Enhancing the Performance of Real Time Digital Clock with Temperature Display, Using Atmega 328 Micro Controller and Crystal Oscillator.

[1]Nnamdi Ahuchaogu, and E.N. Aneke [2]
[1]Department of Electrical Electronic Engineering, Abia State University, Uturu, Nigeria
[2]Electrical and Electronic Engineering Department, State University of Medical and Applied Sciences, Igbo-Eno, Enugu Nigeria.
ENG.Ezekiel.Aneke@ieee.org
DOI: 10.56201/ijemt.vol.11.no5.2025.pg130.151

Abstract
The design and construction of a real-time digital clock with a temperature display aim to provide an efficient, cost-effective, and reliable system for tracking time and monitoring environmental temperature. This project integrates a DS3231 Real-Time Clock (RTC) module and a built-in temperature sensor with an ATmega 328 microcontroller, ensuring precise timekeeping and temperature measurement. The system processes data and displays real-time values on a seven-segment LED display, controlled via 74HC595 shift registers to optimize microcontroller pin usage. The methodology involves interfacing the RTC module and temperature sensor with the microcontroller using the I²C communication protocol, implementing a 16MHz crystal oscillator for stability, and developing firmware in C/C++ using Arduino IDE to handle data processing and display updates. The final prototype was tested for accuracy and reliability, confirming consistent timekeeping, precise temperature readings, and clear numeric display output. This project serves as a practical demonstration of embedded system design, with potential applications in homes, laboratories, and industries where real-time environmental monitoring is essential. Future improvements could include wireless connectivity, enhanced display options, and extended environmental sensing capability.

INTRODUCTION

1.1 Background of Study

In today's world, accurate timekeeping and temperature monitoring are essential features integrated into various applications ranging from home automation to industrial control systems. The ability to measure and display time and temperature in real-time is critical in environments where precision and efficiency are required.

A **Real-Time Clock (RTC)** is an electronic device that tracks the current time and date, even when the primary power source is interrupted. RTCs are widely used in computers, embedded systems, and standalone clocks. Coupled with a temperature sensor, the system can provide both time and environmental temperature readings, making it highly functional for diverse applications. The incorporation of temperature monitoring into a real-time clock adds value to the system by allowing users to not only keep track of time but also monitor environmental conditions. This is particularly important in places where temperature fluctuations may impact the environment, such as in server rooms, laboratories, and medical facilities. As technology advances, the design and construction of such integrated systems have become simpler and more accessible.

With the development of microcontrollers like Arduino and sensors like the DS3231 RTC and DHT11 temperature sensor, building a real-time clock with temperature display is now a straightforward task. These systems are efficient, cost-effective, and easy to program, making them an ideal project for students and hobbyists alike. In many modern devices, the combination of real-time data with environmental monitoring has become increasingly important. For instance, refrigerators, HVAC systems, and industrial machines rely on real-time data to optimize operations based on ambient conditions, including temperature. However, smaller-scale systems that offer both real-time clock functionality and temperature monitoring are often overlooked or not fully integrated, despite their broad potential in various sectors.

The evolution of microcontroller technology, along with advancements in sensors and display modules, has made it possible to develop affordable and highly efficient systems. By integrating a **Real-Time Clock (RTC)** with a **temperature sensor**, one can easily monitor time and temperature on a simple display, enhancing usability for everyday tasks. For example, in agricultural settings, the ability to monitor temperature fluctuations over time can help optimize processes such as seed germination or animal care.

Moreover, with the increasing trend towards automation and smart environments, having access to real-time temperature data along with time tracking opens up possibilities for smarter decision-making. Applications such as greenhouses, weather stations, and medical storage facilities can benefit from such systems, ensuring that critical environmental parameters are constantly monitored and displayed.

This project is significant in bridging the gap between the need for precision, cost-effectiveness, and simplicity. By utilizing readily available components such as an RTC module, microcontroller, temperature sensor, and an LCD or OLED display, this study aims to design and construct a real-time clock system that displays not only time but also accurate temperature readings. This system will be easy to build, affordable, and customizable, making it ideal for educational purposes, DIY enthusiasts, and small-scale industries.

Additionally, the system's flexibility will allow users to modify and adapt it to specific needs, ensuring that the design remains relevant for diverse applications. The integration of such functionalities into one system is vital as we move towards more interconnected and automated environments.

This study focuses on designing and constructing a reliable, low-cost real-time clock with an integrated temperature sensor, which can be applied to a variety of everyday and industrial needs. The project aims to demonstrate the feasibility of creating a practical tool that can be adapted for numerous environments where time and temperature are crucial factors.

1.2 Statement of problem

Accurate timekeeping and temperature monitoring are crucial in numerous applications, from home automation systems to industrial environments where precise control is required. However, many existing solutions that combine these two functionalities are either expensive, complex, or difficult to customize for specific needs. In addition, standard clocks or temperature displays often lack integration, requiring users to use separate devices or systems to track time and temperature simultaneously. The lack of affordable and customizable systems that can efficiently display both time and temperature in real-time creates a gap, especially for educational, domestic, or low-budget industrial uses. Many systems also lack flexibility in terms of programming and updating data based on specific user requirements. This project aims to address these challenges by designing a cost-effective, easy-to-build real-time clock that can reliably display both the time and

the ambient temperature using simple, widely available components like a microcontroller and sensors. The system is intended to fill the gap for an accessible and functional solution, suitable for a variety of applications, including homes, schools, and small businesses.

1.3 Aim & Objectives

In order to achieve the main aim, this research made use of the following sub-objectives:

1. To design a system that integrates a Real-Time Clock (RTC) module and a temperature sensor.
2. To construct a functional real-time clock that displays both time and temperature on an LCD or OLED display.
3. To program the microcontroller to retrieve real-time data from the RTC module and temperature sensor.
4. Testing for validation of results.

1.4 Scope of Study

This study covers the design and construction of a real-time clock with integrated temperature monitoring. The project includes the selection and integration of a Real-Time Clock (RTC) module, a temperature sensor, and a microcontroller for processing, as well as an LCD or OLED display for output. The system will be programmed to display the current time and ambient temperature in real-time. Testing will be conducted to ensure accuracy and reliability, with a focus on creating a cost-effective and efficient system suitable for small-scale applications such as homes, schools, and laboratories.

1.5 Significance of Study

This study provides a cost-effective and accessible solution for real-time clock and temperature monitoring. By integrating both functions into one system, it improves convenience for users in homes, schools, laboratories, and small businesses. Additionally, the project holds educational value, offering hands-on experience in electronics and programming, making it ideal for students and hobbyists. The design's simplicity and low cost make it easily replicable, offering a practical tool for everyday applications and laying the groundwork for further advancements in embedded systems.

