**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT** **THÀNH PHỐ HỒ CHÍ MINH**

**KHOA ĐÀO TẠO CHẤT LƯỢNG CAO**



**ĐỒ ÁN MÔN HỌC 1**

**ĐỒNG HỒ BÁO THỨC THỜI GIAN THỰC SỬ DỤNG VI ĐIỀU KHIỂN PIC16F887**

**SVTH** : **MSSV:**

**ĐƯỜNG VỶ LUÂN 13119098**

**HỒ NHẬT TÂN 13119198**

**GVHD**: **ThS. ĐẬU TRỌNG HIỂN**

**Khoá** : **2013**

**Lớp: 13119CL2**

**Ngành** : **CÔNG NGHỆ KỸ THUẬT MÁY TÍNH**

**Tp. Hồ Chí Minh, tháng 6 năm 2016**

|  |  |
| --- | --- |
| LOGO CLC.JPG | CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM  Độc lập – Tự do – Hạnh phúc  ----\*\*\*---- |

Tp. Hồ Chí Minh, ngày 29 tháng 5 năm 2016

# **NHIỆM VỤ ĐỒ ÁN MÔN HỌC**

|  |  |
| --- | --- |
| Họ và tên sinh viên: Đường Vỷ Luân | MSSV:13119098 |
| Họ và tên sinh viên: Hồ Nhật Tân  Ngành: Công Nghệ Kỹ thuật máy tính | MSSV:13119198  Lớp: 13119CL2 |
| Giảng viên hướng dẫn: ThS. Đậu Trọng Hiển | ĐT: 0983712862 |
| Ngày nhận đề tài: 1/2/2016 | Ngày nộp đề tài: 20/6/2016 |
|  |  |
| 1. Tên đề tài : ĐỒNG HỒ THỜI GIAN THỰC |  |
| 2. Các số liệu, tài liệu ban đầu: |  |
| 3. Nội dung thực hiện đề tài: Thiết kế đồng hồ số thời gian thực sử dụng vi điều khiển. Tìm hiểu và thiết kế phần cứng, viết chương trình điều khiển hoạt động. |  |
| 4. Sản phẩm: Đồng hồ số thời gian thực có báo thức |  |
|  | GIẢNG VIÊN HƯỚNG DẪN |

|  |  |
| --- | --- |
| Logo CLC | **CỘNG HOÀ XÃ HỘI CHỦ NGHĨA VIỆT NAM**  **Độc lập – Tự do – Hạnh Phúc**  \*\*\*\*\*\*\* |
|  |  |

# **PHIẾU NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN**

Họ và tên Sinh viên: MSSV:

Ngành:

Tên đề tài:

Họ và tên Giáo viên hướng dẫn:

**NHẬN XÉT**

1. Về nội dung đề tài & khối lượng thực hiện:

1. Ưu điểm:

1. Khuyết điểm:

1. Đề nghị cho bảo vệ hay không?

1. Đánh giá loại:

1. Điểm:……………….(Bằng chữ: )

Tp*. Hồ Chí Minh, ngày tháng năm 20…*

Giáo viên hướng dẫn

*(Ký & ghi rõ họ tên)*

|  |  |
| --- | --- |
| Logo CLC | **CỘNG HOÀ XÃ HỘI CHỦ NGHĨA VIỆT NAM**  **Độc lập – Tự do – Hạnh Phúc**  \*\*\*\*\*\*\* |

# **PHIẾU NHẬN XÉT CỦA GIÁO VIÊN PHẢN BIỆN**

Họ và tên Sinh viên: MSSV:

Họ và tên Sinh viên: ………………………………………… MSSV:

Ngành:

Tên đề tài:

Họ và tên Giáo viên phản biện:

**NHẬN XÉT**

1. Về nội dung đề tài & khối lượng thực hiện:

1. Ưu điểm:

1. Khuyết điểm:

1. Đề nghị cho bảo vệ hay không?

1. Đánh giá loại:

1. Điểm:……………….(Bằng chữ: )

Tp*. Hồ Chí Minh, ngày tháng năm 20*

Giáo viên phản biện

(Ký & ghi rõ họ tên)

# **LỜI CẢM ƠN**

Trong suốt khoá học (2013-2016) tại Trường Đại học Sư Phạm Kỹ Thuật TP.HCM, em đã nhận được sự tận tình dạy dỗ của quý Thầy Cô về kiến thức chuyên môn cũng như những kiến thức trong cuộc sống. Từ những kiến thức đó đã giúp em hoàn thành Đồ Án trong thời gian cho phép.

Em xin chân thành cảm ơn các Thầy Cô trong Khoa Chất Lượng Cao đã giảng dạy em những kiến thức về chuyên môn và giúp em định hướng theo sự hiểu biết và khả năng để em thực hiện tốt đề tài “Đồng hồ báo thức thời gian thực dùng PIC16F887 hiển thị led 7 đoạn”và tạo điều kiện thuận lợi cho em hoàn tất khoá học.

Em xin gửi lời cảm ơn đến ***Thầy Đậu Trọng Hiển***, người đã hướng dẫn, giúp đỡ chúng em tận tình và tạo mọi điều kiện tốt cho em thực hiện Đồ Án này.

Ngoài ra, em cũng hết lòng biết ơn Gia Đình, Bạn Bè đã hết lòng giúp đỡ, động viên, ủng hộ và đóng góp ý kiến cho em trong suốt thời gian thực hiện Đồ Án này.

Em xin chân thành cảm ơn!

TP. HCM, ngày 29 tháng 5 năm 2016

Sinh viên thực hiện đề tài

# **DANH MỤC CÁC CHỮ VIẾT TẮT**

**LED** : Light Emitting Diode

**PWM***:* Pulse-width modulation

# **DANH MỤC CÁC BẢNG BIỂU**

Bảng 2.1. Cấu hình của các dòng Pic

Bảng 2.2. Bộ nhớ dữ liệu

Bảng 2.3. Tổ chức các thanh ghi thời gian

Bảng 2.4. Mã led 7 đoạn anode chung

# **DANH MỤC CÁC HÌNH ẢNH**

Hình 2.1. Sơ đồ chân của vi điều khiển

Hình 2.2. Hình ảnh thực của PIC

Hình 2.3. . Cấu trúc bên trong của vi điều khiển

Hình 2.4. Bộ nhớ chương trình PIC16F887

Hình 2.5. Các thanh ghi chức năng PIC16F887

Hình 2.6 Cấu trúc của GPIO

Hình 2.7. Bộ định thời

Hình 2.8. Hai gói cấu tạo chip DS1307

Hình 2.9. Kết nối phần cứng DS1307

Hình 2.10. Chân C1815

Hình 2.11. Chân BC548

Hình 2.12. Chân C1815

Hình 2.13 Cấu tạo led 7 đoạn anode

Hình 3.1. Sơ đồ khối hệ thống

Hình 3.2. Adapter 5V, 3A

Hình 3.3 sơ đồ nối chân led 7 đoạn

Hình 3.4 Khối cảnh báo buzzer

Hình 3.5 DS1307

Hình 3.6 các chân nút nhấn

Hình 3.7 Khối xử lý trung tâm

Hình 3.8 Lưu đồ giải mã ngày tháng năm

Hình 3.9 Lưu đồ quét led 7 đoạn

Hình 3.10 Lưu đồ giải mã 4 led 7

Hình 3.11 Lưu đồ lấy thời gian từ IC DS1307

Hình 3.12 Lưu đồ ghi thời gian vào IC DS1307

Hình 3.13 Lưu đồ nút bấm

Hình 3.14 Lưu đồ báo thức sử dụng buzzer

Hình 3.15 Lưu đồ điều chỉnh số BCD

Hình 3.16 Lưu đồ chương trình chính

Hình 4.1 Sơ đồ nguyên lý

Hình 4.2 Sơ đồ mạch in hoàn chỉnh

Hình 4.3 Mạch thực tế khi chưa hoạt động

Hình 4.4 Mạch thực tế khi hoạt động

**MỤC LỤC**

[**NHIỆM VỤ ĐỒ ÁN MÔN HỌC** i](#_Toc454128059)

