Science, Engineering & Education, 9, (1), 2024, 49-58

Development and Programing of LCD Digital Clock with “RTC - Real Time Clock Module”

Yosif Marwan Al-Hamoud, Ivaylo Ivanov*

Vocational School of Electrical Engineering and Electronics
26 Peshtersko Chaussee Blvd., Plovdiv 4002, Bulgaria

Received 05 September 2024, Accepted 11 October 2024
DOI: 10.59957/see.v9.i1.2024.7

ABSTRACT

In the digital era, precise timekeeping is critical across a large variety of applications, from personal devices and hobby electronics to industrial systems. This article presents the development and programming of a digital clock using a Real-Time Clock (RTC) module, a core component that ensures accurate time maintenance even if the device in which it is used is not connected to the power source. My research focuses on the integration of the RTC module using a microcontroller, emphasizing on the reliability of timekeeping provided by this hardware. My article can serve as a guide for hobbyists and enthusiasts looking to create their own digital clocks. It’s also useful for students engaged in electronics and programming. This project gives creatives the opportunity to further develop it or even completely redesign the entire PCB. By combining practical and theoretical knowledge and skills, I aim to demonstrate the importance of RTC modules in modern timekeeping devices and inspire innovation in digital clock designs.

Keywords: device, real time clock, timekeeping, Arduino, PCB, layer, device.

INTRODUCTION

Since time immemorial, people have oriented themselves and measured time. Over the years in its development, inventors have created different ways to measure time. Starting from tracking the sun during the day and the stars at night to the most advanced ways of measuring the time like digital and smart watches. A major leap in timekeeping technology was the first digital clock invented in the 20th century, it replaced traditional dials and hands with displays that showed the time. The first popular digital clock used an LED display to indicate the time. That digital clock used a quartz resonator that generates a signal with a very precise frequency of 1 Hz. However, with the development of technology, inventors have found much better ways to design such a digital clock. One of the most common ways now is to use the RTC - Real time clock module. A real time clock is an electronic device in the form of chip that measures the time with extremely high accuracy [1]. It’s used in personal computers, mobile phones and many other consumer electronic devices. Although you can measure time without RTC modules, using one has a lot of benefits. It can be more accurate than other

**Correspondence to: Ivaylo Ivanov, Vocational School of Electrical Engineering and Electronics, 26 Peshtersko chaussee Blvd., Plovdiv 4002, Bulgaria, E-mail: ivvvolf719@gmail.com*

methos of tracking the time and it has low power consumption. In many cases real time clocks have alternative sources of power. This allows the real time clock to keep track of the time even when the controller is not powered or when the main source of power is off. Another advantage of the RTC module is that it uses I2C communication [2]. I2C is synchronous serial communication used to connect devices to microcontrollers in a short distance. It's simple and it has low manufacturing costs. Speaking of microcontrollers, the one I am using for my project is ATmega328P. The Atmel is 8-bit microcontroller with RISC architecture. It features 32 KB of flash memory, 2 KB of SRAM, and 1 KB of EEPROM. It includes various peripherals such as ADC, timers, and communication interfaces like I2C, SPI, and USART [3]. This controller is well for its use in the Arduino UNO board, making it the perfect controller for DIY electronic projects or prototyping. In this article, we will explore the development and programming of a digital clock using an RTC module and the ATmega328P microcontroller. We will cover the process of designing a PCB. Additionally, we will discuss the practical applications of this project and how it can be a useful learning tool for those that are interested in electronics and programming. The idea of this project is to show the importance of precise timekeeping in modern technology and create a guide for those interested in making a clock of their own [1].

EXPERIMENTAL

The first step in developing the clock was to make a schematic or block diagram of the hardware. Schematic and block diagrams are shown on Fig. 1 and Fig. 2.

The power supply block consists of DC adapter, voltage regulator as well as the in/out filter capacitors. For this project I decided to use surface mount voltage regulator because that makes the use of a heatsink unnecessary. To cool the voltage regulator, I designed a cooper plated PCB. A copper-plated PCB (Printed Circuit Board) refers to a type of circuit board where a layer of copper has been applied or plated onto the board. As we all know, most electric conductors, are good thermal conductors as well. In every way like most metals, copper is both an excellent thermal and electrical conductor. That makes it ideal for cooling the voltage regulator [2].