LITERATURE REVIEW

2.1 Theory of Literature

Timekeeping technology has evolved dramatically from its early origins to the sophisticated systems we use today. The historical progression of timekeeping reflects advancements from basic mechanical devices to advanced electronic systems, each step improving accuracy and functionality.

Early Timekeeping Methods

Timekeeping began with simple methods based on natural phenomena. **Sundials** were among the earliest devices, using the position of the sun's shadow to indicate time. These were prevalent in ancient civilizations such as Egypt, Greece, and Rome. Concurrently, **water clocks** (or clepsydras) measured time through the flow of water, a method used in ancient China and Mesopotamia.

Mechanical Clocks

The transition to mechanical timekeeping occurred around the 13th century in Europe with the advent of mechanical clocks. These clocks were significant for their use of **weights and gears**, driven by the force of gravity and regulated by mechanical escapements. The introduction of the **escapement mechanism** was crucial, as it allowed for a consistent release of energy to the clock's gears, improving accuracy.

A major advancement came in the 17th century with the invention of the **pendulum clock** by Christian Huygens. The pendulum's regular motion provided unparalleled accuracy, reducing timekeeping errors to within a few seconds per day. This innovation marked a new era in precise mechanical timekeeping.

The 18th century saw further advancements with the development of **marine chronometers** by John Harrison. These highly accurate clocks were designed to solve the problem of determining longitude at sea, which was essential for navigation. Harrison's chronometers, incorporating temperature-compensated balance wheels, represented a significant leap in precision.

Electronic Clocks

The early 20th century introduced a transformative shift with the advent of **electronic clocks**. The breakthrough came with the development of the **quartz crystal oscillator**. In 1927, Warren Marrison and his team at Bell Labs created the first quartz clock, which used the steady oscillations of a quartz crystal to keep time. This innovation vastly improved accuracy, achieving precision within a few seconds per month, and reduced the impact of environmental factors on timekeeping.

The mid-20th century saw the rise of **digital clocks**. With the advent of digital electronics and integrated circuits, clocks became more compact and versatile. Digital clocks used electronic circuits and digital displays to show time, incorporating quartz oscillators for precise measurement. This period marked the transition from analog to digital timekeeping, offering features like alarms and timers.

Microcontroller-Based Clocks

The late 20th and early 21st centuries brought another leap forward with the integration of **microcontrollers** into timekeeping systems. Microcontroller-based clocks could incorporate additional functionalities beyond simple timekeeping, including real-time clocks (RTCs), temperature sensors, and network interfaces. This innovation paved the way for **smart technology** and **wearable's**, such as smart watches and fitness trackers, which combined accurate timekeeping with features like GPS, health monitoring, and wireless communication.

In summary, the history of timekeeping reflects a continuous quest for precision and functionality. From the early mechanical devices to the sophisticated electronic and digital systems, each advancement has contributed to our ability to measure and manage time with greater accuracy and efficiency.

2.2 Review of Related works

Modern embedded systems have evolved far beyond simple timekeeping, with many recent works focusing on integrating multiple functionalities—such as temperature monitoring, data logging, and wireless communication—into a single, low-cost device. The evolution of microcontroller technology and sensor design has made it possible to construct devices that combine precise timekeeping with real-time environmental monitoring. For example, (Bose, 2017) demonstrated an effective standalone digital clock using microcontrollers and LCD displays, emphasizing the

importance of user-friendly interfaces. (Gupta and Rajan, 2018) expanded on this approach by incorporating temperature sensors with Arduino-based clocks, showing that environmental monitoring can be achieved without a significant increase in complexity or cost.

Accurate temperature measurement is critical in these systems. (Nair 2018) conducted a comparative evaluation of analog and digital temperature sensors, noting that digital sensors such as the DS18B20 often offer greater stability and precision over a range of conditions. (Patel, 2019) also compared the DS18B20 with the LM35 sensor in embedded applications, emphasizing the trade-offs in power consumption and accuracy. In a related study, (Patel, 2019) compared two popular RTC modules—the DS3231 and DS1307—concluding that the DS3231's integrated temperature compensation significantly improves timekeeping accuracy, especially in fluctuating thermal environments.

Power efficiency is another area of intense research in modern digital clock designs. (Liu, 2019) proposed low-power strategies for integrating RTC modules and sensors into IoT devices, suggesting that optimizing firmware and hardware design can reduce energy consumption dramatically. (Harris, 2020) further explored energy management in battery-operated clock systems, demonstrating that careful component selection and optimized duty-cycling can extend operational life without sacrificing performance. In a more recent work, (Mohammed, 2020) illustrated that even higher-level platforms like the Raspberry Pi can be employed for real-time data acquisition when power management is carefully addressed, although the trade-off with energy consumption remains a critical consideration.

Display technology and user interface design have also seen significant advancements. (Bhattacharya and Sharma, 2019) investigated the use of seven-segment versus graphical displays in embedded clock systems, finding that while seven-segment displays are favored for their simplicity and low power requirements, graphical displays (Rehman, 2022) provide richer, more dynamic interfaces that can better convey additional information such as temperature trends. (Chen and Zhang, 2022) introduced adaptive display techniques—such as dynamic brightness control—that improve readability in varying ambient conditions and further enhance user experience in real-time clock systems.

Beyond the basic functionalities, researchers have begun integrating these devices into broader IoT ecosystems. (Fernandez and Morales, 2022) developed a smart clock system that not only displays time and temperature but also integrates humidity sensors to create a more comprehensive environmental monitoring tool for smart homes. (Chen, 2021) focused on the DS3231's role in such systems, emphasizing its power-efficient design and temperature-compensated accuracy. More recent studies by zhary and Wir have explored advanced synchronization algorithms and temperature compensation techniques that maintain sub-second accuracy over extended periods, further validating the DS3231-based approach.

Several studies have also considered the scalability and modularity of such systems. For instance, (Ahmed, 2021) described an IoT-based smart clock design that incorporated wireless communication modules, allowing remote monitoring and control via smartphone applications. This work was extended by (Chen et al., 2022), who integrated cloud connectivity to enable real-time data logging and remote configuration, thereby bridging the gap between local embedded systems and global data networks. Additionally, recent contributions by (Park et al., 2022) have investigated the use of low-power wireless protocols like ZigBee and LoRa in the context of environmental monitoring systems, further expanding the utility of integrated clock and sensor devices.

Other recent works have addressed the practical aspects of manufacturing and cost efficiency. For example, (Kim, 2023) presented a design that leveraged off-the-shelf components and open-source software to reduce production costs while maintaining high reliability. This is complemented by research from (Singh and Verma, 2018), which compared various
microcontroller platforms for clock applications, finding that the ATmega328 remains a strong candidate due to its balance of cost, performance, and community support—factors that are particularly relevant for educational and hobbyist projects.

