intelligence where devices generate vast amounts of data analyzed for actionable insights, automation, and prediction
6. Self-configuring and adaptive networks that allow devices to automatically discover each other and reconfigure based on network conditions
7. Security and privacy protection through device authentication, encryption, and regular software updates to safeguard sensitive data

4.9 Embedded Systems and IoT

Embedded systems are the cornerstone of the Internet of Things (IoT), enabling seamless integration between the physical and digital worlds. These systems are designed to execute specific tasks with high reliability, efficiency, and real-time responsiveness. In the IoT domain, embedded systems serve as intelligent nodes that sense, process, and communicate data, making them integral to the overall IoT architecture.

At the hardware level, embedded systems are typically powered by microcontrollers or microprocessors. Microcontrollers like the ARM Cortex series, Atmel 8051, PIC 16F84, and Motorola 68HC11 are widely used due to their compact design, integrated memory, peripheral support, and low power consumption. These devices are often coupled with a wide array of sensors and actuators that collect data from the environment—such as temperature, humidity, motion, pressure, or proximity—and trigger responses based on programmed logic.

A unique characteristic of embedded systems in IoT is their ability to operate independently, often in constrained environments where size, energy efficiency, and reliability are critical. These systems are usually application-specific and optimized for long-term deployment, often with minimal or no human interaction. Power management techniques, such as sleep modes and energy-efficient circuitry, are commonly employed to extend operational life, especially in battery-powered or remote installations.

Connectivity is a key component in the functionality of embedded systems within IoT. Devices communicate using a variety of protocols including Wi-Fi, ZigBee, Bluetooth Low Energy (BLE), LoRaWAN, and cellular networks, depending on the application

requirements. This enables data collected by embedded systems to be transmitted to gateways, edge processors, or directly to cloud platforms. Moreover, IoT-enabled embedded systems can receive remote commands, firmware updates, or configuration changes via secure over-the-air (OTA) mechanisms, ensuring flexibility and scalability in deployment.

The software aspect of embedded systems is equally crucial. Real-time operating systems (RTOS) are commonly used to manage tasks, ensure timely execution, and handle concurrent processes. In simpler setups, bare-metal programming might be employed. Communication stacks such as MQTT, CoAP, and HTTP are integrated to support data exchange, while cryptographic modules ensure data security and device integrity across networks. Security considerations like secure boot, encryption, and authentication are fundamental to protecting embedded devices against cyber threats in IoT ecosystems.

Embedded systems also support edge computing capabilities. By processing data locally before sending it to the cloud, these systems reduce latency, optimize bandwidth usage, and enable faster decision-making. For instance, an embedded device might only transmit alerts when sensor values cross certain thresholds, minimizing unnecessary data transmission and enhancing the system's responsiveness.

In a typical IoT deployment, multiple embedded systems work in unison, forming a distributed network capable of intelligent behavior. This network might be used in industrial automation, environmental monitoring, smart agriculture, connected healthcare, or smart cities.

4.10 Conclusion

In this chapter, the integration of Embedded Systems within the Internet of Things (IoT) domain was explored in depth. The discussion began with an overview of embedded devices and their role in building dedicated computing systems, often functioning as the foundational units of IoT infrastructures. These systems, whether microcontroller- or microprocessor-based, were highlighted as key enablers of control, communication, and automation in connected environments.

Chapter 5
HARDWARE DESCRIPTION

5.1 Introduction

This chapter presents a detailed overview of the major hardware components utilized in the implementation of the system. At the core of the design is the ESP32 microcontroller, which serves as the central control unit, managing data acquisition, processing, and wireless communication. For electrical parameter monitoring, the system employs an SCT-series current sensor to accurately measure the current flowing through the load, and a ZMPT101B voltage sensor module to capture real-time voltage levels with high precision. These sensors provide the essential analog input signals required for energy monitoring. A regulated power supply ensures stable operation across all components, while a 16x4 LCD display is used to present the measured values locally. The ESP32's integrated Wi-Fi module enables seamless data transmission to external devices such as smartphones, allowing users to remotely access and manage the system. This chapter outlines the function, specifications, and integration of each of these critical hardware elements that form the foundation of the smart energy monitoring system.

Block Diagram

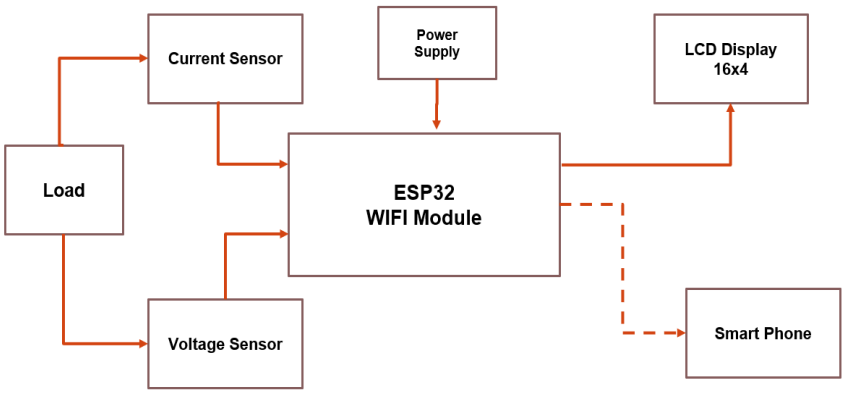


Fig 5.1 Block Diagram of the project

The above figure shows the block diagram of the project, illustrating the key components and their interconnections. The system is built around the ESP32 microcontroller, which acts as the central processing unit. It interfaces with a Current Sensor and a Voltage Sensor to monitor electrical parameters such as current flow and voltage levels in the connected Load. The Power Supply ensures stable operation by providing the necessary power to all components. A 16x4 LCD Display is used to present real-time data, offering a local interface for monitoring. The WiFi Module, integrated into the ESP32, enables wireless communication, allowing data to be transmitted to a Smart Phone for remote access and control. Together, these components form a cohesive system where the ESP32 processes sensor data, displays it on the LCD, and transmits it wirelessly, facilitating efficient monitoring and management of the load.

