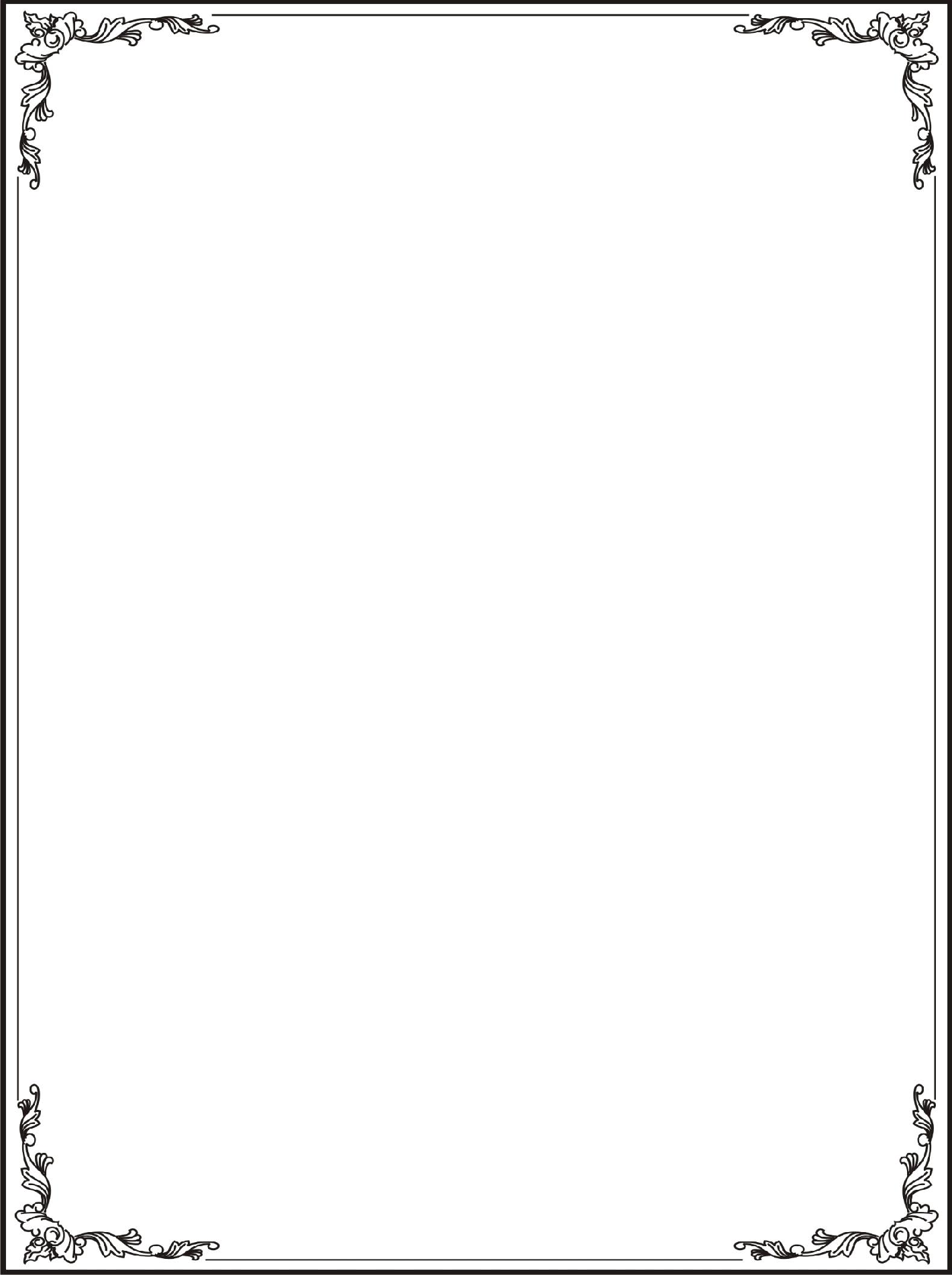
**BỘ GIÁO DỤC VÀ ĐÀO TẠO**



**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP.HCM**

**Báo cáo**

**TT Kiến trúc và tổ chức máy tính**

**GVHD: Huỳnh Hoàng Hà**

**Mã học phần: COOL325364\_23\_1\_10**

**Ngày, tiết học: Thứ sáu tiết 1-5**

**HTSV: Lý Tấn Lộc**

**MSSV: 21139078**

**TP. Hồ Chí Minh – 11/2023**

Mục lục

[A.DS1302 3](#_Toc151101875)

[1.Giới thiệu. 3](#_Toc151101876)

[2.Tính năng. 4](#_Toc151101877)

[3.Mô tả các chân. 4](#_Toc151101878)

[4.Các thành phần chính. 5](#_Toc151101879)

[5.Command Byte. 6](#_Toc151101880)

[6.Các chức năng và mode hoạt động trong DS1302. 6](#_Toc151101881)

[B.CODE VÀ MÔ PHỎNG. 10](#_Toc151101882)

[1.ds1302.c 10](#_Toc151101883)

[2. lcd.c 14](#_Toc151101884)

[3. DS1302Clock LED7 15](#_Toc151101885)

[3.1 main.c 15](#_Toc151101886)

[3.2 Mô phỏng. 19](#_Toc151101887)

[4. DS1302Clock+Button 19](#_Toc151101888)

[4.1 main.c 19](#_Toc151101889)

[4.2 Mô phỏng. 25](#_Toc151101890)

[5. Hiển thị. 25](#_Toc151101891)

[5.1 main.c 25](#_Toc151101892)

[5.2 Mô phỏng. 28](#_Toc151101893)

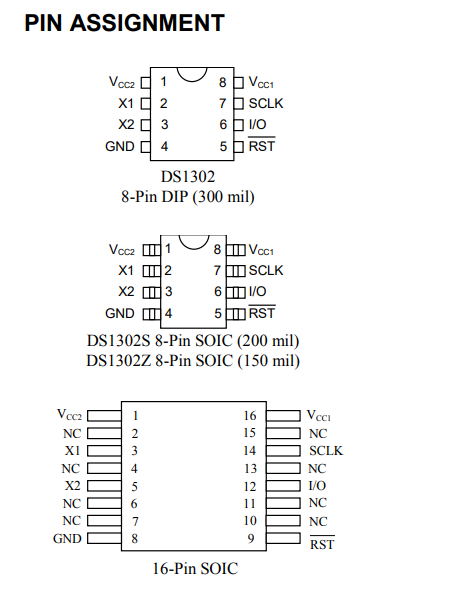
**Thời gian thực**

# A.DS1302

## 1.Giới thiệu.

* Chip chấm công sạc nhỏ giọt DS1302 là mô-đun lịch/đồng hồ thời gian thực với một số tính năng chính
* Máy chấm công: Nó theo dõi giây, phút, giờ, ngày, tháng, năm. Mô-đun tự động điều chỉnh cho những tháng ngắn hơn và những năm nhuận.
* Định dạng: Hoạt động ở định dạng 24 giờ hoặc 12 giờ với chỉ báo AM/PM.
* Bộ nhớ=: Chứa 31 byte RAM tĩnh.
* Giao tiếp nối tiếp: Giao diện với bộ vi xử lý thông qua giao tiếp nối tiếp đồng bộ, chỉ yêu cầu ba dây: RST (Đặt lại), I/O (Dòng dữ liệu) và SCLK (Đồng hồ nối tiếp).
* Truyền dữ liệu: Hỗ trợ truyền theo chế độ một byte hoặc truyền loạt lên tới 31 byte.
* Hoạt động ở mức năng lượng thấp: Lưu giữ dữ liệu và thông tin đồng hồ ở mức dưới 1 microwatt.
* Các tính năng bổ sung: So với phiên bản tiền nhiệm của nó, DS1202, nó cung cấp các chân nguồn kép cho nguồn điện chính và dự phòng, bộ sạc nhỏ giọt có thể lập trình cho VCC1 và bảy byte bộ nhớ Scratchpad bổ sung.