[**PHIẾU NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN**](#_Toc454128062) ii

[**PHIẾU NHẬN XÉT CỦA GIÁO VIÊN PHẢN BIỆN** iii](#_Toc454128065)

[**LỜI CẢM ƠN** iv](#_Toc454128066)

[**DANH MỤC CÁC CHỮ VIẾT TẮT** v](#_Toc454128067)

[**DANH MỤC CÁC BẢNG BIỂU** vi](#_Toc454128068)

[**DANH MỤC CÁC HÌNH ẢNH** vii](#_Toc454128069)

[**CHƯƠNG 1: TỔNG QUAN ĐỀ TÀI** 1](#_Toc454128070)

[**1.1.Đặt vấn đề** 1](#_Toc454128071)

[**1.2. Mục tiêu đề tài** 1](#_Toc454128072)

[**1.3. Đối tượng nghiên cứu** 1](#_Toc454128073)

[**1.4. Phạm vi đề tài** 1](#_Toc454128074)

[**CHƯƠNG 2: CƠ SỞ LÝ THUYẾT** 2](#_Toc454128075)

[**2.1. Giới thiệu vi điều khiển pic16f887** 2](#_Toc454128076)

[**2.1.1 Giới thiệu chung về PIC** 2](#_Toc454128077)

[**2.1.2.Các dòng Pic và cách lựa chọn vi điều khiển Pic** 2](#_Toc454128078)

[**2.1.3. Cấu trúc tổng quát vi điều khiển PIC16F887** 4](#_Toc454128079)

[**2.1.3.1. Cấu hình của Pic16F887** 4](#_Toc454128080)

[**2.1.3.3 Sơ đồ cấu trúc của Pic16F887** 7](#_Toc454128081)

[**2.1.3.4. Tổ chức bộ nhớ của PIC16F887** 8](#_Toc454128082)

[**2.1.3.5. Bộ định thời Timer** 12](#_Toc454128083)

[**2.1.3.6. Bộ chuyển đổi ADC** 13](#_Toc454128084)

[**2.1.3.7. Ngắt** 14](#_Toc454128085)

[**2.1.4 Ngôn ngữ lập trình cho Pic** 14](#_Toc454128086)

[**2.1.5 Mạch nạp Pic** 14](#_Toc454128087)

[**2.2. Các linh kiện dùng trong mạch** 14](#_Toc454128088)

[**2.2.1. IC DS1307** 14](#_Toc454128089)

[**2.2.2 C1815/BC548** 17](#_Toc454128090)

[**2.2.3 Buzzer** 17](#_Toc454128091)

[**2.2.4. Led 7 đoạn** 18](#_Toc454128092)

[**CHƯƠNG 3: THIẾT KẾ VÀ XÂY DỰNG HỆ THỐNG** 19](#_Toc454128093)

[**3.1 Yêu cầu của hệ thống** 19](#_Toc454128094)

[**3.2 Thiết kế phần cứng** 19](#_Toc454128095)

[**3.2.1 Sơ đồ khối hệ thống và chức năng các khối trong hệ thống** 19](#_Toc454128096)

[**3.2.2 Thiết kế các khối trong hệ thống** 21](#_Toc454128097)

[**3.2.2.1 Khối nguồn** 21](#_Toc454128098)

[**3.2.2.2 Khối hiển thị và buzzer** 21](#_Toc454128099)

[**3.2.2.3 Khối thời gian thực DS1307** 23](#_Toc454128100)

[**3.2.2.4 Khối nút nhấn** 24](#_Toc454128101)

[**3.2.2.5 Khối xử lý trung tâm** 25](#_Toc454128102)

[**3.3 Phần mềm cho vi điều khiển** 26](#_Toc454128103)

[**3.3.1 Tính toán các thông số** 26](#_Toc454128104)

[**3.3.1.1 Tính chu kỳ quét led 7 đoạn** 26](#_Toc454128105)

[**3.3.1.2 Tính thông số ngắt timer1** 26](#_Toc454128106)

[**3.3.2 Các giải thuật** 26](#_Toc454128107)

[**3.3.2.1 Giải mã 4 led 7 đoạn** 26](#_Toc454128108)

[**3.3.2.2 Quét 4 led 7 đoạn** 28](#_Toc454128109)

[**3.3.2.3 Lấy thời gian thực/ghi thời gian IC DS1307** 31](#_Toc454128110)

[**3.3.2.4 Điều chỉnh thời gian thực sử dụng nút bấm** 33](#_Toc454128111)

[**3.3.2.5 Báo thức sử dụng buzzer** 34](#_Toc454128112)

[**3.3.2.6 Điều chỉnh số BCD tăng/giảm** 35](#_Toc454128113)

[**3.3.2.7 Chương trình chính** 36](#_Toc454128114)

[**CHƯƠNG 4: KẾT QUẢ** 37](#_Toc454128115)

[**4.1 Sơ đồ nguyên lý** 37](#_Toc454128116)

[**4.2 Sơ đồ mạch in** 38](#_Toc454128117)

[**4.3 Mô hình sau khi thi công** 38](#_Toc454128118)

[**4.3 Vận hành** 39](#_Toc454128120)

[**CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN** 41](#_Toc454128121)

[**5.1. Kết luận** 41](#_Toc454128122)

[**5.2. Hướng phát triển** 41](#_Toc454128123)

[**TÀI LIỆU THAM KHẢO** 42](#_Toc454128124)

[**PHỤ LỤC** 43](#_Toc454128125)

[**1. Các lệnh lập trình PIC16F887** 43](#_Toc454128126)

[**2. Code điều khiển** 45](#_Toc454128127)

# **CHƯƠNG 1: TỔNG QUAN ĐỀ TÀI**

## **1.1.Đặt vấn đề**

Nhóm mong muốn tạo ra một đồng hồ số nhỏ gọn, có thể hiển thị ngày giờ, kể cả ngày tháng năm. Là một phiên bản thu nhỏ của các loại đồng hồ số vạn niên treo tường cồng kềnh. Đồng hồ này thuận tiện cho việc xem giờ giấc ngay ở trên bàn làm việc, vẫn hiển thị đủ giờ phút, ngày tháng, có cả chức năng báo thức vô cùng tiện lợi. Điều quan trọng là ngay cả khi mất điện đồng hồ vẫn có thể tự đếm được.

## **1.2. Mục tiêu đề tài**

- Tìm hiểu về IC thời gian thực DS1307

- Thiết kế và thi công hệ thống đồng hồ báo thức

## **1.3. Đối tượng nghiên cứu**

- Vi điều khiển Pic16F887

- IC thời gian thực DS1307

- Buzzer

- Led 7 đoạn

## **1.4. Phạm vi đề tài**

**-** Hiển thị thời gian thực trên 2 led 7 đoạn.

-Có hẹn giờ báo thức trong vòng 1 phút.

# **CHƯƠNG 2: CƠ SỞ LÝ THUYẾT**

## **2.1. Giới thiệu vi điều khiển pic16f887**

### **2.1.1 Giới thiệu chung về PIC**

PIC bắt nguồn là chữ viết tắt của "Programmable Intelligent Computer" (Máy tính khả trình thông minh) là một sản phẩm của hãng General Instruments đặt cho dòng sản phẩm đầu tiên của họ là PIC1650. Lúc này, PIC1650 được dùng để giao tiếp với các thiết bị ngoại vi cho máy chủ 16bit CP1600, vì vậy, người ta cũng gọi PIC với cái tên "Peripheral Interface Controller" (Bộ điều khiển giao tiếp ngoại vi). CP1600 là một CPU tốt, nhưng lại kém về các hoạt động xuất nhập, và vì vậy PIC 8-bit được phát triển vào khoảng năm 1975 để hỗ trợ hoạt động xuất nhập cho CP1600. PIC sử dụng microcode đơn giản đặt trong ROM, và mặc dù, cụm từ RISC chưa được sử dụng thời bây giờ, nhưng PIC thực sự là một vi điều khiển với kiến trúc RISC, chạy một lệnh một chu kỳ máy (4 chu kỳ của bộ dao động).