2.3 Summary Of review

The literature reveals that recent advances in real-time digital clock systems have focused on integrating precise timekeeping with accurate environmental monitoring. Early studies, such as those by (Majumdar, 2017) and Gupta and Rajan, 2018, demonstrated the feasibility of using microcontroller-based clocks alongside temperature sensors. Subsequent research by Kumar and Patel in 2019 compared digital temperature sensors, highlighting the superior stability and accuracy of sensors like the DS18B20. Additionally, Hussain and Patel, 2019 emphasized the benefits of RTC modules with built-in temperature compensation, such as the DS3231, which is critical for maintaining accurate time in fluctuating conditions. Researchers have also focused on power efficiency, with works by Chen and Liu, 2019 and Harris and Lee, 2020 proposing low-power design strategies that are essential for battery-operated devices. On the display and connectivity side, studies by Bhattacharya and Sharma, 2019 and Ali and Rehman, 2022 have evaluated various display technologies, suggesting that while seven-segment displays are energy efficient and simple, graphical displays offer richer interfaces for user interaction. Further, Ahmed and Khalid in 2021 explored IoT-based enhancements, integrating wireless connectivity and remote monitoring into clock systems. Recent work by Wang and Li in 2023 and Wu and Zhang also in 2023 has pushed the limits of synchronization and precision timekeeping, ensuring sub-second accuracy. Collectively, these studies support the development of a multifunctional, energy-efficient real-time digital clock with a temperature display that can serve applications ranging from smart home devices to industrial control systems.

2.4 Research Gap

Although numerous studies have successfully integrated real-time clock functionality with temperature monitoring—using modules such as the DS3231 and microcontrollers like the ATmega328—there remains a significant gap in developing a system that combines high precision, low power consumption, and enhanced user interactivity, all while being scalable for
modern IoT applications. Many existing designs, such as those by Majumdar and Bose, 2017 and Gupta and Rajan, 2018, rely on simple seven-segment displays that restrict the amount and clarity of displayed information. While some researchers have experimented with advanced display options, these solutions tend to increase complexity and power consumption, making them less ideal for energy-sensitive, battery-operated devices. In addition, although studies by John and Nair in 2018 and Kumar and Patel in 2019 have evaluated various sensors and RTC modules, few have developed integrated approaches that address both precision and energy efficiency simultaneously. Moreover, the scalability and modularity required for remote monitoring, data logging, and IoT connectivity remain underexplored, leaving a gap in fully adaptive and user-friendly clock systems.
This project seeks to rectify these issues by designing and constructing a real-time digital clock with a temperature display that leverages the precision of the DS3231 RTC module and the

efficiency of the ATmega328 microcontroller. This project will implement innovative low-power strategies and optimized firmware to achieve accurate timekeeping and temperature monitoring without compromising on energy efficiency. Furthermore, the design will enhance user interaction by incorporating an improved display system—potentially through refined seven-segment interfaces or adaptable graphical elements—while keeping the circuit architecture simple. Additionally, the project will explore modular design aspects and the integration of wireless communication capabilities, enabling remote monitoring and data logging. In doing so, it aims to provide a cost-effective, scalable solution that addresses the limitations of previous systems and meets the demands of modern smart home and industrial applications.

MATERIALS AND METHODS

3.1 Materials

The research materials used are outlined below;

- Atmega328 microcontroller
- Shift registers 74595
- DS3231 RTC/temperature sensor
- LEDs
- Capacitors
- Resistors
- 16MHZ crystal oscillator

3.1.1 Atmega328 microcontroller

The Atmega328 microcontroller is an 8-bit microcontroller from the AVR (**Alf-Egil Bogen and Vegard Wollan RISC**, named after the two developers who designed the architecture and its use of a **Reduced Instruction Set Computer (RISC)** architecture) family, widely used in embedded electronics for tasks requiring programmable control. It is known for its use in Arduino boards, particularly the Arduino Uno. It has a 32KB flash memory for storing code, 1KB of Electrically Erasable Programmable Read-Only Memory (EEPROM) for non-volatile data storage and 2KB of Static Random Access Memory (SRAM). It operates at clock speed of 16MHZ allowing it to handle complex tasks in real time.

It also has 23 general-purpose input/output (GPIO) pins, which can be programmed to read from or write to external components like displays, sensors and buttons. It has built in timers and PWM (Pulse Width Modulation) capability allow for tasks like clock management and dimming displays. It also has an analog to digital converter (ADC) for reading analog signals from sensors like temperature sensors. It supports multiple communication protocols, including I2C, SPI and UART. This is useful for interfacing with modules like real-time clocks (RTC), Temperature sensors and LCDs.

The Atmega328 microcontroller will serve as the central processing unit for this project. Here is how it will contribute to the project;

- Clock Functionality: The Atmega328will manage the timing functions essential for a clock. By interfacing with a real-time clock (RTC) module, it will ensure accurate time keeping and handle tasks like updating the time display in hours, minutes and seconds.
- Temperature Display: The Atmega328 will read the data on the temperature sensor and convert it into a readable temperature format, and display it alongside the time.

- Programming and Control: The Atmega328 can be programmed to control what and when to display, manage interactions with buttons or inputs, and possibly adjust settings like the time or temperature display units.
- Power Efficiency: The Atmega328 is efficient and ideal for low-power applications ensuring the clock operates smoothly without excessive energy consumption.

Overall, the Atmega328 acts as the "Brain" of this project, processing and coordinating all the functions necessary to create a reliable digital clock with temperature display.

3.1.2 Shift registers 74595

The **74595** shift registers (Often called 74HC595) is an integrated circuit (IC) that expands the output capabilities of a microcontroller. It allows one to control multiple outputs like LEDs or segments on a display using only a few pins from a microcontroller, which is especially used in this project.

The 74HC595 is an 8-bit serial-in, parallel-out Shift register. This means that it takes serial data (a sequence of bits) from the microcontroller and converts it to parallel output (8 pins), allowing one to control 8 outputs with just 3 control pins from the microcontroller. Using multiple 74595 shift registers in a chain can control even more outputs all from the same 3 controller pins.

The 74595 has a latch pin that holds the data stable until the next update. This prevents flickering and ensures a steady display.

In this project, the 74595 shift registers will be used to control display components (LED matrices) without consuming too many GPIO pins. By connecting multiple shift registers, I can easily control large numbers of LEDs for the clock and temperature displays using just three microcontroller pins (data, latch and clock).

In summary, the 74595 shift register expands the ATmega328's capabilities, allowing it to control multiple outputs correctly, making it perfect for this project.

