TRƯỜNG ĐẠI HỌC NHA TRANG **KHOA CÔNG NGHỆ THÔNG TIN**



**THIẾT KẾ ĐỒNG HỒ THỜI GIAN THỰC SỬ DỤNG MODULE DS1307**

**GVHD : Ths. ĐOÀN VŨ THỊNH**

**SVTH : Nguyễn Thành Huynh\_59131001**

**Nguyễn Ngọc Hoàng\_59130800**

**Lớp : 59.CNTT-1**

Khánh Hòa, tháng 01 năm 2020

MỤC LỤC

[TÓM TẮT 3](#_Toc30164835)

[1. GIỚI THIỆU 4](#_Toc30164836)

[1.1. Màn hình LCD 4](#_Toc30164839)

[1.2. Module LCD I2C 6](#_Toc30164840)

[1.3. Module thời gian thực DS1307 7](#_Toc30164841)

[1.4. Board mạch Arduino UNO R3 11](#_Toc30164843)

[1.5. IC lập trình được Atmega16u2 13](#_Toc30164844)

[1.6. Phần mềm Fritzing 14](#_Toc30164845)

[1.7. Phần mềm Arduino IDE 15](#_Toc30164846)

[2. PHƯƠNG PHÁP NGHIÊN CỨU 16](#_Toc30164847)

[2.1. Mô phỏng nguyên lí hoạt động trên phần mềm Fritzing 16](#_Toc30164848)

[2.2. Lập trình 18](#_Toc30164849)

[2.2.1. Giao tiếp RTC với Arduino 19](#_Toc30164850)

[2.2.2. Hiển thị thời gian lên màn hình LCD 19](#_Toc30164851)

[3. KẾT QUẢ 21](#_Toc30164852)

[THẢO LUẬN 22](#_Toc30164853)

[PHỤ LỤC 23](#_Toc30164854)

# TÓM TẮT

Đồng hồ là một công cụ tiện lợi giúp đo đạc những mốc thời gian hay để theo dõi được ngày, tháng, năm, năm âm lịch, năm dương lịch và sắp xếp thời gian biểu một cách hợp lí. Yêu cầu của bài toán là thiết kế, thi công, lắp đặt sản phẩm đồng hồ thời gian thực hiển thị các thông số bao gồm: Thứ, ngày, tháng, năm giờ, phút, giây. Thời gian tự động cập nhật ngay cả khi không có nguồn điện cung cấp. Để thực hiện các công việc này cần sử dụng đến các thành phần, linh kiện điện tử như: module lập trình Arduino UNO R3, module thời gian thực DS1307, màn hình LCD1602, module chuyển đổi I2C cho LCD, các thành phần khác bổ trợ khác như điện trở, nguồn cung cấp.

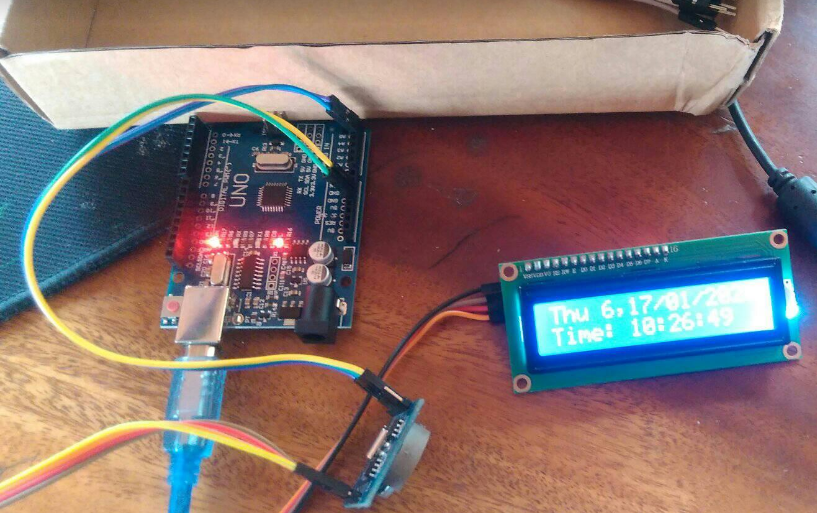
Quá trình thiết kế bắt đầu với việc sử dụng phần mềm Fritzing 0.9.3b để xây dựng sơ đồ nguyên lý và tiến hành mô phỏng trạng thái hoạt động của hệ thống. Thuật toán được lập trình trên phần mềm Arduino IDE ver1.8.10.

Kết quả thực hiện cho thấy sản phẩm có thể hiển thị thời gian với các thông số như yêu cầu đặt ra, thời gian được cập nhật sau mỗi giây. Toàn bộ sản phẩm được đóng gói, mã nguồn được trình bày trong phụ lục và được upload theo địa chỉ: https://github.com/thinhdoanvu/ThucTapTHCS\_2019/tree/master/NguyenNgocHoang\_NguyenThanhHuynh.

# 1. GIỚI THIỆU

Đồng hồ là một công cụ tiện lợi giúp đo đạc những mốc thời gian hay để theo dõi được ngày, tháng, năm, năm âm lịch, năm dương lịch và sắp xếp thời gian biểu một cách hợp lí. Những loại đồng hồ dùng trong kĩ thuật thường có độ chính xác cao và cấu tạo rất phức tạp. Trong khi đó, người ta có thể tạo ra những đồng hồ nhỏ gọn để dễ dàng mang theo bên mình.

Hình bên dưới là một ví dụ minh họa việc hiển thị thời gian thực, nhiệt độ và độ ẩm lên màn hình LCD 1602. Thành phần cơ bản bao gồm bộ hiển thị cho biết thời gian như: ngày, tháng, năm, giờ, phút, giây và còn hiển thị nhiệt độ và độ ẩm lên LCD. Để có thể tạo ra một đồng hồ như vậy cần vận dụng các kiến thức về lập trình thiết bị nhúng, cụ thể là KIT Arduino, kiến thức về điện tử số và điện tử tương tự để tính toán các thành phần điện tử cơ bản và thực hiện các thao tác ghép nối các thành phần điện tử với nhau. Ngoài ra, còn một thành phần không thể thiếu đó là module thời gian thực DS1307 (<https://www.electroschematics.com/ds1307-datasheet/>) được dùng để đọc ghi thời gian hiện hành và hiển thị lên màn hình LCD.



Hình 1.1. Đồng hồ hiển thị thời gian thực trên LCD

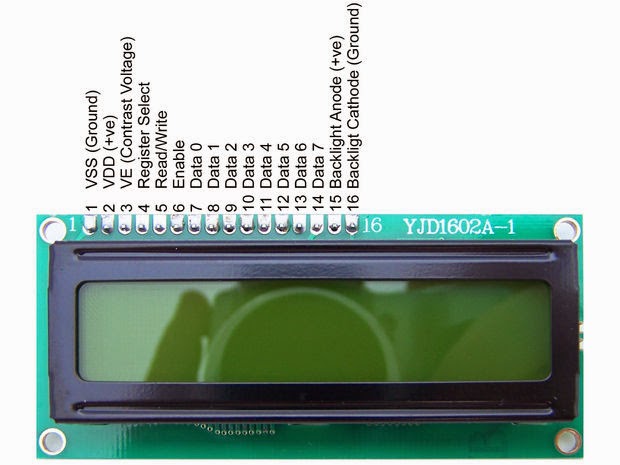
## 1.1. Màn hình LCD

Ngày nay, thiết bị hiển thị LCD 1602 được sử dụng trong rất nhiều ứng dụng của vi điều khiển với nhiều ưu điểm so với các dạng hiển thị khác như: khả năng hiển thị kí tự đa dạng, dễ dàng đưa vào mạch ứng dụng theo nhiều giao thức khác giao tiếp khác nhau, tiêu tốn ít tài nguyên hệ thống và dễ sử dụng…

Màn hình LCD1602 xanh lá sử dụng driver HD44780, có thể hiển thị 2 dòng với 16 kí tự hay 32 kí tự cùng một lúc. Màn hình có độ bền cao, được dùng phổ biến, nhiều code mẫu, dễ sử dụng, thích hợp với những người mới học và làm dự án.

