|  |  |
| --- | --- |
|  | [Document title] |
|  |  |
|  | [Author]  [Course title]  [Date] |

Contents

[Phần 1. Cơ sở lý thuyết 4](#_Toc183271615)

[1. Hiển thị Led 7 đoạn 4](#_Toc183271616)

[2. Đọc dữ liệu từ module RTC DS1307 6](#_Toc183271617)

[Phần 2. Nội dung thực hành 18](#_Toc183271618)

[1. Sơ đồ kết nối mạch 18](#_Toc183271619)

[2. Cách hoạt động của mô hình 18](#_Toc183271620)

[3. Lập trình chương trình nhúng 19](#_Toc183271621)

[4. Mô hình thử nghiệm phần cứng 20](#_Toc183271622)

[Phần 3. Kết luận 21](#_Toc183271623)

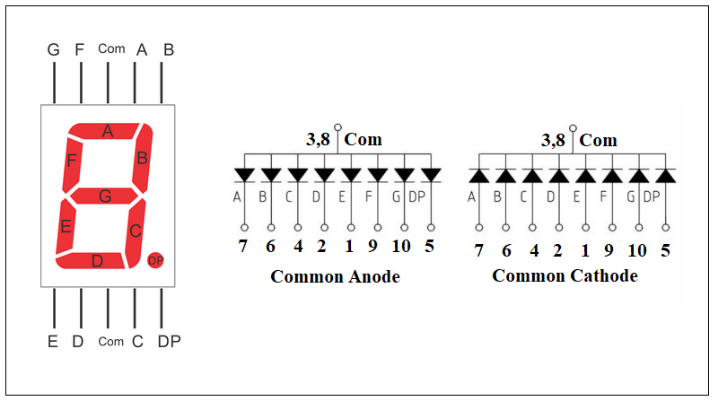
[Phần 4. Nâng cao 22](#_Toc183271624)

# Cơ sở lý thuyết

## Hiển thị Led 7 đoạn

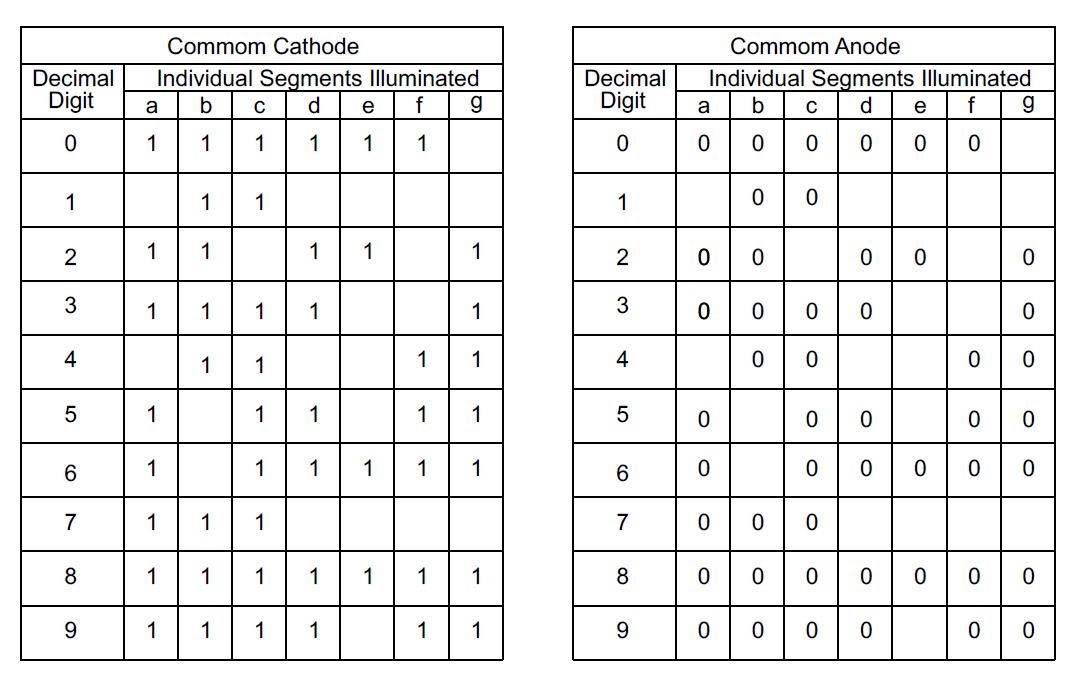
LED là linh kiện cho phép dòng điện đi qua một chiều và phát sáng. Điện áp rơi trên LED phụ thuộc vào màu sắc của LED, thông thường LED có điện áp rơi từ khoảng 1.8V đến 2.2V với dòng trung bình khoảng 20mA. Điện áp của vi điều khiển AVR là 5V và dòng ra 20mA. Cần phải hạ áp cho LED bằng điện trở. Điện trở hạ áp cho LED từ 5V thường là 300Ω đến 1kΩ.

LED 7 thanh được cấu tạo bởi 8 LED đơn ký hiệu A, B, C, D, E, F, G và DP với mục đích hiển thị tùy ý. Có 2 loại là Anode chung và Cathode chung. Sơ đồ minh họa của 2 loại LED như sau:



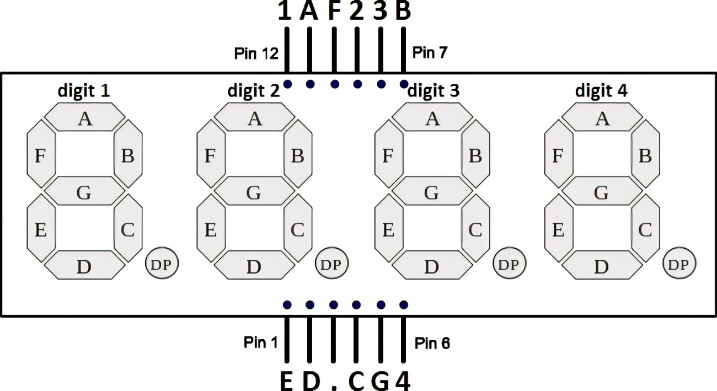
Hình 1. Led 7 segment common cathode/common anode.

Tùy từng loại LED 7 đoạn là Anode chung hay Cathode chung mà có mã để hiển thị với từng ký tự. Dưới đây là mã hiển thị của từng loại:



Hình 2. Led 7 segment truth table.

Ngoài loại LED 7 đoạn riêng biệt, còn có loại thanh gồm nhiều LED 7 đoạn nối với nhau, tùy theo loại sử dụng là Anode chung hay Cathode chung mà các chân còn lại của LED 7 đoạn được nối chung với nhau.



Hình 3. 4 digit 7 segment LED

Loại LED này được sử dụng bằng phương pháp quét led, bằng cách hiển thị từng led trong khoảng thời gian mà mắt không nhận thấy được sự nhấp nháy (24FPS) sẽ làm cho mắt người cảm thấy là hiển thị liên tục. Để điều khiển Led, cần một công tắc điện tử (có thể là BJT, MOSFET, opto quang, …) để có thể thông mạch led.