## 2.Tính năng.

* Giờ hiện hành : Theo dõi chính xác giây, phút, giờ, ngày, tháng, ngày trong tuần và năm với mức bù năm nhuận lên tới 2100.
* Bộ nhớ : Bao gồm RAM 31 x 8 để lưu trữ dữ liệu.
* I/O nối tiếp : Giảm thiểu việc sử dụng pin để giao tiếp hiệu quả.
* Dải điện áp : Hoạt động trong khoảng 2,0–5,5V.
* Hiệu suất năng lượng : Sử dụng ít hơn 300 nA ở 2.0V.
* Truyền dữ liệu : Hỗ trợ cả chuyển chế độ byte đơn và chế độ truyền liên tục.
* Bao bì : Có sẵn dạng DIP hoặc SOIC 8 chân để gắn trên bề mặt.
* Giao diện : Giao diện 3 dây đơn giản, tương thích với TTL.
* Phạm vi nhiệt độ : Phạm vi công nghiệp tùy chọn từ –40°C đến +85°C.
* Khả năng tương thích : Tương thích DS1202 và được Phòng thí nghiệm Underwriters công nhận.

## 3.Mô tả các chân.

|  |  |
| --- | --- |
| X1,X2 | 32.768 kHz Crystal Pins |
| GND | Nối đất. |
|  | Reset. |
| I/O | Kết nối với các thiết bị bên ngoài. |
| SCLK | Cung cấp thời gian cho giao tiếp nối tiếp. |
| VCC1, VCC2 | Các chân này cấp nguồn cho mô-đun. |

* VCC1: Cung cấp hoạt động ở mức năng lượng thấp cho cả hệ thống nguồn đơn và hệ thống chạy bằng pin. Trong các hệ thống sử dụng bộ sạc nhỏ giọt, nguồn năng lượng sạc được sẽ kết nối với chân này.
* VCC2: Chân cấp nguồn chính trong cấu hình nguồn kép. VCC1 được kết nối với nguồn dự phòng để duy trì ngày giờ mà không cần nguồn điện chính. DS1302 hoạt động từ VCC1 hoặc VCC2 lớn hơn.
* SCLK (Đầu vào đồng hồ nối tiếp): Đồng bộ hóa chuyển động dữ liệu trên giao diện nối tiếp.
* I/O (Đầu vào/Đầu ra dữ liệu): Chân dữ liệu hai chiều cho giao diện 3 dây.
* RST (Reset): Phải được xác nhận ở mức cao trong quá trình đọc hoặc ghi.
* X1, X2: Kết nối với tinh thể thạch anh 32,768 kHz. Bộ dao động bên trong được thiết kế cho các tinh thể có điện dung tải 6 pF. DS1302 cũng có thể được điều khiển bởi bộ tạo dao động 32,768 kHz bên ngoài.

## 4.Các thành phần chính.

Shift Register: Tạo điều kiện thuận lợi cho việc truyền dữ liệu nối tiếp vào và ra khỏi thiết bị.

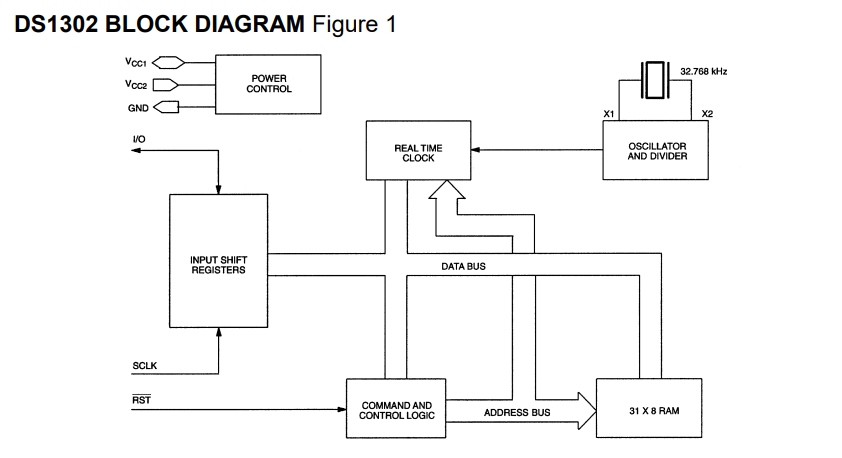
Control Logic: Quản lý các hoạt động của mô-đun, bao gồm chức năng truyền dữ liệu và đồng hồ.

Oscillator: Cung cấp tín hiệu định thời cho đồng hồ thời gian thực, thường sử dụng tinh thể 32,768 kHz bên ngoài được kết nối với các chân X1 và X2.

Real Time Clock: Duy trì thông tin ngày giờ chính xác.

RAM: 31 byte RAM tĩnh dùng để lưu trữ dữ liệu.

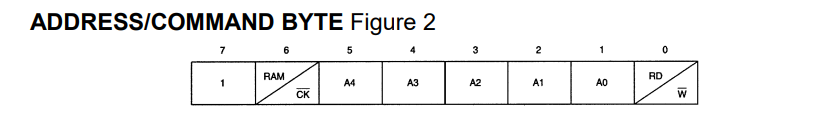
Các thành phần này hoạt động cùng nhau để đảm bảo việc chấm công và truyền dữ liệu đáng tin cậy với bộ vi xử lý thông qua giao diện nối tiếp đơn giản.



## 5.Command Byte.

Byte lệnh trong mô-đun DS1302 RTC rất quan trọng để bắt đầu truyền dữ liệu. Cấu trúc của nó như sau:

* Bit 7 (MSB): Phải được đặt thành logic 1. Nếu được đặt thành 0, nó sẽ vô hiệu hóa việc ghi vào DS1302.
* Bit 6: Chỉ định dữ liệu đồng hồ/lịch (logic 0) hoặc dữ liệu RAM (logic 1).
* Bit 1 đến 5: Chỉ định các thanh ghi được chỉ định cho đầu vào hoặc đầu ra.
* Bit 0 (LSB): Chỉ định thao tác ghi (đầu vào) nếu logic 0 hoặc thao tác đọc (đầu ra) nếu logic 1.
* Byte lệnh là đầu vào bắt đầu bằng LSB (bit 0). Cấu trúc này cho phép kiểm soát chính xác quá trình truyền dữ liệu đến và đi từ DS1302.