Ưu điểm: Hiển thị dễ dàng, có thể linh động trong việc hiển thị thời gian, kết nối đơn giản, mạch điện không phức tạp.

Khuyết điểm: Không thu hút được sự chú ý tốt như led 7 đoạn, giá thành cao.



Hình 1.2. Màn hình LCD 1602

(Nguồn: <http://arduino.vn/bai-viet/531-dieu-khien-lcd-bang-arduino-uno>)

**Cấu tạo:** Màn hình LCD1602 gồm 16 chân (hình 1.2) bao gồm:

VSS: Chân đất (0V)

VDD: Cấp nguồn 5V cho LCD

VEE: Điều khiển độ tương phản của LCD

RS (Register Select): Dùng để chọn thanh ghi, nếu giá RS=0 thì bus DB0 - DB7 sẽ nối với thanh lệnh IR của LCD (ở chế độ ghi) hoặc nối với bộ đếm địa chỉ của LCD (ở chế độ đọc), ngươc lại nếu giá trị là 1 thì bus DB0 – DB7 sẽ nối với thanh dữ liệu DR bên trong LCD.

R/W (Read/Write): Dùng để chọn chế độ đọc hoặc ghi dữ liệu, nếu giá trị R/W = 0 thì sẽ chọn chế độ ghi và ngược lại là 1 để chọn chế độ đọc dữ liệu.

ENABLE: Cho phép ghi vào LCD

D0 - D7: 8 chân trao đổi dữ liệu với các vi điều khiển, với 2 chế độ sử dụng

+ Chế độ 8 bit: Dữ liệu được truyền trên cả 8 đường, với bit MSB là bitDB7.

+ Chế độ 4 bit: Dữ liệu được truyền trên 4 đường từ DB4 tới DB7, bit MSB là DB7.

Backlight (Backlight Anode (+) và Backlight Cathode (-)): dùng để tắt/bật đèn màn hình LCD.

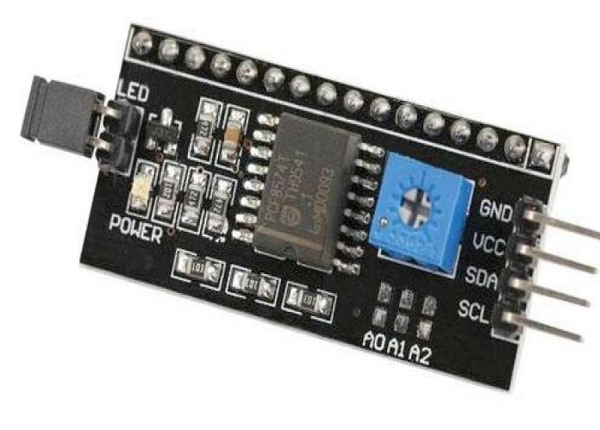
|  |  |
| --- | --- |
| Điện áp hoạt động | 5V |
| Điện áp MAX | 7V |
| Điện áp MIN | -0.3V |
| Hoạt động ổn định | 2.7 – 5.5V |
| Điện áp ra mức cao | >2.4 |
| Điện áp ra mức thấp | < 0.4V |
| Dòng điện cáp nguồn | 350uA - 600uA |
| Nhiệt độ hoạt động | -30 – 75 độ C |

Bảng 1.1. Thông số kĩ thuật của màn hình LCD 1602

## 1.2. Module LCD I2C

Thông thường, để sử dụng màn hình LCD sẽ mất rất nhiều chân trên Arduino để điều khiển. Do vậy, để đơn giản hóa việc này, một loại mạch điều khiển màn hình LCD sử dụng giao tiếp I2C được tạo ra. Thay vì sử dụng tối thiểu 6 chân của vi điều khiển để kết nối với LCD (RS, EN, D7, D6, D5 và D4) thì với module chuyển đổi chỉ cần sử dụng 2 chân (SCL, SDA) để kết nối.

Module chuyển đổi I2C hỗ trợ cho các loại LCD sử dụng driver HD44780 (ví dụ: LCD 1602, LCD 2004,…), kết nối với vi điều khiển thông qua giao tiếp I2C, tương thích với hầu hết các vi điều khiển hiện nay.



Hình 1.3. Module LCD I2C

Nguồn: <https://www.amazon.in/>

Cấu tạo: Module này gồm có 4 chân và các chân của nó được mô tả như sau:

Chân GND: nối đất 0V

Chân VCC: cấp nguồn 5V cho giao tiếp I2C.

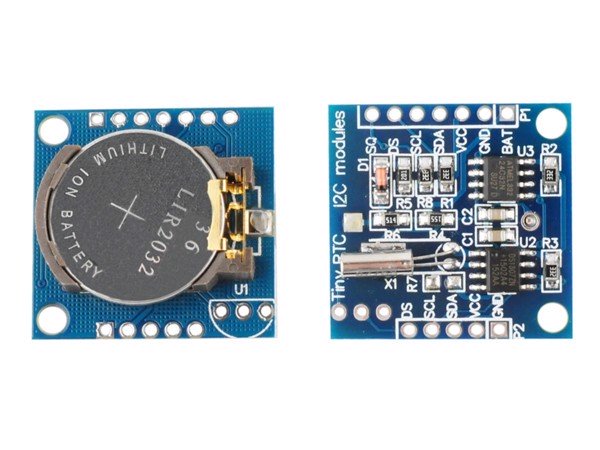
SCL và SDA, trong đó SCL (serial clock) và SDA (serial data) là hai bus dữ liệu của DS1307. Thông tin truyền và ghi đều được truyền qua 2 đường truyền này theo chuẩn I2C.

Ngoài ra trên module chuyển đổi I2C còn có các chân địa chỉ (A0 – A2) để Jumper chọn địa chỉ từ A0 - A2 của màn hình LCD. Nếu Jumper ngắt kết nối thì địa chỉ logic là 1, nếu được kết nối sẽ trở thành 0.

## 1.3. Module thời gian thực DS1307

Tiếp theo, để có thể cấp dữ liệu ngày giờ cho LCD thì ta cần phải dùng đến module thời gian thực DS1307.