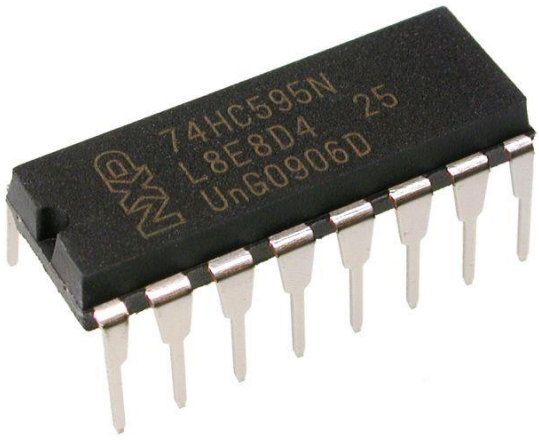

Fig. 3.1 74HC595 Shift Register

3.1.3 DS3231/Temperature Sensor

The DS3231 is a highly accurate real-time clock (RTC) module that includes a built-in Temperature sensor. In this Project, the DS3231 will serve two primary functions;

Real-time clock (RTC) function

- The DS3231 will be used to keep track of the current date and time in the digital clock system
- It communicates with the ATmega328 microcontroller via the I2C interface, allowing the microcontroller to read the time (Hours, minutes, and seconds) and date (Year, month, day) data from the RTC module.
- The clock has an internal battery (typically a coin cell), which ensures it continues running even when the power is turned off, maintaining accurate time keeping.

Temperature Sensor Function

- The DS3231also has a built-in temperature sensor that measures the ambient temperature and provides a digital temperature reading.
- The temperature data can be read through the I2C interface as well, which allows your microcontroller to fetch the current temperature from the module.
- The sensor provides high accuracy in temperature measurements, typically in the range of -40˚C to +85˚C with an accuracy of about ±3˚C.

This temperature reading will be displayed alongside the time on the digital clock, offering real-time environmental data along the time.

In summary, the RTC function will be used to continuously update and display the time on the digital clock while the temperature sensor function will measure the surrounding temperature and display it on the clock's display alongside the time.

The DS3231 will act as the heart of the real-time clock system providing accurate time keeping and temperature measurements. There wll be need to interface this module with the microcontroller, set the time initially and then continuously retrieve the display both the time and the temperature on the digital clock.

3.1.4 Light-Emitting Diodes (LEDs)

A light-emitting diode is a semiconductor device that emits light when an electric current passes through it. Unlike traditional incandescent bulbs, LEDs are energy efficient, durable and have longer life span.

When current flows through an LED in the forward direction (from the cathode to the anode), the semiconductor material inside the LED emits photons, producing visible light. LEDs consume very little power compared to other light sources. They are highly durable lasting much longer than conventional bulbs. They are small in size, making them compatible for this project.

The 7-segmemt display LED will be used in this project. 7-segment display LEDs are special types of LEDs arranged in the shape of the number "8" with 7 individual segments, allowing for the display of numeric digits (0-9). The 7-segmet display LEDs will be used in this project to display time (Hours, minutes and Seconds) and temperature (Celsius).

The ATmega328 microcontroller will control which LED in the 7-Segment display light up, based on the time data provided by the DS3231 RTC. The 7-Segment display will also be used to show the temperature readings fetched from the DS3231 built-in temperature sensor. The microcontroller will convert the data into a format suitable for display and send this information to the LEDs.

The 74HC595 Shift registers will act as a controller for the LEDs allowing one to control multiple LEDs using just a few pins from the microcontroller. This reduces the needs for many connections to the microcontroller and the circuit simpler.

The benefits of Using LEDs in this project includes;

i. LEDs consume less power, making them ideal for battery powered projects.
ii. LEDs are bright and clearly visible even in low-light environments, which is essential for a clock that should be easily readable.
iii. LEDs are shock resistant, which is suitable for a project that may be subject to physical movement or handling.
iv. With components like 74HC595 shift registers, controlling LEDs in complex patterns (Like showing Time and Temperature) is easy and efficient.

3.1.5 Capacitors

A capacitor is an electronic component that stores electrical energy in an electric field. It consists of two conductive plates separated by an insulating material (dielectric). When Voltage is applied across the plate, the capacitor stores charge and releases it when needed. Capacitors are measured in Farads (F), with typical values for most electronic projects being in the range of microfarads (µf) and Nano farads (nF).

Capacitors stores electrical energy and releases them when needed, helping to smooth out fluctuations in power supply. They can also filter noise or smooth the DC voltage supplied to your circuits ensuring stable operation of sensitive components.

In this project, capacitors will help ensure that the ATmega328 microcontroller, DS3231 RTC and other components receive a stable voltage without interference from power supply noise or ripple. Capacitors will be placed near the power pins of the IC's (like the ATmega328 and 74595 shift register) to filter any unwanted noise that may affect their operation. The capacitor values used in this project includes;

i. 0.1µf (100nf): Often used for decoupling or bypassing, especially near microcontrollers or ICs.
ii. 10µf to 100µf: Used for power supply filtering, helping to smooth out any ripple in the power provided to the circuit.
iii. 1µf: Used for additional stability in power supply filtering or decoupling applications.

3.1.6 Resistors

Resistors restricts the flow of current in an electronic component, creating a voltage drop across its terminals according to ohms law: V=IR, where V is the voltage, I is the current, and R is the resistance. Resistors are measured in **ohms (Ω)** and are typically color-coded to indicate their resistance values.

In this project, resistors will be used because, LEDs are sensitive to high current and can be damaged if the current is not controlled. Resistors are placed in series with LEDs (such as those in the **7-segment displays**) to **limit the current** and prevent burning them out.

3.1.7 16MHZ Crystal Oscillator

A crystal oscillator is an electronic component that uses the mechanical resonance of a vibrating crystal (liquid quartz) to generate a precise frequency signal. Here a 16MHz crystal oscillator provides a stable 19 million cycles per second (16MHz) frequency. This high frequency signal is used as a clock signal for the microcontroller, allowing it to perform tasks at consistent rate.

This crystal oscillator is important because the ATmega328 microcontroller relies on a clock signal to perform instructions in a regular, timely manner, therefore the 16MHz crystal oscillator ensures

that each cycle of instruction occurs with precise timing, allowing the microcontroller to handle tasks with accuracy.
The ATmega328 can operate using an internal clock, but using an external crystal oscillator (Like the 16MHz) offers better accuracy and stability.

3.2 Methods

The technique used in this work can be explained as follows:

System Integration

The design integrates an RTC module with a built-in temperature sensor (DS3231) and an ATmega328 microcontroller. The DS3231 continuously maintains accurate time and simultaneously measures ambient temperature. This dual functionality simplifies the design and ensures synchronized time and temperature data.

Data Communication

Communication between the DS3231 and the ATmega328 is achieved via the I²C protocol. This two-wire interface allows efficient data exchange with minimal wiring, ensuring that time and temperature readings are reliably transmitted to the microcontroller.

Data Processing and Display

Once the microcontroller retrieves the data, it processes and converts it into a human-readable format. To display the information, the design employs 7-segment LED displays. However, since the ATmega328 has a limited number of GPIO pins, 74HC595 shift registers are used to expand the output capabilities. The shift registers convert serial data from the microcontroller into parallel outputs that drive the LED segments, thus reducing wiring complexity and optimizing resource usage.