## 6.Các chức năng và mode hoạt động trong DS1302.

**RESET AND CLOCK CONTROL**

Chức năng Đặt lại (RST) và Điều khiển Đồng hồ trong mô-đun DS1302 RTC như sau:

Đầu vào RST: Việc bắt đầu bất kỳ quá trình truyền dữ liệu nào đều yêu cầu đẩy đầu vào RST lên cao. Đầu vào này có hai chức năng chính:

Kích hoạt logic điều khiển, cho phép truy cập vào thanh ghi thay đổi để sắp xếp thứ tự địa chỉ/lệnh.

Cung cấp một cách để chấm dứt việc truyền dữ liệu một hoặc nhiều byte.

Chu kỳ đồng hồ: Được xác định bởi một cạnh giảm theo sau là một cạnh tăng. Dữ liệu đầu vào phải hợp lệ trong cạnh tăng của đồng hồ và các bit dữ liệu được xuất ra ở cạnh giảm.

Trạng thái RST thấp: Khi RST ở mức thấp, tất cả quá trình truyền dữ liệu sẽ dừng và chân I/O trở thành trở kháng cao.

Điều kiện bật nguồn: Khi bật nguồn, RST phải ở mức logic 0 cho đến khi VCC vượt quá 2,0 volt. Ngoài ra, SCLK phải ở mức logic 0 khi RST được điều khiển về mức logic 1.

Các cơ chế này đảm bảo việc truyền dữ liệu được kiểm soát đến và đi từ DS1302, duy trì tính toàn vẹn và độ tin cậy của dữ liệu chấm công.

**DATA INPUT**

Sau 8 chu kỳ SCLK cho một byte lệnh ghi, một byte dữ liệu sẽ được đưa vào ở cạnh lên của 8 chu kỳ SCLK tiếp theo.Dữ liệu đầu vào bắt đầu bằng bit 0.Các chu kỳ SCLK bổ sung ngoài những chu kỳ này sẽ bị bỏ qua.

**DATA OUTPUT**

Sau 8 chu kỳ SCLK cho một byte lệnh đọc, một byte dữ liệu được xuất ra ở cạnh xuống của 8 chu kỳ SCLK tiếp theo.Bit dữ liệu đầu tiên được truyền ở cạnh xuống đầu tiên sau bit cuối cùng của byte lệnh.Có thể đọc ở chế độ chụp liên tục miễn là RST vẫn ở mức cao.

**BURST MODE**

Cho phép đọc hoặc ghi một chuỗi byte dữ liệu trong các thanh ghi đồng hồ/lịch hoặc RAM.Bắt đầu bằng cách đánh địa chỉ vị trí 31 với tất cả các bit trong lệnh được đặt thành logic 1.

**CLOCK/CALENDAR**

Dữ liệu được lưu trữ ở định dạng thập phân mã hóa nhị phân (BCD) trên bảy thanh ghi ghi/đọc.

**CLOCK HALT FLAG**

Bit 7 của thanh ghi giây. Đặt bit này thành logic 1 sẽ dừng đồng hồ và đặt DS1302 ở chế độ năng lượng thấp.

**AM-PM/12-24 MODE**

Bit 7 của thanh ghi giờ điều khiển định dạng 12/24 giờ. Bit 5 biểu thị AM/PM ở chế độ 12 giờ.

**WRITE PROTECT BIT**

Bit 7 của thanh ghi điều khiển. Khi được đặt, nó sẽ ngăn việc ghi vào đồng hồ hoặc RAM.

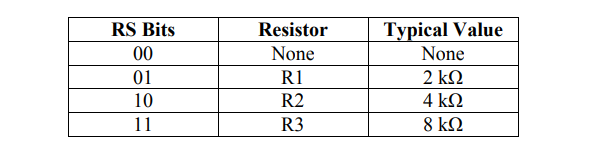
**TRICKLE CHARGE REGISTER**

Thanh ghi sạc nhỏ giọt trong DS1302 kiểm soát các tính năng sạc nhỏ giọt của nó:

Trickle Charge Select (TCS) Bits (Bit 4-7): Điều khiển việc kích hoạt bộ sạc nhỏ giọt. Chỉ có mẫu 1010 mới kích hoạt được; tất cả các mẫu khác sẽ vô hiệu hóa nó. DS1302 khởi động với bộ sạc nhỏ giọt bị vô hiệu hóa.

Diode Select (DS) Bits (Bit 2-3): Xác định việc kết nối các diode giữa VCC2 và VCC1. Nếu DS là 01, một diode được chọn hoặc nếu DS là 10, hai diode được chọn. Nếu DS là 00 hoặc 11, bộ sạc nhỏ giọt sẽ bị vô hiệu hóa bất kể cài đặt TCS.

Resistor Select (RS) Bits (Bit 0-1): Chọn điện trở kết nối giữa VCC2 và VCC1. Các giá trị cụ thể của điện trở được xác định bởi các bit RS này.



Vd: Nếu RS là 00, bộ sạc nhỏ giọt được vô hiệu hóa độc lập với TCS. Lựa chọn diode và điện trở được người dùng xác định dựa trên dòng điện tối đa mong muốn cho việc sạc pin hoặc super cap. Dòng sạc tối đa có thể được tính toán như trong ví dụ sau: Giả sử nguồn cấp hệ thống 5V được áp dụng tới VCC2 và một super cap được kết nối với VCC1. Giả sử bộ sạc nhỏ giọt được kích hoạt với một diode và điện trở R1 giữa VCC2 và VCC1. Dòng tối đa Imax sẽ được tính như sau: Imax = (5.0V - sụt điện áp diode) / R1 ~ (5.0V - 0.7V) / 2 kΩ ~ 2.2 mA. Rõ ràng, khi super cap sạc, sụt điện áp giữa VCC2 và VCC1 sẽ giảm và do đó dòng sạc cũng sẽ giảm.