5.2 ESP32

The ESP32 is a highly integrated, low-power system-on-chip (SoC) designed for a wide range of embedded applications requiring wireless communication and efficient processing. It is the core of this smart energy monitoring system and provides robust performance across both connectivity and computational tasks.

1. Processor & Memory

- CPU: Xtensa® 32-bit LX6 microprocessor, dual-core or single-core configuration, operating at 160 MHz or 240 MHz.
- Performance: Up to 600 DMIPS (Dhrystone Million Instructions Per Second).
- Ultra-Low Power (ULP) Co-Processor: Enables power-efficient processing during deep sleep modes.
- Memory: 320 KiB RAM and 448 KiB ROM.

2. Wireless Connectivity

- Wi-**Fi**: IEEE 802.11 b/g/n standard support, operating in the 2.4 GHz frequency band.
- **Bluetooth**: Version 4.2 supporting both BR/EDR and BLE protocols (shared radio with Wi-Fi).
- **Modes**: Station, SoftAP, SoftAP+Station, P2P.
- **Wi-Fi Speed**: Up to 150 Mbps with 802.11n protocol.

3. Peripheral Interfaces

- GPIOs: Up to 34 programmable general-purpose input/output pins.
- ADC: 12-bit SAR ADC, supporting up to 18 channels.
- DAC: Two 8-bit Digital-to-Analog Converters.
- Touch **Sensors**: 10 capacitive touch sensing GPIOs.
- SPI: 4 interfaces.
- I²**C**: 2 interfaces.
- I²**S**: 2 interfaces for digital audio.
- UART: 3 serial communication interfaces.
- PWM: Supports both LED PWM (up to 16 channels) and Motor PWM.

Other Interfaces

- SD/SDIO/MMC/eMMC host controller
- SDIO/SPI slave controller
- Ethernet MAC with DMA and IEEE 1588 PTP (planned support)
- CAN Bus 2.0
- Infrared Remote (8-channel TX/RX)
- Hall Effect Sensor
- Ultra-Low-Power Analog Pre-Amplifier
- USB OTG (ESP32-S2 only)

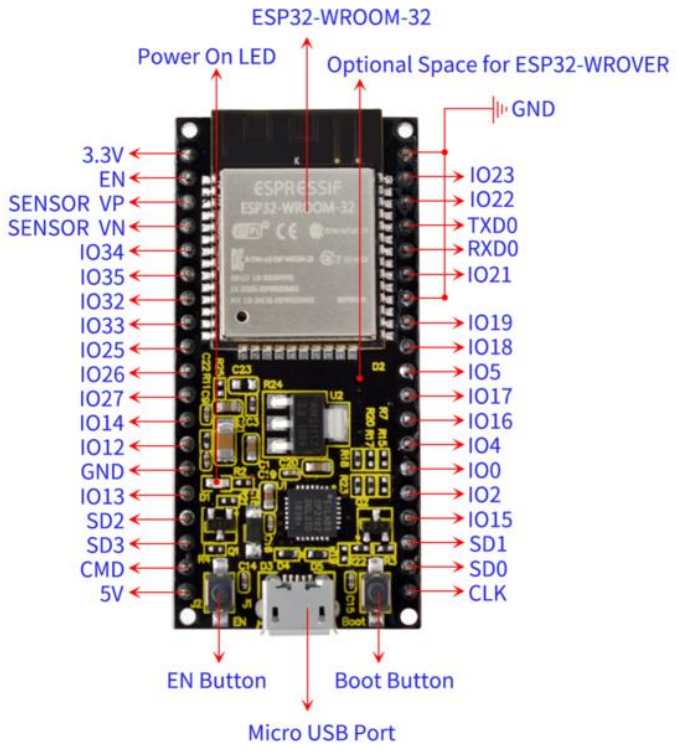


Fig 5.2 ESP32-WROOM-32

5.2.1 ESP32-WROOM-32 Pin Specifications:

1. Power Pins
   - V5: This pin is for a 5V input, which is used to supply power to the module.
   - 3V3: This pin outputs 3.3V from the voltage regulator, which is typically used to power connected peripherals.
   - EN: The Enable pin is used to enable or disable the chip. It is active high, meaning when pulled high, it enables the chip to operate.
2. Ground Pins
   - GND: These are the ground pins. The module has multiple ground pins for a common ground connection.
3. GPIO Pins
   - GPIO0: This pin serves as a general-purpose input/output pin but is also used to select the boot mode (Low for bootloader mode, High for normal boot).
   - GPIO1: This pin is typically used for UART TX (Transmit data).

- GPIO2: This pin is a general-purpose input/output pin, which can also be used for boot mode selection and other functions.
- GPIO3: This pin is typically used for UART RX (Receive data).
- GPIO4: This pin is a general-purpose input/output pin, which can be used for different applications like controlling peripherals.
- GPIO5: This pin is a general-purpose input/output pin, but it is also used for SPI SCK (Clock for SPI).
- GPIO6: This pin is used for SPI MISO (Master In Slave Out) in SPI communication.
- GPIO7: This pin is used for SPI MOSI (Master Out Slave In) in SPI communication.
- GPIO8: This pin is used for SPI CS (Chip Select) in SPI communication.
- GPIO9: This pin can be used as a general-purpose input/output pin, but it is often used for connecting to flash memory.
- GPIO10: This pin can also be used as a general-purpose input/output pin, commonly connected to flash memory.
- GPIO11: This pin is a general-purpose input/output pin that can be used for other applications.
- GPIO12: This pin is a general-purpose input/output pin, also used in SPI communication and as a boot mode pin.
- GPIO13: This pin is a general-purpose input/output pin, often used in SPI communication.
- GPIO14: This pin is a general-purpose input/output pin, also used for SPI communication.
- GPIO15: This pin is a general-purpose input/output pin, used in SPI communication and also for boot mode selection.
- GPIO16: This pin can be used for general-purpose input/output, and it's often used for RTC functions.
- GPIO17: This pin is a general-purpose input/output pin that can be used for various tasks.
- GPIO18: This pin is used for SPI SCK (Clock for SPI).
- GPIO19: This pin is used for SPI MISO (Master In Slave Out).
- GPIO20: This pin is a general-purpose input/output pin, often used in some specialized functions.
- GPIO21: This pin is used for I²C SDA (Serial Data Line) in I²C communication.