The crystal oscillator block consists of 16 MHz oscillator and two load capacitors. As we can see the real time clock module has a Li-ion battery that will power it while the microcontroller is not

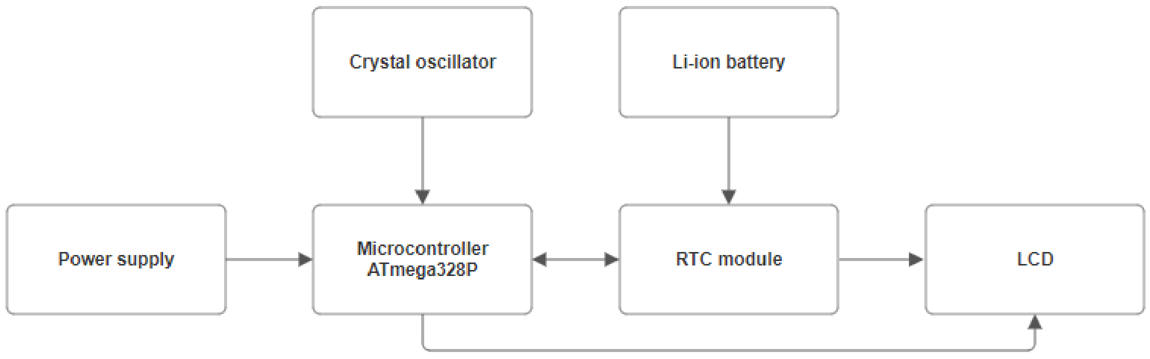

Fig. 1. Block diagram.

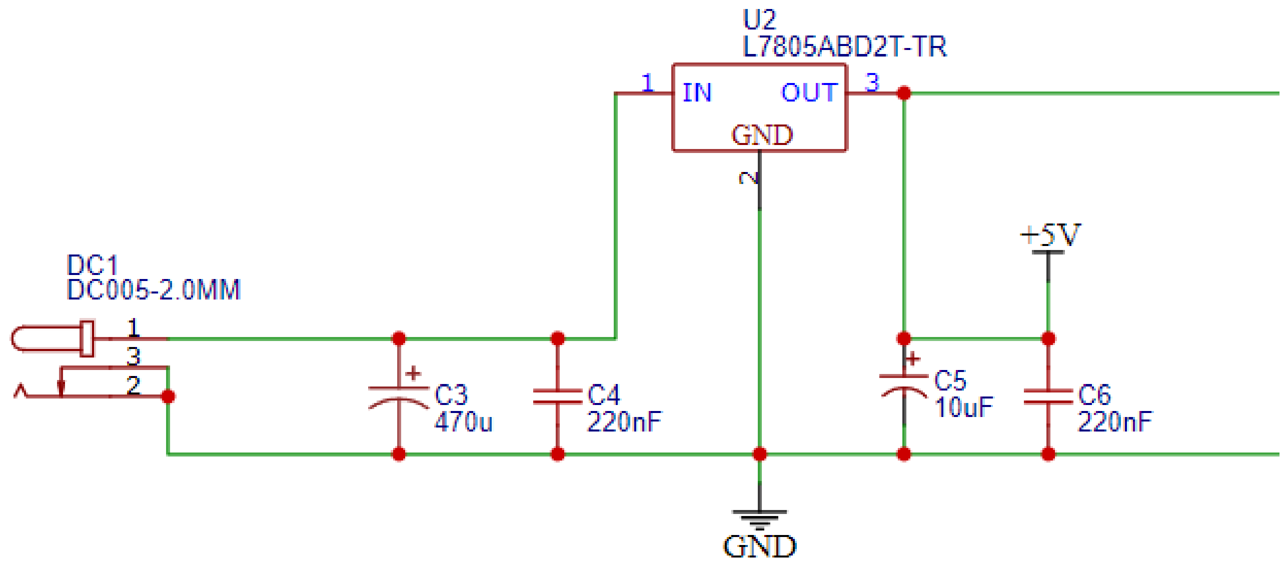

Fig. 2. Schematic diagram of the designed power supply unit.

providing power [4]. Real-time clocks (RTCs) need power to maintain accurate timekeeping even when the main power supply to a device is turned off. Lithium-ion (Li-ion) batteries are often used for this purpose due to several reasons such as stable voltage, low self-discharge rate, compact size and many more [5]. The LCD block consists of display with I2C module attached to it. The model of the display is 1602A ver. 5.5 and its 16 by 2 liquid crystal display.

Before completing the overall schematic diagram of the device, I analyzed every pin of each integrated circuit I used. That helped me make the table of used elements and it also gave me a better understanding of the components I'm working with. Starting from the voltage regulator, that chip has 3 pins, one "in" and one "out" pin. The third pin is connected to the ground. Like most LCD displays the one I'm using has 16 pins. GND and VCC pins are used to connect the display to the power supply. The pin V0 is used to control the contrast of the display. Usually, a voltage ranging from 0 to 5V is applied to this pin (depending on how much contrast we want the display to have). The contrast of a display refers to the difference in luminance or brightness between the lightest (white) and the darkest (black) parts of the screen. Next is the RW pin. RW stands for "Read/Write". The RW pin is used to control whether the display is in read or write mode. In most cases this pin is connected to the ground, so that it operates in write mode displaying data sent from the controller. The E pin on LCD displays stands for "Enable". It is used to initiate the data transfer between the LCD and the microcontroller. In essence this pin's role is to time the data transactions. Next are the D pins (D0 - D7) D stands for data. These pins are used to carry the actual image information that needs to be displayed on the screen. The last two pins on the LCD are the BLA and BLK (Back light anode and back light cathode) these pins are used to connect the backlight LED to the power supply. The back light plays every important role in LCD displays, because it provides illumination, allowing the content on the display to be visible. However, since we are using I2C communication module the LCD display has only 4 pins (GND, VCC, SDA, SCL) that are attached on the integrated module that is soldered on the back of the display. The real time clock has 6 pins. Two of them are used to power the module (GND and VCC). Pins SCL and SDA (serial clock and serial data) are connected to the clock signal line and data signal line. The SCL line is used to synchronize data transfer between master and slave devices. The SDA line is used to carry the actual data that is being transmitted between the master and slave devices. Both the master and slave devices can send or receive data over this line. The 32K pin is used for output pin for a 32 kHz square wave signal. The last pin is SQW this is programmable pin and it generates square wave signals with a frequency of 1 Hz, 4 kHz, 8 kHz or 32 kHz. The microcontroller has 28 pins [1]. Pins. Port B is an 8-bit I/O port with internal pull-up resistors. The pins in Port B (PB0 to PB7) can be used for general-purpose I/O and have several alternative functions. Table 1 presents a detailed list of components including their names, designations, values, quantities, and additional notes.