Năm 1985, General Instruments bán bộ phận vi điện tử của họ, và chủ sở hữu mới hủy bỏ hầu hết các dự án - lúc đó đã quá lỗi thời. Tuy nhiên PIC được bổ sung EEPROM để tạo thành 1 bộ điều khiển vào ra khả trình. Ngày nay rất nhiều dòng PIC được xuất xưởng với hàng loạt các module ngoại vi tích hợp sẵn (như USART, PWM, ADC...), với bộ nhớ chương trình từ 512 Word đến 32K Word.

Họ vi điều khiển này có thể tìm mua dễ dàng tại thị trường Việt Nam. Giá thành không quá đắt. Có đầy đủ các tính năng của một vi điều khiển khi hoạt động độc lập. Là một sự bổ sung rất tốt về kiến thức cũng như về ứng dụng cho họ vi điều khiển mang tính truyền thống như họ vi điều khiển 8051. Hiện nay tại Việt Nam cũng như trên thế giới có một số lượng lớn người sử dụng họ vi điều khiển PIC, vì vậy họ vi điều khiển này được sử dụng khá rộng rãi. Điều này tạo nhiều thuận lợi trong quá trình tìm hiểu và phát triển các ứng dụng, cũng như dễ dàng trao đổi, học tập, dễ dàng tìm được sự chỉ dẫn khi gặp khó khăn,…đồng thời cũng được sự hỗ trợ của nhà sản xuất về trình biên dịch, các công cụ lập trình, nạp chương trình từ đơn giản đến phức tạp.

### **2.1.2.Các dòng Pic và cách lựa chọn vi điều khiển Pic**

* Các sản phẩm vi điều khiển PIC của Microchip có gần 100 loại sản phẩm từ họ 10Fxxx đến các họ 12Cxxx, 12Fxxx, 16Cxx, 17Cxx, 16Fxx, 16Fxxx, 16FxxxA, 16LFxxxA, 18Fxxx, 18LFxxx, 18Fxxxx, 18LFxxxx,…**Cách phân loại PIC theo chữ cái**
* Các họ PIC xxCxxx được đưa vào một nhóm, gọi là OTP (One Time Programmable): chúng ta chỉ có thể lập trình và nạp chương trình cho nó được một lần duy nhất.
* Nhóm thứ hai có chữ cái F hoặc LF: chúng ta gọi nhóm này là nhóm Flash, nhóm này cho phép ghi xóa nhiều lần bằng các mạch điện thông thường.
* **Cách phân loại theo hai con số đầu tiên của sản phẩm**

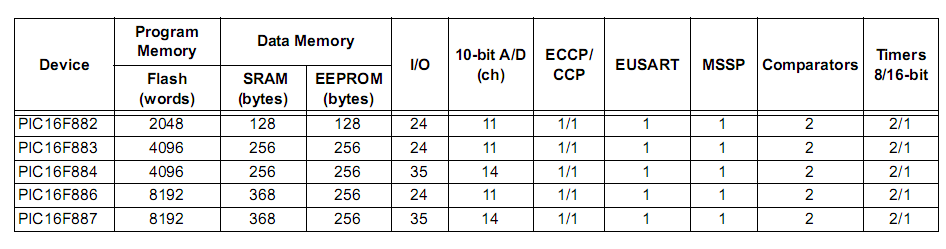
Loại thứ nhất là dòng PIC cơ bản( Base – line ), gồm các PIC 12Cxxx, có độ dài 12 bit

* Loại thứ hai là các dòng PIC 10F, 12F va 16F, gọi là dòng phổ thông ( Mid – Range ), có dộ dài 14 bit.
* Loại thứ ba là dòng PIC 18( High – End ), có độ dài 16 bit.
* **Cách lựa chọn một vi điều khiển PIC phù hợp:**
* Ở Việt Nam phổ biến nhất là các họ vi điều khiển PIC do hãng Microchip sản xuất. Cách lựa chọn một vi điều khiển PIC phù hợp: Trước hết cần chú ý đến số chân của vi điều khiển cần thiết cho ứng dụng. Có nhiều vi điều khiển PIC với số lượng chân khác nhau, thậm chí có vi điều khiển chỉ có 8 chân,ngoài ra còn có các vi điều khiển 18, 28, 40, 44, … chân. Cần chọn vi điều khiển PIC có bộ nhớ flash để có thể nạp xóa chương trình được nhiều lần hơn. Tiếp theo cần chú ý đến các khối chức năng được tích hợp sẵn trong vi điều khiển, các chuẩn giao tiếp bên trong. Sau cùng cần chú ý đến bộ nhớ chương trình mà vi điều khiển cho phép.
* Trước hết ta cần chú ý số chân của vi điều khiển PIC cần thiết cho ứng dụng. Có nhiều vi điều khiển PIC có số lượng chân khác nhau, thậm chí có VĐK chỉ có 8 chân, ngoài ra còn có các vi điều khiển PIC 28, 40, 44,… chân.
* Nên lựa chọn VĐK có bộ nhớ flash để có thể nạp xóa chương trình được nhiều lần hơn.
* Tiếp theo cần chú ý các khối chức năng được tích hợp sẵn trong VĐK, các chuẩn giao tiếp bên trong
* Sau cùng chú ý đến bộ nhớ chương trình mà VĐK cho phép.

*Ngoài ra, mọi thông tin về lựa chọn VĐK PIC có thể được tìm thấy trong cuốn sách “ Select PIC guide” do nhà sản xuất Microchip cung cấp.*

### **2.1.3. Cấu trúc tổng quát vi điều khiển PIC16F887**

#### **2.1.3.1. Cấu hình của Pic16F887**



*Bảng 2.1.* Cấu hình của các dòng Pic 16fxxx

Đây là vi điều khiển thuộc họ PIC16xxxx với tập lệnh gồm 35 lệnh có độ dài 14 bit. Mỗi lệnh đều được thực thi trong một chu kỳ xung clock. Tốc độ hoạt động tối đa cho phép là 20Mhz với một chu kỳ lệnh là 200ns. Bộ nhớ flash chương trình là 8192 words và bộ nhớ dữ liệu là 368 bytes SRAM + 256 bytes EEPROM. Số PORT I/O là 5 với 33 pin I/O.

* **Các đặc tính ngoại vi bao gồm các khối chức năng sau:**

-Timer0: bộ nhớ 8 bit với bộ chia tần số 8 bit.

-Timer1: bộ đếm 16 bit với bộ chia tần số có thể thực hiện chức năng đếm dựa vào xung clock ngoại vi ngay khi vi điều khiển hoạt động ở chế độ sleep.

-Timer2: bộ đếm 8 bit với bộ chia tần số, bộ postcaler.

-Hai bộ Capture/ so sánh/ điều chế độ rộng xung.

-Các chuẩn giao tiếp nối tiếp SSP (Synchronuos Serial Port), ISP và I2C.

-Chuẩn giao tiếp nối tiếp USART với 9 bit địa chỉ.

-Cổng giao tiếp song song PSP (Parallel Slave Port) với các chân điều khiển RD, WR, CS ở bên ngoài.

* **Các đặc tính Analog:**

-14 kênh chuyển đổi ADC 10 bit

-2 bộ so sánh

Bên cạnh đó là một vài đặc tính khác của vi điều khiển như:

-Bộ nhớ flash có khả năng ghi xóa được 100 000 lần.

-Bộ nhớ EEPROM có khả năng ghi xóa được 1 000 000 000 lần.

-Dữ liệu bộ nhớ EEPROM có thể lưu trữ trên 40 năm.