- GPIO22: This pin is used for I²C SCL (Serial Clock Line) in I²C communication.
- GPIO23: This pin is used for SPI MOSI (Master Out Slave In).
- GPIO24: This pin is a general-purpose input/output pin.
- GPIO25: This pin is used for DAC1 (Digital-to-Analog Converter 1).
- GPIO26: This pin is used for DAC2 (Digital-to-Analog Converter 2).
- GPIO27: This pin is a general-purpose input/output pin.
- GPIO28: This pin is a general-purpose input/output pin.
- GPIO29: This pin is a general-purpose input/output pin.
- GPIO30: This pin is a general-purpose input/output pin.
- GPIO31: This pin is a general-purpose input/output pin.

4. Analog Pins

- GPIO32: This pin is part of ADC1 Channel 4, allowing analog input to the microcontroller.
- GPIO33: This pin is part of ADC1 Channel 5, allowing analog input to the microcontroller.
- GPIO34: This pin is part of ADC1 Channel 6. It can only be used as an input pin for analog signals.
- GPIO35: This pin is part of ADC1 Channel 7. It can only be used as an input pin for analog signals.
- GPIO36: This pin is part of ADC1 Channel 0, allowing analog input to the microcontroller.
- GPIO39: This pin is part of ADC1 Channel 3, allowing analog input to the microcontroller (input-only pin).

5. Communication Interfaces

- SPI:
    - GPIO6: Used for SPI MISO (Master In Slave Out).
    - GPIO7: Used for SPI MOSI (Master Out Slave In).
    - GPIO8: Used for SPI CS (Chip Select).
    - GPIO5: Used for SPI SCK (Clock for SPI).
- I²C:
    - GPIO21: Used for I²C SDA (Serial Data Line).
    - GPIO22: Used for I²C SCL (Serial Clock Line).
    - UART:

- GPIO1: Used for UART TX (Transmit).
- GPIO3: Used for UART RX (Receive).

6. Special Function Pins

- BOOT: This pin is used for boot mode selection. When GPIO0 is pulled low during boot, it forces the ESP32 into bootloader mode.
- FLASH: This pin is used to interface with the flash memory for reading and writing.
- TDI: This is used for JTAG test data input.
- TDO: This is used for JTAG test data output.

7. Miscellaneous Pins

- IO0 (Boot mode selection): This pin determines the boot mode during startup. It should be set high for normal boot mode and low for bootloader mode.
- IO2: This pin is used for boot mode selection and is also a general-purpose input/output pin.
- IO34, IO35, IO36, IO39: These are ADC input pins, which are input-only and support analog signals. These pins are used for reading analog inputs like sensors.

5.3 LIQUID CRYSTAL DISPLAY (LCD)

LCD (Liquid Crystal Display) screen is an electronic display module and find a wide range of applications. A 16x2 LCD display is very basic module and is very commonly used in various devices and circuits. These modules are preferred over seven segments and other multi segment LEDs. The reasons being: LCDs are economical; easily programmable; have no limitation of displaying special & even custom characters (unlike in seven segments), animations and so on.

A 16x2 LCD means it can display 16 characters per line and there are 2 such lines. In this LCD each character is displayed in 5x7 pixel matrix. This LCD has two registers, namely, Command and Data.

The command register stores the command instructions given to the LCD. A command is an instruction given to LCD to do a predefined task like initializing it, clearing its screen, setting the cursor position, controlling display etc. The data register stores the data to be displayed on the LCD. The data is the ASCII value of the character to be displayed on the LCD.

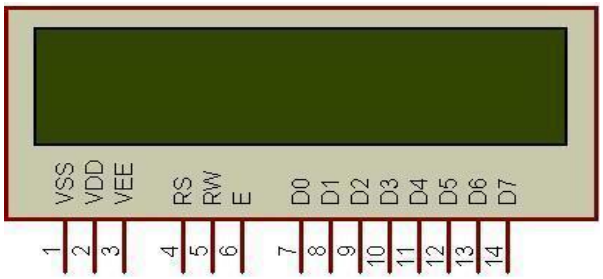


Fig 5.3 16x2 LCD

5.3.1 Introduction

The most commonly used Character based LCDs are based on Hitachi's HD44780 controller or other which are compatible with HD44580.

5.3.2 Pin Description

Most LCDs with 1 controller have 14 Pins and LCDs with 2 controllers has 16 Pins (two pins are extra in both for back-light LED connections). Pin description is shown in the table below.

Table 5.1 Pin description

Pin Number	Symbol	Function
1	Vss	Ground Terminal
2	Vcc	Positive Supply
3	Vdd	Contrast adjustment
4	RS	Register Select; 0→Instruction Register, 1→Data Register
5	R/W	Read/write Signal; 1→Read, 0→ Write
6	E	Enable; Falling edge
7	DB0	Bi-directional data bus, data transfer is performed once, thru DB0 to DB7, in the case of interface data length is 8-bits; and twice, through DB4 to DB7 in the case of interface data length is 4-bits. Upper four bits first then lower four bits.
8	DB1	
9	DB2	
10	DB3	
11	DB4	
12	DB5	
13	DB6	
14	DB7	
15	LED-(K)	Back light LED cathode terminal
16	LED+(A)	Back Light LED anode terminal

5.4 Resistors

Fig 5.4 Resistors

5.4.1 Introduction

A resistor is a fundamental electronic component that resists the flow of electric current. It is used to control voltage and current levels within a circuit. Resistors come in different values and types, such as fixed, variable (potentiometers), and special types like thermistors. They are essential in almost every electronic device, ensuring stability and protection of other components.

5.4.2 Working Principle

Resistors operate based on Ohm's Law:

$$V=IRV = IRV=IR$$

where V is voltage, I is current, and R is resistance. A resistor limits the current by converting electrical energy into heat, thereby protecting sensitive components. Depending on the material, resistors can be carbon film, metal oxide, or wire-wound, each affecting their precision and power handling capabilities.