Pins PB6 and PB7 are used to connect the external oscillator. Port C is a 7-bit bi-directional I/O port with internal pull-up resistors. The pins in Port C (PC0 to PC6) can be used for general-purpose I/O and have several alternative functions. PC4 (SDA) and PC5 (CLK) are used for the I2C the communication. PC 6 is used for the function RESET. Port D (PD0-PD7) is an 8-bit bi-directional I/O port with internal pull-up resistors. The pins in Port D (PD0 to PD7) can be used for general-purpose I/O and have several alternative functions. The VCC and GND pin are used to connect the controller to the power source [6]. AVCC is the supply voltage pin for the ADC and a few other analogy peripherals. AREF is the analogy reference voltage pin for the ADC. After we analysed the key components of our LCD digital clock, we can now focus on developing the schematic. This next figure will illustrate how we connected every component, to make a functional digital clock. Fig. 3 shows the schematic diagram.

The next step in our design process involves translating this schematic into PCB that can be manufactured and assembled. This is achieved through the creation of detailed 2D and 3D models of the printed circuit board (PCB). These models not only provide a visual representation of the component placement and routing, but it also helps us visualize the product. Let’s begin the PCB design process, with the 2D layout and then converting it to 3D model that brings our digital clock to life. Firstly, we must put the crystal oscillator as close as possible to the

Table 1. Table of used elements.

№	Name	Designation	Values	Quantity	Note
1	Microcontroller ATMEGA 328P	U1	-	1	DIP28
2	Voltage regulator 7805	U2	5V	1	SMD
3	LCD display	H1	16x2	1	1602A ver5.5
4	Real-time clock (RTC) module	MOD1	-	1	DS3231
5	Crystal oscillator	Q1	16 Mhz	1	Quartz
6	Capacitors	C1, C2	22 pF	2	Ceramic
7	Capacitor	C3	470 uF	1	Electrolytic
8	Capacitor	C4, C6	220 nF	2	Ceramic
9	Capacitor	C5	10 uF	1	Electrolytic
10	DC adapter	DC1	12V	1	-