-Khả năng tự nạp chương trình với sự điều khiển của phần mềm.

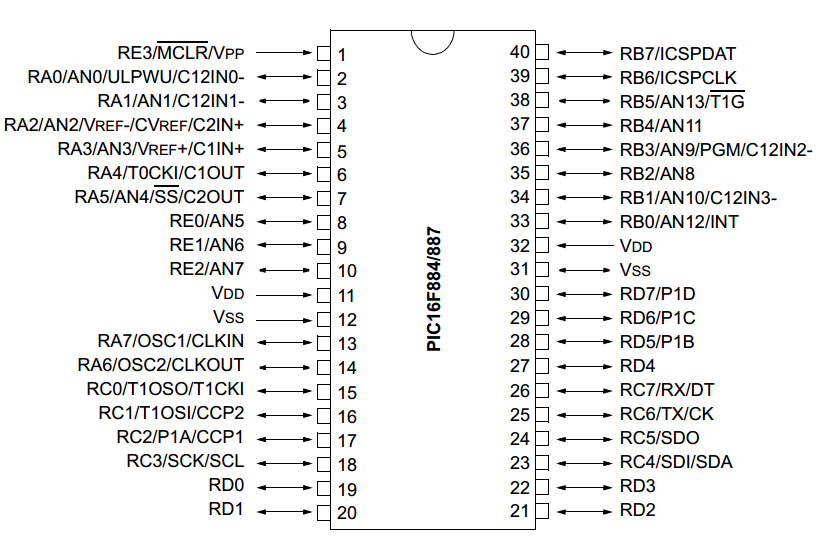
-Nạp được chương trình ngay trên mạch điện ICSP (In circuit Serial Programming) thông qua chân 2.

-Watchdog Timer với bộ dao động trong.

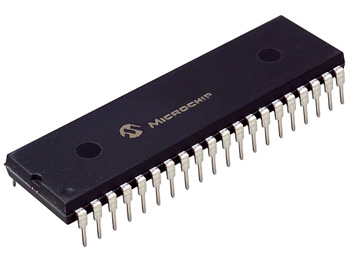
-Chức năng bảo mật mã chương trình.

-Chế độ sleep.

-Có thể hoạt động với nhiều dạng Oscillator khác nhau

* **2.1.3.2 Khảo sát sơ đồ chân của Pic16F887**Chức năng chân của vi điều khiển PIC 16F887

*Hình 2.1. Sơ đồ chân của vi điều khiển PIC16F887*

****

*Hình 2.4. Hình dạng thực tế của vi điều khiển PIC 16F887PIC16F887*

*Hình 2.2 : hình ảnh thực của PIC*

* Port A*:* PortA (RA0 🡪 RA5) có số chân từ chân số 2 đến chân số 7.

PortA (RPA) bao gồm 6 I/O pin. Đây là các chân hai chiều (bidirectional pin), nghĩa là có thể xuất và nhập được. Chức năng I/O này được điều khiển bởi thanh ghi TRISA (địa chỉ 85h).

* PortB: PortB (RB0 🡪 RB7) có số chân từ chân số 33 đến chân số 40.

PortB (RPB) gồm 8 pin I/O. Thanh ghi điều khiển xuất nhập tương ứng là TRISB. Bên cạnh đó một số chân của Port B còn được sử dụng trong quá trình nạp chương trình cho vi điều khiển với các chế độ nạp khác nhau. PortB còn liên quan đến ngắt ngoại vi và bộ Timer0. PortB còn được tích hợp chức năng điện trở kéo lên được điều khiển bởi chương trình.

* PortC: PortC(RC0 🡪 RC7) có số chân từ chân số 15 đến chân số 18 và chân số 23 đến chân số 26.

PortC (RPC) gồm 8 pin I/O. Thanh ghi điều khiển xuất nhập tương ứng là TRISC. Bên cạnh đó PortC còn chứa các chân chức năng của bộ so sánh, bộ Timer1, bộ PWM và các chuẩn giao tiếp nối tiếp I2C, SPI, SSP, USART.

* PortD: Port D (RD0 🡪 RD7) có số chân từ chân số 33 đến chân số 40.

Port D (RPD) gồm 8 chân I/O, thanh ghi điều khiển xuất nhập tương ứng là TRISD. Port D còn là cổng xuất dữ liệu của chuẩn giao tiếp PSP(Parallel Slave Port).

* PortE: PortE (RE0 🡪 RE2) có số chân từ chân số 19 đến chân số 22 và chân số 27 đến chân số 30.

PortE (RPE) gồm 3 chân I/O. Thanh ghi điều khiển xuất nhập tương ứng là TRISE. Các chân của PortE có ngõ vào analog. Bên cạnh đó PortE còn là các chân điều khiển của chuẩn giao tiếp PSP. Chân 11,12,31,32 là các chân cung cấp nguồn cho vi điều khiển. Chân 13, 14 là chân được đấu nối thạch anh với bộ dao động xung clock bên ngoài cung cấp xung clock cho chip hoạt động. Chân 1 là chân RET: là tín hiệu cho phép thiết lập lại trạng thái ban đầu cho hệ thống, và là tín hiệu nhập là mức tích cực mức cao.

#### **df2.1.3.3 Sơ đồ cấu trúc của Pic16F887**

*Hình 2.3. Cấu trúc bên trong của vi điều khiển PIC 16F887PIC16F887*

#### **2.1.3.4. Tổ chức bộ nhớ của PIC16F887**

Cấu trúc bộ nhớ của vi điều khiển PIC 16F887 bao gồm bộ nhớ chương trình (Program memory) và bộ nhớ dữ liệu (data memory).

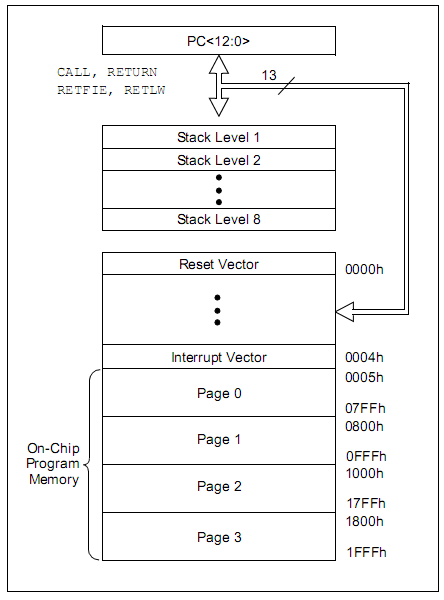
Bộ nhớ chương trình (Programmemory)

Bộ nhớ chương trình của vi điều khiển PIC 16F887 là bộ nhớ flash, dung lượng là 8 Kword (1 word = 14 bit) và được phân thành nhiều trang (từ page0 đến page3).

Như vậy bộ nhớ chương trình có khả năng chứa được 8x1024 = 8192 lệnh (vì mỗi lệnh sau khi mã hóa sẽ có dung lượng 14 bit = 1 word).

Để mã hóa được địa chỉ của 8 Kword chương trình, bộ đếm chương trình có dung lượng 13 bit. Khi vi điều khiển được reset bộ đếm chương trình sẽ chỉ đến địa chỉ 0000H (Reset vector). Khi có ngắt xảy ra, bộ đếm chương trình sẽ chỉ đến địa chỉ 0004H (Interrupt vector).