Module thời gian thực DS1307 là chip thời gian thực hay RTC (Read Time Clock). Đây là một IC tích hợp cho thời gian bởi vì tính chính xác về thời gian tuyệt đối cho thời gian: Giờ, phút, giây, thứ, ngày, tháng, năm. Module DS1307 được chế tạo bởi Dallas. Chip này có 7 thanh ghi 8 bit mỗi thanh ghi này chứa: Thứ, ngày, tháng, năm, giờ, phút, giây. Ngoài ra DS1307 còn chứa 1 thanh ghi điều khiển ngõ ra phụ và 56 thanh ghi trống các thanh ghi này có thể dùng như là RAM. DS1307 được đọc thông qua chuẩn truyền thông I2C nên do đó để đọc được và ghi từ DS1307 thông qua chuẩn truyền thông này. Do nó được giao tiếp chuẩn I2C nên cấu tạo bên ngoài nó rất đơn giản.



Hình 1.4. Module thời gian thực RTC DS1307

(Nguồn: <https://www.indiamart.com/proddetail/ds1307-i2c-real-time-clock-module-19179517430.html>)

**Cấu tạo:**

X1 và X2: là đầu vào dao động cho DS1307, được nối với thạch anh có tần số 32.768kHz.

Vbat là nguồn nuôi cho chip. Nguồn này có điện áp từ 2V- 3.5V ta. Đây là nguồn cho chip hoạt động liên tục khi không có nguồn VCC

Chân VCC: là nguồn cung cấp cho giao tiếp I2C, điện áp cung cấp là 5V chuẩn và được dùng chung với vi xử lý. Nếu mất nguồn VCC mà Vbat vẫn có nguồn thì module DS1307 vẫn hoạt động bình thường nhưng mà không ghi và đọc được dữ liệu.

Chân GND: là nguồn đất (Mass) chung cho cả VCC và Vbat.

Chân SQW/OUT (square wave/output driver):là một ngõ ra phụ tạo xung dao động (xung vuông), được sử dụng để tạo xung, thiết lập bộ đếm cho các thiết bị khác. Khi được kích hoạt thì bit SQWE được thiết lập 1 chân SQW/OUT phát đi 1 trong 4 tần số (1Hz, 4kHz, 8kHz, 32kHz). Chân này sẽ hoạt động khi cả VCC và Vbat được cấp.

Chân SCL (Serial Clock): được sử dụng để đồng bộ sự chuyển dữ liệu trên đường dây nối tiếp.

Chân SDA (Serial Data): là chân vào ra cho 2 đường dây nối tiếp, được thiết kế theo kiểu cực máng hở, đòi hỏi phải có 1 điện trở kéo trong khi hoạt động.

### Bảng 1.2. Địa chỉ thanh ghi giá trị thời gian của DS1307

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Địa chỉ** | **BIT 7** | **BIT 6** | **BIT 5** | **BIT 4** | **BIT 3** | **BIT 2** | **BIT 1** | **BIT 0** | **Hàm** | **Phạm vi** |
| **00h** | CH | 10 Seconds | | | Seconds | | | | Seconds | 00-59 |
| **01h** | 0 | 10 Minutes | | | Minutes | | | | Minutes | 00-59 |
| **02h** | 0 | 12 | 10 Hour | 10  Hour | Hour | | | | Hour | 1-12 +AM/PM  00-23 |
| 24 | PM/AM |
| **03h** | 0 | 0 | 0 | 0 | 0 | DAY | | | Day | 01-07 |
| **04h** | 0 | 0 | 10 Date | | Date | | | |  | 01-31 |
| **05h** | 0 | 0 | 0 | 10 Month | Month | | | | Month | 01-12 |
| **06h** | 10 Year | | | | Year | | | | Year | 00-99 |
| **07h** | OUT | 0 | 0 | SQWE | 0 | 0 | RS1 | RS0 | Control |  |
| **08h-3Fh** |  | | | | | | | | RAM  56 x 8 |  |

Để hiểu rõ hơn về module DS1307 thì trước tiên phải tìm hiểu về giao tiếp I2C.

I2C (Inter-Integrated Circuit) là một loại bus nối tiếp được phát triển bởi hãng sản xuất linh kiện điện tử Philips. I2C sử dụng hai đường truyền tín hiệu: một đường xung nhịp đồng hồ (SCL) và một đường dữ liệu (SDA). SCL và SDA luôn được kéo lên nguồn bằng một điện trở kéo lên có giá trị xấp xỉ 4,7 KOhm. Các chế độ hoạt động của I2C bao gồm:

+ Chế độ chuẩn (Standard Mode) hoạt động ở tốc độ 100 Kbit/s.

+ Chế độ tốc độ thấp (Low-Speed Mode) hoạt động ở tốc độ 10 Kbit/s.

Tần số xung nhịp đồng hồ có thể xuống 0 Hz. I2C sử dụng 7 bit để định địa chỉ, do đó trên một bus có thể định địa chỉ tới 112 nút, 16 địa chỉ còn lại được sử dụng vào mục đích riêng. Điểm mạnh của I2C chính là hiệu suất và sự đơn giản của nó: một khối điều khiển trung tâm có thể điều khiển cả một mạng thiết bị mà chỉ cần hai lối ra điều khiển. Ban đầu, loại bus này chỉ được dùng trong các linh kiện điện tử của Philips. Sau đó, do tính ưu việt và đơn giản của nó, I2C đã được chuẩn hóa và được dùng rộng rãi trong các module truyền thông nối tiếp của vi mạch tích hợp ngày nay. Vì Arduino đã có hỗ trợ sẵn điện trở kéo lên nên các bạn chỉ cần kết nối vào cách chân SDA và SCL.

Có thể bắt gặp những module sử dụng giao tiếp I2C như LCD, DS1307, MPU6050… Như các bạn có thể thấy thì trong hình Arduino có SDA SCL tại chân A4/A5 và kế AREF cũng có 2 chân.

Module này có 1 viên pin để giữa cho việc cập nhật thời gian không bị dừng. để thời gian được cập nhật liên tục như thế thì module RTC có việc hỗ trợ cập nhật thời gian là linh kiện Thạch anh có trên DS1307.

Thạch anh là linh kiện điện tử tạo giao động ổn định cho bộ điều khiển. Ứng dụng của thạch anh trong điện tử đa phần để tạo ra tần số được ổn định vì tần số của thạch anh tạo ra rất ít bị ảnh hưởng bởi nhiệt độ hơn là các mạch dao động RC…. Vì vậy để tăng thời gian liên tục cần phải có tần số tạo giao động thời gian sẽ được tăng lên.

Module DS1307 Mạch sử dụng 1 viên pin CMOS có thể sử dụng đến 10 năm để chạy thời gian khi máy đã tắt nguồn.

Để có thể sử dụng giao tiếp I2C, chúng ta dùng thư viện “*Wire.h”*và không cần cài đặt vì đã có sẵn trong Arduino IDE.

Trong bộ nhớ của DS1307 có tất cả 64 thanh ghi địa chỉ từ 0 đến 63 và được bắt đầu từ 0x00 đến 0x3F nhưng trong đó chỉ có 8 thanh ghi đầu là thanh ghi thời gian thực nên chúng ta sẽ đi sâu vào 8 thanh ghi ( chức năng và địa chỉ thanh ghi thời gian thực này). Nhìn vào (bảng 1.5) ở phần giới thiệu trong datasheet ta sẽ thấy như sau: Giây, phút, giờ, thứ, ngày, tháng, năm và bắt đầu từ thanh ghi giây (00h) và kết thúc bằng thanh ghi năm (06h). Riêng thanh ghi Control dùng để điều khiển ngõ ra của chân SQW/OUT nên trong thực tế nên không mấy ai sử dụng thanh ghi này trong thời gian thực nên chúng ta bỏ qua thanh ghi này.