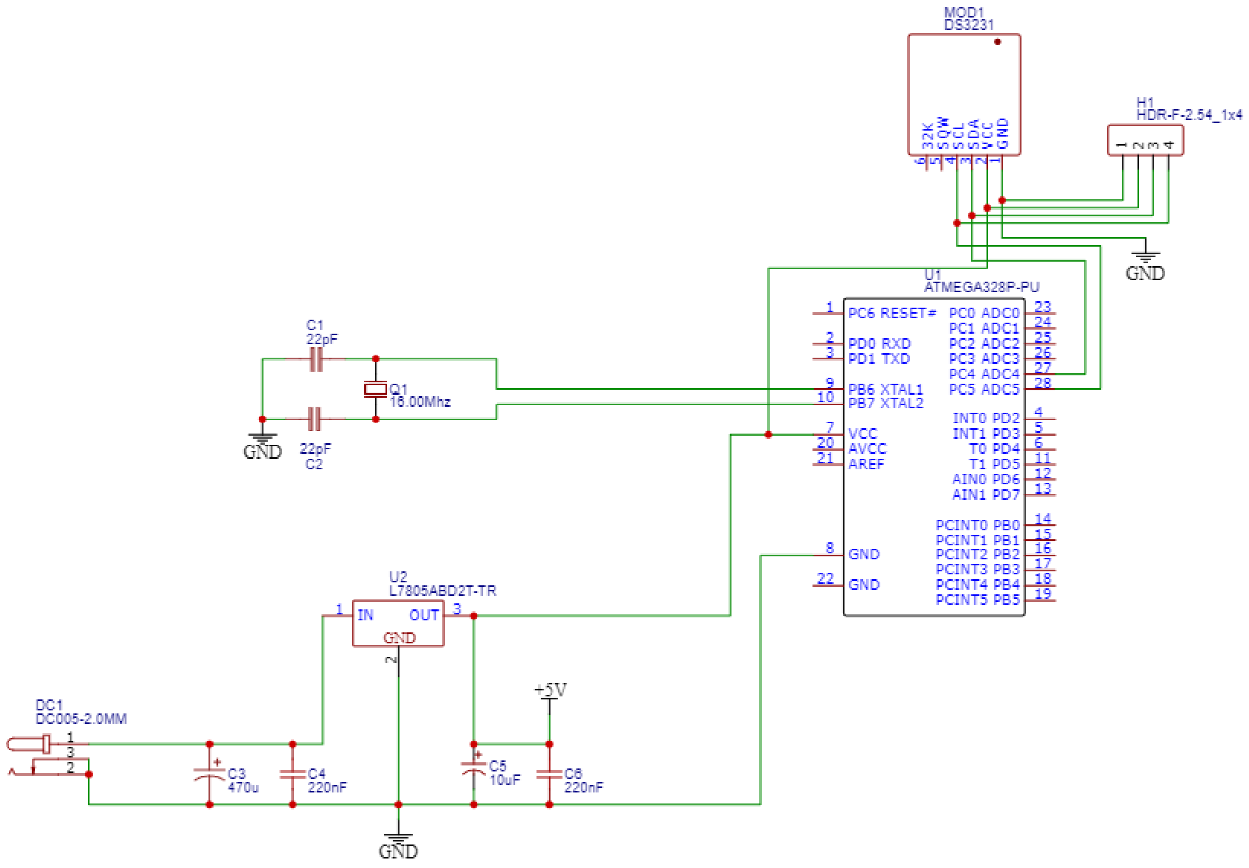

Fig. 3. Schematic diagram of the digital clock.

microcontroller. Long signal paths can introduce parasitic inductance and capacitance, which can distort the clock signal. We need to place the barrel jack in a way that makes it accessible and convenient to use. And since we are making a multilayer PCB we have to place vias. A via is a small hole that is drilled through the PCB and then plated with copper to create electrical connections between different layers of the board. Vias are shown on Fig. 5. It's crucial since it allows signals to flow through layers. Thermal vias are placed close to the voltage regulator to ensure better heat dissipation to other layers. Thermal vias are shown on Fig. 4. All vias are connected to the ground (GND).

To make the 2D and 3D model I'm using the

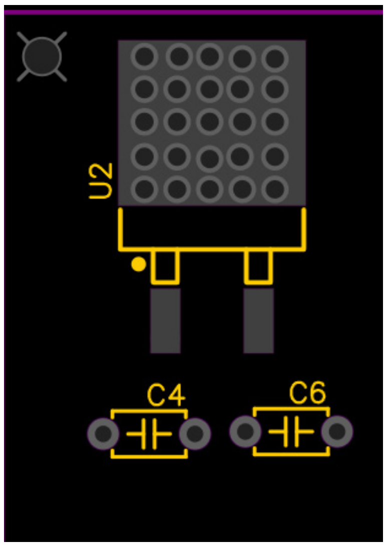

Fig. 4. Thermal Vias.

free PCB design software EasyEda. Figures (Figs. 6, 7, 8, 9, 10, 11) show the design of the PCB. Fig. 12 and Fig. 13 show the design of the 3D model.

After we are done with the PCB, we must make a block diagram of the code we will be using for this project. The block diagram of the code is shown on Fig. 14. Once the physical layout of the PCB is finalized, including the placement of components and routing of electrical connections, it becomes essential to define how these components will interact through software. That's why it's important to make a block diagram that will represent the code structure. Our block diagram aims to show the flow of data between every key module or component. By creating a block diagram, we can ensure that the code will be well organized.

Testing the digital clock on a breadboard is an important step for verifying the functionality of our design before committing to a PCB. Bread

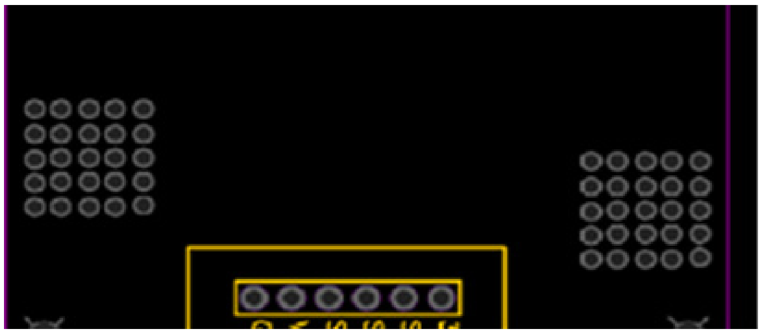
Fig. 5. Via placements.