**CLOCK/CALENDAR BURST MOD**

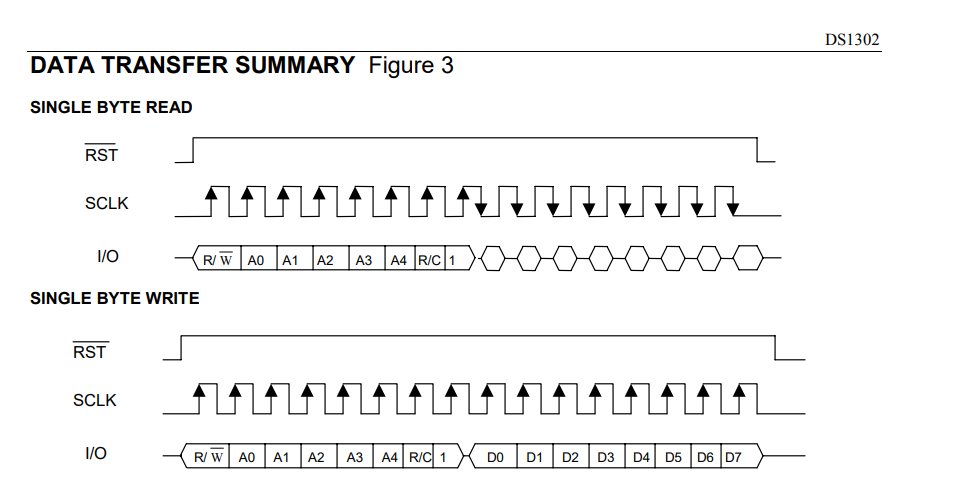
Chế độ Burst của Clock/Calendar và các tính năng khác của mô-đun DS1302 hoạt động như sau:

Chế độ Burst Clock/Calendar: Cho phép đọc hoặc viết liên tiếp tám thanh ghi đầu tiên của đồng hồ/lịch, bắt đầu từ bit 0 của địa chỉ 0. Nếu bit bảo vệ ghi được thiết lập cao, không có truyền dữ liệu nào xảy ra.

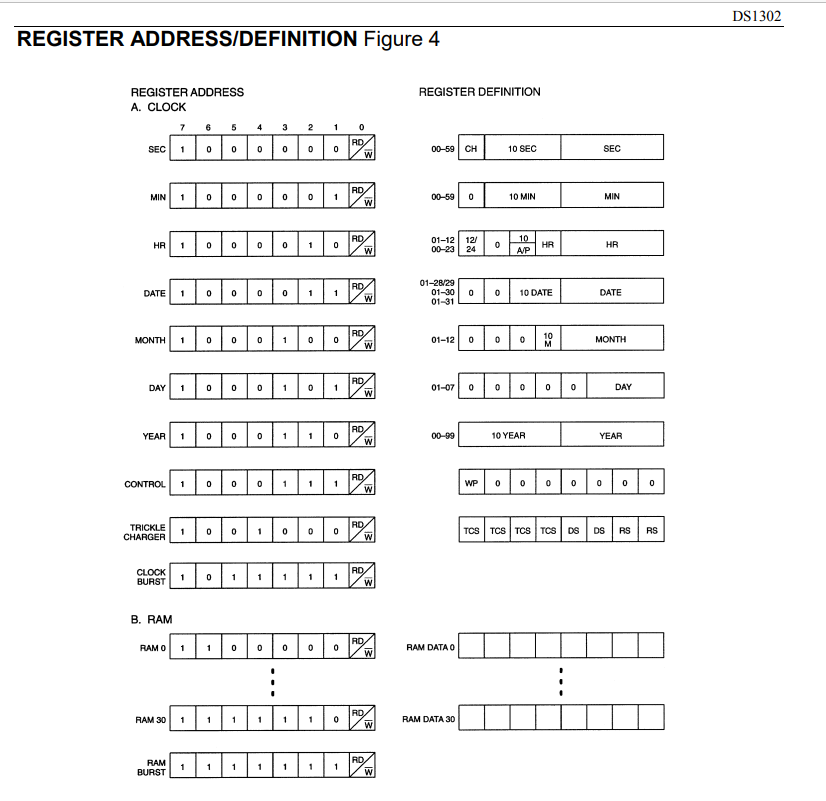
RAM: Bao gồm 31 x 8 byte, có thể địa chỉ hóa liên tiếp trong không gian RAM.

Chế độ Burst RAM: Cho phép đọc hoặc viết liên tiếp 31 thanh ghi RAM.

Lựa Chọn Crystal: Một crystal 32.768 kHz với dung kháng quy định là 6 pF có thể được kết nối trực tiếp qua X1 và X2. Chi tiết hơn có thể tìm thấy trong Ghi chú Ứng dụng 58, "Crystal Considerations with Dallas Real Time Clocks."



Ở chế độ cụm, được giữ ở mức cao và các chu kỳ SCLK bổ sung được gửi cho đến khi kết thúc cụm.



# B.CODE VÀ MÔ PHỎNG.

## 1.ds1302.c

#include "ds1302.h"

//---Các địa chỉ để đọc và ghi thời gian từ DS1302---//

//---Thứ tự thời gian là giây, phút, giờ, ngày, tháng, năm, ngày trong tuần;---

uchar code READ\_RTC\_ADDR[7] = {0x81, 0x83, 0x85, 0x87, 0x89, 0x8b, 0x8d};

uchar code WRITE\_RTC\_ADDR[7] = {0x80, 0x82, 0x84, 0x86, 0x88, 0x8a, 0x8c};

//---Khởi tạo thời gian ban đầu là 01/01/2013 12:00:00---//

//---Thời gian được lưu theo định dạng BCD---

uchar TIME[7] = {0, 0, 0x10, 0x01, 0x11, 0x10, 0x13};

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Hàm: Ds1302Write

\* Chức năng: Ghi lệnh và dữ liệu vào DS1302 (địa chỉ + dữ liệu)

\* Tham số đầu vào: addr (địa chỉ), dat (dữ liệu)

\* Trả về: Không có

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void Ds1302Write(uchar addr, uchar dat)

{

    uchar n;

    RST = 0;

    \_nop\_();

    SCLK = 0;  // Đặt chân SCLK xuống mức thấp để bắt đầu giao tiếp.

    \_nop\_();

    RST = 1;  // Đặt chân RST (CE) lên mức cao để bắt đầu truyền dữ liệu.

    \_nop\_();

    for (n = 0; n < 8; n++)  // Bắt đầu gửi địa chỉ (addr) bit từng bit

    {

        DSIO = addr & 0x01;  // Gửi dữ liệu từ bit thấp nhất

        addr >>= 1;

        SCLK = 1;  // Gửi dữ liệu khi SCLK ở mức cao, DS1302 lấy dữ liệu

        \_nop\_();

        SCLK = 0;

        \_nop\_();

    }

    for (n = 0; n < 8; n++)  // Gửi 8 bit dữ liệu

    {

        DSIO = dat & 0x01;

        dat >>= 1;

        SCLK = 1;

        \_nop\_();

        SCLK = 0;

        \_nop\_();

    }

    RST = 0;  // Kết thúc việc gửi dữ liệu.

    \_nop\_();

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Hàm: Ds1302Read

\* Chức năng: Đọc dữ liệu từ DS1302 ở một địa chỉ cụ thể

\* Tham số đầu vào: addr (địa chỉ)

\* Trả về: dat (dữ liệu đọc được)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

uchar Ds1302Read(uchar addr)

{

    uchar n, dat, dat1;

    RST = 0;

    \_nop\_();

    SCLK = 0;  // Đặt chân SCLK xuống mức thấp để bắt đầu giao tiếp.

    \_nop\_();

    RST = 1;  // Đặt chân RST (CE) lên mức cao để bắt đầu truyền dữ liệu.