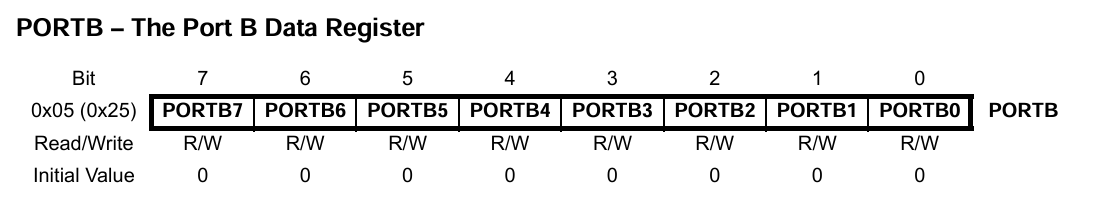
Trong bài thực hành của nhóm, nhóm sử dụng 2 thanh led 7 đoạn 4 ký tự .

Vi điều khiển AVR của nhóm dùng là Atmega328p.

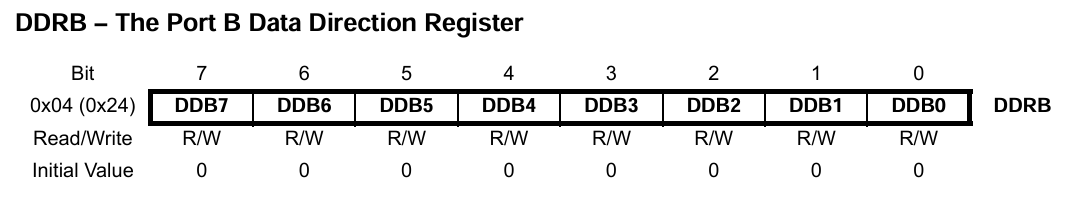
Để điều khiển trạng thái của đầu ra/đầu vào vi điều khiển AVR dùng để điều khiển LED, trong AVR có 3 thanh ghi là DDRx, PORTx, PINx. Tùy vào từng thiết lập mà sẽ có từng chức năng đối với I/O Pin của AVR.

Ví dụ cấu trúc của 3 thanh ghi DDRB, PORTB, PINB như sau:

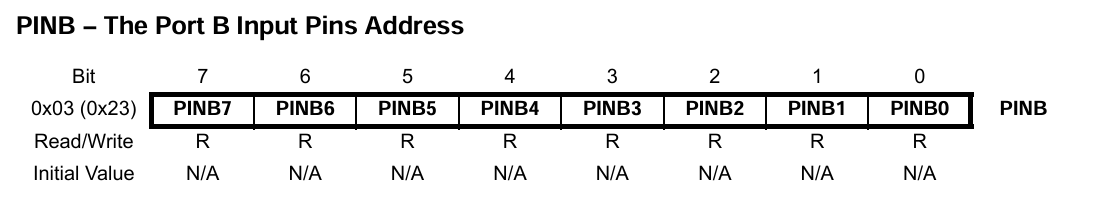
* PORTB:



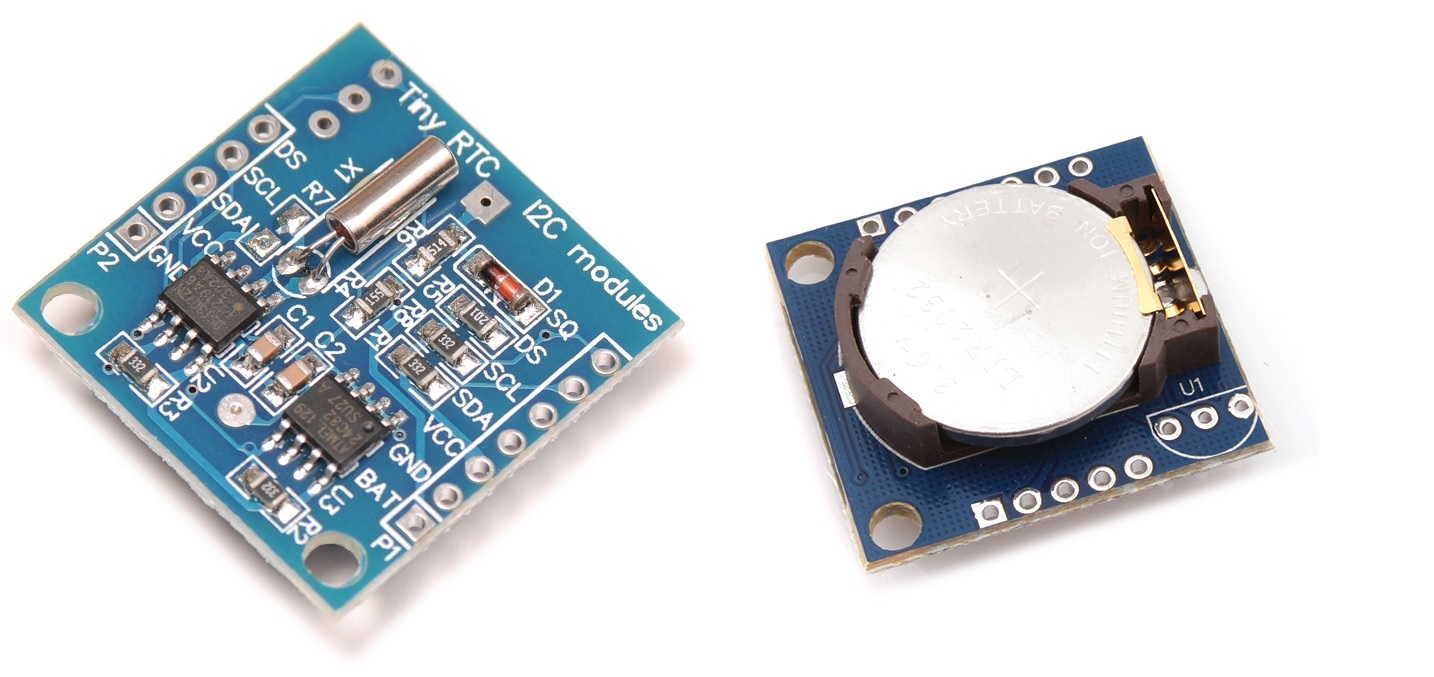
* DDRB:



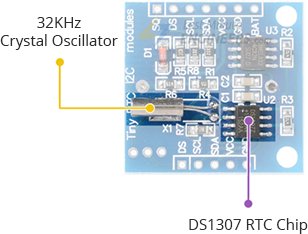
* PINB:



## Đọc dữ liệu từ module RTC DS1307



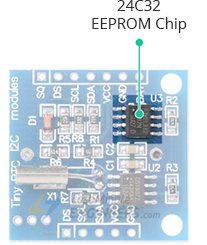
Hình 4. Module RTC DS1307.

Module được xây dựng từ 2 IC chính là DS1307 RTC và AT24C32 EEPROM.