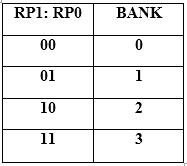
Bộ nhớ chương trình không bao gồm bộ nhớ stack và không được địa chỉ hóa bởi bộ đếm chương trình. Bộ nhớ stack sẽ được đề cập cụ thể trong phần sau:



*Hình 2.4. Bộ nhớ chương trình Pic16F887 16F887PIC16F887*

* **Bộ nhớ dữ liệu (Data memory)**

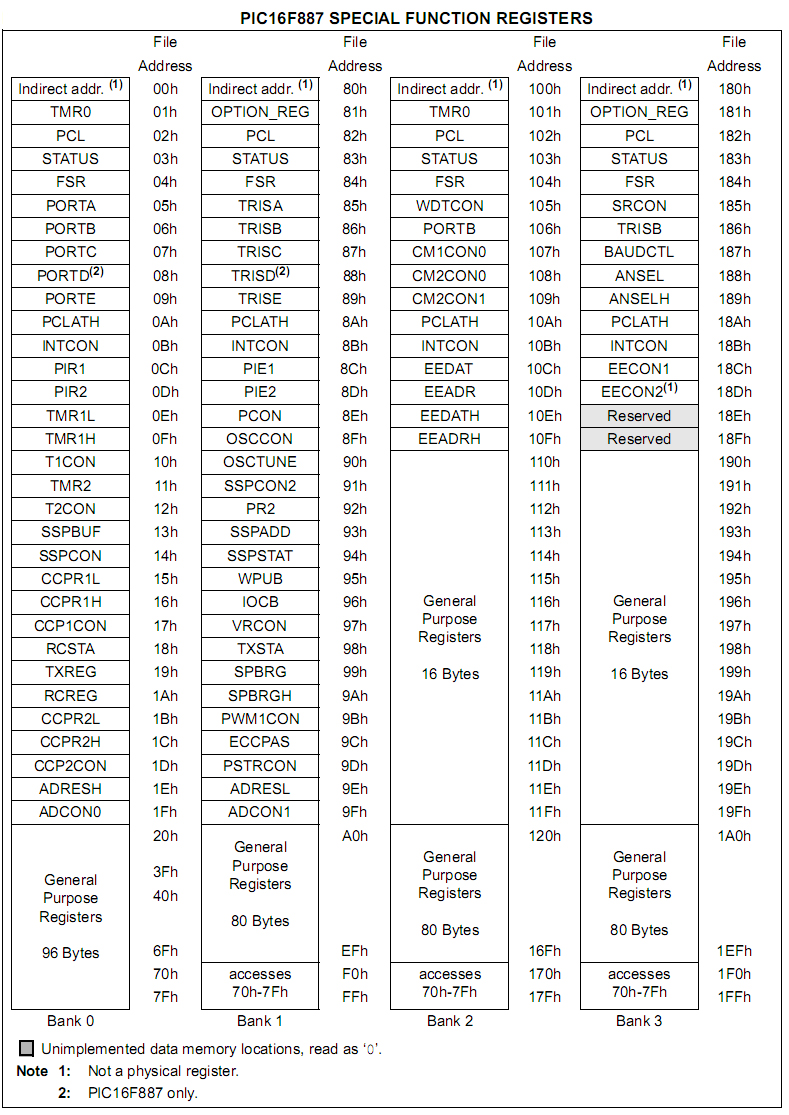
Bộ nhớ dữ liệu của PIC và bộ nhớ EEPROM được chia ra làm nhiều bank. Đối với vi điều khiển PIC 16F887 thì bộ nhớ dữ liệu được chia là 4 bank. Bank được chọn phụ thuộc vào bit RP1 và RP0 (bit thứ 6 và bit thứ 5) của thanh ghi trạng thái status.



*Bảng 2.2. Bộ nhớ dữ liệu*

Mỗi bank có dung lượng 128 byte, bao gồm các thanh ghi có chức năng đặc biệt SFR (Special Function Register) nằm ở 32 vị trí đầu tiên của mỗi bank và các thanh ghi mục đích chung GPR (General Purpose Register) nằm ở 96 vị trí cuối cùng của mỗi bank, đóng vai trò như Static RAM.

Các thanh ghi SFR thường xuyên được sử dụng (ví dụ như thanh ghi STATUS) sẽ được đặt ở tất cả các bank của bộ nhớ dữ liệu, giúp thuận tiện trong quá trình truy xuất và làm giảm bớt lệnh chương trình. Sơ đồ cụ thể của bộ nhớ dữ liệu vi điều khiển PIC 16F887 như sau:



*Hình 2.5. Các thanh ghi chức năng Pic16F887 16F887PIC16F887*

* **Các thanh ghi đặc biệt**

Các thanh ghi được sử dụng bởi CPU hoặc dùng để thiết lập điều khiển các khối chức năng tích hợp trong vi điều khiển. Phân chia thanh SFR làm 2 loại: thanh ghi SFR liên quan đến các chức năng trong CPU và thanh ghi SFR dùng để thiết lập và điều khiển các khối chức năng bên ngoài.

Các thanh ghi liên quan đến bên trong

Thanh ghi STATUS (03h,83h,103h,183h): thanh ghi chứa kết quả thực hiện phép toán của khối ALU, trạng thái Reset và các bit chọn bank cùng truy suất trong bộ nhớ dữ liệu.

Thanh ghi OPTION\_REG (81h, 181h): thanh ghi này cho phép đọc và ghi, cho phép điều khiển các chức năng puled ma trận up của các chân PortB, xác lập các tham số xung tác động, cạnh tác động của ngắt ngoại vi và bộ đếm Timer0.

Thanh ghi INTCON (0Bh, 8Bh, 10Bh, 18Bh): thanh ghi cho phép đọc và ghi, chứa các bit điều khiển và các bit cờ hiệu khi Timer0 tràn, ngắt ngoại vi RB0/INT và ngắt interrup-on-change tại các chân của PortB.

Thanh ghi PIE1(8Ch): chứa các bit điều khiển chi tiết các ngắt của khối chức năng ngoại vi.

Thanh ghi PIR1(0Ch): chứa cờ ngắt của các khối chức năng ngoại vi , các ngắt này được cho phép bởi các bit điều khiển chứa trong thanh ghi PIE1.

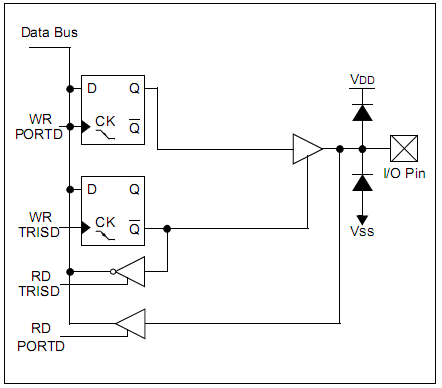
Thanh ghi PIE2 (8Dh): chứa các bit điều khiển các ngắt của khối chức năng CCP2, SSP bus, ngắt của bộ so sánh và ngắt ghi vào bộ nhớ EEPROM.

Thanh ghi PIR2(0Dh): chứa các cờ ngắn của các khối chức năng ngoại vi , các ngắt này được cho phép bởi các bit điều khiển chứa trong thanh ghi PIE2.

Thanh ghi PCON(8Eh): chứa các cờ hiệu cho biết trạng thái các chế độ Reset của vi điều khiển.

Thanh ghi mục đích chung GPR: các thanh ghi này có thể truy suất trực tiếp hoặc gián tiếp thông qua thanh ghi FSG (File Select Rersister). Đây là các thanh ghi dữ liệu thông thường, người sử dụng có thể tùy theo mục đích chương trình mà có thể dùng các thanh ghi này để chứa các biến số, hằng số, kết quả hoặc tham số phục vụ cho chương trình.

