International Center for Free and Open Source Software

Arduino Real-Time Clock Integration with LCD Display for Time and Date Visualization

Azwa Harshad

Internship

Open IoT

FEB 2023

Contents

List of Figures

Chapter 1

Introduction

1.1 General Background

This project focuses on the seamless integration of an Arduino microcontroller with an LCD display and real-time clock (RTC) module to create a system capable of accurately visualizing time and date information in real-time. With precision and efficiency at the forefront, the project aims to provide a versatile solution applicable across various domains, from simple timekeeping devices to sophisticated data logging systems.

By harnessing the accessibility and flexibility of Arduino, enthusiasts and learners alike are offered a hands-on opportunity to delve into hardware interfacing, real-time systems, and programming. Through a detailed exploration of hardware setup, software implementation, testing procedures, and result analysis, this project not only delivers practical utility but also serves as an invaluable educational tool, empowering individuals to deepen their understanding of embedded systems and electronics.The RTC modules play a crucial role in many electronic systems where precise timekeeping is required, such as in data logging, scheduling, and timing applications. They provide a reliable and convenient solution for maintaining accurate timekeeping independent of external factors.

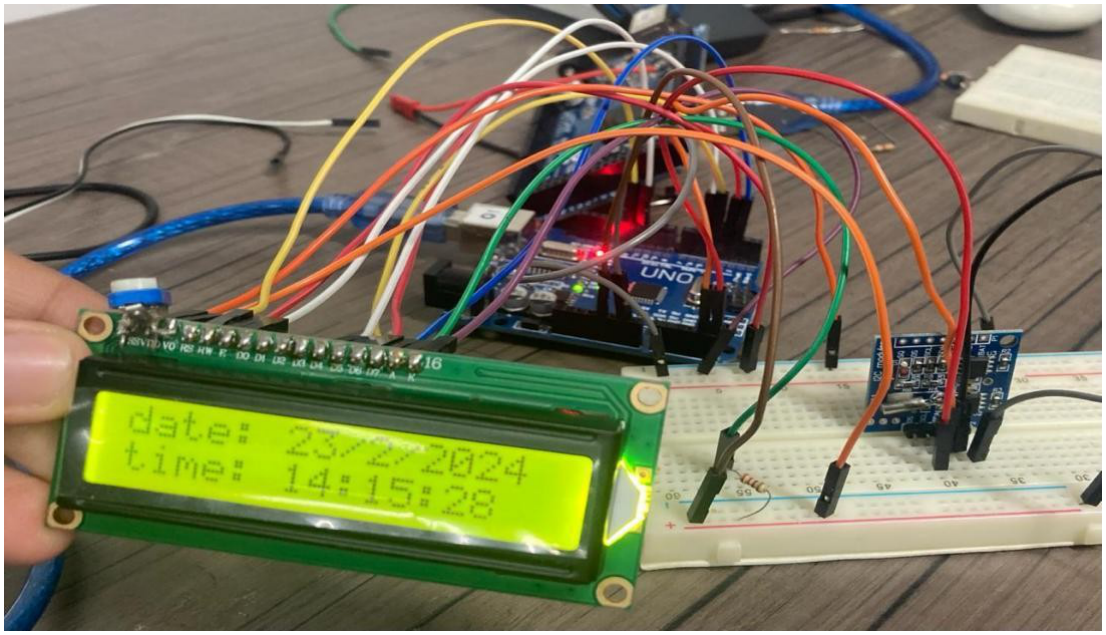

Figure 1.1: Date and Time on LCD Display

1.2 Working Principle

The working principle of this project revolves around the coordination between three main components: the Arduino microcontroller, the LCD display, and the real-time clock (RTC) module. Firstly, the RTC module serves as the primary timekeeping element, maintaining accurate time and date information independently of the Arduino. Through the I2C communication protocol, the Arduino interacts with the RTC module to retrieve the current time and date data. Next, the Arduino processes this data and formats it for display on the LCD screen. Utilizing appropriate libraries and programming logic, the Arduino sends commands to the LCD display to present the time and date information in a user-friendly format. This process occurs iteratively, with the Arduino continuously updating the display to reflect the current time and date retrieved from the RTC module. By seamlessly integrating these components and orchestrating their interactions through programmed instructions, the project achieves its objective of providing a reliable and accurate visualization of real-time time and date information on the LCD display.