    \_nop\_();

    for (n = 0; n < 8; n++)  // Bắt đầu gửi địa chỉ (addr) bit từng bit

    {

        DSIO = addr & 0x01;  // Gửi dữ liệu từ bit thấp nhất

        addr >>= 1;

        SCLK = 1;  // Gửi dữ liệu khi SCLK ở mức cao, DS1302 lấy dữ liệu

        \_nop\_();

        SCLK = 0;

        \_nop\_();

    }

    \_nop\_();

    for (n = 0; n < 8; n++)  // Đọc 8 bit dữ liệu

    {

        dat1 = DSIO;  // Đọc dữ liệu từ bit thấp nhất

        dat = (dat >> 1) | (dat1 << 7);

        SCLK = 1;

        \_nop\_();

        SCLK = 0;

        \_nop\_();

    }

    RST = 0;

    \_nop\_();  // Kết thúc việc đọc dữ liệu.

    SCLK = 1;

    \_nop\_();

    DSIO = 0;

    \_nop\_();

    DSIO = 1;

    \_nop\_();

    return dat;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Hàm: Ds1302Init

\* Chức năng: Khởi tạo DS1302

\* Tham số đầu vào: Không có

\* Trả về: Không có

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void Ds1302Init()

{

    uchar n;

    Ds1302Write(0x8E, 0X00);  // Tắt việc bảo vệ ghi, chỉ đọc - Tạm thời vô hiệu hóa ghi

    for (n = 0; n < 7; n++)  // Ghi dữ liệu thời gian mặc định (năm 2013, tháng 1, ngày 1, giờ 12:00:00)

    {

        Ds1302Write(WRITE\_RTC\_ADDR[n], TIME[n]);

    }

    Ds1302Write(0x8E, 0x80);  // Bật lại việc bảo vệ ghi

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Hàm: Ds1302ReadTime

\* Chức năng: Đọc thời gian từ DS1302

\* Tham số đầu vào: Không có

\* Trả về: Không có

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void Ds1302ReadTime()

{

    uchar n;

    for (n = 0; n < 7; n++)  // Đọc 7 giá trị thời gian từ DS1302 và lưu vào mảng TIME

    {

        TIME[n] = Ds1302Read(READ\_RTC\_ADDR[n]);

    }

}

## 2. lcd.c

#include "lcd.h"

/\*

 \* Hàm: Lcd1602\_Delay1ms

 \* Ch?c nang: T?o d? tr? 1ms

 \* Tham s? d?u vào: c (s? l?n d?i)

 \* Tr? v?: Không có

 \* Ghi chú: Hàm này dùng d? t?o d? tr? 1ms, tùy thu?c vào t?n s? xung nh?p h? th?ng.

 \*/

void Lcd1602\_Delay1ms(uint c)

{

    uchar a, b;

    for (; c > 0; c--)

    {

        for (b = 199; b > 0; b--)

        {

            for (a = 1; a > 0; a--);

        }

    }

}

/\*

 \* Hàm: LcdWriteCom

 \* Ch?c nang: G?i l?nh d?n màn hình LCD

 \* Tham s? d?u vào: com (l?nh c?n g?i)

 \* Tr? v?: Không có

 \*/

#ifndef LCD1602\_4PINS

void LcdWriteCom(uchar com)

{

    // B?t d?u truy?n l?nh

    LCD1602\_E = 0;

    LCD1602\_RS = 0;

    LCD1602\_RW = 0;

    // G?i l?nh qua các chân d? li?u

    LCD1602\_DATAPINS = com;

    Lcd1602\_Delay1ms(1);

    // Hoàn thành vi?c g?i l?nh

    LCD1602\_E = 1;

    Lcd1602\_Delay1ms(5);

    LCD1602\_E = 0;

}

#else

void LcdWriteCom(uchar com)

{

    // B?t d?u truy?n l?nh (ch? d? 4-bit)

    LCD1602\_E = 0;

    LCD1602\_RS = 0;

    LCD1602\_RW = 0;

    // G?i 4-bit d?u tiên c?a l?nh

    LCD1602\_DATAPINS = com;

    Lcd1602\_Delay1ms(1);

    // G?i 4-bit còn l?i

    LCD1602\_E = 1;

    Lcd1602\_Delay1ms(5);

    LCD1602\_E = 0;

    LCD1602\_DATAPINS = com << 4;

    Lcd1602\_Delay1ms(1);

    LCD1602\_E = 1;

    Lcd1602\_Delay1ms(5);

    LCD1602\_E = 0;

}

#endif

## 3. DS1302Clock LED7

### 3.1 main.c

#include "lcd.h"

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Hàm: Lcd1602\_Delay1ms

\* Chức năng: Delay mili giây

\* Tham số đầu vào: c (thời gian delay)

\* Trả về: Không có

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void Lcd1602\_Delay1ms(uint c)

{

    uchar a, b;

    for (; c > 0; c--)

    {

        for (b = 199; b > 0; b--)

        {

            for (a = 1; a > 0; a--);

        }

    }

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Hàm: LcdWriteCom

\* Chức năng: Gửi lệnh đến màn hình LCD

\* Tham số đầu vào: com (lệnh)

\* Trả về: Không có

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#ifndef LCD1602\_4PINS

void LcdWriteCom(uchar com)

{

    LCD1602\_E = 0;

    LCD1602\_RS = 0;

    LCD1602\_RW = 0;

    LCD1602\_DATAPINS = com;

    Lcd1602\_Delay1ms(1);

    LCD1602\_E = 1;

    Lcd1602\_Delay1ms(5);

    LCD1602\_E = 0;

}

#else

void LcdWriteCom(uchar com)

{

    LCD1602\_E = 0;

    LCD1602\_RS = 0;

    LCD1602\_RW = 0;

    LCD1602\_DATAPINS = com;

    Lcd1602\_Delay1ms(1);

    LCD1602\_E = 1;

    Lcd1602\_Delay1ms(5);

    LCD1602\_E = 0;

    Lcd1602\_Delay1ms(1);

    LCD1602\_DATAPINS = com << 4;

    Lcd1602\_Delay1ms(1);

    LCD1602\_E = 1;

    Lcd1602\_Delay1ms(5);

    LCD1602\_E = 0;

}

#endif

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Hàm: LcdWriteData

\* Chức năng: Gửi dữ liệu đến màn hình LCD

\* Tham số đầu vào: dat (dữ liệu)

\* Trả về: Không có

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#ifndef LCD1602\_4PINS

void LcdWriteData(uchar dat)

{

    LCD1602\_E = 0;

    LCD1602\_RS = 1;

    LCD1602\_RW = 0;

    LCD1602\_DATAPINS = dat;

    Lcd1602\_Delay1ms(1);

    LCD1602\_E = 1;

    Lcd1602\_Delay1ms(5);

    LCD1602\_E = 0;

}

#else

void LcdWriteData(uchar dat)

{

    LCD1602\_E = 0;

    LCD1602\_RS = 1;