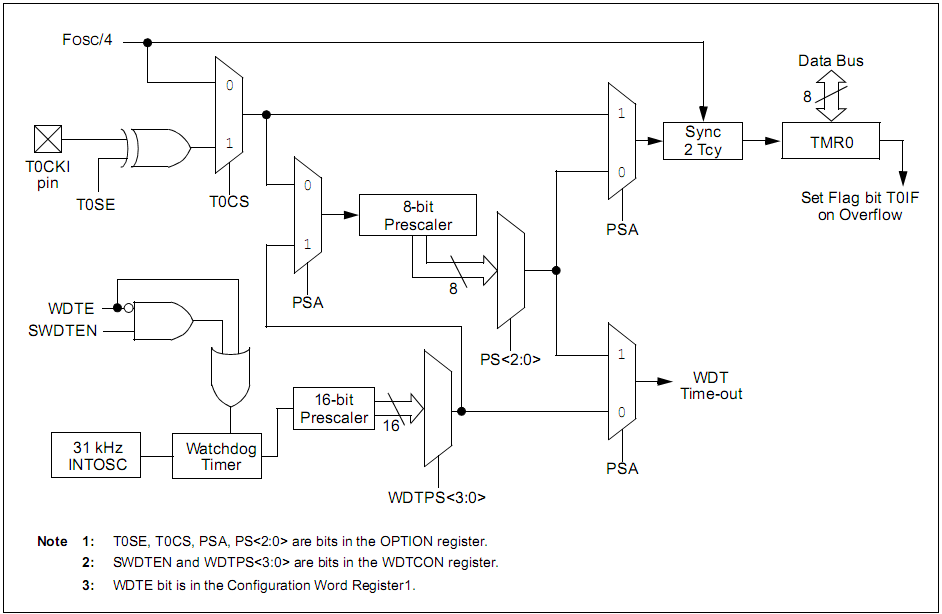
* **Cấu trúc của GPIO:**



*Hình 2.6. Cấu trúc của GPIO*

#### **2.1.3.5. Bộ định thời Timer**

* **Timer 0**

Là một trong 3 bộ đếm hoặc bộ định thời của vi điều khiển PIC 16F887. Timer0 là bộ đếm 8 bit được kết nối với bộ chia tần số (prescaler) 8 bit. Cấu trúc của Timer0 cho phép ta lựa chọn xung clock tác động và cạnh tích cực của xung clock. Ngắt Timer0 sẽ xuất hiện khi Timer0 bị tràn. Bit TMR0IE (INTCON < 5) là bit điều khiển của Timer0. TMR0IE = 1 cho phép ngắt Timer0 tác động, TMR0IE = 0 không cho phép ngắt Timer0 tác động. Sơ đồ khối của Timer0 như sau:

*Hình 2.7. Bộ định thời Timer0*

Bộ chia tần số (prescaler) được chia sẽ giữa Timer0 và WDT (Watchdog Timer), điều đó có nghĩa là nếu prescaler được sử dụng cho Timer0 thì WDT sẽ không có được sự hỗ trợ của prescaler và ngược lại. Prescaler được điều chỉnh bởi thanh ghi OPTION\_REG. Bit PSA (OPTION\_REG < 3 >) xác định đối tượng tác động của prescaler. Các bit PS2:PS0 (OPTION\_REG<2:0>) xác định tỷ số chia tần số của prescaler. Xem lại thanh ghi OPTION\_REG để xác định lại một cách chi tiết về các bit điều khiển trên.

Các lệnh tác động lên giá trị thanh ghi TMR0 sẽ xóa chế độ hoạt động của prescaler. Khi đối tượng tác động là Timer0, tác động lên giá trị thanh ghi TMR0 sẽ xóa prescaler nhưng không làm thay đổi đối tượng tác động của prescaler. Khi đối tượng tác động là WDT, lệnh CLR WDT sẽ xóa prescaler, đồng thời prescaler sẽ ngưng tác vụ hỗ trợ cho WDT.

* **Bộ Timer1**

Bộ Timer1 là bộ định thời 16 bit, giá trị của Timer1 sẽ được lưu trong thanh ghi (TMR1H:TMR1L). Cờ ngắt của Timer1 là bit TMR1IF. Bit điều khiển của Timer1 là TMR1IE. Tương tự như Timer0, Timer1 cũng có 2 chế độ hoạt động: chế độ định thời và chế độ xung kích là xung clock của osciled ma trận ator. Tần số Timer bằng tần số của osciled ma trận ator và chế độ đếm(counter) với xung kích là xung phản ánh các sự kiện cần đếm lấy từ bên ngoài thông qua chân RCO/T1CKI/T1OSO. Việc lựa chọn chế độ hoạt động của Timer được điều khiển bởi bit TMR1CS.

* **Bộ Timer2**

Bộ Timer2 là bộ định thời 8 bit và được hỗ trợ hai bộ chia tần số Prescaler và postscaler. Thanh ghi chứa giá trị đếm của Timer2 là TMR2. Bit cho phép ngắt Timer2 tác động là TMR2ON. Cờ ngắt của Timer2 là bit TMR2IF. Xung ngõ vào được đưa qua bộ chia tần số prescaler 4 bit (với các tỉ số chia tần 1:1, 1:4 hoặc 1:6) và được điều khiển bởi các bit T2CKPS1:T2CKPS0.

#### **2.1.3.6. Bộ chuyển đổi ADC**

ADC (Analog to Digital Converter) là bộ chuyển đổi tín hiệu giữa hai dạng tương tự và số . PIC16F887 có 14 ngõ vào analog (RA5:RA0, RE2:RE0 và RB5:RB0). Hiệu điện thế chuẩn có thể được chọn hay hiệu điện thế chuẩn được xác lập trên 2 chân RA2 và RA3. Kết quả chuyển đổi từ tín hiệu tương tự sang tín hiệu số là 10 bit tương ứng và được lưu trong 2 thanh ghi ADRESH:ADRESL. Khi không sử dụng bộ chuyển đổi ADC thì các thanh ghi này có thể sử dụng các thanh ghi thông thường khác. Khi quá trình chuyển đổi hoàn tất, kết quả sẽ được lưu vào 2 thanh ghi ADRESH:ADRESL.

#### **2.1.3.7. Ngắt**

Khi giá trị đếm trong thanh ghi TMR0 tràn từ FFh về 00h thì phát sinh ngắt, cờ báo ngắt TMR0IF lên 1. Ngắt có thể ngăn bằng bit cho phép ngắt TMR0IE.

Trong chương trình con phục vụ ngắt Timer0 phải xóa cờ báo ngắt TMR0IF. Ngắt của TMR0 không thể kích CPU thoát khỏi chế độ ngủ vì bộ định thời sẽ ngừng khi CPU ở chế độ ngủ.

Thanh ghi INTCON nằm trong vùng nhớ RAM có địa chỉ là 0xB0.

### **2.1.4 Ngôn ngữ lập trình cho Pic**

Ngôn ngữ lập trình cho PIC rất đa dạng. Ngôn ngữ lập trình cấp thấp có MPLAB (được cung cấp miễn phí bởi nhà sản xuất Microchip), các ngôn ngữ lập trình cấp cao hơn bao gồm C, Basic, Pascal, … Ngoài ra còn có một số ngôn ngữ lập trình được phát triển dành riêng cho PIC như PICBasic, MikroBasic. Trong đồ án này, nhóm sử dụng Pic C Compiler CCS để viết chương trình và biên dịch.

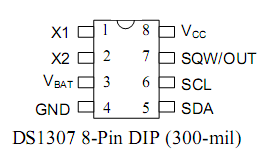
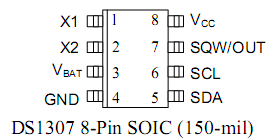
### **2.1.5 Mạch nạp Pic**

Đây cũng là một dòng sản phẩm rất đa dạng dành cho vi điều khiển PIC. Có thể sử dụng các mạch nạp được cung cấp bởi nhà sản xuất là hãng Microchip như: PICSTART plus, MPLAB ICD 2, MPLAB PM 3, PRO MATE II. Có thể dùng các sản phẩm này để nạp cho vi điều khiển khác thông qua chương trình MPLAB. Dòng sản phẩm chính thống này có ưu thế là nạp được cho tất cả các vi điều khiển PIC, tuy nhiên giá thành rất cao và thường gặp rất nhiều khó khăn trong quá trình mua sản phẩm.