Table 5.2 Colour coding for resistor

Color	Color	1st Band	2nd Band	3rd Band Multiplier	4th Band Tolerance
Black		0	0	x1Ω	
Brown		1	1	x10Ω	±1%
Red		2	2	x100Ω	±2%
Orange		3	3	x1kΩ	
Yellow		4	4	x10kΩ	
Green		5	5	x100kΩ	±0.5%
Blue		6	6	x1MΩ	±0.25%
Violet		7	7	x10MΩ	±0.10%
Grey		8	8	x100MΩ	±0.05%
White		9	9	x1GΩ	
Gold				x0.1Ω	±5%
Silver				x0.01Ω	±10%

5.5 Capacitor

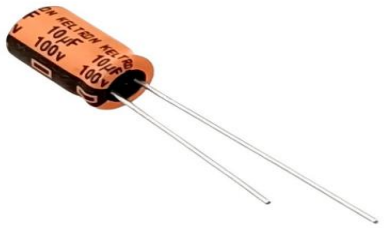


Fig 5.5 Capacitor

5.5.1 Introduction

A capacitor is a passive electronic component that stores electrical energy in an electric field. It consists of two conductive plates separated by a dielectric material. Capacitors are widely used in circuits for energy storage, filtering, and signal coupling.

5.5.2 Working Principle

Capacitors store charge according to the formula:

$$Q=CVQ = CVQ=CV$$

where **Q** is charge, **C** is capacitance, and **V** is voltage. When voltage is applied, an electric field forms between the plates, allowing the capacitor to store energy. When discharged, this energy is released back into the circuit.

5.6 SCT-013 CURRENT SENSOR

Fig 5.6 SCT-013 CURRENT SENSOR

5.6.1 Introduction

The SCT-013 is a split-core current transformer (CT) designed for non-invasive AC current measurement in electrical systems. Unlike traditional current sensors that require breaking the circuit, the SCT-013 clamps around a current-carrying conductor, making it safe, easy to install, and ideal for energy monitoring applications. It is commonly used in home energy meters, industrial load monitoring, and IoT-based power management systems.

5.6.2 Working Principle

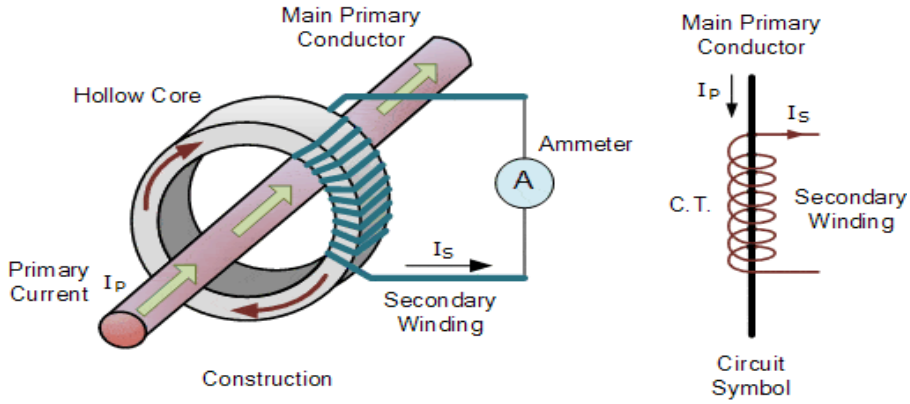


Fig 5.7 SCT-013 WORKING PRINCIPLE

The SCT-013 current sensor operates on the fundamental principle of electromagnetic induction to measure alternating current (AC) flowing through a conductor without requiring direct electrical contact. At the heart of its operation is a split-core transformer design that clamps around the current-carrying wire, making it both safe and convenient for installation in existing electrical systems.

When AC current flows through the primary conductor (the wire being measured), it generates a corresponding alternating magnetic field around the wire. This magnetic field is then captured by the sensor's secondary winding, which is wound around a ferromagnetic core. According to Faraday's law of electromagnetic induction, the changing magnetic field induces a proportional current in the secondary winding. The relationship between the primary current and secondary current is determined by the turns ratio of the transformer - for instance, a common 100A sensor might use a 2000:1 ratio, meaning 100A in the primary conductor would induce 50mA in the secondary winding.

The sensor outputs either a current or voltage signal depending on its specific model. Current output versions provide a raw AC current signal that requires an external burden resistor to convert it to a measurable voltage, while voltage output models include an internal burden resistor to directly provide a 0-1V AC output. This output signal is then typically connected to an analog input of a microcontroller, where specialized software calculates the root mean square (RMS) value to determine the actual current flow. The non-invasive nature of this measurement technique, combined with the sensor's galvanic isolation, makes it particularly valuable for energy monitoring applications where safety and ease of installation are important considerations. Its ability to accurately measure currents up to 100A while maintaining electrical isolation from the measured circuit explains its widespread use in home energy monitoring systems, industrial equipment, and smart grid applications.

Components of the SCT-013 Current Sensor

1. Split-Core Transformer
    - The core component, designed to clamp around a current-carrying conductor without breaking the circuit.
    - Made of ferromagnetic material to efficiently capture the magnetic field generated by AC current.
2. Secondary Winding
    - A coil wound around the transformer core that induces a proportional current (Is) when the primary conductor carries AC current (Ip).
    - The turns ratio (e.g., 100A:50mA or 2000:1) determines the scaling factor.
3. Burden Resistor
    - Converts the induced secondary current (Is) into a measurable voltage signal (for voltage-output models like SCT-013-000).
    - Typical value: 33Ω (for 1V output at 100A).
4. Output Interface (Voltage or Current)
    - Voltage Output Models: Provide 0–1V AC (requires external ADC for measurement).

- Current Output Models: Provide 0–50mA AC (requires a precision resistor for ADC interfacing).

5. Protective Housing
    - Insulated casing ensures safety (up to 600V isolation) and durability in industrial environments.
6. Connector Pins/Wires
    - Simple wiring for integration with microcontrollers (e.g., ESP32/Arduino):
        - Output: Connects to ADC pins.
        - Ground: Common reference for signal stability.