    LCD1602\_RW = 0;

    LCD1602\_DATAPINS = dat;

    Lcd1602\_Delay1ms(1);

    LCD1602\_E = 1;

    Lcd1602\_Delay1ms(5);

    LCD1602\_E = 0;

    LCD1602\_DATAPINS = dat << 4;

    Lcd1602\_Delay1ms(1);

    LCD1602\_E = 1;

    Lcd1602\_Delay1ms(5);

    LCD1602\_E = 0;

}

#endif

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Hàm: LcdInit

\* Chức năng: Khởi tạo màn hình LCD

\* Tham số đầu vào: Không có

\* Trả về: Không có

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#ifndef LCD1602\_4PINS

void LcdInit()

{

    LcdWriteCom(0x38);  // Hiển thị 2 dòng, font 5x7

    LcdWriteCom(0x0c);  // Tắt con trỏ

    LcdWriteCom(0x06);  // Con trỏ di chuyển về bên phải sau mỗi lần ghi

    LcdWriteCom(0x01);  // Xóa màn hình và đặt con trỏ về vị trí ban đầu

    LcdWriteCom(0x80);  // Đặt con trỏ về vị trí đầu tiên của dòng 1

}

#else

void LcdInit()

{

    LcdWriteCom(0x32);  // Chuyển 8-bit bus sang 4-bit bus

    LcdWriteCom(0x28);  // Hiển thị 2 dòng, font 5x7

    LcdWriteCom(0x0c);  // Tắt con trỏ

    LcdWriteCom(0x06);  // Con trỏ di chuyển về bên phải sau mỗi lần ghi

    LcdWriteCom(0x01);  // Xóa màn hình và đặt con trỏ về vị trí ban đầu

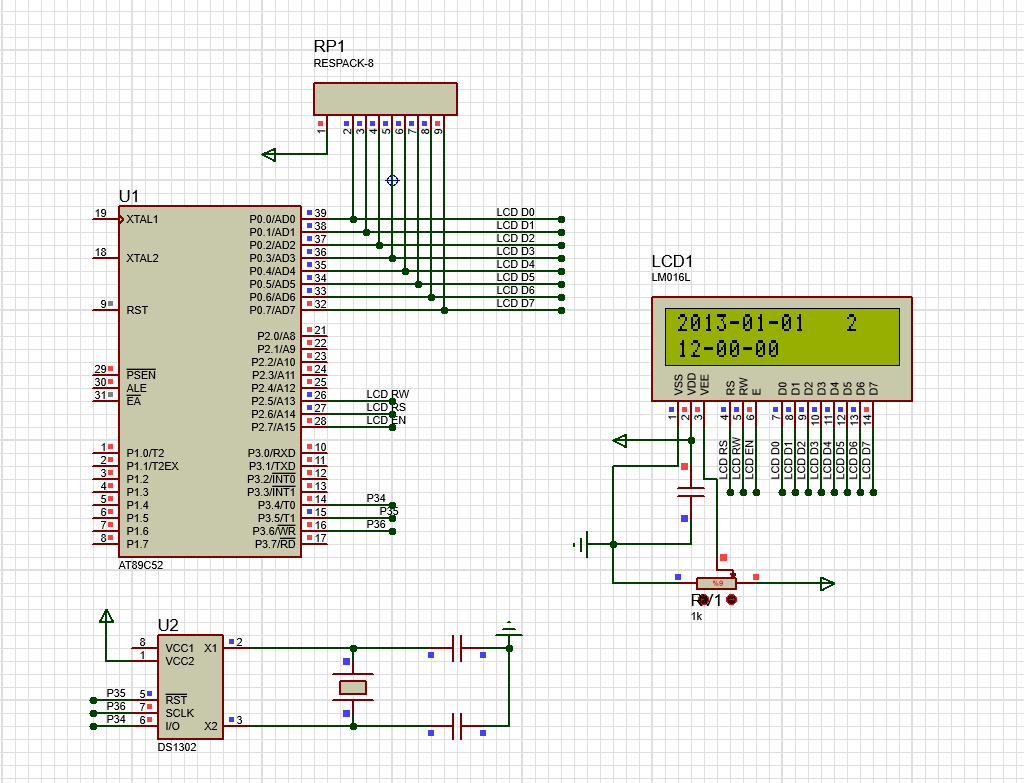
    LcdWriteCom(0x80);  // Đặt con trỏ về vị trí đầu tiên của dòng 1

}

#endif

Đây là một chương trình đơn giản để hiển thị thời gian từ mô-đun DS1302 lên một màn hình LCD 16x2. Hàm LcdDisplay() được sử dụng để hiển thị các giá trị thời gian lên màn hình LCD.

### 3.2 Mô phỏng.



## 4. DS1302Clock+Button

### 4.1 main.c

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Tên dự án       : Đồng hồ thời gian thực

\* Sử dụng các chân IO :

\* Chức năng        : Hiển thị thời gian trên màn hình LCD 16x2, K3 chuyển giữa trạng thái đặt giờ và hiển thị giờ, K1 và K2 để đặt thời gian

\* Lưu ý             : Không có

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include<reg51.h>

#include"lcd.h"

#include"ds1302.h"

sbit K1 = P3^1;

sbit K2 = P3^0;

sbit K3 = P3^2;

sbit K4 = P3^3;

void Int0Configuration();

void LcdDisplay();

unsigned char SetState, SetPlace;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Hàm: main

\* Chức năng: Hàm chính của chương trình

\* Đầu vào: Không có

\* Đầu ra: Không có

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void main()

{

    unsigned char i;

    Int0Configuration();

    LcdInit();

    Ds1302Init();

    while(1)

    {

        if(SetState == 0)

        {

            Ds1302ReadTime();

        }

        else

        {

            if(K1 == 0)     // Kiểm tra xem K1 có được nhấn không

            {

                Delay10ms();    // Đợi chống nhiễu

                if(K1 == 0)

                {

                    SetPlace++;

                    if(SetPlace >= 7)

                        SetPlace = 0;

                }

                while((i < 50) && (K1 == 0))     // Kiểm tra xem nút đã được thả chưa

                {

                    Delay10ms();

                    i++;

                }

                i = 0;

            }

            if(K2 == 0)     // Kiểm tra xem K2 có được nhấn không

            {

                Delay10ms();    // Đợi chống nhiễu

                if(K2 == 0)

                {

                    TIME[SetPlace]++;

                    if((TIME[SetPlace] & 0x0f) > 9)                 // Xử lý tràn BCD

                    {

                        TIME[SetPlace] = TIME[SetPlace] + 6;

                    }

                    if((TIME[SetPlace] >= 0x60) && (SetPlace < 2))    // Giờ chỉ cho phép đến 59

                    {

                        TIME[SetPlace] = 0;

                    }

                    if((TIME[SetPlace] >= 0x24) && (SetPlace == 2))   // Giờ chỉ cho phép đến 23

                    {

                        TIME[SetPlace] = 0;