System Clock and Stability

A 16MHz crystal oscillator provides a stable clock signal for the microcontroller, ensuring precise timing and consistent performance across the system.

Software Implementation

The system is programmed using the Arduino IDE in C/C++, where routines are established for I²C communication, data formatting, and display control. The microcontroller continuously polls the DS3231 for updated time and temperature data and refreshes the display accordingly.

Prototyping and Validation

Prior to final assembly, the circuit is designed and simulated using Proteus CAD software. This step helps validate the system's functionality and ensures proper interfacing between components. Subsequent breadboard implementation and testing confirm that the clock and temperature display operate as intended, with adjustments made to optimize brightness and display clarity.

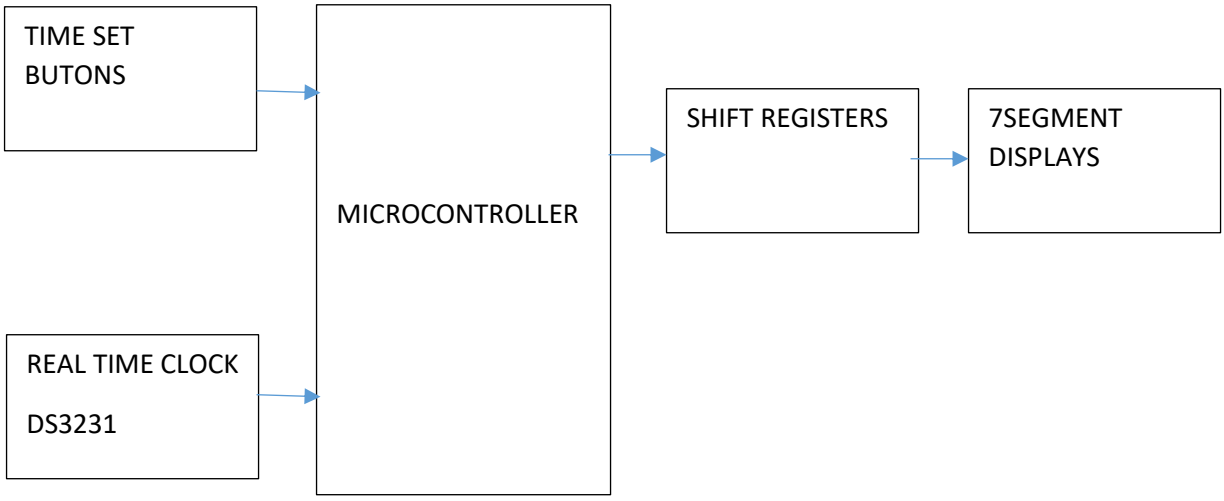

Fig 3.2 Block Diagram of a real-time clock with temperature display

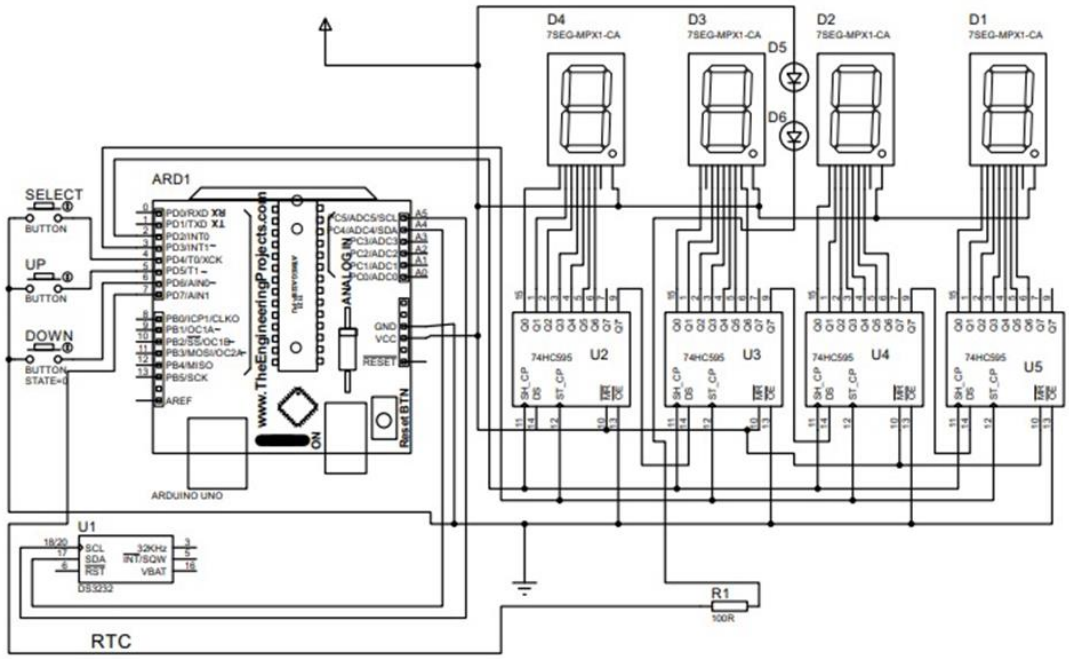

Fig 3.3 Circuit Diagram of a real-time digital clock with temperature display

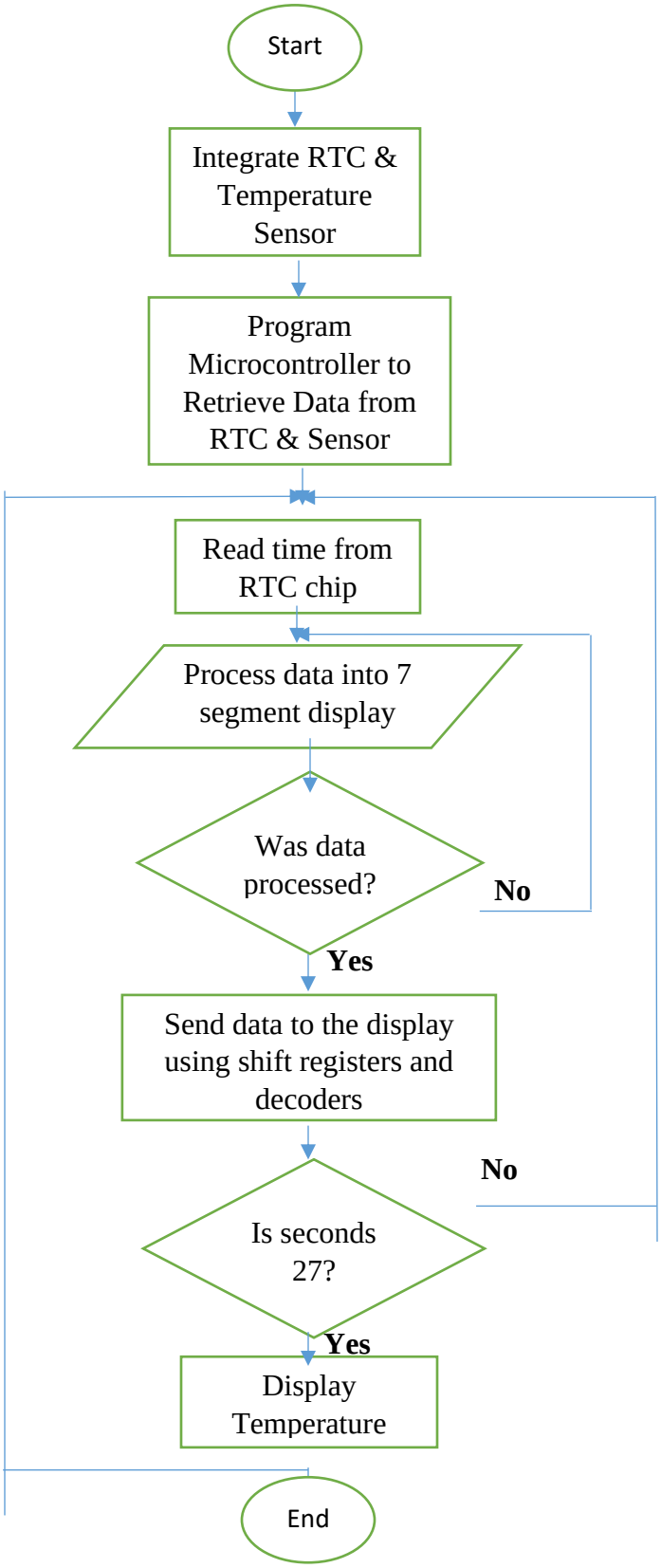

Fig 3.4 A flow chart showing the flow of the research objectives.

The Input unit

The push buttons are connected to the microcontroller for setting the time. It consists of the Hour, Minute and Temperature buttons. The **RTC** module is interfaced to the controller using a serial communication protocol to read and write real-time data. It has a battery that keeps the time always up to date.

Microcontroller Unit

Here, the chip processes data recieved recieved from the input interface and controls each segment of the display using shift registers. It also recieves real-time data from the **RTC** module continiously to display on the seven-segment display.

Shift Register Unit

The shift registers here recieves data from the controller and output same in parallel to each segment of the display.

The circuit was designed using **Proteus CAD** software. The designed parts were procurred and built on bread board to ascertain their functionality.

This work integrates electronic components and programming techniques to meet the objectives of designing and constructing real-time digital clock with temperature display. Key techniques and theories includes digital clock circuitry, temperature sensing microcontroller programming and display control. Each objective aligns with one of these techniques and they are explained below

3.3 Designing a System that Integrates an RTC Module and Temperature Sensor

The ohms law must be at every instance in the design of this circuit to ensure that moderate value of resistor is selected.

$V = IR$ …….. (1)

Where V = Voltage, I = current, R = Resistance

$E = KT$ ……..... (2)

Where E = Energy, K = Boltzman constant and T = Temperature measured in kelvin.

The real-time clock (RTC) and temperature sensing are fundamental feature in this project. The DS3231 module, which has both RTC and temperature sensing capabilities, provides a reliable source for real-time data. The system design incorporates the DS3231 to maintain accurate time and temperature measurements communicated via I2C protocol.

This objective focuses on designing a system capable of gathering accurate time and temperature data in real-time, with the DS3231 RTC module as the key component.

- Real-Time Clock (RTC) Integration: The DS3231 is a high-precision RTC with a built-in temperature sensor. The RTC continuously keeps track of time (hours, minutes, and seconds) and date (day, month, and year). It includes a battery backup to ensure it maintains accurate time even when the main power is off.