Trong quá trình ghi dữ liệu cho các thanh ghi thời gian thực chúng ta cũng phải chuyển đổi thành mã BCD tương ứng sau đó mới ghi vào cho DS1307.

Tiếp theo chân X1 và chân X2 của RTC sẽ được kết nối với thạch anh 32.768KHz để tạo dao động cho DS1307. Chân SDA có chắc năng đường dữ liệu trong chuẩn giao tiếp I2C, chân SLC có chắc năng đường xung nhịp trong chuẩn giao tiếp I2C, chân SQW(OUT) tạo xung vuông với tầng số có thể lập trình được.

Trong đó:

**OUT:** lựa chọn mức logic xuất ra tại chân **SQW/OUT** khi chức năng **SQW** không được kích hoạt (**SQW=0**).

**OUT** = 1: xuất ra mức logic 1​

**OUT** = 0: xuất ra mức logic 0​

**SQW**: = 1, cho phép tạo xung vuông ra trên chân **SQW/OUT**, tần số xung vuông được thiết lập bởi **RS0**, **RS1**:

Bảng 1.3. Bảng tầng số trạng thái chân RS1 và RS0

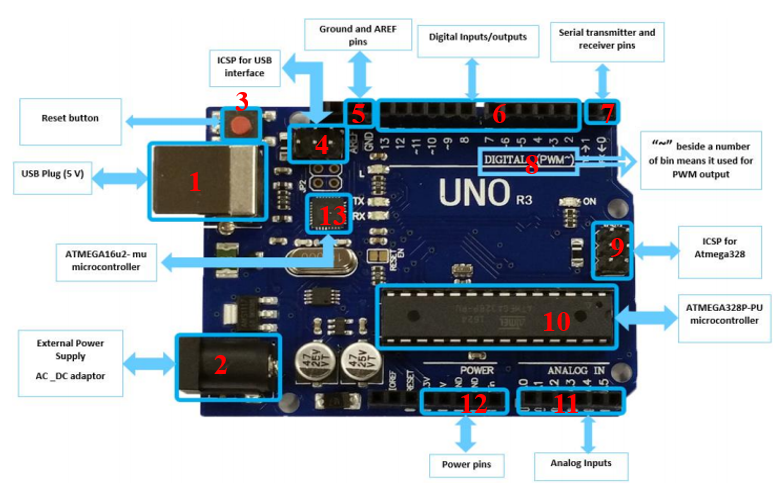
|  |  |  |
| --- | --- | --- |
| RS1 | RS0 | Tần số |
| 0 | 0 | 1Hz |
| 0 | 1 | 4.096KHz |
| 1 | 0 | 8.192KHz |
| 1 | 1 | 32,768KHz |

Bảng 1.4. Thanh ghi Control(0x07)

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | | OUT | 0 | 0 | SQW | 0 | 0 | RS1 | RS0 |   Bảng 1.8. Tên chức năng các thanh ghi thời gian của DS1307 |
| |  |  |  | | --- | --- | --- | | **Tên thanh ghi** | **Địa chỉ thanh ghi** | **Chức năng** | | **SECONDS** | 0x00 | **-Bit0** – Bit chứa giá trị giấy  **-Bit7** (CH):= 1 vô hiệu hóa giao động trong DS1307 | | **MINUTES** | 0x01 | Chứa giá trị phút | | **HOURS** | 0x02 | **-Bit0** – Bit4 chứa giá trị đơn vị của giờ  **-Bit6**=1: Chế độ 12h  =0: chế độ 24h  -Trong chế độ:  **24h: Bit5 – Bit4:** chứa giá trị chục của giờ  **12h:**  Bit4: chứa giá trị chục của giờ  Bit5:=0:AM  =1:PM | | **DAY** | 0x03 | Chứa giá trị thứ | | **DATE** | 0x04 | Chứa giá trị ngày | | **MONTH** | 0x05 | Chứa giá trị tháng | | **YEAR** | 0x06 | Chứa giá trị năm | |

## 1.4. Board mạch Arduino UNO R3

Để đọc dữ liệu từ module thời gian thực DS1307 và hiển thị lên màn hình LCD cần sử dụng Arduino Uno R3. Nó là một máy tính nhỏ để người dùng có thể lập trình và thực hiện các dự án điện tử mà không cần phải có các công cụ chuyên biệt để phục vụ việc nạp code.



Hình 1.5. Giới thiệu chi tiết về board mạch Arduino UNO R3

Nguồn: <http://www.fecegypt.com/uploads/dataSheet/1522237550_arduino%20uno%20r3.pdf>)

Arduino UNO sử dụng vi điều khiển họ 8bit AVR Hình (1.2.1) là ATmega8, gồm:

+14 chân digital I/O(0 - 13): dùng để đọc hoặc xuất tín hiệu với dòng vào/ra tối đa trên mỗi chân là 40mA.

+2 chân RXD và TXD: được dùng trong truyền thông nối tiếp. Khi 2 chân này cũng có thể được cấu hình làm đầu vào hoặc đầu ra tín hiệu số.

+Chân PWM - Pulse Width Modulation (3, 5, 6, 9, 10, và 11): có kí hiệu là ~ cho phép xuất ra chuỗi xung vuông với độ phân giải 8bit (0 → 28-1).

+Chân giao tiếp SPI – Serial Peripheral Interface (10 - SS), 11 - MOSI, 12 - MISO, 13 - SCK). Ngoài các chức năng thông thường, 4 chân này còn dùng để truyền phát dữ liệu bằng giao thức SPI với các thiết bị khác.

+LED 13 (ký hiệu L): được dùng để hiển thị trạng thái 1/HIGH/ON khi xuất tín hiệu ra chân 13 với giá trị là 1 và ngược lại là 0/LOW/OFF .

+6 chân tín hiệu đầu vào Analog (A0 → A5): cung cấp độ phân giải tín hiệu 10bit (0 → 210-1) để đọc giá trị điện áp trong khoảng 0V → 5V.

+AREF và GND: GND hay chân Mass hay chân đất dùng để cấp nguồn 0V cho các module khác có kết nối với Arduino. Trong khi chân AREF được dùng để phối hợp với các chân Analog Input (11) để điều chỉnh dải điện áp đầu vào cho ADC 10 bit.Chân A4 (SDA – Serial Data) và A5 (SCL-Serial Clock -Line) hỗ trợ giao tiếp I2C/TWI với các thiết bị khác.

+USB: Cổng giao tiếp USB có 2 chức năng: cấp nguồn cho board mạch và truyền thông nối tiếp với máy tính trong việc nạp chương trình hay giao tiếp nối tiếp.

+RESET: Đặt lại trạng thái ngay khi nạp chương trình.

+ISCP: Chân giao tiếp với USB, tín hiệu giao tiếp có thể giám sát từ đây.

+ICSP: Giao tiếp với Atmega328.