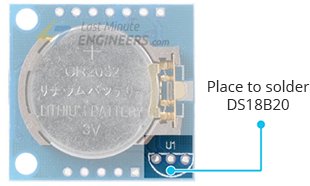
Trung tâm của module là chip RTC, cực kỳ chính xác từ Maxim — DS1307. Nó xử lý tất cả các chức năng giữ giờ và giao tiếp với bộ vi điều khiển qua I2C.

DS1307 có thể theo dõi giây, phút, giờ, thứ, ngày, tháng và năm. Nó có thể hoạt động ở định dạng 12 giờ hoặc 24 giờ và có chỉ báo AM/PM. Đối với những tháng có ít hơn 31 ngày, tính năng này sẽ tự động điều chỉnh ngày vào cuối tháng, bao gồm cả việc sửa đổi năm nhuận (có hiệu lực đến năm 2100).

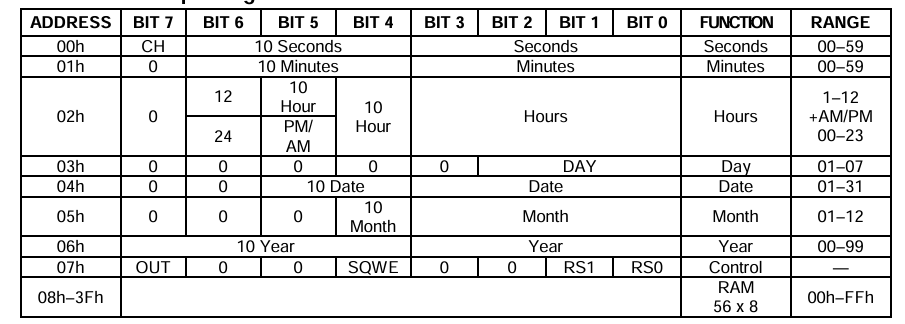
Một tính năng thú vị khác của DS1307 là chân SQW, có thể được lập trình để xuất ra một trong bốn tần số sóng vuông: 1Hz, 4kHz, 8kHz hoặc 32kHz. DS1307 cần một tinh thể 32kHz bên ngoài để tính giờ, tần số của tinh thể này dễ bị ảnh hưởng bởi nhiệt độ bên ngoài. Mặc dù sự thay đổi tần số này không đáng kể, nhưng nó vẫn cộng dồn. DS1307 cần một tinh thể 32kHz bên ngoài để tính giờ, tần số của tinh thể này dễ bị ảnh hưởng bởi nhiệt độ bên ngoài. Mặc dù sự thay đổi tần số này không đáng kể, nhưng nó vẫn cộng dồn.

Mô-đun RTC DS1307 cũng bao gồm chip AT24C32 EEPROM 32 byte (4K x 8 bit) với 1.000.000 chu kỳ ghi. Con chip này thực ra không liên quan gì đến RTC, nhưng nó có thể hữu ích cho những việc như ghi dữ liệu hoặc lưu trữ bất kỳ dữ liệu nào khác mà muốn không bị mất.

EEPROM 24C32 giao tiếp qua I2C và chia sẻ cùng bus I2C với DS1307. Địa chỉ của 24C32 là 0x50.

Có thể lắp đặt DS18B20 bằng ba lỗ lắp ở góc trên bên phải, gần giá đỡ pin (được đánh dấu U1). Sau khi cài đặt DS18B20,có thể lấy được số liệu nhiệt độ từ chân DS. Những số liệu này có thể được sử dụng để bù cho sự trôi thời gian dựa trên nhiệt độ.

Nội dung của Timekeeper của IC DS1307 được định dạng theo định dạng BCD (giá trị thập phân được mã hóa nhị phân). Có tổng cộng tám thanh ghi trong thanh ghi thời gian để cài đặt giây, phút, giờ, ngày, ngày, tháng, năm và kiểm soát. Sau khi chúng ta thiết lập giá trị cho các thanh ghi này, chúng sẽ tiếp tục tự cập nhật và chúng ta có thể đọc các thanh ghi này để lấy các giá trị cập nhật.



Hình 5. Timekeeper Registers

1. **Địa chỉ 00H:02H: Clock Registers**

**Địa chỉ - 00H:**

Trong thanh ghi này, bit 7 là bit CH, được sử dụng để bật/tắt bộ dao động tinh thể. Khi giá trị của bit này là 0, bộ dao động tinh thể được kích hoạt; ngược lại, nếu bit này không bằng 0, bộ dao động sẽ không được kích hoạt. Do đó, chúng ta luôn đặt bit này bằng 0 khi sử dụng RTC.

Các bit còn lại được sử dụng để đọc/ghi giá trị của giây. Vì thanh ghi thời gian lưu trữ giá trị ở định dạng BCD, nên:

* Bit 4 đến Bit 6 lưu trữ chữ số BCD cao của giây (giá trị từ 0 đến 5).
* Bit 0 đến Bit 3 lưu trữ chữ số BCD thấp của giây (giá trị từ 0 đến 9).

Giá trị của giây bắt đầu từ 00 và kết thúc ở 59.

**Địa chỉ - 01H:**

Địa chỉ này được sử dụng để đọc/ghi giá trị của phút.

* Chữ số BCD cao của phút được lưu trữ trong các bit từ 4 đến 6.
* Chữ số BCD thấp của phút được lưu trữ trong các bit từ 0 đến 3.

**Địa chỉ - 02H:**

Địa chỉ này được sử dụng để đọc/ghi giá trị của giờ. Đồng hồ có thể hoạt động ở định dạng 12 giờ hoặc 24 giờ.

* **Định dạng 12 giờ:**
  + Để thiết lập đồng hồ ở định dạng 12 giờ, cần đặt bit 6 là logic 1.
  + Trong định dạng 12 giờ, bit 5 chỉ thị AM/PM: logic 1 là PM và logic 0 là AM.
  + Bit 4 biểu thị hàng chục giờ (10 giờ), dùng để lưu chữ số cao hơn của giá trị giờ, có thể là 0 hoặc 1 trong hệ thống 12 giờ.
  + Bit 0 đến bit 3 lưu giá trị chữ số thấp của giờ (giá trị từ 0 đến 9).
* **Định dạng 24 giờ:**
  + Để thiết lập đồng hồ ở định dạng 24 giờ, cần đặt bit 6 là logic 0.
  + Bit 4 và bit 5 biểu thị hàng chục giờ (10 giờ), dùng để lưu chữ số cao hơn của giá trị giờ, có thể là từ 0 đến 2 trong hệ thống 24 giờ.
  + Bit 0 đến bit 3 lưu giá trị chữ số thấp của giờ (giá trị từ 0 đến 9).

1. **Địa chỉ 03H: 06H: Calendar Register**

**Địa chỉ - 03H:**

Địa chỉ này được sử dụng để đọc/ghi giá trị ngày trong tuần, từ 1 đến 7. Các bit từ 0 đến 2 được sử dụng để đọc/ghi giá trị của ngày.