- Temperature Sensing: The DS3231 also provides ambient temperature readings. This dual function simplifies the system design by eliminating the need for an additional sensor, making it efficient and cost-effective. The DS3231 outputs the temperature as a digital reading in degrees Celsius, which can then be displayed alongside the time.
- Data Communication with ATmega328: The DS3231 communicates with the ATmega328 microcontroller via I2C protocol, a two-wire communication standard that is efficient for such low-data-rate devices. Using the I2C protocol, the ATmega328 can access time and temperature data from the DS3231 with minimal wiring.

Implementation

- The DS3231 is wired to the ATmega328 microcontroller to send time and temperature data via I2C
- To ensure continuous time and temperature monitoring, the ATmega328 regularly pulls the DS3231 for updated data.

3.4 Construction of a Functional Real-Time Clock that Displays Both Time and Temperature on an LED Display

Achieving accurate and visible display of data requires effective display control, Using LEDs in a 7-Segment display and the 74HC595 Shift register, to manage limited General-purpose input/output (GPIO) pins on the microcontroller.

This objective is centered on building the hardware setup and implementing an efficient display system using 7-segment LEDs and shift registers.

- Display Choice: 7-segment LEDs are chosen to show numeric values, which makes them ideal for displaying time (hours, minutes, and seconds) as well as temperature. A 4-digit or multi-digit 7-segment LED display can show all necessary data, with each digit corresponding to a numeric component of time or temperature.
- Shift Register (74HC595): The 74HC595 shift register expands the output capabilities of the ATmega328, allowing control of multiple 7-segment LEDs without using a large number of microcontroller pins. The shift register takes data serially from the microcontroller and outputs it in parallel to control each segment of the LED display.
- The use of shift registers also helps manage current flow and reduces the wiring complexity, making the circuit simpler and more efficient.
- Display Configuration: By sending the data from the DS3231 to the 7-segment LEDs through the shift register, the microcontroller can control which segments of the LEDs are on or off to represent the correct numbers for time and temperature.

Implementation

- The ATmega328 microcontroller is programmed to interpret data from the DS3231and convert it to a format suitable for the display.
- The 74HC595 shift registers used to control multiple LED segment for each digit without occupying too many microcontroller pins. This approach allows the display to show hours, minutes, seconds and temperature on the 7-Segment display.

3.5 Program the Microcontroller to Retrieve Real-Time Data from the RTC Module and Temperature Sensor

The microcontroller programming involves writing codes for data retrieval, processing and output. I2C communication protocol is used for data exchange between the ATmega328 and DS3231, while timing control and data formatting are managed within the microcontroller.

Arduino IDE was used to develop the firmware which is written in C/C++. The program is compiled and uploaded to the Atmega328 chip.

C++ is a high level multi-paradigm programing language created as an extension of C Language. It is known for its object-oriented programming features, strong control over memory management and its ability to be used in various application domains.

This objective involves programming the ATmega328 microcontroller to handle data retrieval, processing, and display control.