                    }

                    if((TIME[SetPlace] >= 0x32) && (SetPlace == 3))   // Ngày chỉ cho phép đến 31

                    {

                        TIME[SetPlace] = 0;

                    }

                    if((TIME[SetPlace] >= 0x13) && (SetPlace == 4))   // Tháng chỉ cho phép đến 12

                    {

                        TIME[SetPlace] = 0;

                    }

                    if((TIME[SetPlace] >= 0x7) && (SetPlace == 5))    // Thứ chỉ cho phép đến 7

                    {

                        TIME[SetPlace] = 1;

                    }

                }

                while((i < 50) && (K2 == 0))     // Kiểm tra xem nút đã được thả chưa

                {

                    Delay10ms();

                    i++;

                }

                i = 0;

            }

        }

    LcdDisplay();

    }

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Hàm: LcdDisplay

\* Chức năng: Hiển thị thời gian trên màn hình LCD

\* Đầu vào: Không có

\* Đầu ra: Không có

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void LcdDisplay()

{

    LcdWriteCom(0x80 + 0X40);

    LcdWriteData('0' + TIME[2] / 16);        // Giờ

    LcdWriteData('0' + (TIME[2] & 0x0f));

    LcdWriteData('-');

    LcdWriteData('0' + TIME[1] / 16);        // Tháng

    LcdWriteData('0' + (TIME[1] & 0x0f));

    LcdWriteData('-');

    LcdWriteData('0' + TIME[0] / 16);        // Ngày

    LcdWriteData('0' + (TIME[0] & 0x0f));

    LcdWriteCom(0x80);

    LcdWriteData('2');

    LcdWriteData('0');

    LcdWriteData('0' + TIME[6] / 16);       // Năm

    LcdWriteData('0' + (TIME[6] & 0x0f));

    LcdWriteData('-');

    LcdWriteData('0' + TIME[4] / 16);       // Phút

    LcdWriteData('0' + (TIME[4] & 0x0f));

    LcdWriteData('-');

    LcdWriteData('0' + TIME[3] / 16);       // Giây

    LcdWriteData('0' + (TIME[3] & 0x0f));

    LcdWriteCom(0x8D);

    LcdWriteData('0' + (TIME[5] & 0x07));  // Thứ

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Hàm: Int0Configuration

\* Chức năng: Cấu hình ngắt ngoại vi 0

\* Đầu vào: Không có

\* Đầu ra: Không có

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void Int0Configuration()

{

    // Cấu hình INT0

    IT0 = 1;    // Ngắt theo cạnh xuống để phát hiện nhấn nút

    EX0 = 1;    // Cho phép ngắt INT0

    EA = 1;     // Cho phép ngắt toàn cầu

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Hàm: Int0

\* Chức năng: Ngắt ngoại vi 0 xử lý khi nút K3 được nhấn

\* Đầu vào: Không có

\* Đầu ra: Không có

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void Int0() interrupt 0

{

    Delay10ms();

    if(K3 == 0)

    {

        SetState = ~SetState;   // Chuyển đổi trạng thái đặt giờ và hiển thị giờ

        SetPlace = 0;           // Thiết lập vị trí đặt giờ về 0

        Ds1302Init();           // Khởi tạo lại DS1302 để cập nhật thời gian

    }

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Hàm: Delay10ms

\* Chức năng: Tạo độ trễ 10ms

\* Đầu vào: Không có

\* Đầu ra: Không có

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void Delay10ms(void)

{

    unsigned char a, b, c;

    for(c = 1; c > 0; c--)

        for(b = 38; b > 0; b--)

            for(a = 130; a > 0; a--);

}

Chức năng chính của chương trình là:

Hiển thị thời gian trên màn hình LCD: Sử dụng hàm LcdDisplay để hiển thị giờ, phút, giây, ngày, tháng, năm và thứ lên màn hình LCD.

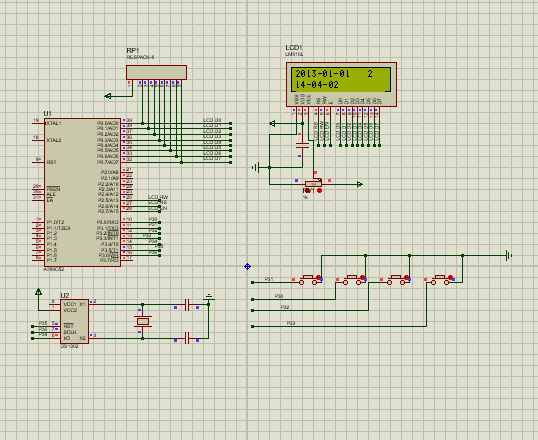
Đọc và cài đặt thời gian từ module DS1302: Sử dụng các hàm Ds1302ReadTime, Ds1302Write, Ds1302Init để đọc thời gian từ DS1302 và cài đặt thời gian mặc định nếu ở chế độ đặt giờ

Xử lý các nút nhấn: Sử dụng nút nhấn K1, K2 và K3 để thực hiện các chức năng sau:

* K1: Chuyển giữa các vị trí của thời gian (giờ, phút, giây, ngày, tháng, năm, thứ).
* K2: Tăng giá trị ở vị trí được chọn của thời gian.
* K3: Chuyển giữa chế độ hiển thị thời gian và chế độ đặt giờ.

Cài đặt ngắt ngoại vi INT0: Sử dụng hàm Int0Configuration để cấu hình ngắt ngoại vi INT0 để xử lý sự kiện khi nút K3 được nhấn.

### 4.2 Mô phỏng.



## 5. Hiển thị.

### 5.1 main.c

#include <reg51.h>

#include "ds1302.h"

// Định nghĩa thanh ghi xuất 7 đoạn LED và các chân chọn đoạn

#define DIG P0

sbit LSA = P2^2;

sbit LSB = P2^3;

sbit LSC = P2^4;

// Mã hiển thị của các số từ 0 đến 9

unsigned char code DIG\_CODE[10] = {0x3f, 0x06, 0x5b, 0x4f, 0x66, 0x6d, 0x7d, 0x07, 0x7f, 0x6f};

unsigned char Num = 0;

unsigned int disp[8] = {0x3f, 0x3f, 0x3f, 0x3f, 0x3f, 0x3f, 0x3f, 0x3f};

// Khai báo các hàm

void LcdDisplay();

void Timer0Configuration();

// Hàm chính

void main()

{

    // Khởi tạo DS1302 và cấu hình Timer0

    Ds1302Init();

    Timer0Configuration();

    // Vòng lặp chính

    while (1)

    {

        // Đọc thời gian từ DS1302

        Ds1302ReadTime();

        // Thiết lập mã hiển thị cho từng đoạn LED

        disp[7] = DIG\_CODE[TIME[0] & 0x0f];

        disp[6] = DIG\_CODE[TIME[0] >> 4];

        disp[5] = 0X40;

        disp[4] = DIG\_CODE[TIME[1] & 0x0f];

        disp[3] = DIG\_CODE[TIME[1] >> 4];

        disp[2] = 0X40;

        disp[1] = DIG\_CODE[TIME[2] & 0x0f];

        disp[0] = DIG\_CODE[TIME[2] >> 4];