**Địa chỉ - 04H:**

Địa chỉ này được sử dụng để đọc/ghi giá trị ngày trong tháng.

* Bit 4 và Bit 5 được sử dụng để đọc/ghi chữ số cao của ngày (giá trị từ 0 đến 3).
* Bit 0 đến Bit 3 được sử dụng để đọc/ghi chữ số thấp của ngày (giá trị từ 0 đến 9).

**Địa chỉ - 05H:**

Địa chỉ này được sử dụng để đọc/ghi giá trị của tháng.

* Bit 4 được sử dụng để đọc/ghi chữ số cao của tháng, có thể là 0 hoặc 1.
* Bit 0 đến Bit 3 được sử dụng để lưu chữ số thấp của tháng (giá trị từ 0 đến 9).

**Địa chỉ - 06H:**

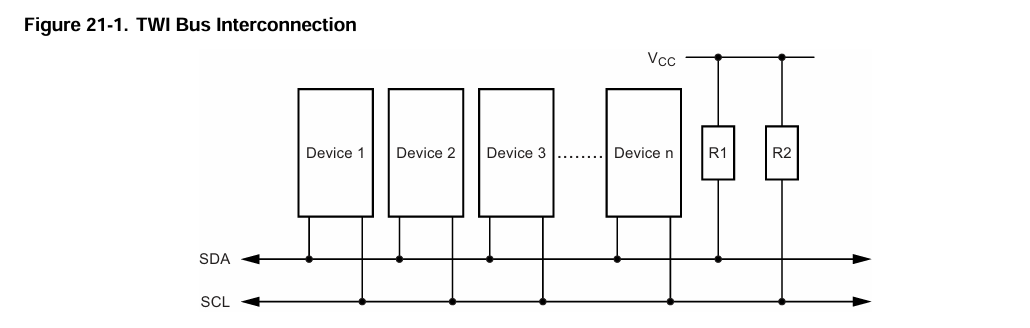
Địa chỉ này được sử dụng để đọc/ghi giá trị của năm, chỉ bao gồm hai chữ số cuối của năm.

* Bit 0 đến Bit 3 lưu chữ số thấp của năm.
* Bit 4 đến Bit 7 lưu chữ số cao của năm.

1. **Address-07H: Control Register**Thanh ghi này sử dụng để thiết lập xung tại chân SQW.

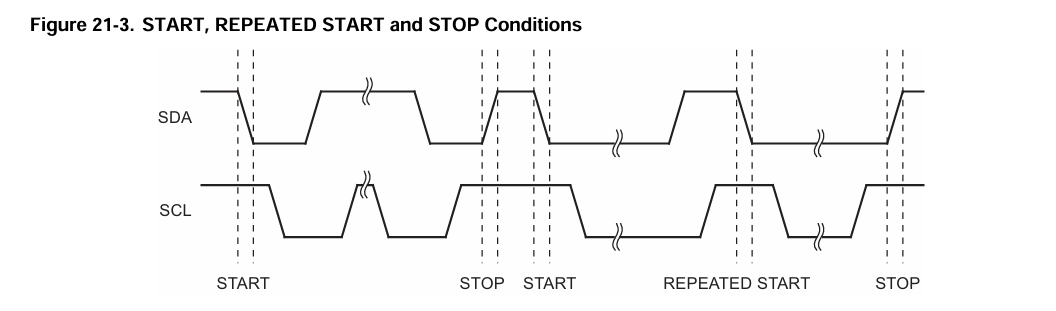
## Giao tiếp I2C (TWI) của AVR

Giao diện nối tiếp 2 dây (TWI) lý tưởng cho các ứng dụng vi điều khiển điển hình. Giao thức TWI cho phép hệ thống kết nối tới 128 thiết bị khác nhau chỉ bằng hai đường bus hai chiều, một cho đồng hồ (SCL) và một cho dữ liệu (SDA). Phần cứng bên ngoài duy nhất cần thiết để triển khai bus là một điện trở kéo lên duy nhất cho mỗi đường bus TWI. Tất cả các thiết bị được kết nối với bus đều có địa chỉ riêng và các cơ chế giải quyết tranh chấp xe buýt vốn có trong nguyên mẫu TWI.



Hình 6. TWI Bus Interconnection

Master bắt đầu và chấm dứt truyền dữ liệu. Quá trình truyền được bắt đầu khi chủ đưa ra điều kiện START trên bus và nó bị chấm dứt khi chủ đưa ra điều kiện STOP. Giữa điều kiện START và STOP, bus được coi là bận rộn và không Master nào khác nên cố gắng nắm quyền kiểm soát xe buýt. Trường hợp đặc biệt xảy ra khi điều kiện START mới được ban hành giữa điều kiện START và STOP. Điều này được gọi là điều kiện REPEATED START và được sử dụng khi Master muốn bắt đầu chu trình chuyển dữ liệu mới mà không từ bỏ quyền kiểm soát bus. Sau khi REPEATED START, bus được coi là bận rộn cho đến STOP tiếp theo. Điều này giống hệt với START và do đó START được sử dụng để mô tả cả START và REPEAT START. Như được mô tả bên dưới, các điều kiện START và STOP được báo hiệu bằng cách thay đổi mức của đường SDA khi đường SCL cao.

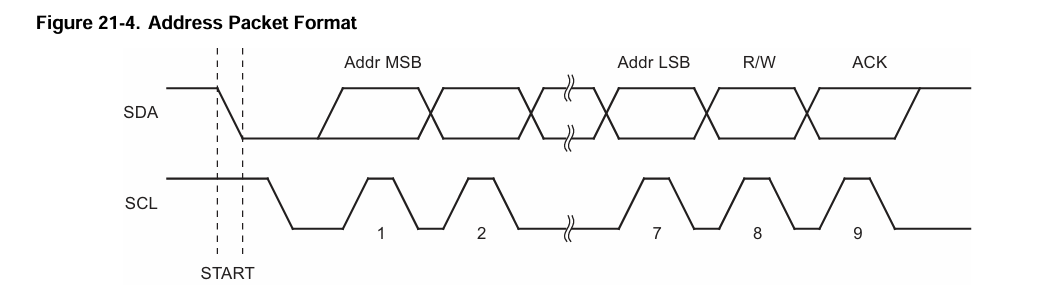


Hình 7. START, REPEATED START and STOP Conditions

1. Address Packet Format

Tất cả các gói địa chỉ được truyền trên bus TWI dài 9 bit, bao gồm 7 bit địa chỉ (tương đương 128 thiết bị), một bit điều READ/WRITE Control và bit xác nhận (acknowledge). Nếu bit READ/WRITE được set, thao tác đọc sẽ được thực hiện, nếu không sẽ thực hiện thao tác ghi. Khi một Slave nhận ra rằng nó đang được gọi, nó nên thừa nhận bằng cách kéo SDA xuống thấp trong xung SCL thứ 9 (ACK). Nếu Slave được gọi đang bận, dòng SDA nên được để cao trong chu kỳ xung nhịp ACK. Sau đó, máy chủ có thể truyền điều kiện STOP hoặc điều kiện REPEATED START để bắt đầu đường truyền móei. Một gói địa chỉ bao gồm một địa chỉ Slave và một bit READ/WRITE được gọi là SLA + R hoặc SLA + W, tương ứng.