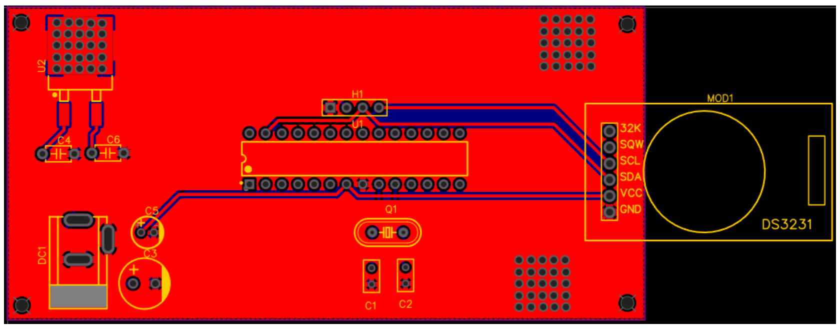

Fig. 6. PCB layout Top Layer.

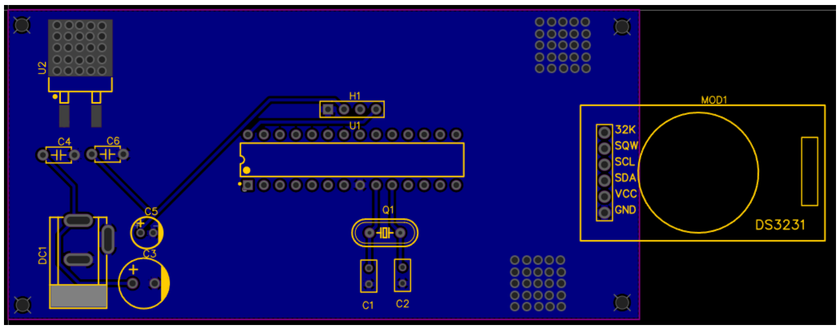

Fig. 7. PCB layout Bottom Layer.

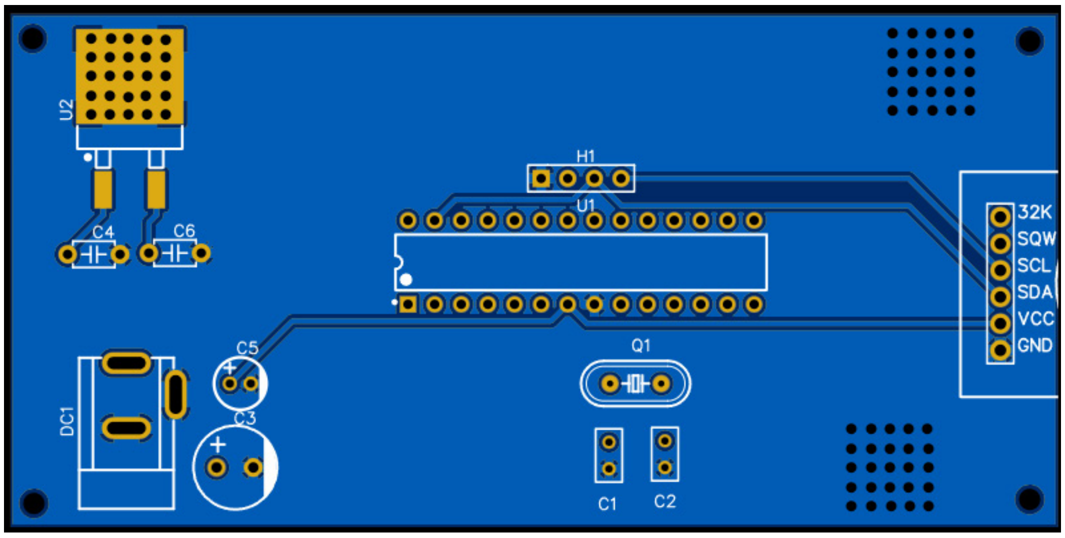

Fig. 8. 2D render of the PCB (Top side).

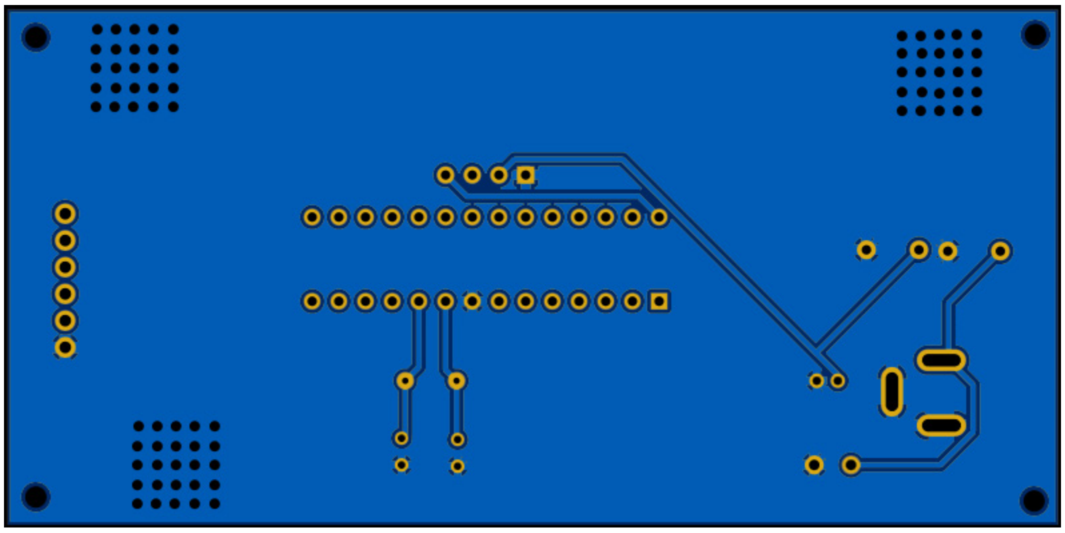

Fig. 9. 2D render of the PCB (Bottom side).