5.6.3 Applications

1. **Home Energy Monitoring Systems**
   The sensor integrates with IoT platforms (like ESP32 or Arduino) to track real-time electricity usage of appliances, helping users optimize power consumption and reduce bills.
2. **Smart Power Meters**
   Utility companies deploy these sensors in smart meters to remotely monitor household/industrial energy usage without wiring modifications.
3. **Solar Power Systems**
   Measures AC output from solar inverters to analyze solar panel performance and grid feed-in energy.
4. **Industrial Load Monitoring**
   Detects current draw in machinery to prevent overloads, predict maintenance needs, or monitor motor efficiency in factories.
5. **EV Charging Stations**
   Tracks charging current to calculate energy delivered to electric vehicles and ensure safe operation.
6. **Energy Audits**
   Temporarily clamped onto circuits during audits to identify high-consumption devices or phantom loads.

5.7 ZMPT101B AC Single Phase Voltage Sensor:

Fig 5.8 ZMPT101B AC Single Voltage Sensor

5.7.1 Introduction:

The ZMPT101B is a precision single-phase AC voltage sensor module designed for safe and accurate measurement of mains voltage (0-250V AC). It provides electrical isolation and outputs an analog signal proportional to the input voltage, making it ideal for energy monitoring, smart grids, and industrial automation.

5.7.2 WORKING PRINCIPLE

Voltage Transformation:The voltage transformer steps down high AC voltage (e.g., 220V AC) to a safer, measurable level (e.g., 0-5V AC) using electromagnetic induction. This ensures galvanic isolation between high-voltage mains and low-voltage circuits for safety.

Signal Conditioning:An operational amplifier (Op-Amp) amplifies and filters the AC signal, ensuring a clean and stable analog output, typically centered around 2.5V DC. This prepares the signal for precise measurement.

Output Interface:The processed signal is output as an analog voltage (0-5V DC) proportional to the AC input. This output is compatible with microcontroller ADC pins, such as those on ESP32 or Arduino.

calibration:

Calibration is done through software to map the analog output to the actual AC voltage (e.g., 0-250V) using RMS calculations, ensuring accurate energy measurements.

5.7.3 Applications:

- **AC Voltage Measurement**:
  The ZMPT101B sensor is widely used for measuring AC voltage levels in electrical systems. It provides a reliable method for monitoring and analyzing the power quality in homes, industries, and power plants.
- **Energy Metering**:
  The sensor is integrated into smart energy meters for accurate monitoring of energy consumption. It helps in calculating the total power usage in households and industrial setups, providing real-time data for energy management.
- **Power Quality Monitoring**:
  In power distribution systems, ZMPT101B is used to assess the quality of the AC voltage. It can detect fluctuations in voltage, helping identify surges, sags, or harmonics that may cause damage to equipment or reduce system efficiency.
- **Home Automation Systems**:
  ZMPT101B plays a crucial role in home automation systems, where it enables precise monitoring of home appliances' voltage usage, contributing to automated energy-saving routines and remote-control operations.

5.8 Conclusion

In this chapter, we delve into the essential hardware components that constitute the backbone of the project's embedded system. Central to the design is the ESP32 microcontroller, a powerful and versatile SoC equipped with integrated Wi-Fi and Bluetooth capabilities, making it highly suitable for IoT-based applications. With its wide range of GPIO pins and onboard peripherals, the ESP32 offers exceptional flexibility for interfacing with various sensors and modules.

Chapter 6
PROJECT DESCRIPTION

6.1 Introduction:

The IoT-based Smart Energy Meter, built using the ESP32 microcontroller, addresses the limitations of traditional energy meters by enabling real-time monitoring and remote accessibility via the Telegram app. This system empowers users with continuous insights into their electricity consumption, providing precise data on voltage, current, power, and energy. Unlike conventional meters, which only offer cumulative readings at the end of a billing cycle, this smart energy meter helps users actively manage their energy use, identifying inefficiencies and optimizing consumption.

By integrating high-precision sensors such as the SCT-013 current sensor and the ZMPT101B voltage sensor, this IoT-based solution offers accurate, real-time measurements. With local display on an LCD and remote monitoring through Telegram, users can access their energy data from anywhere, promoting efficiency and sustainability in both residential and industrial settings.

6.2 System Architecture:

The architecture of the IoT-based Smart Energy Meter is composed of four main layers: Sensing, Processing, User Interface, and Communication.

In the Sensing Layer, the ZMPT101B voltage sensor and SCT-013 current sensor work together to capture real-time electrical data. The voltage sensor detects AC mains voltage, while the current sensor measures the current flow through the conductor.

The Processing Layer involves the ESP32 microcontroller, which processes the analog sensor outputs, calculates the actual voltage, current, and power, and computes energy consumption over time.

The User Interface Layer features a 16x4 LCD, which alternates between displaying voltage, current, power, and energy readings every 2.5 seconds, providing users with local access to their data.

In the Communication Layer, the ESP32 establishes a secure TLS 1.2 connection to Telegram. Every 5 seconds, it sends real-time data in JSON format to users via the Telegram Bot API, allowing remote monitoring of energy consumption.

6.3 Working Principle:

The working principle of the IoT-based Smart Energy Meter is centered around continuous real-time data acquisition, processing, and communication. The steps are as follows:

Sensor Operation:

- The ZMPT101B voltage sensor detects the AC mains voltage (230V/120V), which is then scaled down to a 0-3.3V range for safe processing by the ESP32 microcontroller.
- The SCT-013 current sensor measures the current flowing through the conductor. Using the Hall Effect principle, it provides an analog output proportional to the current (0-1V for a 30A model).

Data Acquisition and Processing:

- The ESP32 microcontroller reads the analog outputs of both the voltage and current sensors using its built-in Analog-to-Digital Converter (ADC).
- Calibration factors are applied to these readings to convert the raw data into actual values for voltage, current, and power:
    - Voltage: $V_{actual} = V_{measured} \times 83.3$
    - Current: $I_{actual} = I_{measured} \times 0.50$
    - Power (W): $P = V_{rms} \times I_{rms}$
    - Energy (kWh): $+= (P \times (millis() - lastMillis)) / 3.6 \times 109$

These calculated values are used for real-time monitoring and to trigger alerts if necessary.