+IC lập trình ATMEGA16u2

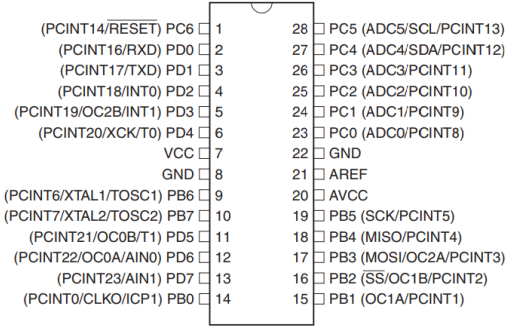
+POWER: Cung cấp nguồn 5V, GND cho các thành phần mở rộng.

Bảng 1.5. Thông số của Arduino UNO R3

|  |  |
| --- | --- |
| Vi điều khiển | ATmega328 họ 8bit |
| Điện áp hoạt động | 5V DC (chỉ được cấp qua cổng USB) |
| Tần số hoạt động | 16 MHz |
| Dòng tiêu thụ | khoảng 30mA |
| Điện áp vào khuyên dùng | 7-12V DC |
| Điện áp vào giới hạn | 6-20V DC |
| Số chân Digital I/O | 14 (6 chân hardware PWM |
| Số chân Analog | 6 (độ phân giải 10bit) |
| Dòng tối đa trên mỗi chân I/O | 30 mA |
| Dòng ra tối đa (5V) | 500 mA |
| Dòng ra tối đa (3.3V) | 50 mA |
| Bộ nhớ flash | 32 KB (ATmega328) với 0.5KB dùng bởi bootloader |
| SRAM | 2 KB (ATmega328) |
| EEPROM | 1 KB (ATmega328) |

## 1.5. IC lập trình được Atmega16u2

Arduino R3 không thể hoạt động mà không cần đến bộ não của board mạch này. Atmega16u2được sản xuất theo kiến trúc RISC (Reduce Instruction Set Computer)tiên tiến bao gồm: 125 tập lệnh đơn lệnh đơn chu kỳ; 32 thanh ghi 8 bit dùng chung; Tối đa 16 tập lệnh MIPS (Microprocessor without Interlocked Pipeline Stages)với tần số cho phép 16 MHz.Một bộđếm 8bit (Timer/Counters)và 1 bộ đếm 16 bit (Timer/Counter)dùng cho điều biến độ rộng xung PWM. Giao tiếp USART với chế độ SPI master vàchế độ kiểm soát phần cứng (RTS/CTS). Bộ chuyển đổi tín hiệu ADC, công cụ thời gian thực và khả năng hoạt động ở mức điện áp thấp trong chế độ tiết kiệm năng lượng.



Hình 1.6. Sơ đồ chân IC Atmega16u2

*(Nguồn:* [*http://www.fecegypt.com/uploads/dataSheet/1522237550\_arduino%20uno%20r3.pdf*](http://www.fecegypt.com/uploads/dataSheet/1522237550_arduino%20uno%20r3.pdf)*)*

**Cấu tạo:**

22 chân đa dụng vừa làm tín hiệu vào vừa làm tín hiệu ra tín hiệu số (PB0 - PB5 ~ chân 14 – chân 19, PC0 – PC6 ~ chân 23 – chân 28 và chân 1, PD0 – PD7 ~ chân 2 – chân 6 và chân 11 – chân 13).

Chân 8, 9: là 2 chân nối với bộ dao động thạch anh, tần số dao động 16MHz.

Chân 7, 8, 22: là 3 chân nguồn VCC và GND cấp nguồn 5V và Mass.

Chân 23 đến chân 28: là 6 chân ADC (Analog Digital Converter): bộ chuyển đổi ADC 10 bit phối hợp với chân 21(AREF) hiệu chỉnh giá trị điện áp đầu vào cho ADC.

## 1.6. Phần mềm Fritzing

Trước khi thi công mạch sản phẩm thực tế cần trải qua quá trình mô phỏng để kiểm tra thuật toán cũng như hoạt động logic của các linh kiện điện tử. Fritzing là phần mềm thiết kế mạch điện tử tự động giúp mô phỏng, thiết kế mạch nguyên lí, sơ đồ in,xây dựng trên test board, chạy mạch PCB, lập trình,...

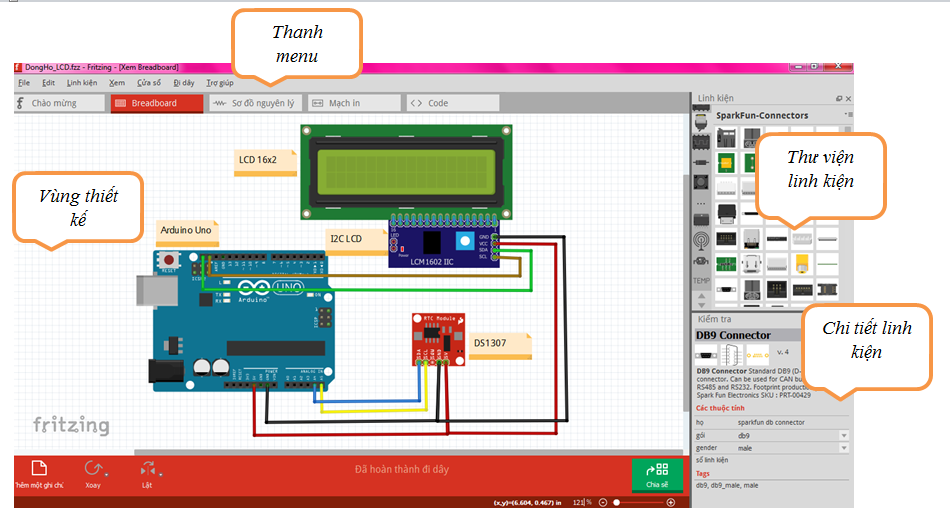
Với phần mềm Fritzing sẽ hỗ trợ những tính năng như sau:

+Sơ đồ cắm linh kiện trực tiếp dùng để kiểm tra mạch với Breadboard.

+Sơ đồ mạch nguyên lí với Schematic.

+Hỗ trợ chạy mạch PCB tự động.

+Hỗ trợ chạy code dùng cho sơ đồ mạch (nếu có lập trình).



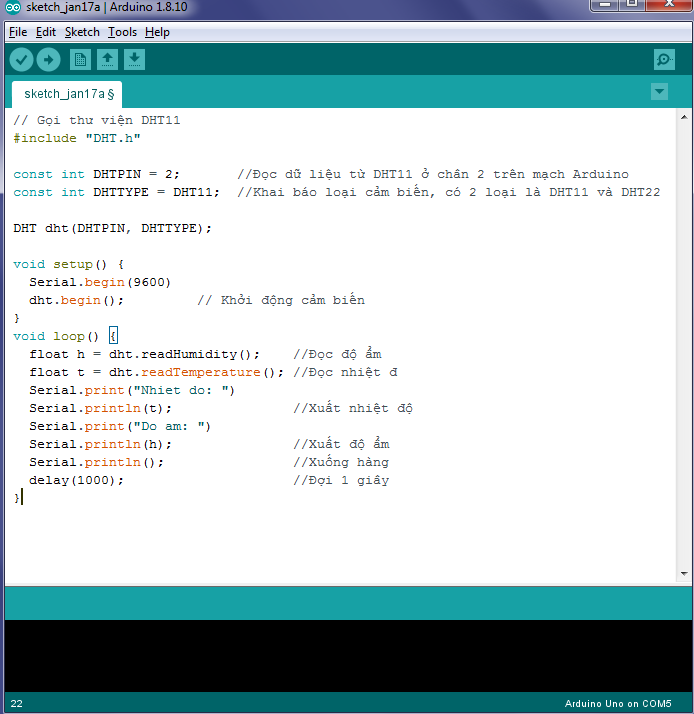
Hình 1.7. Phần mềm thiết kế mạch Fritzing

## 1.7. Phần mềm Arduino IDE

Để lập trình được cho các board mạch Arduino, các bạn cần phải có một công cụ hỗ trợ cho việc nạp code lên board mạch gọi là Arduino IDE(<https://www.arduino.cc/>).Công cụ này được đội ngũ kĩ sư của Arduino phát triển và có thể chạy trên Windows, MAC OS X và Linux.