## **2.2. Các linh kiện dùng trong mạch**

### **2.2.1. IC DS1307**

DS1307 là một sản phẩm của Dallas Semiconductor (một công ty thuộc Maxim Integrated Products). Chip này có 7 thanh ghi 8-bit chứa thời gian là: giây, phút, giờ, thứ (trong tuần), ngày, tháng, năm. Ngoài ra DS1307 còn có 1 thanh ghi điều khiển ngõ ra phụ và 56 thanh ghi trống có thể dùng như RAM. DS1307 được đọc và ghi thông qua giao diện nối tiếp I2C.

*Hình 2.8. Hai gói cấu tạo chip DS1307*

Các chân của DS1307 được mô tả như sau:

- X1 và X2: là 2 ngõ kết nối với 1 thạch anh 32.768KHz làm nguồn tạo dao động cho chip.

- VBAT: cực dương của một nguồn pin 3V nuôi chip.

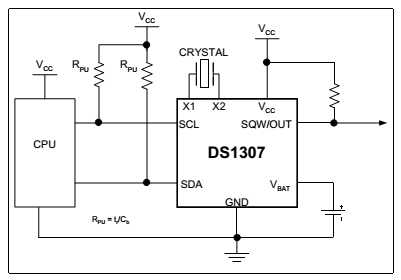
- GND: chân mass chung cho cả pin 3V và Vcc.

- Vcc: nguồn cho giao diện I2C, thường là 5V và dùng chung với vi điều khiển. Chú ý là nếu Vcc không được cấp nguồn nhưng VBAT được cấp thì DS1307 vẫn đang hoạt động (nhưng không ghi và đọc được).

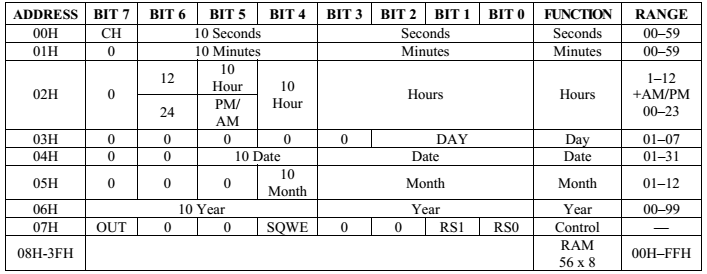
- SQW/OUT: một ngõ phụ tạo xung vuông (Square Wave / Output Driver), tần số của xung được tạo có thể được lập trình. Như vậy chân này hầu như không liên quan đến chức năng của DS1307 là đồng hồ thời gian thực, chúng ta sẽ bỏ trống chân này khi nối mạch.

- SCL và SDA là 2 đường giao xung nhịp và dữ liệu của giao diện I2C

Thời gian lưu trong các thanh ghi theo dạng BCD. BCD là viết tắt của cụm từ Binary-Coded Decimal, tạm dịch là các số thập phân theo mã nhị phân.



*Hình 2.9 . Kết nối phần cứng DS1307*



*Bảng 2.3. Tổ chức các thanh ghi thời gian*

**Thanh ghi giây (SECONDS):** thanh ghi này là thanh ghi đầu tiên trong bộ nhớ của DS1307, địa chỉ của nó là 0x00. Bốn bit thấp của thanh ghi này

chứa mã BCD 4-bit của chữ số hàng đơn vị của giá trị giây. Do giá trị cao nhất của chữ số hàng chục là 5 (không có giây 60 ) nên chỉ cần 3 bit (các bit SECONDS6:4) là có thể mã hóa được (số 5 =101, 3 bit). Bit cao nhất, bit 7, trong thanh ghi này là 1 điều khiển có tên CH (Clock halt – treo đồng hồ),

nếu bit này được set bằng 1 bộ dao động trong chip bị vô hiệu hóa, đồng hồ không hoạt động. Vì vậy, nhất thiết phải reset bit này xuống 0 ngay từ đầu.

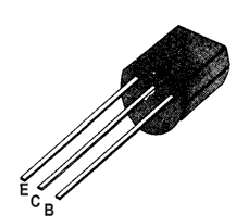
Thanh ghi phút (MINUTES): có địa chỉ 0x01, chứa giá trị phút của đồng hồ. Tương tự thanh ghi SECONDS, chỉ có 7 bit của thanh ghi này được dùng lưu mã BCD của phút, bit 7 luôn luôn bằng 0.

**Thanh ghi giờ (HOURS):**Thanh ghi này có địa chỉ 0x02. Trước hết 4-bits thấp của thanh ghi này được dùng cho chữ số hàng đơn vị của giờ. Do DS1307 hỗ trợ 2 loại hệ thống hiển thị giờ là 12h và 24h, bit6 xác lập hệ thống giờ. Nếu bit6=0 thì hệ thống 24h được chọn, khi đó 2 bit cao 5 và 4 dùng mã hóa chữ số hàng chục của giá trị giờ. Do giá trị lớn nhất của chữ số hàng chục trong trường hợp này là 2 nên 2 bit 5 và 4 là đủ để mã hóa. Nếu bit6=1 thì hệ thống 12h được chọn, với trường hợp này chỉ có bit 4 dùng mã hóa chữ số hàng chục của giờ, bit 5 chỉ buổi trong ngày, AM hoặc PM. Bit5 =0 là AM và bit5=1 là PM. Bit 7 luôn bằng 0.

Thanh ghi thứ (DAY – ngày trong tuần): nằm ở địa chĩ 0x03. Thanh ghi DAY chỉ mang giá trị từ 1 đến 7 tương ứng từ Chủ nhật đến thứ 7 trong 1 tuần. Vì thế, chỉ có 3 bit thấp trong thanh ghi này có nghĩa.Các thanh ghi còn lại có cấu trúc tương tự, DATE chứa ngày trong tháng (1 đến 31), MONTH chứa tháng (1 đến 12) và YEAR chứa năm (00 đến 99). Chú ý, DS1307 chỉ dùng cho 100 năm, nên giá trị năm chỉ có 2 chữ số, phần đầu của năm do người dùng tự thêm vào.

Ngoài các thanh ghi trong bộ nhớ, DS1307 còn có một thanh ghi khác nằm riêng tại địa chỉ 0x08 đến 0x3F, là vùng ram cho người dùng tự lưu trữ thông tin cá nhân.

### **2.2.2 C1815/BC548**

* Transistor C1815,bc548 là thuộc loại transistor NPN.
* C1815 có Uc cực đại = 50V dòng Ic cực đại = 150mA. BC548 có Uc= 30V ,Ic= 100mA
* Hệ số khuếch đại hFE của C1815 trong khoảng 25 đến 100, của BC548 trong khoảng 450 đến 900.
* Thứ tự các chân từ trái qua phải của C1815: E C B
* Thứ tự các chân từ trái qua phải của BC548: E B C

*Hình 2.11. Chân BC548*

*Hình 2.10 . Chân C1815*

### **2.2.3 Buzzer**

* Nguồn:3.5V-5.5V
* Dòng điện tiêu thụ: <25mA
* Biên độ âm thanh: >80dB
* Nhiệt độ hoạt động: -200C đến 700C

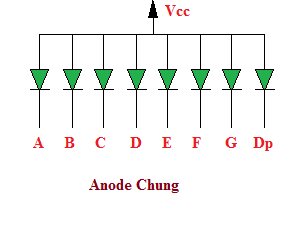
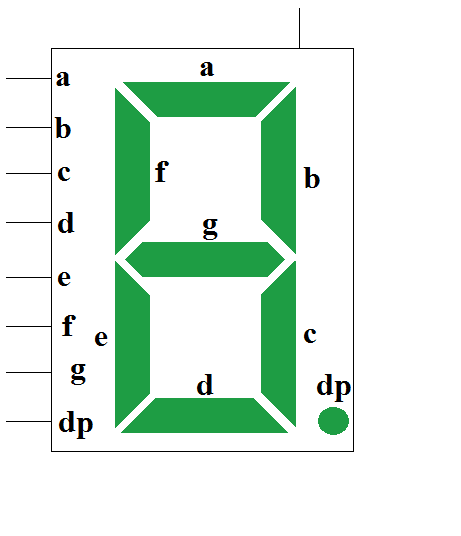
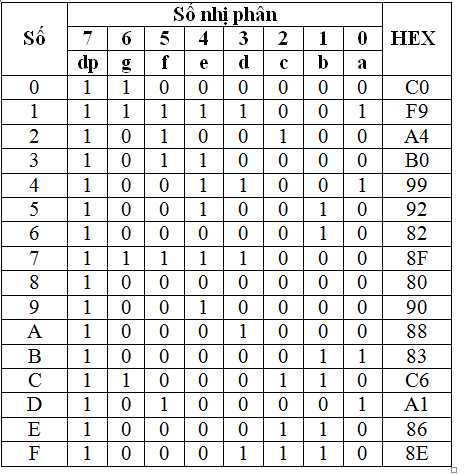