    }

}

// Hàm cấu hình Timer0

void Timer0Configuration()

{

    TMOD = 0X02;  // Chế độ 2 - 8-bit tự động nạp lại

    TH0 = 0X9C;   // Thiết lập giá trị đặt trước cho Timer0

    TL0 = 0X9C;   // (để tạo độ trễ và tần số ngắt)

    ET0 = 1;      // Cho phép ngắt Timer0

    EA = 1;       // Cho phép ngắt chung

    TR0 = 1;      // Khởi động Timer0

}

// Ngắt Timer0 - Hiển thị lần lượt trên 7 đoạn LED

void DigDisplay() interrupt 1

{

    DIG = 0;  // Tắt tất cả đoạn LED

    // Chọn đoạn LED dựa trên biến Num

    switch (Num)

    {

        case (7): LSA = 0; LSB = 0; LSC = 0; break;

        case (6): LSA = 1; LSB = 0; LSC = 0; break;

        case (5): LSA = 0; LSB = 1; LSC = 0; break;

        case (4): LSA = 1; LSB = 1; LSC = 0; break;

        case (3): LSA = 0; LSB = 0; LSC = 1; break;

        case (2): LSA = 1; LSB = 0; LSC = 1; break;

        case (1): LSA = 0; LSB = 1; LSC = 1; break;

        case (0): LSA = 1; LSB = 1; LSC = 1; break;

    }

    DIG = disp[Num];  // Hiển thị số tương ứng trên đoạn LED

    Num++;

    if (Num > 7)

        Num = 0;  // Reset Num để quét từ đoạn LED đầu tiên

}

Chương trình này có chức năng chính là hiển thị thời gian từ module DS1302 lên một hiển thị 7 đoạn LED. Cụ thể:

Đọc Thời Gian từ DS1302:

Sử dụng hàm Ds1302Init để khởi tạo module DS1302.Trong vòng lặp chính (while(1)), sử dụng hàm Ds1302ReadTime để đọc thời gian từ DS1302 và lưu vào mảng TIME.

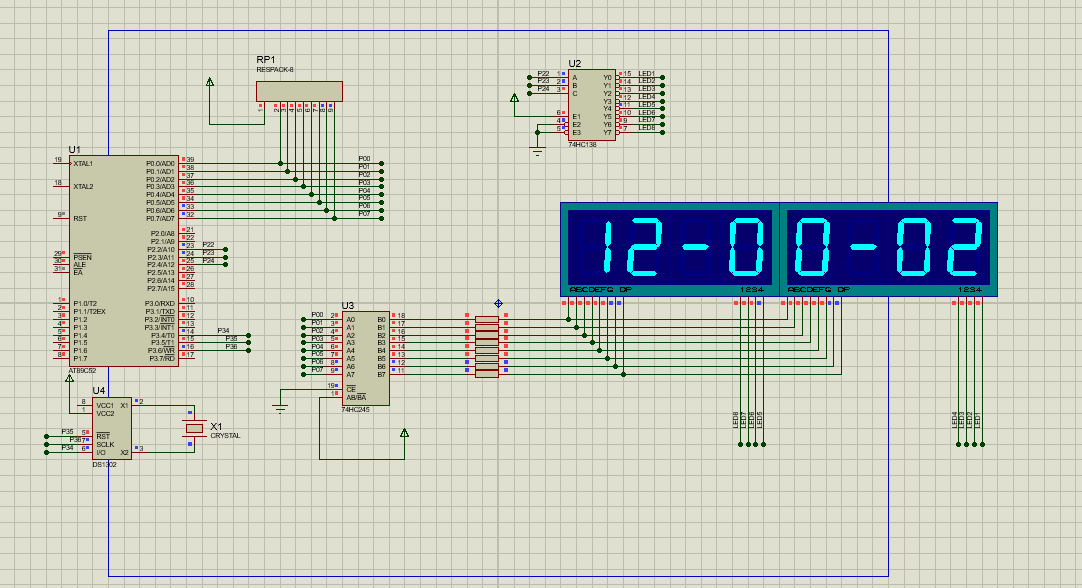
Hiển Thị Thời Gian trên 7 Đoạn LED:

Sử dụng ngắt Timer0 và hàm DigDisplay để quét và hiển thị lần lượt các số giờ, phút, giây trên 7 đoạn LED.Mã hiển thị của các số từ 0 đến 9 được lưu trong mảng DIG\_CODE.Các số được lấy từ mảng TIME và chuyển đổi thành mã hiển thị tương ứng để hiển thị lên đoạn LED.

Cấu Hình Timer0:

Sử dụng hàm Timer0Configuration để cấu hình Timer0 ở chế độ 2 (8-bit tự động nạp lại).Timer0 được sử dụng để tạo ngắt định kỳ, quét lần lượt các đoạn LED để hiển thị thời gian.Chức năng chính của chương trình là duy trì hiển thị thời gian thực lên 7 đoạn LED và cập nhật liên tục theo thời gian từ module DS1302.

### 5.2 Mô phỏng.



ở bài này dùng thêm 2 con ic là 74hc138 và 74hc245.

74HC138:

Tác dụng chính: Là một IC giải mã 3-tới-8, 74HC138 được sử dụng để mở rộng các cổng đầu ra của AT89C52 để kết nối với các LED 7-segment.Cách hoạt động: AT89C52 sẽ tạo ra 3 tín hiệu đầu vào cho 74HC138 (ví dụ: A0, A1, A2), và 74HC138 sẽ chuyển đổi chúng thành 8 đường đầu ra tương ứng. Điều này giúp kiểm soát 8 LED trên mỗi LED 7-segment với số lượng chân đầu ra ít hơn.

74HC245:

Tác dụng chính: Là một IC truyền dữ liệu 8-bit, 74HC245 được sử dụng để truyền dữ liệu từ AT89C52 đến LED 7-segment thông qua các dây dẫn.Cách hoạt động: Khi AT89C52 cần truyền dữ liệu đến LED 7-segment, nó sẽ gửi dữ liệu qua 74HC245, và sau đó 74HC245 sẽ chuyển đưa dữ liệu này đến các LED 7-segment thông qua các chân đầu ra.

LED 7-segment:

Tác dụng chính: Hiển thị thông tin thời gian từ DS1302 và AT89C52 dưới dạng các số 0-9 trên mỗi LED 7-segment.

Cách hoạt động: AT89C52 sẽ gửi dữ liệu số lên LED 7-segment thông qua 74HC245 và sử dụng 74HC138 để kiểm soát việc hiển thị số trên các LED.

Quy trình tổng quan:

AT89C52 đọc thời gian từ DS1302.

Dữ liệu số cần hiển thị được chuyển đến LED 7-segment thông qua 74HC245.

74HC138 được sử dụng để kiểm soát việc hiển thị số trên LED 7-segment.