Arduino IDE (Arduino Integrated Development Environment) là một chương trình phần mềm mã nguồn mở được viết bằng ngôn ngữ C/C++ có khả năng tương thích hầu hết với các board mạch Arduino. Chương trình viết trên IDE có thể là C hoặc C++, sau khi biên dịch được nạp trực tiếp lên board mạch lập trình thông qua cổng USB.

Hình 1.10 là ví dụ về giao diện của Arduino IDE được sử dụng để hiển thị nhiệt độ và độ ẩm lên màn hình Serial. Trong đó có 2 hàm chính là hàm setup dùng để khai báo và mở cổng hoạt động, hàm loop là hàm chạy chính thao tác các thuật toán và được chạy lặp lại liên tục cho đến khi mạch tắt nguồn.



Hình 1.8. Minh họa giao diện phần mềm Arduino

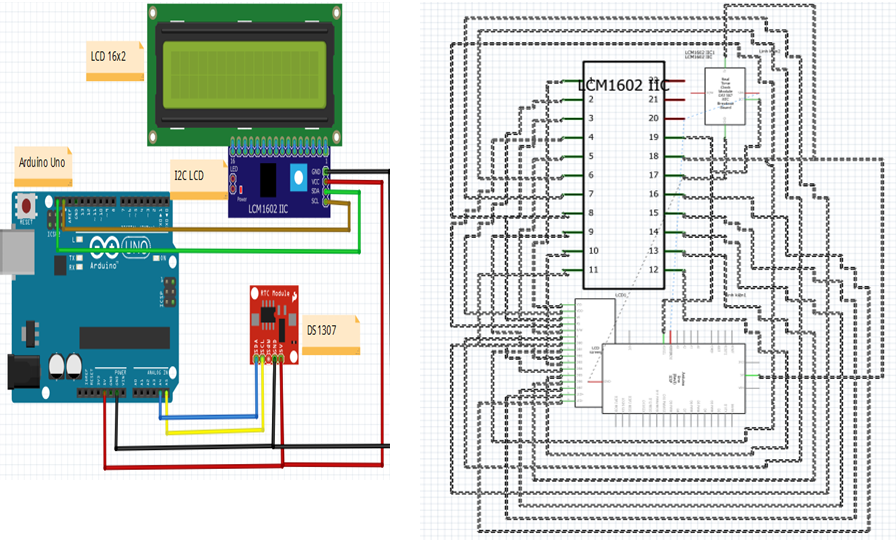
# 

# 2. PHƯƠNG PHÁP NGHIÊN CỨU

Như đã đề cập, đồng hồ hiển thị thời gian thực LCD bắt đầu từ những những linh kiện để hiển thị, kit Arduino, module cung cấp dữ liệu thời gian để mạch Arduino đọc dữ liệu. Tất cả các linh kiện và thiết bị cùng nhau phối hợp để tạo nên một sản phẩm hoàn chỉnh.

## 2.1. Mô phỏng nguyên lí hoạt động trên phần mềm Fritzing

Để tiến hành làm sản phẩm trước hết cần phải sử dụng đến một phần mềm hỗ trợ việc mô phỏng, giải thuật để giúp kiểm tra các chi tiết nhằm tránh những sai sót không đáng có. Đầu tiên ta sử dụng phần mềm fritzing để mô phỏng các linh kiện và cho mạch hoạt động.



Hình 2.1. Sơ đồ mô phỏng và nguyên lý cho đồng hồ thời gian thực

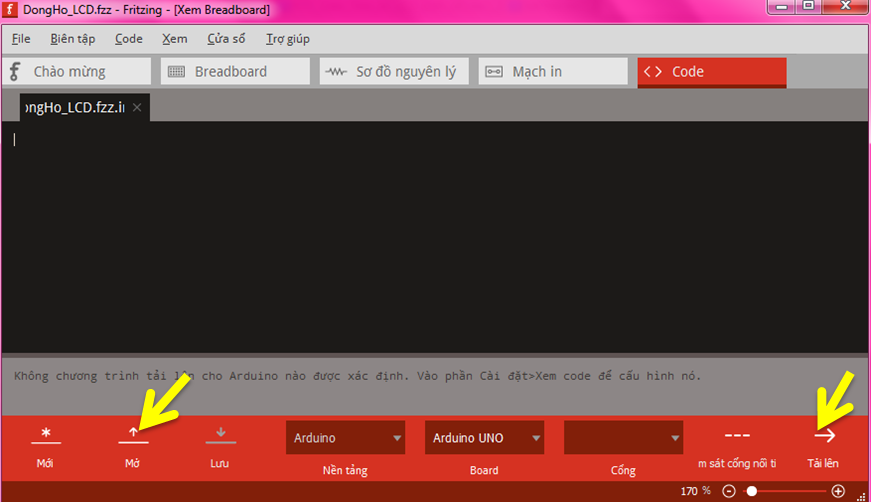
Trong hình 2.1, màn hình LCD được dùng để hiển thị các thành phần của thời gian như: thứ, ngày, tháng, năm, giờ, phút, giây. Mạch chuyển đổi I2C (kết nối trực tiếp với 16 chân của LCD) để tiết kiệm được các chân của mạch Arduino UNO R3. I2C có 2 chân SDA và SCL, trong đó SCL được sử dụng đồng bộ hóa việc truyền dữ liệu giữa Arduino UNO R3 với LCD 1302; chân SDA dùng cho việc truyền dữ liệu data. Module thời gian thực (khung màu đỏ) được dùng cho mô phỏng thời gian của đồng hồ. Board Arduino (màu xanh) là module dùng để lập trình mô phỏng.

. Module thời gian thực (khung màu đỏ) được dùng cho mô phỏng thời gian của đồng hồ. Board Arduino (màu xanh) là module dùng để lập trình mô phỏng. Các bước tiến hành nạp code như sau:

**Bước 1:** Hoàn thiện việc lấy linh kiện cần thiết và nối dây nối liên kết.

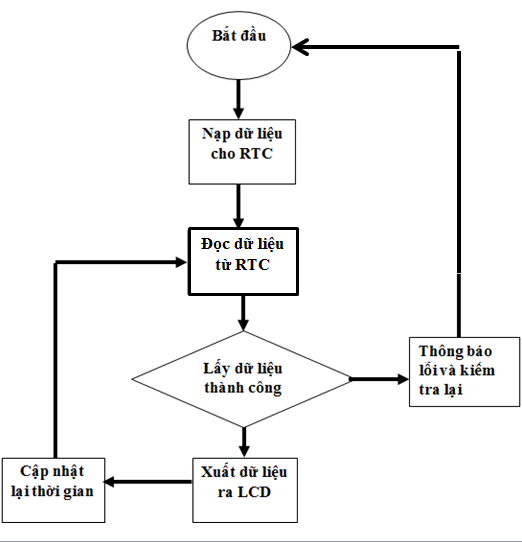
**Bước 2.** Nạp code cho Arduino bằng cách Click vào biểu tượng “<> Code”, nhấp vào nút mở và lựa chọn tập tin có phần mở rộng .INO để thực thi.