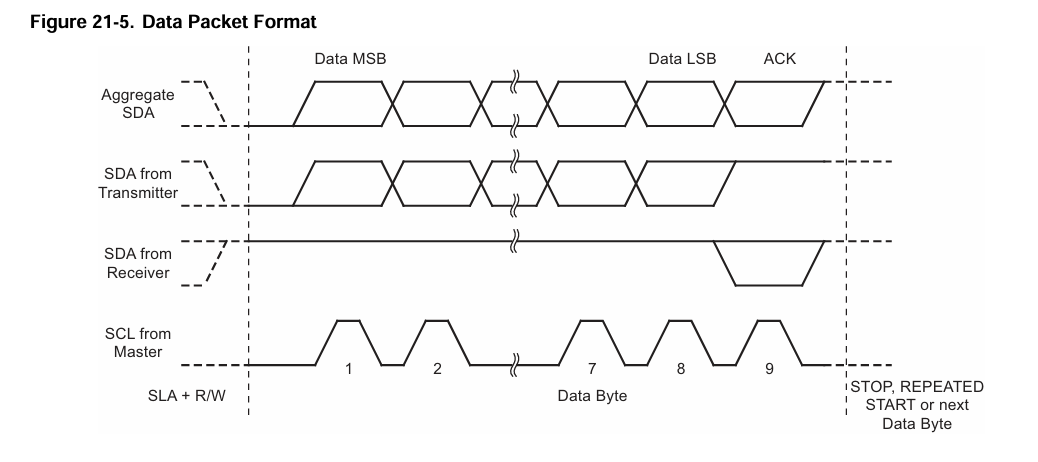
MSB của byte địa chỉ được truyền trước. Địa chỉ nô lệ có thể được phân bổ tự do bởi nhà thiết kế, nhưng địa chỉ 0000 000 được dành riêng cho một địa chỉ gọi chung. Khi một cuộc gọi chung được đưa ra, tất cả các Slave nên trả lời bằng cách kéo đường SDA xuống thấp trong chu kỳ ACK. Một cuộc gọi chung được sử dụng khi một chủ muốn truyền cùng một thông điệp đến một số Slave trong hệ thống. Khi địa chỉ cuộc gọi chung theo sau là một bit WRITE được truyền trên bus, tất cả các Slave được thiết lập để xác nhận cuộc gọi chung sẽ kéo đường SDA xuống thấp trong chu kỳ ACK. Các gói dữ liệu sau đó sẽ được nhận bởi tất cả các nô lệ đã nhận cuộc gọi chung. Lưu ý rằng việc truyền địa chỉ cuộc gọi chung theo sau là một bit READ là vô nghĩa, vì điều này sẽ gây ra xung đột nếu một số nô lệ bắt đầu truyền dữ liệu khác nhau.



Hình 8. Address Packet Format

1. Data Packet Format

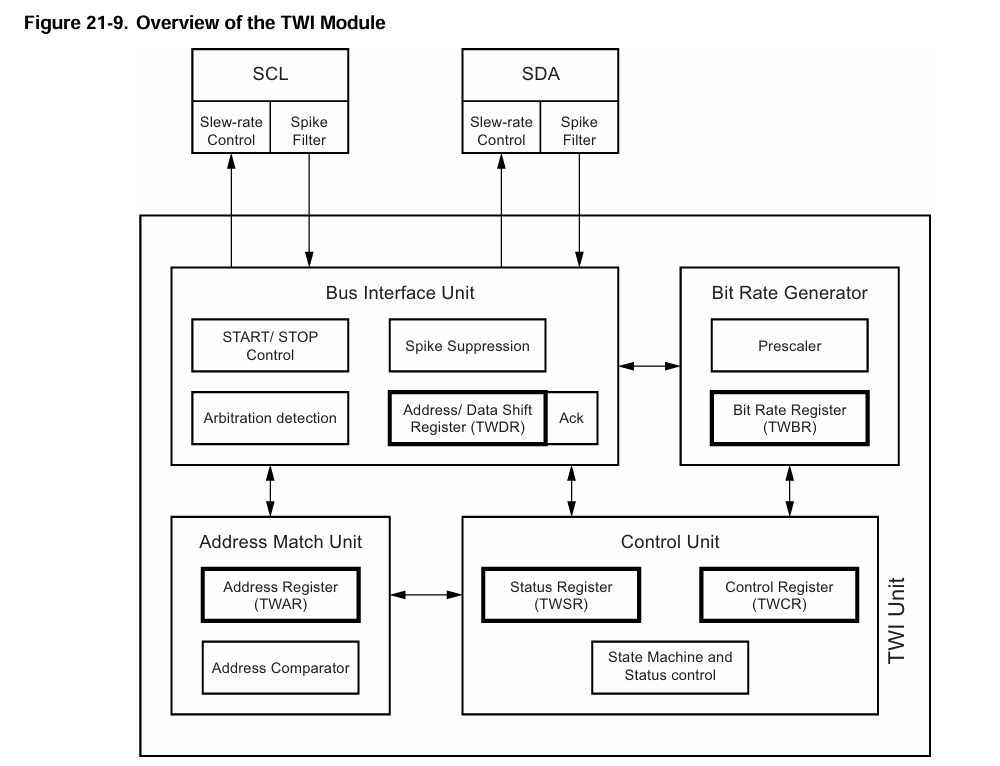
Tất cả các gói dữ liệu được truyền trên bus TWI dài 9 bit, bao gồm một byte dữ liệu và một bit xác nhận. Trong quá trình truyền dữ liệu, chủ tạo ra clock và các điều kiện START và STOP, trong khi người nhận chịu trách nhiệm xác nhận việc nhận. Xác nhận (ACK) được báo hiệu bởi bên thu kéo đường SDA xuống thấp trong chu kỳ SCL thứ chín. Nếu máy thu rời khỏi đường SDA cao, NACK sẽ được báo hiệu. Khi người nhận đã nhận được byte cuối cùng, hoặc vì lý do nào đó không thể nhận thêm bất kỳ byte nào, nó sẽ thông báo cho máy phát bằng cách gửi NACK sau byte cuối cùng. MSB của byte dữ liệu được truyền trước.