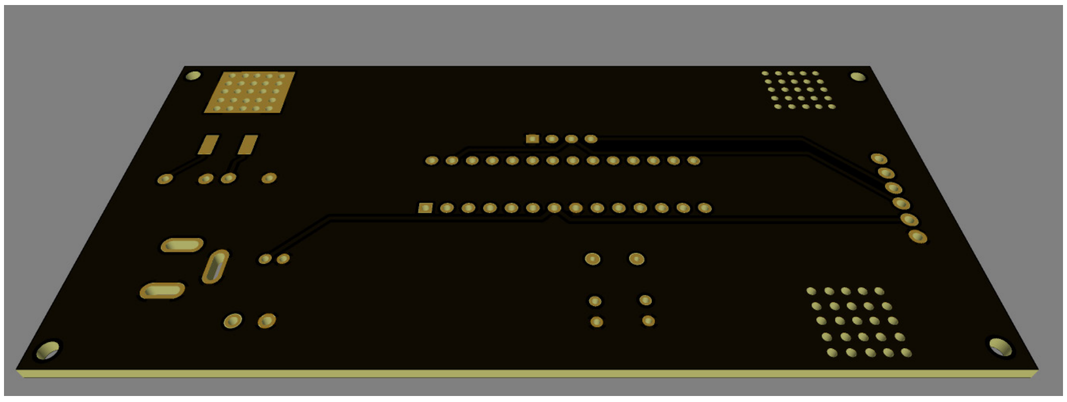

Fig. 10. 3D render of the PCB without components and silk layer (Top side).

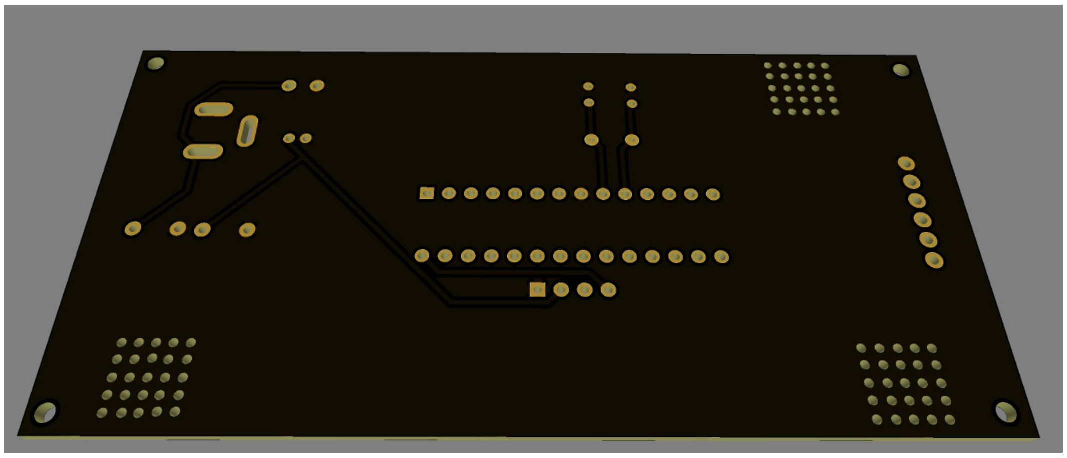
Fig. 11. 3D render of the PCB without components and silk layer (Bottom side).