**Bước 3:** Nhấn tải lên để nạp code cho Arduino.



Hình 2.2. Hướng dẫn nạp chương trình cho Arduino

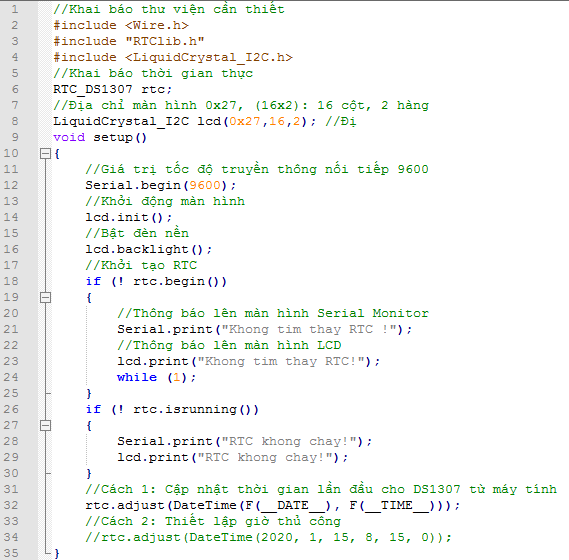
## 2.2. Lập trình



Hình 2.3. Sơ đồ hoạt động của chương trình đồng hồ thời gian thực

## 2.2.1. Giao tiếp RTC với Arduino

Để cho RTC (của module DS1307) giao tiếp được với Arduino thì phải cần phải sử dụng đến thư viện *“RTClib.cpp” và “RTClib.h”,*công dụng của thư viện này giúp cho việc lấy dữ liệu từ bộ nhớ của RTC ra xử lý và nạp vào vi điều khiển.



Hình 2.4. Trình tự từng bước khởi tạo RTC

**Bước 1:** Khởi tạo RTC: *rtc.begin();*

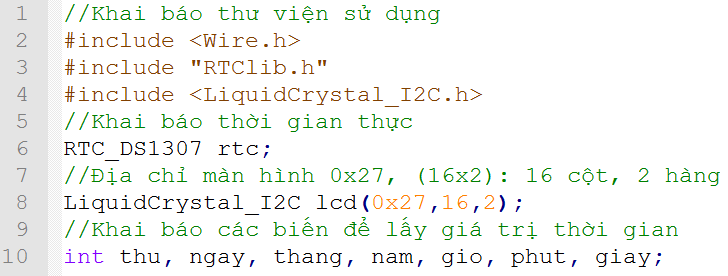
**Bước 2:** Kiểm tra trạng thái hoạt động của RTC:*!rtc.isrunning();* kết quả trả về giá trị “RTC không chạy” lên màn hình Serial Monitor và LCD khi RTC chưa hoạt động

**Bước 3**: Hàm cập nhật thời gian tự động cho RTC từ máy tính: *rtc.adjust(DateTime(F(\_DATE\_), F(\_TIME\_)));*

## 2.2.2. Hiển thị thời gian lên màn hình LCD

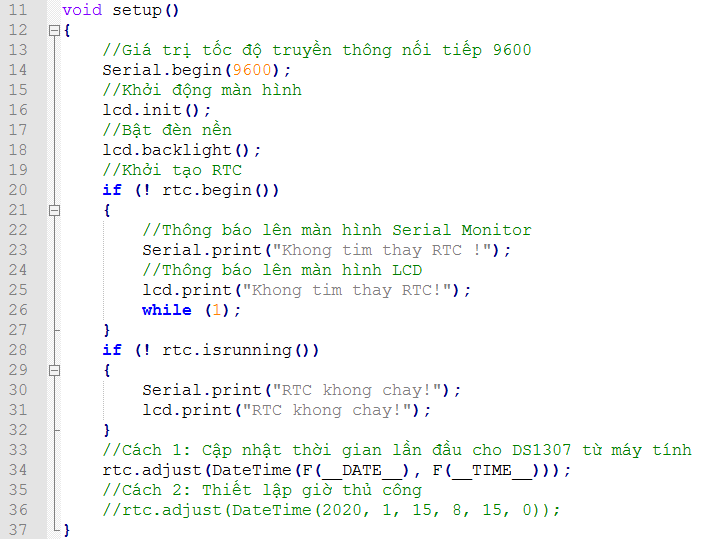
Thời gian được hiển thị lên màn hình LCD với các thông số của hàng thứ nhất là: thứ, ngày, tháng , năm. Thông số của hàng thứ 2: giờ, phút, giây và được cập nhật liên tục sau mỗi giây.

Khai báo các thư viện và các biến, gồm: Các thư viện <Wire.h>, <RTClib.h> để giao tiếp với module DS1307, thư viện <LiquidCrystal\_I2C.h> để thao tác trên màn hình. Đặt mặc định các chân của màn hình LCD và địa chỉ cho LCD là (0x27), (16, 2) là số cột của màn hình LCD là 16 và số hàng là 2. Khai báo các biến để lấy giá trị thời gian từ các hàm tính thời gian mặc định, gồm: thứ, ngày, tháng, năm, giờ, phút, giây.



Hình 2.5. Khai báo thư viện và các biến

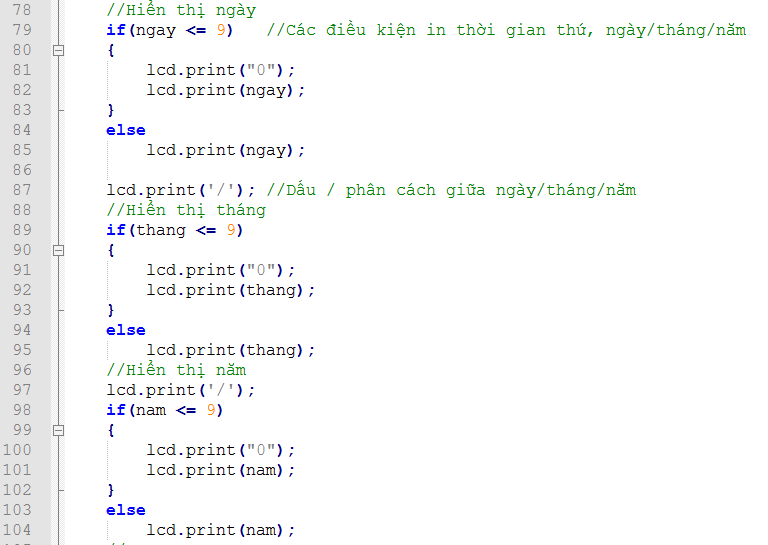
Hàm setup: khai báo các biến, thiết lập các thông số như: truyền giá trị tốc độ truyền thông nối tiếp 9600, hàm khởi động màn hình LCD, hàm chế độ bật đèn nền cho LCD, khởi tạo và chạy RTC, hàm cập nhật thời gian bằng 2 cách: cách 1 cập nhật thời gian lần 1 lần duy nhất từ máy tính, cách 2 thiết lập thời gian thủ công bằng tham số tự truyền vào.