Hình 9. Data Packet Format

1. Overview of the TWI Module

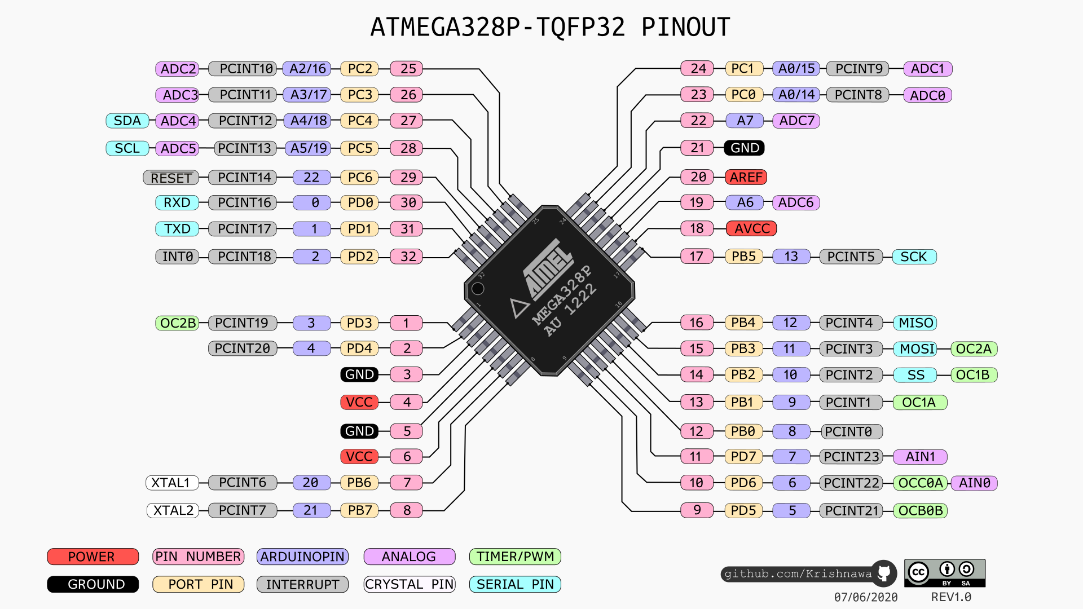
Mô-đun TWI bao gồm một số mô-đun phụ, như thể hiện trong Hình 21-9. Tất cả các thanh ghi được vẽ trong một đường đậm có thể truy cập thông qua bus dữ liệu AVR.



Hình 10. Overview of the TWI Module

1. SCL and SDA Pins

Chân SDA và SCL được thể hiện trong hình dưới.



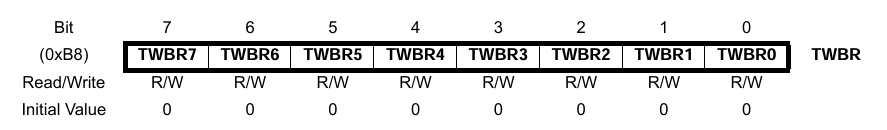
Hình 11. Atmega328p Pinout

1. Bit Rate Generator Unit

Bộ phận kiểm soát khoảng thời gian của SCL khi hoạt động ở chế độ chính. Chu kỳ của SCL được kiểm soát bởi thanh ghi tốc độ bit TWI (TWBR) và các bit prescaler trong thanh ghi trạng thái TWI (TWSR). Hoạt động của Slave không phụ thuộc vào tốc độ bit hoặc prescale, nhưng tần số xung nhịp CPU trong Slave phải cao hơn ít nhất 16 lần so với tần số SCL. Tần số SCL được tạo theo phương trình sau:

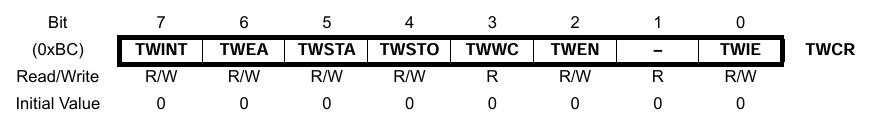
1. Các thanh ghi được dùng của TWI

* TWBR – TWI Bit Rate Register



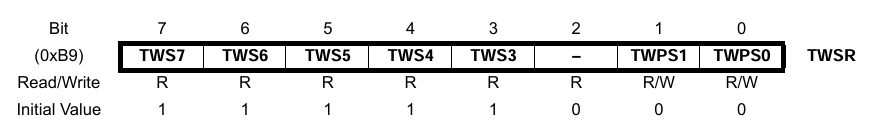
Thanh ghi chứa giá trị của TWBR.

* TWCR – TWI Control Register



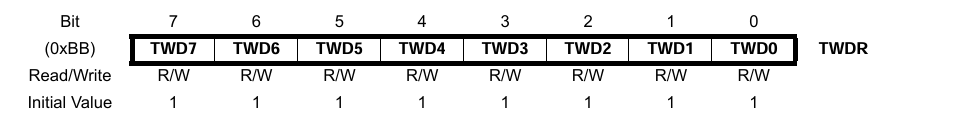
TWCR được sử dụng để kiểm soát hoạt động của TWI. Nó được sử dụng để kích hoạt TWI, để bắt đầu truy cập chính bằng cách áp dụng điều kiện START cho bus, để tạo xác nhận người nhận, tạo điều kiện dừng và kiểm soát việc dừng bus trong khi dữ liệu được ghi vào bus được ghi vào TWDR. Nó cũng chỉ ra xung đột ghi nếu dữ liệu được cố gắng ghi vào TWDR trong khi thanh ghi không thể truy cập được.

* TWSR – TWI Status Register



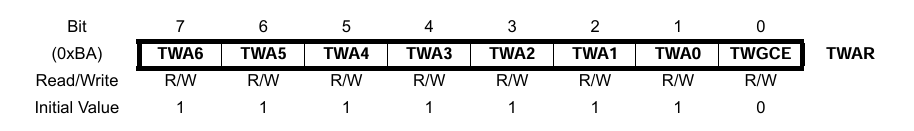
Thanh ghi này phản ánh trạng thái của bus TWI và bit thiết lập prescale của SCL.

* TWDR – TWI Data Register



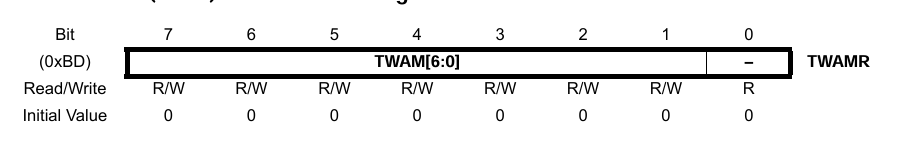
Chứa byte dữ liệu truyền hoặc nhận.

* TWAR – TWI (Slave) Address Register



Chứa địa chỉ của Slave được gọi và bit TWGCE (TWI General Call Recognition Enable Bit).