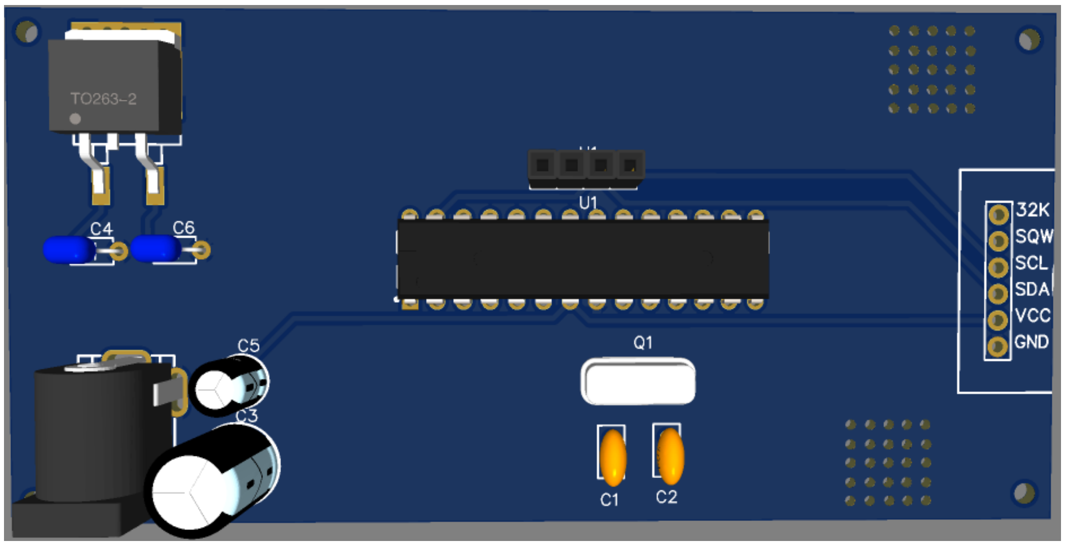

Fig. 12. 3D render of the PCB (Top side).

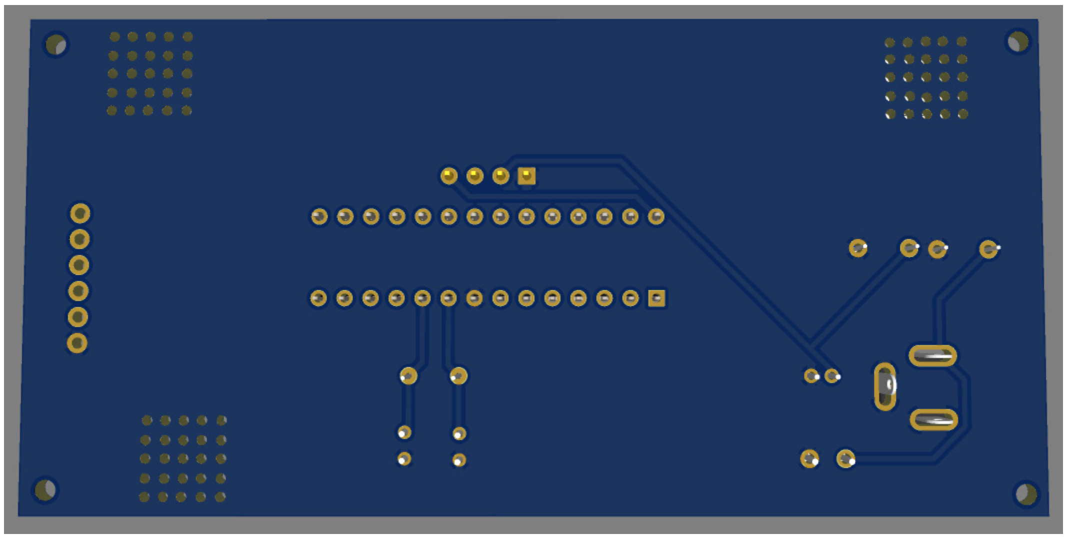
Fig. 13. 3D render of the PCB (Bottom side).

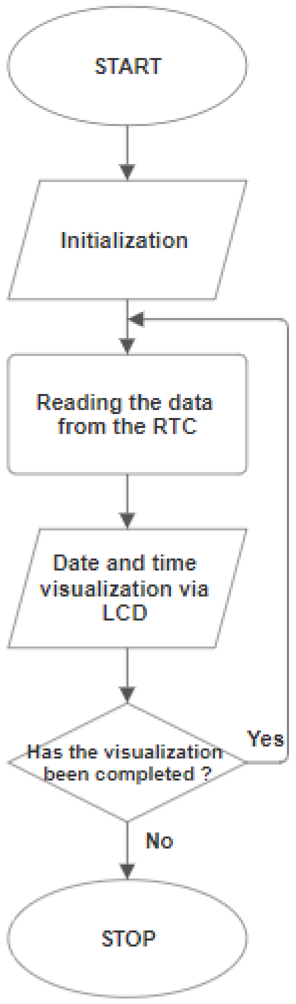
START
Initialization
Reading the data from the RTC
Date and time visualization via LCD
Has the visualization been completed ?
Yes
No
STOP

Fig. 14. Block diagram of the code.

board verification is shown on Fig. 15 and Fig.16. This allows us to confirm that everything works the way we envisioned it and designed it on paper, and that the circuit behaves as expected. By assembling the device on a breadboard, you can easily adjust and troubleshoot any issues that arise. After connecting everything with jumper wires and powering the device with a battery we can ensure its working capacity. That will also help us see the accuracy of the clock (the accuracy of the timekeeping). If everything goes as anticipated, we make certain that the final product will function correctly.

The next critical step of my project is to solder all components on the PCB. This process requires attention to the detail to ensure that every component is soldered correctly. This step requires precision and steady hands to avoid cold solder joints or shorts that can compromise the device's functionality. With the soldering process complete, the only one thing left was to make a case for the digital clock. The completed device is shown on Fig. 17 and Fig. 18. The case is shown on Fig. 19 and Fig. 20.