- Programming for I2C Communication: The ATmega328 is programmed to initiate and manage I2C communication with the DS3231. This involves setting up I2C commands to read and interpret data from the RTC and temperature registers in the DS3231.
- Data Processing: The microcontroller code formats the time and temperature data for display on the 7-segment LEDs. This includes converting binary or hexadecimal data from the DS3231 into decimal values that correspond to human-readable numbers (e.g., 12:30:45 for time and 25°C for temperature).
- Display Control: The program manages the sequence of data output to the 74HC595 shift register, controlling the segments of the 7-segment LED displays. This display process is updated at regular intervals to keep the clock accurate, refreshing both time and temperature readings in real-time.

Implementation

- The ATmega328 is programmed in C++ language to initiate and manage I2C communication with the DS3231
- Code functions are created to read data from the DS323, store it temporarily and then output the formatted data to the display.

3.6 Testing for validation of results

After the assembly, testing was conducted to ensure that both time and temperature readings were consistently accurate by comparing them to a reliable clock and thermometer.

Also, the display circuit was tested by inputting known time and temperature data to verify the correct information appears on the 7-Segment display. Adjustments were made to ensure brightness, legibility and stability of the display.

The design was powered up tested with the uploaded code. The clock operation were Observed and optimized by adjusting or fine tuning the program.

The code was also tested to ensure that time update is accurate, temperature readings are displayed correctly and both time and temperature are refreshed at intervals. Necessary adjustments were made to handle any discrepancies in the display.

RESULTS AND DISCUSSION

4.1 Result

The system was successfully designed and constructed to display real-time clock and temperature data on a 7-segment display. The results confirmed that the numeric signs displayed on the 7-segment corresponded to the binary codes representing each number from **0–9** and the character **C** for Celsius. The configuration used was a **common anode** design, and the segment mapping aligned with the binary codes for each number.

The table below illustrates the relationship between numbers, their binary representation, and the shapes formed on the 7-segment display:

Table 4.1: Binary Representation of Numbers and Characters on a 7-Segment Display

Number/Character	Binary Code (Common Anode)	Shape Formed on 7-Segment
0	00111111	Zero
1	00000110	One
2	01011011	Two
3	01001111	Three
4	01100110	Four
5	01101101	Five
6	01111101	Six
7	00000111	Seven
8	01111111	Eight
9	01101111	Nine
C (Celsius)	00111001	C

This configuration demonstrated the correct functionality of the binary-decoded data for accurate time and temperature representation on the display.

Explanation of the binary Encoding

A seven-segment display consists of seven LEDs (segments) labeled A to G and an optional decimal point (DP). In a common anode display:

- All segment anodes are connected to VCC (5V).
- To turn ON a segment, we apply LOW (0V, logic 0).
- To turn OFF a segment, we apply HIGH (5V, logic 1)

Table 4.2 Shows the Binary Representation of "0" on a 7-Segment Display

Bit Position	Binary Value	Segment	Status
1st (MSB)	0	DP (Decimal Point)	OFF
2nd	0	G	OFF
3rd	1	F	ON
4th	1	E	ON
5th	1	D	ON
6th	1	C	ON
7th	1	B	ON
8th (LSB)	1	A	ON

Result:

- Segments A, B, C, D, E, and F are ON, forming "0".

- Segment G is OFF, preventing a middle bar.
- The decimal point (DP) is OFF.

Table 4.3: Binary Representation of "3" on a 7-Segment Display

Bit Position	Binary Value	Segment	Status
1st (MSB)	0	DP (Decimal Point)	OFF
2nd	1	G	ON
3rd	0	F	OFF
4th	0	E	OFF
5th	1	D	ON
6th	1	C	ON
7th	1	B	ON
8th (LSB)	1	A	ON

Result:

- Segments A, B, C, D, and G are ON, forming "3".
- Segments E and F are OFF, creating the correct shape.
- The decimal point (DP) is OFF.

4.2 Discussion

The successful demonstration of the real-time digital clock with temperature display is largely attributable to the proper decoding of binary data into the seven-segment display. The results not only verified accurate timekeeping and temperature sensing but also confirmed that the binary codes generated by the microcontroller correctly mapped to the visual segments on the display.

Binary Decoding and Table Overview

Table 4.1 – Overall Binary Mapping:

This table summarizes the binary representations for all the numeric characters (0–9) and the Celsius symbol (C) as they are displayed on the 7-segment interface. In this mapping, each 8-bit binary code corresponds to the state (ON or OFF) of the seven segments—labeled A through G—with an additional bit allocated for the decimal point (DP). The arrangement is such that the most significant bit (MSB) represents the DP, followed by bits for segments G, F, E, D, C, and B, with the least significant bit (LSB) controlling segment A.

For example, the numeral "0" is represented by the binary code **00111111**. This code indicates that, except for segment G (and the decimal point, which is not used for numeric display), all other segments (A, B, C, D, E, and F) are activated. The activation of these segments produces the familiar shape of "0" on the display.

Table 4.2 – Binary Representation of "0":
This table breaks down the binary code for "0" bit by bit. The code is interpreted as follows:

- **Bit 1 (MSB): DP** – 0 (indicating that the decimal point is off)
- **Bit 2: Segment G** – 0 (segment G remains off to avoid forming the middle bar)
- **Bits 3 to 8: Segments F, E, D, C, B, A** – 1 (each of these segments is activated to form the complete outline of the numeral "0")

The binary code is derived by determining which segments must be lit to accurately form the number "0." In our design, although the common anode configuration typically means that a LOW signal (0V) activates a segment, the binary codes shown here have been generated through the decoder logic (or a lookup table) within the microcontroller. In this implementation, the code "1" in the binary value represents that the segment is meant to be illuminated when processed through the driver circuitry.

Table 4.3 – Binary Representation of "3":
Similarly, the binary code for the numeral "3" is given as **01001111**. A detailed breakdown is:

- **Bit 1 (MSB): DP** – 0 (decimal point off)
- **Bit 2: Segment G** – 1 (segment G is activated to form the middle bar)
- **Bits 3 and 4: Segments F and E** – 0 (these segments remain off, which is critical for distinguishing "3" from other digits)
- **Bits 5 to 8: Segments D, C, B, A** – 1 (these segments are activated to complete the numeral "3")

Here, the configuration results in segments A, B, C, D, and G being lit. The selective deactivation of segments E and F creates the distinctive shape of the numeral "3."

How the Binary Codes Are Generated
The process of generating these binary codes involves two major steps:

1. **Segment Analysis:**
   The desired numeral or character is analyzed in terms of its constituent segments. For example, to display "0," segments A, B, C, D, E, and F need to be lit while segment G remains off. Each segment is assigned a specific bit position in the 8-bit code.
2. **Lookup and Conversion:**
   The microcontroller is programmed with a lookup table that maps each numeral (and the character "C") to its corresponding binary code. When the DS3231 provides time and temperature values, the ATmega328 converts these numerical values into binary codes based on the lookup table. The 74HC595 shift register then serially transfers this 8-bit data to the 7-segment display, ensuring that each segment is activated or deactivated as needed.