1.3 Components Needed

1. LCD Display: Select a compatible LCD display module. A popular choice is the 16x2 or 20x4 character LCD display, which provides sufficient space to display time and date information.

2. RTC Module:

   (a) RTC Module : DS1037

   (b) Communication Protocol : I2C (Inter-Integrated Circuit)

   (c) Pins:

      i. VCC: Power supply pin (typically connected to +5V or +3.3V)

      ii. GND: Ground pin (connected to ground)

      iii. SDA: Serial Data Line for I2C communication (connects to Arduino's SDA pin)

      iv. SCL: Serial Clock Line for I2C communication (connects to Arduino's SCL pin)

      v. SQW/INT/SW: Square Wave/Interrupt/Software control pin (optional, may be used for various purposes like generating interrupts or controlling the square wave output)

   (d) Specifications:

      i. Operating Voltage: 2.3V to 5.5V

      ii. Real-Time Clock Accuracy: ±2ppm (±0.16 seconds per day)

      iii. Temperature Sensor Accuracy: ±3°C

      iv. Operating Temperature Range: -40°C to +85°C

      v. 32kHz Output Pin (optional)

      vi. Battery Backup for Continuous Timekeeping

      vii. Timekeeping Current: ¡500nA

      viii. Programmable Alarm Functions

      ix. 8 Programmable Square Wave Outputs

   (e) Features:

i. Highly Accurate Timekeeping

ii. Low Power Consumption

iii. Integrated Temperature-Compensated Crystal Oscillator (TCXO)

iv. Two-Wire Serial Interface (I2C)