Hình 2.6. Hàm setup() trong chương trình

Hàm loop:

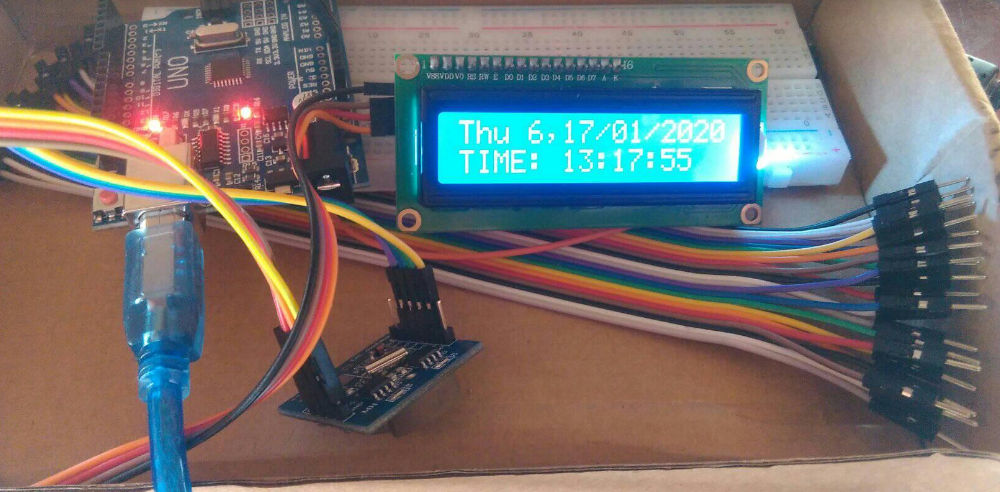
Bắt đầu bằng việc thiết lập thời gian hiện tại từ máy tính, sau đó gán từng giá trị cho các biến đã được khai báo ở ngoài. Tiếp theo, sử dụng hàm lcd.print để in lần lượt các giá trị thời gian thứ, ngày, tháng, năm, giờ, phút, giây ra màn hình LCD.



Hình 2.7. Hàm loop() trong chương trình

# 3. KẾT QUẢ

Chương trình được nạp vào board mạch Arduino, thời gian được hiển thị đồng thời lên màn hình LCD từ trái sang phải từ trên xuống dưới tương ứng với thứ, ngày, tháng, năm, giờ phút, giây. Hình 2.6 là đồng hồ thời gian thực hiển thị lên màn hình LCD tương ứng với thứ ngày tháng năm, giờ phút giây.



Hình 2.8. Sản phẩm hoàn thiện với thời gian được hiển thị lên màn hình LCD

# THẢO LUẬN

Sản phẩm hoàn thiện với các chức năng đã được trình bày ở phần phương pháp. Toàn bộ quy trình thiết kế, thi công sản phẩm được hoàn thiện trong thời gian 5 tuần của đợt thực tập cơ sở. Sản phẩm về cơ bản đáp ứng được các yêu cầu đặt ra trong yêu cầu của Giáo viên hướng dẫn. Tuy nhiên, sản phẩm vẫn chưa hoàn thiện về mặt hình thức và vẫn còn thiếu quy trình thi công mạch in.

# PHỤ LỤC

//Khai báo thư viện sử dụng

#include <Wire.h>

#include "RTClib.h"

#include <LiquidCrystal\_I2C.h>

//Khai báo thời gian thực

RTC\_DS1307 rtc;

//Địa chỉ màn hình 0x27, (16x2): 16 cột, 2 hàng

LiquidCrystal\_I2C lcd(0x27,16,2);

//Khai báo các biến để lấy giá trị thời gian

int thu, ngay, thang, nam, gio, phut, giay;

void setup()

{

//Giá trị tốc độ truyền thông nối tiếp 9600

Serial.begin(9600);

//Khởi động màn hình

lcd.init();

//Bật đèn nền

lcd.backlight();

//Khởi tạo RTC

if (! rtc.begin())

{

//Thông báo lên màn hình Serial Monitor

Serial.print("Khong tim thay RTC !");

//Thông báo lên màn hình LCD

lcd.print("Khong tim thay RTC!");

while (1);

}

if (! rtc.isrunning())

{

Serial.print("RTC khong chay!");

lcd.print("RTC khong chay!");

}

//Cách 1: Cập nhật thời gian lần đầu cho DS1307 từ máy tính

rtc.adjust(DateTime(F(\_\_DATE\_\_), F(\_\_TIME\_\_)));

//Cách 2: Thiết lập giờ thủ công

//rtc.adjust(DateTime(2020, 1, 15, 8, 15, 0));

}

void loop ()

{

//Thiết lập thời gian hiện tại

DateTime now = rtc.now();

//Truyền giá trị cho các biến thứ, ngày, tháng, năm

thu = now.dayOfTheWeek();

ngay = now.day();

thang = now.month();

nam = now.year();

//Truyền giá trị cho các biến thứ, ngày, tháng, năm

gio = now.hour();

phut = now.minute();

giay = now.second();

//Hàng đầu tiên hiển thị thứ, ngày, tháng, năm

lcd.setCursor(0, 0);

//Hiển thị thứ trong tuần

switch(thu)

{

case 0:

lcd.print("CN,");

break;

case 1:

lcd.print("Thu 2,");

break;

case 2:

lcd.print("Thu 3,");

break;

case 3:

lcd.print("Thu 4,");

break;

case 4:

lcd.print("Thu 5,");

break;

case 5:

lcd.print("Thu 6,");

break;

case 6:

lcd.print("Thu 7,");

break;

}

//Hiển thị ngày

if(ngay <= 9) //Các điều kiện in thời gian thứ, ngày/tháng/năm

{

lcd.print("0");

lcd.print(ngay);

}

else

lcd.print(ngay);

lcd.print('/'); //Dấu / phân cách giữa ngày/tháng/năm

//Hiển thị tháng

if(thang <= 9)

{

lcd.print("0");

lcd.print(thang);

}

else

lcd.print(thang);

//Hiển thị năm

lcd.print('/');

if(nam <= 9)

{

lcd.print("0");

lcd.print(nam);

}

else

lcd.print(nam);

//

//Hàng 2 hiển thị thông tin giờ/phút/giây

lcd.setCursor(0, 1);

lcd.print("TIME: ");

//Hiện thị giờ

if(gio <= 9) //Điều kiện hiển thị

{

lcd.print("0");

lcd.print(gio);

}

else

lcd.print(gio);

lcd.print(':'); //Dấu : phân cách giữa giờ/phút/giây

//Hiển thị phút

if(phut <= 9)

{

lcd.print("0");

lcd.print(phut);

}

else

lcd.print(phut);

lcd.print(':');

//Hiển thị giây

if(giay <= 9)

{

lcd.print("0");

lcd.print(giay);

}

else

lcd.print(giay);

}