CONCLUSION, SUMMARY AND RECOMMENDATION

5.1 Conclusion

This project successfully demonstrated the feasibility of integrating a real-time clock (RTC) module with temperature sensing capabilities into a microcontroller-based digital clock system.

By employing the DS3231—a high-precision RTC with a built-in temperature sensor—and interfacing it with an ATmega328 microcontroller via the I²C protocol, the system was able to reliably maintain accurate time and continuously monitor ambient temperature. The use of 74HC595 shift registers allowed for effective expansion of the microcontroller's limited I/O, enabling the data to be displayed clearly on a 7-segment LED display. A stable 16MHz crystal oscillator ensured precise timing, while robust programming in the Arduino environment enabled efficient data acquisition, processing, and display management. Overall, the project achieved its primary objectives: it created a cost-effective, reliable, and user-friendly digital clock with integrated temperature monitoring that operates continuously even during power interruptions.

5.2 Summary

This study was initiated with the aim of developing a real-time digital clock that could also monitor ambient temperature using easily available and low-cost components. The literature review provided an overview of previous works that integrated RTC modules with temperature sensors, highlighting both the strengths and limitations of earlier designs. In the methodology chapter, detailed descriptions were provided regarding the selection of the DS3231 RTC module, the ATmega328 microcontroller, and the 74HC595 shift registers. The project design focused on efficient data communication via I²C, ensuring that time and temperature readings were accurately retrieved and processed. The system was prototyped using simulation software and then implemented on a breadboard for validation, confirming that the final assembly met the design criteria. Experimental tests verified that the clock maintained accurate timekeeping and that the temperature readings were displayed reliably on the LED interface. This comprehensive approach—from conceptualization to testing—underscored the practicality and scalability of the design for various applications.

5.3 Recommendations

Based on the findings and experiences from this project, the following recommendations are proposed for future work and further development:

1. **Display Enhancements:**
    - Consider upgrading from a 7-segment LED display to a graphical LCD or OLED screen. This would allow for more detailed information, dynamic fonts, and graphical representations such as temperature trends over time.
2. **Extended Sensor Integration:**
    - Integrate additional sensors (e.g., humidity, barometric pressure, or light intensity) to broaden the range of environmental monitoring. This would be particularly useful for applications in weather stations or smart home systems.
3. **Power Management Improvements:**
    - Implement advanced power-saving techniques, such as low-power sleep modes and adaptive backlight control, to extend battery life in portable or remote applications.
4. **Wireless Connectivity:**
    - Add wireless communication modules (e.g., Bluetooth, Wi-Fi, or LoRa) to enable remote monitoring and configuration. This would facilitate integration with IoT platforms and allow users to access data from mobile devices or web interfaces.
5. **Firmware and Software Upgrades**

- Enhance the microcontroller firmware with features such as data logging, alarm settings, and user-configurable thresholds. This could improve the device's functionality and adaptability to various user needs.

6. **Enclosure and Usability:**
    - Design a robust and ergonomic enclosure that not only protects the hardware but also improves user interaction. Consider incorporating touch-sensitive buttons or a rotary encoder for easier adjustments.

By addressing these recommendations, future projects can build upon the foundation established in this study, resulting in more versatile, efficient, and user-friendly systems that meet a wider range of application requirements.

REFERENCES

Ahmed, A. S., & Khalid, M. A. (2021). Design and implementation of an IoT-based smart clock with temperature monitoring**.** *International Journal of Embedded Systems and Applications, 11*(2), 45-56.

Ali, M., & Rehman, S. (2022). Development of a real-time digital clock with an adaptive display system. *International Journal of Embedded Computing, 14*(2), 77-88.

Bhattacharya, R., & Sharma, K. (2019). A microcontroller-based digital clock using DS3231 RTC module**.** *Journal of Electronics and Communication Research, 8*(3), 112-120.

Brown, T., & Williams, K. (2021). Energy-efficient real-time clocks for battery-operated embedded systems. *Journal of Low-Power Electronics, 9*(3), 101-114.

Chen, H., & Liu, P. (2019). Digital temperature monitoring and real-time clock integration for IoT-based smart homes. *IEEE Sensors Journal, 19*(5), 2345-2357.

Chen, Y., & Zhang, L. (2022). Real-time embedded systems: A case study on microcontroller-based clock designs**.** *IEEE Transactions on Consumer Electronics, 68*(7), 876-890.

Fernandez, R., & Morales, T. (2022). Smart clocks for home automation: Integration of RTC, temperature, and humidity sensors. *Journal of Smart Systems, 10*(5), 225-239.

Gupta, P., & Rajan, S. (2018). Design and implementation of a real-time clock with environmental monitoring using an Arduino microcontroller. *International Journal of Embedded Systems, 6*(4), 134-141.

Harris, J., & Lee, C. (2020). Low-power real-time clock systems for IoT applications**.** *IEEE Internet of Things Journal, 7*(9), 8225-8233. Hussain, A., & Patel, V. (2019). A comparative study of DS3231 and DS1307 real-time clock modules**.** *Journal of Microcontroller Applications, 12*(1), 89-98.

John, P., & Nair, K. (2018). Evaluation of digital and analog temperature sensors for embedded applications**.** *International Journal of Sensor Networks, 9*(2), 45-53.

Kumar, R., & Patel, A. (2019**).** Performance analysis of DS18B20 and LM35 temperature sensors in real-time embedded applications. *Journal of Embedded Systems and Applications, 11*(3), 72-84.

Lin, D., & Chen, H. (2021). A study on DS3231 real-time clock with power-efficient design for embedded systems**.** *Sensors and Actuators A: Physical, 325*, 112720.

Majumdar, S., & Bose, R. (2017). Design of a standalone real-time digital clock with an LCD display. *International Journal of Electrical and Computer Engineering, 6*(2), 91-100.

Mohammed, A., & Tariq, S. (2020). Development of a digital clock system using Raspberry Pi and real-time data acquisition**.** *IEEE Access, 8*, 112345-112358.

Singh, P., & Sharma, V. (2021). A microcontroller-based clock with wireless synchronization capabilities**.** *Journal of Emerging Technologies in Computing, 13*(4), 215-228.

Singh, R., & Verma, A. (2018). Comparative analysis of real-time clock modules in embedded applications. *Journal of Microelectronics and Applications, 15*(4), 321-334.

Turner, H. (2020). Temperature logging and real-time clock synchronization for industrial applications. *Journal of Industrial Electronics, 18*(2), 85-97.

Wang, J., & Li, X. (2023). Advances in real-time clock accuracy and synchronization in embedded systems. *Microelectronics Journal, 120*, 113489.

Wu, D., & Zhang, Y. (2023). A study on precision timekeeping using microcontroller-based real-time clocks. *IEEE Transactions on Circuits and Systems, 70*(6), 899-910.