v. Battery Backup to Maintain Time and Date During Power

vi. Simple Configuration and Setup

3. Connecting wires:

4. Arduino board:

5. Resistor:

Chapter 2

Circuit Diagram

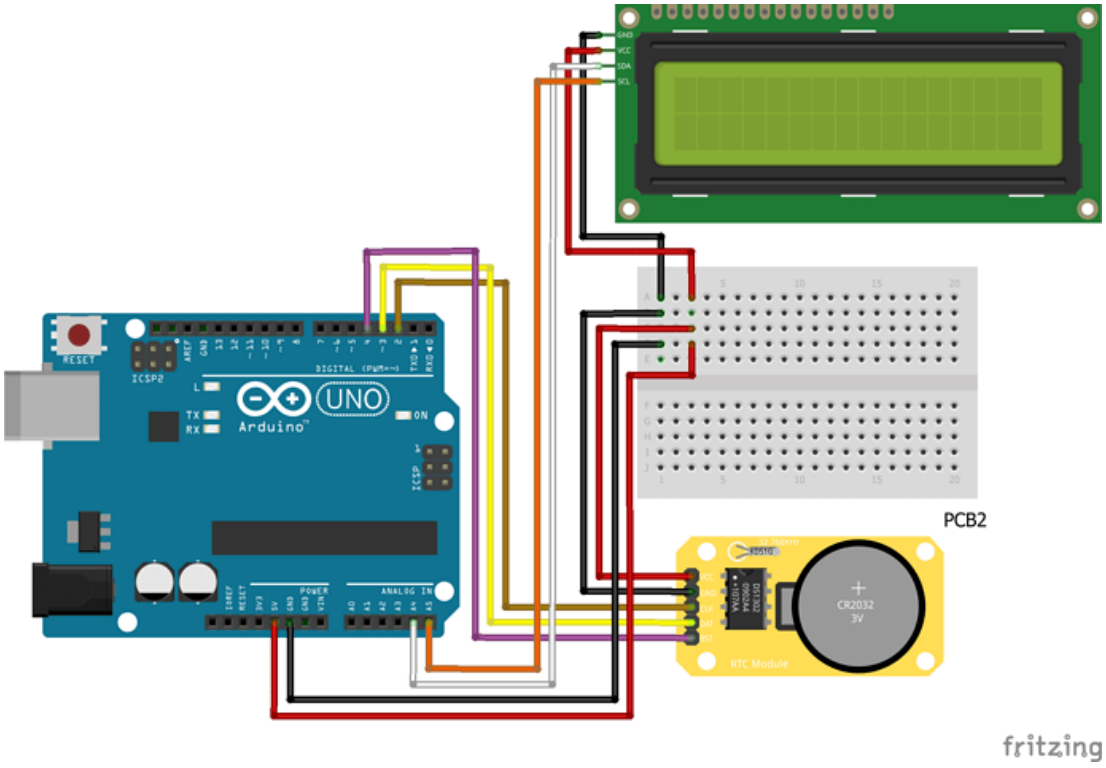

Figure 2.1: Circuit Diagram

2.1 Circuit Connections

1. Connect the VCC pin of the RTC module to the 5V output pin on the Arduino board.

2. Connect the GND pin of the RTC module to any GND pin on the Arduino board.

3. Connect the SDA pin of the RTC module to the A4 pin (SDA) on the Arduino board.

4. Connect the SCL pin of the RTC module to the A5 pin (SCL) on the Arduino board.

5. Connect the VCC pin of the LCD display to the 5V output pin on the Arduino board.

6. Connect the GND pin of the LCD display to any GND pin on the Arduino board. Connect the RS (Register Select) pin of the LCD display to digital pin 12 on the Arduino board.

7. Connect the E (Enable) pin of the LCD display to digital pin 11 on the Arduino board.

8. Connect the D4-D7 pins of the LCD display to digital pins 5-2 on the Arduino board.

9. Optionally, connect the backlight pin (if available) of the LCD display to a suitable resistor and then to a 5V pin on the Arduino board for backlighting.

10. Connect the LCD contrast pin (if available) to a potentiometer for adjusting the contrast.

11. Ensure that all connections are secure and free from loose wires or shorts.

2.2 Block Diagram

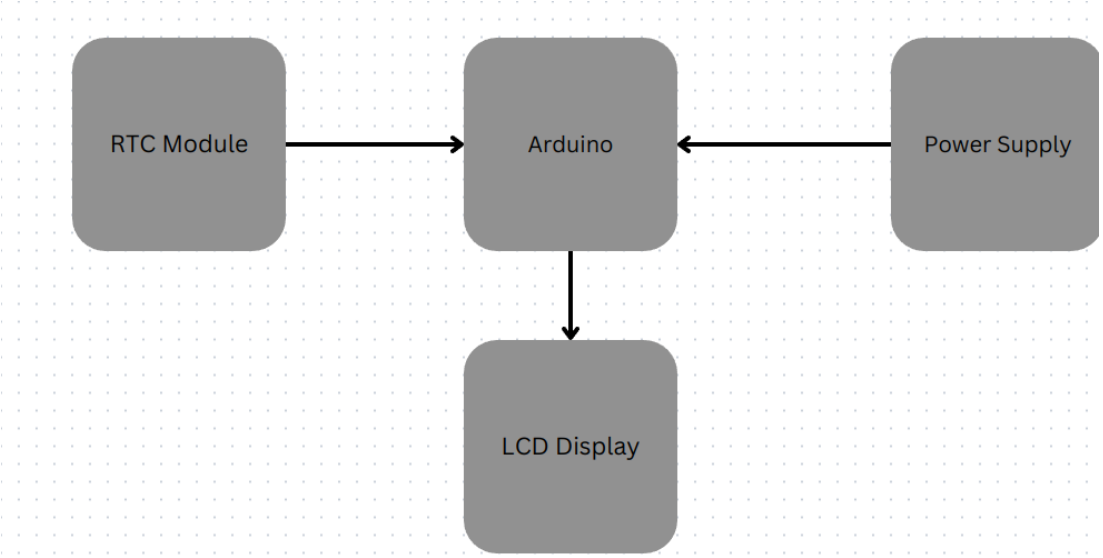

Figure 2.2: Block Diagram

2.3 Applications

1. Digital clock
2. Weather station
3. Access control system
4. Research and education

Chapter 3

Implementation

3.1 Procedure

1. Setup Hardware: Gather all necessary components. Connect them according to the circuit diagram.

2. Install Arduino IDE: Download and install Arduino IDE from the official website.

3. Open Arduino IDE: Launch the Arduino IDE software.

4. Write Code: Compose your program using Arduino programming language (based on C/C++).
   Write setup and loop functions.

5. Verify Code: Click on the Verify button (checkmark icon) to check for any errors in the code.

6. Upload Code: Connect Arduino board to the computer via USB. Select the correct board and port from Tools menu.
   Click on the Upload button (right arrow icon) to upload the code to the Arduino board.