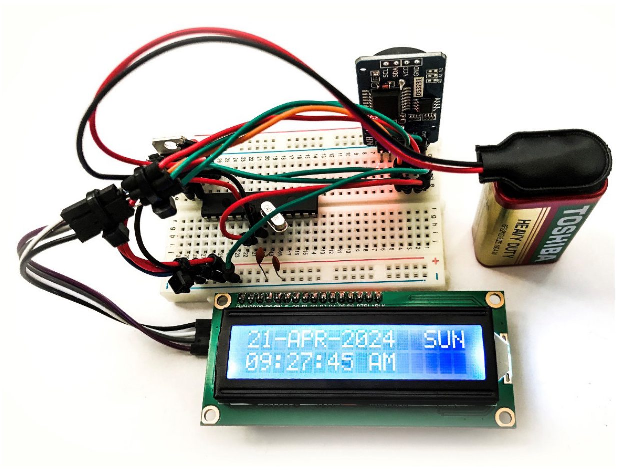

Fig. 15. Breadboard verification.

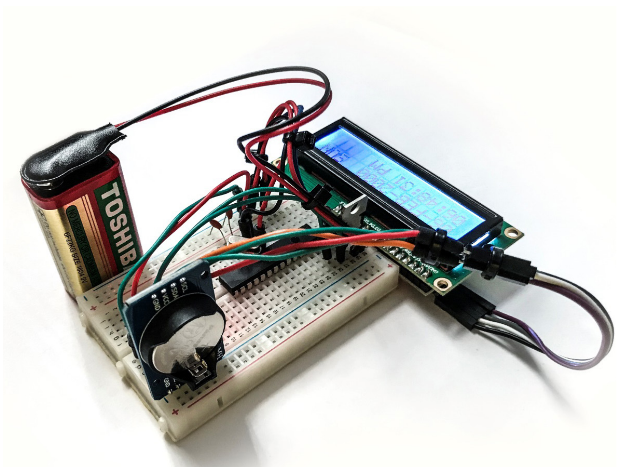

Fig. 16. Breadboard verification.

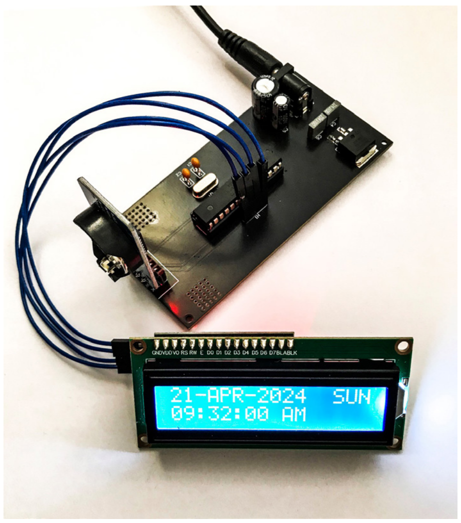

Fig. 17. Completed device.

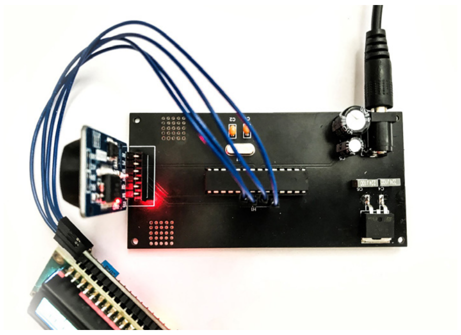

Fig. 18. Complete device.

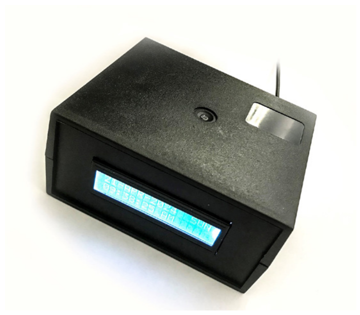

Fig. 19. Complete device.

Fig. 20. Complete device.

CONCLUSIONS

A digital clock has been developed using the Atmega328P microcontroller. The device can be integrated into various household and hobby electronics. It has been tested in different simulation environments as well as on a breadboard, proving its functionality. In the future, the project can be further developed in various directions to enhance its functionality. Different sensors, such as humidity and temperature sensors, or a buzzer for implementing an alarm, can be added to the clock. The device can be used independently or as part of a larger system. This project is highly useful and can find widespread application in laptops, phones, modern smart air conditioners, stoves, washing machines, and many other devices.

REFERENCES

1. S. Enkov, Programming in the Arduino environment - a practical guide, "Paisiy Hilendarski" University Publishing House, Plovdiv, ISBN 978-619-202-261-7, 2017.
2. D. Petkov, Microcontrollers - Architecture and principle of operation, Progstarter, Sofia, 2015.
3. What is an Arduino? Sparkfun. https://learn.sparkfun.com/tutorials/what-is-an-arduino/all, Available on 29.07.2024.
4. H. Barman, Differences between a Microprocessor and a Microcontroller, Centre for Management Studies, Dibrugarh University (CMSDU), 1-6, 2021.
5. N. Stefanov, Guide to the design of power supply devices, Sofia, Technika, 1988.
6. Schematic Capture Features. https://www.labcenter.com/schematic/, Available on 11.08.2024.