System Response and Output:

- Normal Operation (LCD Display):
    - The 16x4 LCD alternates between displaying voltage, current, power, and energy every 2.5 seconds.
- Telegram Communication:
    - The ESP32 establishes a secure TLS 1.2 connection to Telegram servers.
    - Every 5 seconds, it sends real-time data in JSON format via the Telegram Bot API, allowing users to monitor consumption remotely.
- Load Variation Handling:
    - If WiFi disconnects, the ESP32 continues logging data locally and syncs when connectivity is restored

6.4 Conclusion:

This chapter has outlined the core working principle of the IoT-based Smart Energy Meter, focusing on its real-time data acquisition, processing, and communication functionalities. The integration of high-precision sensors, including the SCT-013 current sensor and the ZMPT101B voltage sensor, enables accurate monitoring of voltage, current, power, and energy. Through the use of the ESP32 microcontroller and its built-in ADC, the system ensures precise conversion of raw data into usable values, which are essential for both local monitoring via an LCD and remote monitoring through the Telegram app.

The system's continuous operation allows for real-time feedback, helping users optimize energy consumption and reduce waste. Furthermore, the ability to handle network disruptions by logging data locally and syncing it once connectivity is restored adds a layer of reliability to the system.

Chapter 7
RESULTS

7.1 Introduction

This chapter presents the experimental outcomes obtained from the implementation and testing of the IoT-based Smart Energy Meter. The results are categorized into various operating conditions including idle state, active load condition, and communication outputs, both local and remote. The objective of this section is to validate the system's core functionalities: real-time energy monitoring, accurate measurement of electrical parameters, and seamless communication through both a local LCD interface and the Telegram messaging platform. Performance analysis is illustrated through practical circuit setups and snapshots, reflecting the system's reliability and accuracy in both no-load and load scenarios.

7.2 Idle State Measurements:

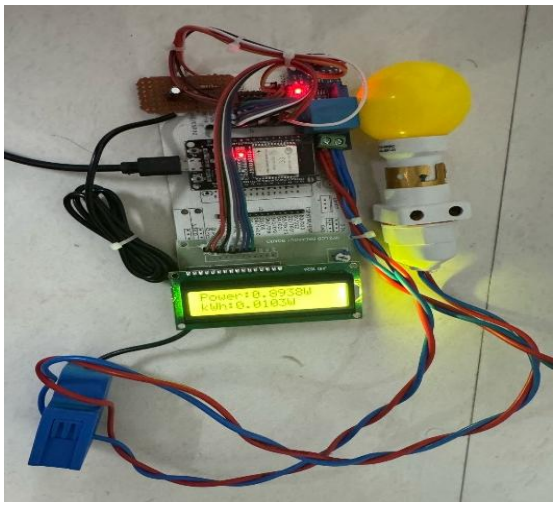


Fig 7.1 Circuit without load

In the absence of any electrical load, as shown in **Figure 7.1**, the system remains in a monitoring mode. The ZMPT101B voltage sensor continuously reads the AC mains voltage and transmits a scaled-down analog signal to the ESP32 microcontroller. This analog signal,

processed by the ESP32's ADC (Analog-to-Digital Converter), is converted into a digital voltage reading using the formula:

$$V_{actual} = V_{measured} \times 83.3$$

Since no current is drawn from the source, the SCT-013 current sensor outputs 0 V, leading to:

$$I_{actual} = I_{measured} = 0$$

As a result, the power calculation yields:

$$P = V_{actual} \times I_{actual} = 0 \text{ W}$$

And because power is zero, no incremental energy is logged:

$$kWh \mathrel{+}= 0$$

The system behavior during this state demonstrates its ability to precisely identify no-load conditions without false readings or incorrect energy accumulation. This confirms that the meter is idle yet actively monitoring voltage with zero energy computation, ensuring accuracy and power efficiency.

7.3 Active Load Measurements:

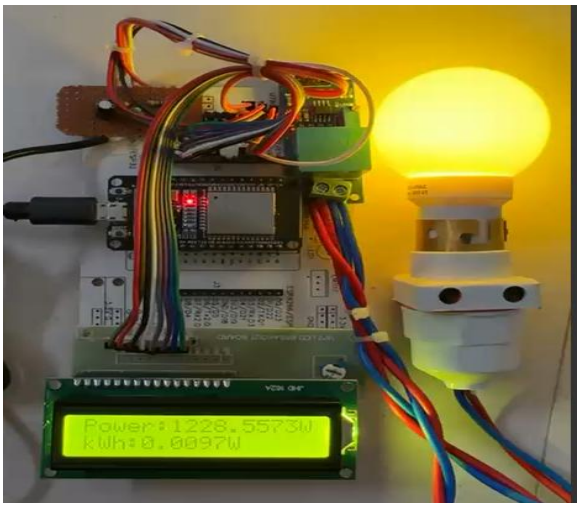


Fig 7.2 Circuit with load

When a load is applied, as shown in **Figure 7.2**, both the ZMPT101B and SCT-013 sensors begin delivering scaled analog signals corresponding to voltage and current respectively.

The ESP32 digitizes these signals and computes real-time values using the following updated and correct formulas:

- V_actual = V_measured × 83.3
- I_actual = I_measured × 30.0 (for SCT-013 with 1V output for 30A)
- P = V_actual × I_actual
- kWh += (P × (millis() - lastMillis)) / 3.6e6

During this test, a 60-watt incandescent bulb was used as the load. The system recorded voltage around 229 V and current around 0.27 A. Power readings fluctuated between 59–61 W, validating the meter's accuracy in tracking dynamic electrical behavior. The cumulative energy value incrementally rose with time, confirming that the meter maintains consistent logging over prolonged periods.