* TWAMR – TWI (Slave) Address Mask Register

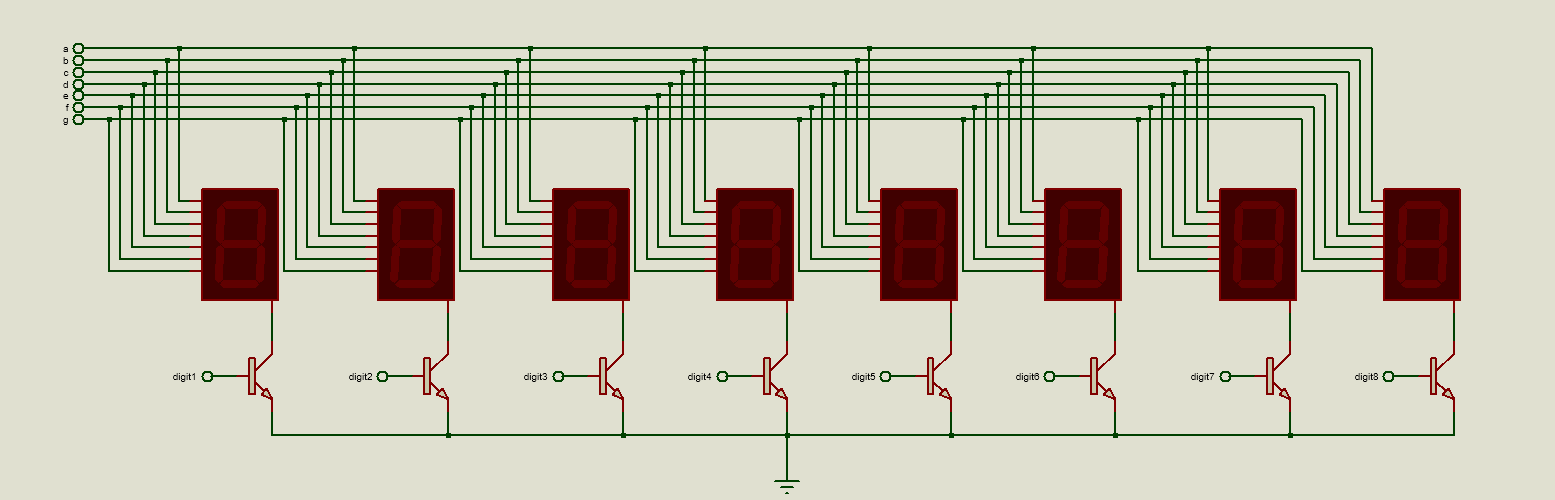
TWAMR có thể được tải với mặt nạ địa chỉ nô lệ 7 bit. Mỗi bit trong TWAMR có thể che (tắt) các bit địa chỉ tương ứng trong sổ đăng ký địa chỉ TWI (TWAR).

# Nội dung thực hành

## Sơ đồ kết nối mạch

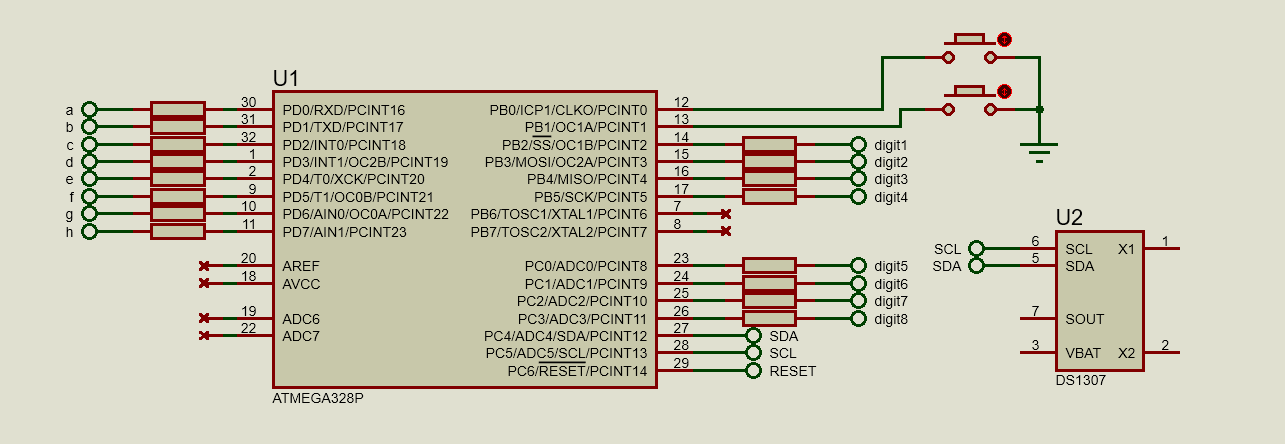
Mạch hiển thị 8 Led 7 đoạn:

Từng led con của led 7 đoạn được kết nối với các chân a, b, c, d, e, f, g. Từng chân cathode chung của led 7 đoạn được điều khiển bằng 1 transistor C1815.



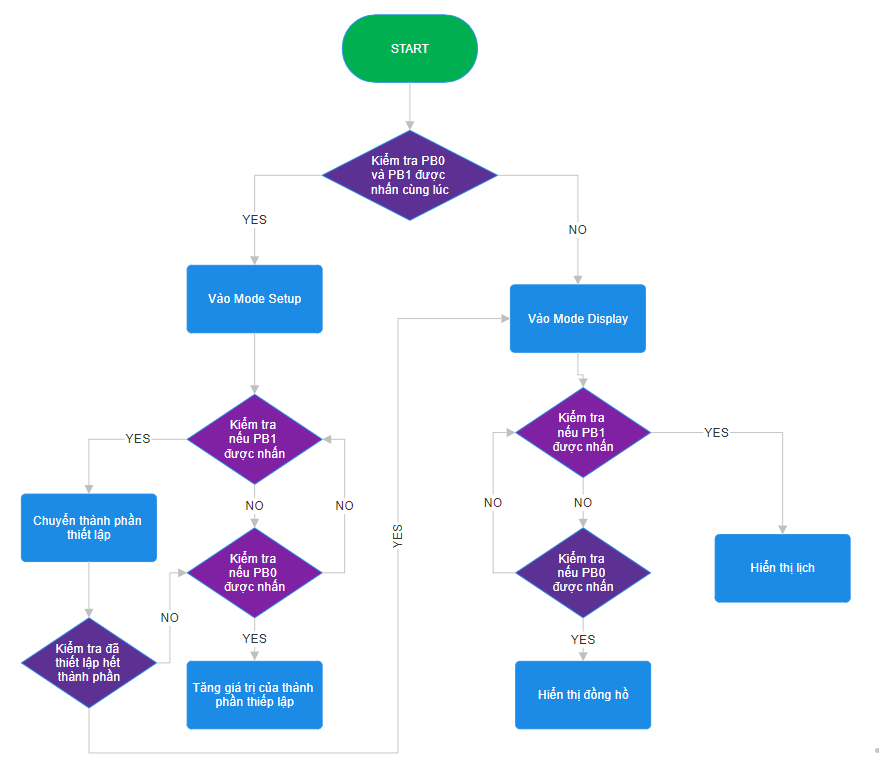
Mạch điều khiển:

PORTD được sử dụng để điều khiển từng led con của led 7 đoạn, được hạ áp qua điện trở 390 Ohm. Các chân PB2:5 được sử dụng để điều khiển 4 led 7 đoạn đầu tiên, các chân PC0:PC3 được sử dụng để điều khiển 4 led 7 đoạn sau cùng, hạ áp tại chân B của C1815 bằng điện trở 1k Ohm. 2 nút bấm được kết nối với PB0:1 được sử dụng như đầu vào cài đặt của hệ thống. Module DS1307 được kết nối tới TWI của vi điều khiển.