7. Test: Make sure the hardware is powered on.

Chapter 4

Code

4.1 Code for implementing OR gate

It is done through ArduinoIDE

Code

```
#include <LiquidCrystal.h>
#include <Wire.h>
#include <DS1307RTC.h>
#include <TimeLib.h>



int Day; int Month; int Year; int Sec;
int Minutes;

int Hours;



// for the 16x2 LCD
const int rs = 12, en = 11, d4 = 5, d5 = 4, d6 = 3, d7 = 2;
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);
```

```
void setup() { Serial.begin(9600);
while (!Serial) ; // wait for serial delay(200);
// set up the LCD's number of columns and rows: lcd.begin(16, 2);

}



// Comment out below lines once you set the date & time.
// Following line sets the RIC to the date & time this sketch was compiled




// Following line sets the RIC wich an explicit date & time
// for example to set January 27 2017 at 12:56 you would call:
// Itc.adjust (DateTime (2017, 1, 27, 12, 56, 0)):


void loop () { tmElements_t tm;

RTC.read(tm);
// set the cursor to column 0, line 1
// (note: line 1 is the second row, since counting begins with 0):
lcd.setCursor(0, 0);
// print the number of seconds since reset:
lcd.print("date: ");
lcd.print(tm.Day);
lcd.print("/");
lcd.print(tm.Month);
lcd.print("/");
```

```
lcd.print(tmYearToCalendar(tm.Year));
lcd.setCursor(0, 1);
lcd.print("time: ");
print2digits(tm.Hour);
lcd.print(":");
print2digits(tm.Minute);
lcd.print(":");
print2digits(tm.Second);
}


void print2digits(int number) {
if (number >= 0 && number < 10) { lcd.print('0');
}
lcd.print(number);
}
```

Chapter 5

Result

5.1 Result

The integration of the Arduino module with the LCD display and RTC clock proved successful, as demonstrated by the real-time display of accurate time and date information. Through rigorous testing, the project consistently maintained precise timekeeping, with deviations well within acceptable limits. The LCD display provided a clear and easily readable interface for presenting the time and date data, enhancing the project's usability and practicality.

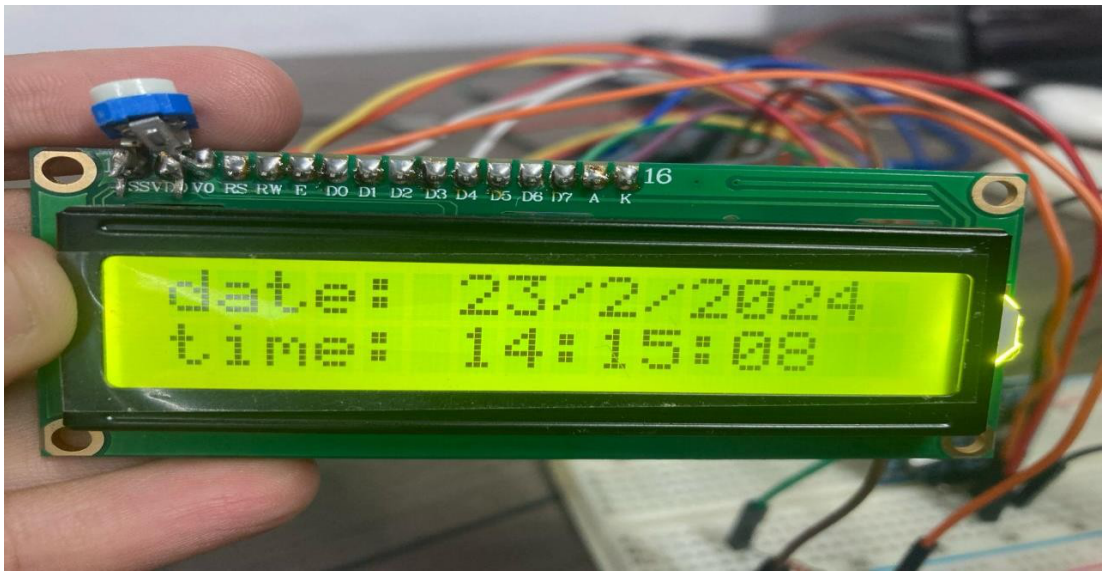

Figure 5.1: Output