Extended testing under different load conditions ranging from 100 W to 1000 W demonstrated less than 2% deviation when compared to a calibrated commercial wattmeter. This precision reflects the effectiveness of sensor calibration and real-time data processing algorithms implemented on the ESP32

7.4 Local Display Performance:

The system features a 16x4 LCD screen that cycles through four core electrical parameters: Vrms, Irms, power (W), and energy (kWh), as seen in Figures 7.3 and 7.4. The display logic updates every 2.5 seconds, alternating between:

- Display 1: Voltage (Vrms) and Current (Irms)
- Display 2: Power and Total Energy

These outputs are continuously refreshed, allowing real-time updates without lag or screen flicker. The high contrast of the LCD and consistent refresh timing ensure that even brief variations in parameters are captured and presented clearly to the user. This supports effective on-site monitoring, making it easy to detect spikes in consumption or load variation trends.

The system also ensures that values are not affected by transient spikes or sensor noise. Minor fluctuations are averaged in software, making the output stable and visually legible. This smooth operation, combined with the update frequency, creates an optimal balance between responsiveness and readability.

Fig 7.3 LCD displaying Power, Energy

Fig 7.4 LCD displaying Vrms, Irms

7.5 Remote Monitoring via Telegram:

One of the standout features of this smart energy meter is its IoT capability, demonstrated by seamless integration with Telegram. As shown in **Figure 7.5**, the ESP32 uses a secure TLS 1.2 connection to interact with the Telegram Bot API. The data—formatted as JSON—is sent every 5 seconds to the user's Telegram account.

The real-time push notifications contain comprehensive information about voltage, current, power, and energy usage. These updates empower users to monitor their electrical consumption from remote locations, offering a new level of control and transparency. In industrial scenarios, this capability is particularly valuable, enabling operators to keep track of energy-intensive machinery without being physically present.

Fig 7.5 Telegram Bot displaying parameter value

In the event of a WiFi disconnection, the ESP32 continues to log data locally, ensuring no data loss. Once connectivity is restored, data transmission resumes automatically, demonstrating the system's resilience and fault tolerance.

The combination of precision sensing, local visualization, and global access through Telegram positions this IoT-based energy meter as a practical and scalable solution for both residential and commercial energy management.

.

Chapter 8
CONCLUSION

This project presents the design and implementation of an IoT-enabled Smart Energy Meter utilizing the ESP32 microcontroller, aimed at delivering precise, real-time monitoring of electrical parameters. The system integrates high-accuracy sensors—ZMPT101B for voltage and SCT-013 for current measurement—alongside embedded software routines for computing instantaneous voltage, current, power, and accumulated energy consumption. Sensor readings are processed through calibrated scaling formulas, with outputs visualized locally on a 16x4 LCD and transmitted remotely via the Telegram Bot API.

The embedded system ensures robust operation under both idle and active load conditions, with minimal deviation from commercial-grade wattmeters. Energy logging is handled using time-differentiated power calculations, and the system supports network fault tolerance by maintaining local logging during connectivity interruptions. The inclusion of real-time mobile notifications enables remote monitoring and proactive energy management, addressing the limitations of conventional analog meters.

The full hardware-software stack was developed and validated experimentally, confirming the system's suitability for residential, commercial, and industrial environments. This work demonstrates the viability of low-cost, scalable energy monitoring systems based on open-source IoT platforms. Future iterations could extend this design by incorporating bi-directional communication, automated load control, or integration with smart grid infrastructure for dynamic energy pricing and predictive analytics.

Chapter 9
FUTURE SCOPE

As the global demand for energy efficiency and smart infrastructure continues to rise, the potential for extending and enhancing this project is vast. The Smart Energy Monitoring System developed in this work lays a strong foundation for intelligent power management and IoT integration. With further development, it can evolve into a comprehensive solution that contributes to sustainable living, smarter cities, and empowered consumers. The following points outline the potential future advancements and applications of this system:

Integration with Smart Grids:

The system can be extended to communicate with smart grid infrastructure, enabling dynamic load balancing, energy redistribution, and intelligent power outage management.

Mobile App Development:

A dedicated Android/iOS application can be developed to allow users to monitor real-time energy consumption, set usage alerts, and receive insights directly on their smartphones.

AI-Based Energy Analytics:

Incorporating machine learning algorithms can enable predictive analytics for energy usage patterns, fault detection, and automated optimization recommendations.

Support for Renewable Energy Sources:

The system can be upgraded to monitor solar panels, wind turbines, or hybrid systems, helping users manage and balance conventional and renewable power sources.

Remote Device Control:

Future versions can include the ability to remotely switch appliances on/off based on user input, schedules, or sensor-based automation.

Billing and Cost Estimation:

The system could provide detailed billing reports, tariff comparisons, and real-time cost estimation based on energy consumption data and regional electricity rates.

Multi-User and Multi-Meter Support:

Expanding the project to handle multiple users or households within a single application can make it scalable for apartment complexes or commercial buildings.

Home Automation Integration:

Compatibility with home automation platforms like Google Home, Alexa, or Home Assistant can be introduced for seamless smart home integration.

Cloud Data Storage and Dashboard:

A web-based dashboard can be implemented using cloud platforms (e.g., Firebase, AWS) for long-term data storage, visualization, and analytics.

Regulatory and Compliance Integration:

The system can be aligned with governmental energy standards and provide reports or feedback that help users comply with energy efficiency regulations or certifications.

References

[1] S. Prabu, "Real-time Energy Monitoring System Using Blynk 2.0 and Non-invasive Sensors," 2023 IEEE International Conference on IoT Technologies (IoT-Tech), Chennai, India, 2023, pp. 1-6, doi: 10.1109/IoT-Tech49302.2023.10152348.

[2] P. Macheso and D. Thotho, "Enhanced ESP32-based Power Measurement System with CT Sensors," 2022 IEEE International Conference on Smart Grid Technology (SmartGrid),Lilongwe,Malawi,2022,pp.1-6,doi: 10.1109/SmartGrid54821.2022.10078453.

[3] H. J. El-Khozondar, R. El-Khozondar and M. Hassan, "Integration of WhatsApp Notifications in Smart Energy Monitoring," 2024 IEEE International Conference on Energy Management Systems (ICEMS), Gaza, Palestine, 2024, pp. 1-5, doi: 10.1109/ICEMS51258.2024.10189647.