## Cách hoạt động của mô hình

Dưới đây là cách hoạt động của chương trình:



Hình 12. Operation

Chương trình có hai chế độ chính là *Mode\_Setup* và *Mode\_Display*, được truy cập bằng việc ấn cả hai nút PB0 và PB1 cùng một lúc, mặc định khi khởi động, chương trình ở *Mode\_Display*.

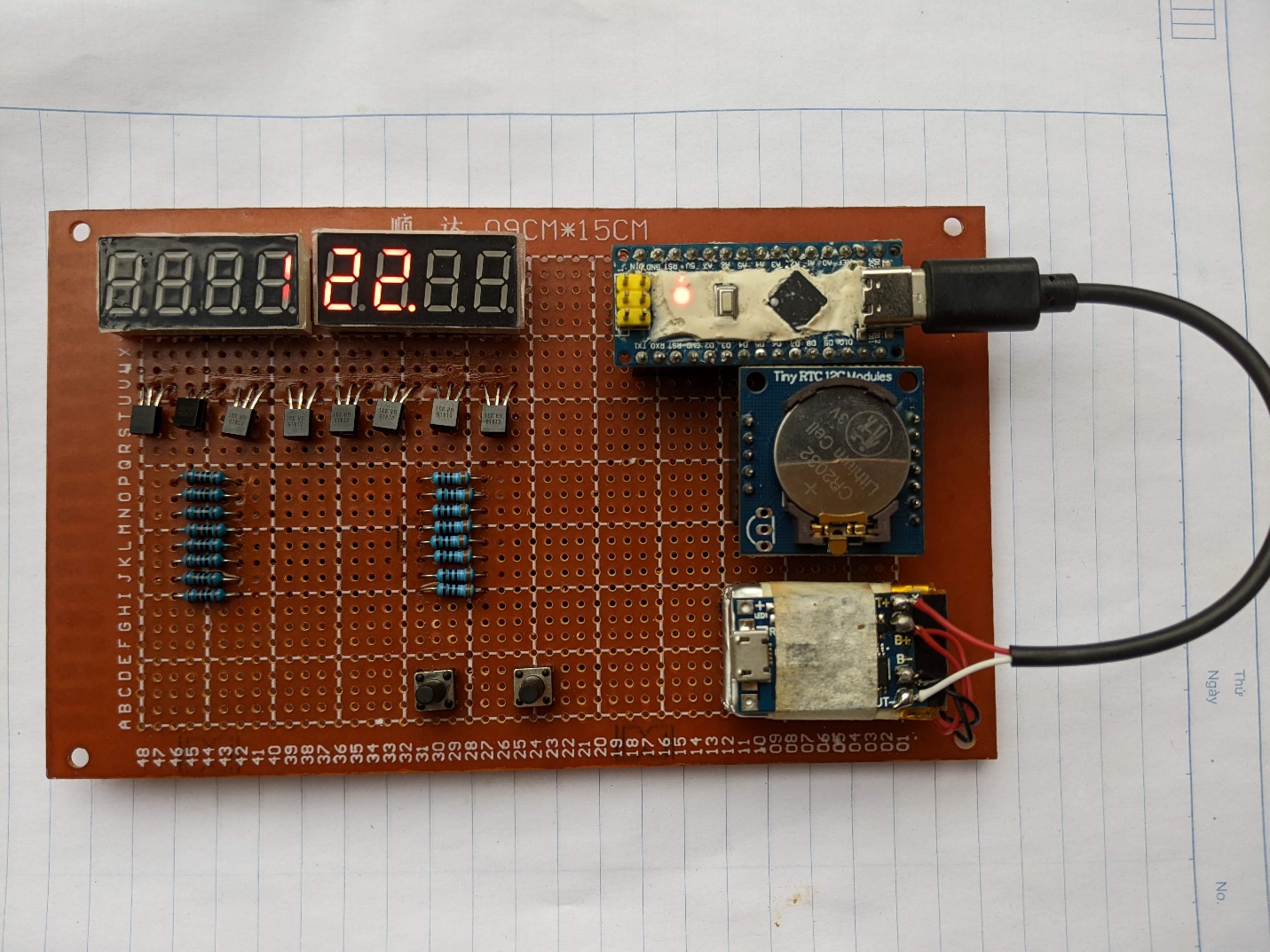
* 1. Ở *Mode\_Setup*, chương trình kiểm tra nút PB1 dùng để chỉnh tham số thành phần thiết lập bao gồm giờ, phút, giây, ngày, tháng, năm, ngày trong tuần. Sử dụng nút PB0 để tăng giá trị của biến đang thiết lập. Khi thiết lập xong toàn bộ tham số thành phần thì tự thoát *Mode\_Setup* và vào *Mode\_Display*.
  2. Ở *Mode\_Display*, chương trình kiểm tra hai nút PB0 và PB1 để chọn chế độ hiển thị là giờ/phút/giây hoặc ngày/tháng/năm.

## Lập trình chương trình nhúng

Cấu trúc Folder chương trình gồm các File thư viện cho Led 7 đoạn và Module Ds1307 như sau:

|  |  |
| --- | --- |
| main.c | Cấu trúc chương trình chính |
| Led\_7Segment\_Master.h + Led\_7Segment\_Master.c | Thư viện giao tiếp hiển thị với 8 led 7 đoạn |
| I2C\_Master.h + I2C\_Master.c | Thư việc giao tiếp I2C |
| DS1307\_Master.h + DS1307\_Master.c | Thư viện giaot tiếp với module DS1307 |

## Mô hình thử nghiệm phần cứng



# Kết luận

Yêu cầu 1: Mạch điện tử được thiết kế trên máy tính bằng phần mềm

* Sơ đồ nguyên lý và kết nối mạch được thiết kế bằng Proteus 8.13
* Mô phỏng hoạt động của mô hình bằng Proteus 8.13

Yêu cầu 2: Hệ thống hiển thị được thời gian thực ở 2 chế độ tùy theo người dùng: ngày/tháng/năm và giờ/phút/giây

* Chế độ giờ/phút/giây và ngày/tháng/năm được chọn bằng 2 nút PB0 và PB1

Yêu cầu 3: Sai số tối đa 5 giây/ngày

* Sai số ở điều kiện thường so với thời gian tham chiếu tại web Time.is không có sai lệch.

Yêu cầu 4: Trên hệ thống có nguồn dự phòng

* Trên hệ thống có nguồn dự phòng là pin 400Mah, thời gian hoạt động 30 phút.

Yêu cầu 5: Trên hệ thống có phím chỉnh thời gian

* Trên hệ thống có chức năng chỉnh thời gian.

# Nâng cao