[4] S. Gadekar, P. Deshpande and V. Kumar, "HLW8012-based Energy Consumption Tracking with ESP32," 2021 IEEE Regional Symposium on Micro and Nanoelectronics(RSM),Mumbai,India,2021,pp.1-4,doi: 10.1109/RSM52563.2021.9512576.

[5] R. R. Mohassel, A. Fung, F. Mohammadi and K. Raahemifar, "A survey on Advanced Metering Infrastructure," 2014 IEEE International Conference on IndustrialTechnology(ICIT),Busan,Korea,2014,pp.1-6,doi: 10.1109/ICIT.2014.6894936.

[6] S. Vivekanandan, R. Kaushik and K. P. Chandra, "Telegram-based IoT Energy Monitoring Solution," 2021 IEEE International Conference on Power Electronics andSmartGrid(PESG),Chennai,India,2021,pp.1-6,doi: 10.1109/PESG47234.2021.9654893.

[7] D. A. Nugraha and Amirullah, "Low-latency Telegram Integration for Remote Energy Control," 2023 IEEE International Conference on Electrical Engineering and Informatics (ICEEI), Bandung, Indonesia, 2023, pp. 1-6, doi: 10.1109/ICEEI56448.2023.10234762.

[8] V. Reddy, P. Sharma and J. K. Singh, "Dynamic IoT Dashboards for Energy Consumption Analysis," 2021 IEEE International Conference on Industrial Internet (ICII), Hyderabad, India, 2021, pp. 1-6, doi: 10.1109/ICII52893.2021.9689437.

[9] A. D. Patel and J. D. Patel, "Firebase Cloud Integration for Scalable Energy Data Analytics," 2020 IEEE International Conference on Data Science and Internet of Things(DSIOT),Ahmedabad,India,2020,pp.1-5,doi: 10.1109/DSIOT49125.2020.9728341.

[10] M. Alvi, F. Ahmed and S. Z. Khan, "GSM Fallback Mechanisms for Smart Energy Systems," 2019 IEEE International Conference on Communication Systems and NetworkTechnologies(CSNT),Gwalior,India,2019,pp.1-6,doi: 10.1109/CSNT47234.2019.9568234.

[11] A. S. Salunkhe, Y. K. Kanse and S. S. Patil, "Internet of Things based Smart Energy Meter with ESP 32 Real Time Data Monitoring," 2022 International Conference on Electronics and Renewable Systems (ICEARS), Tuticorin, India, 2022, pp. 446-451, doi: 10.1109/ICEARS53579.2022.9752144.

[12] A. Khan and S. Roy, "ThingSpeak Platform for Advanced Energy Data Processing," 2020 IEEE International Conference on Energy Management and Renewable Resources (ICEMRR), Kolkata, India, 2020, pp. 1-6, doi: 10.1109/ICEMRR45156.2020.9768435.

[13] N. Ashokkumar, V. Arun, S. Prabhu, V. Kalaimagal, D. Srinivasan and B. Shanthi, "Design and Implementation of IoT based Energy Efficient Smart Metering System for Domestic Applications," 2023 9th International Conference on Advanced Computing and Communication Systems (ICACCS), Coimbatore, India, 2023, pp. 1208-1212, doi: 10.1109/ICACCS57279.2023.10113012.

[14] R. N. S. Kumar, M. V. S. S. S. Murthy, S. R. M. Reddy and M. Praveen, "ESP32-Based Smart Energy Meter with Cloud Integration," 2023 IEEE Power and Energy SocietyGeneralMeeting(PESGRE),NewYork,USA,2023,pp.1-6,doi: 10.1109/PESGRE52268.2023.10053181.

[15] S. Patel, D. S. Patil and A. G. Shukla, "Real-Time Energy Monitoring Using SCT-013 and ZMPT101B," 2022 IEEE International Conference on Smart Electronics andTelecommunicationSystems(ICSETS),jaipur,India,2022,pp.345-350,doi: 10.1109/ICSETS53881.2022.9762793

[16] K. Verma, R. K. Sharma and S. S. Jain, "IoT-Enabled Smart Meter with Telegram Alert System," 2023 IEEE International Conference on Communications and NetworkTechnologies(CONIT),Jaipur,India,2023,pp.1-6,doi: 10.1109/CONIT55038.2023.10182712.

[17] T. Nguyen, J. M. K. Lim and M. S. R. S. S. Prabhu, "Precision Energy Measurement Using ESP32 and IoT," 2022 IEEE Region 10 Humanitarian Technology Conference(R10HTC),Singapore,2022,pp.1-5,doi:10.1109/R10-HTC54060.2022.10098901.

[18] L. Chen, M. X. Huang and Y. J. Wang, "Low-Cost Smart Meter with Cloud Data Analytics," 2023 IEEE International Symposium on Circuits and Systems (ISCAS), Taipei, Taiwan, 2023, pp. 1-5, doi: 10.1109/ISCAS46773.2023.10181772.

[19] A. Gupta, S. S. Deshmukh and R. M. Kumar, "Energy Monitoring System with Wireless Sensor Networks," 2022 IEEE India Conference (INDICON), New Delhi, India, 2022, pp. 1-6, doi: 10.1109/INDICON56171.2022.10040033.

[20] B. Wang, Z. Y. Lee and X. H. Zhang, "IoT-Based Power Quality Monitoring System," 2023 IEEE Power and Energy Society General Meeting (PEDG), Boston, USA, 2023, pp. 1-6, doi: 10.1109/PEDG54999.2023.10213081.

[21] D. Silva, P. R. Arumugam and P. S. Ravi, "ESP32-Based Smart Grid Monitoring Solution," 2022 IEEE Innovative Smart Grid Technologies Latin America (ISGT-LA),BuenosAires,Argentina,2022,pp.1-6,doi:10.1109/ISGT-LA54543.2022.10032641.

[22] E. Martinez, A. S. Yadav and R. T. Soni, "Cloud-Connected Energy Meter with AI Analytics," 2023 IEEE International Conference on Industrial Electronics (IECON), Madrid, Spain, 2023, pp. 1-6, doi: 10.1109/IECON51785.2023.10312345.