*Hình 2.12 .Buzzer*

### **2.2.4. Led 7 đoạn**

Led 7 đoạn có cấu tạo bao gồm 7 led đơn xếp theo hình phía trên và có thêm một led đơn hình tròn nhỏ thể hiện dấu chấm tròn ở góc dưới, bên phải của led 7 thanh

Nếu led 7 đoạn có Anode (cực +) chung, đầu chung này được nối với +Vcc, các chân còn lại dùng để điều khiển trạng thái sáng tắt của các led đơn, led chỉ sáng khi tín hiệu đặt vào các chân này ở mức 0.



*Hình 2.13. Cấu tạo led 7 đoạn anode chung*

*Bảng 2. 4. Mã led 7 đoạn anode chung*

# **CHƯƠNG 3: THIẾT KẾ VÀ XÂY DỰNG HỆ THỐNG**

## **3.1 Yêu cầu của hệ thống**

* Phần cứng phải hiển thị được thời gian thực qua 2 led 7 đoạn đôi
* Hẹn giờ như yêu cầu
* Báo thức thông qua buzzer
* Sử dụng nguồn Adapter

## **3.2 Thiết kế phần cứng**

### **3.2.1 Sơ đồ khối hệ thống và chức năng các khối trong hệ thống**

* Sơ dồ khối gồm các khối chính
* Khối IC DS1307 Real-Time
* Khối Vi điều khiển
* Khối hiển thị
* Khối buzzer
* Khối nút nhấn
* Khối nguồn

KHỐI NGUỒN

KHỐI VI ĐIỀU KHIỂN PIC 16F887

KHỐI IC DS1307 REAL-TIME

KHỐI HIỂN THỊ

KHỐI NÚT NHẤN

KHỐI BUZZER

I2C

*Hình 3.1. Sơ đồ khối hệ thống*

* **Chức năng từng khối**
* **Khối IC DS1307 Real-Time:** Sau khi nạp giá trị ban đầu, ic tiếp tục tự chạy giờ phút giây, ngày tháng năm. Có sử dụng nguồn phụ 3.3V nhằm đảm bảo việc chạy liên tục khi bị mất nguồn.
* **Khối Vi điều khiển:** Sử dụng Vi điều khiển Pic 16F887 để điều khiển, xử lý mọi hoạt động của hệ thống như lấy thông tin từ IC DS1307, nạp thông tin vào lại IC DS1307, hiển thị ra led 7 đoạn. Điều khiển buzzer kêu khi có hẹn giờ.
* **Khối hiển thị:** Sử dụng phương pháp quét hiển thị ra bốn led 7 đoạn, nhằm hiển thị thông tin như giờ phút giây, ngày tháng năm.
* **Khối buzzer:** Đồng hồ có chức năng báo thức vì vậy buzzer có nhiệm vụ tạo ra tiếng kêu
* **Khối nút nhấn:** Các nút nhấn sử dụng để điều chỉnh giờ phút giây, ngày tháng năm. Thiết lập hẹn giờ và chuyển đổi thông tin giữa giờ phút giây và ngày tháng năm trên led 7 đoạn.
* **Khối nguồn:** Tạo ra điện áp 5V một chiều từ dòng xoay chiều 220V cho hệ thống hoạt động

### **3.2.2 Thiết kế các khối trong hệ thống**

#### **3.2.2.1 Khối nguồn**

Nhận điện áp ngõ vào 220VAC/50Hz, cung cấp điện áp ổn định 5V, 12V/3A cho toàn bộ hệ thống

* Tính toán linh kiện

Do áp lấy ra 5V nên cần nguồn 5V, 3A cung cấp dòng điện cho cả mạch

*Hình3.2 Adapter 5V,3A*

#### **3.2.2.2 Khối hiển thị và buzzer**

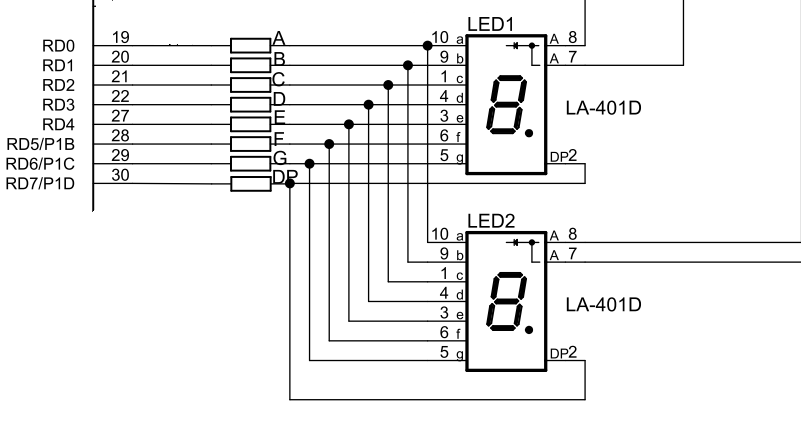
* **Khối hiển thị**

Có nhiều cách để hiển thị thông tin từ Vi điều khiển sau khi xử lý: dùng LCD, dùng Led 7 đoạn…

Vì nhu cầu sử dụng là xuất ra giờ, phút, ngày tháng, năm nên sử dụng 2 Led 7 đoạn đôi, các linh kiện hỗ trợ cho 2 đèn ít nên tiết kiệm diện tích. Giây được hiển thị bằng lượt đếm của 2 led đơn xen giữa 2 led 7 đoạn đôi , hiển thị được đơn vị giây của thông tin mà không cần thêm 1 led 7 đoạn nữa. Do đó ở đây tôi sử dụng led 7 đoạn nhằm hiển thị trực quan hơn thông tin cần cung cấp cho người dùng sau khi xử lý.

Kết nối led 7 đoạn cấu trúc dạng 8 bits với vi điều khiển, ở dạng này, ta sử dụng 7 chân từ D0 đến D6 để xuất dữ liệu 7 thanh led tương ứng, còn D7 sẽ được nối vào DP .

Trước khi nối vào port D, các led 7 đoạn phải đc mắc qua trở 330V.

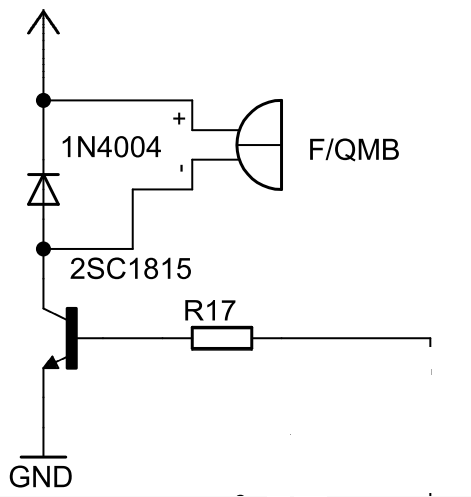
*hình 3.3: sơ đồ nối chân led 7 đoạn*

* **Buzzer**

Khối cảnh báo được thiết kế sử dụng buzzer.

Với điện áp vào khoảng 3,5V, áp dẫn trên transistor vào khoảng 0,7V, từ đó điện áp rơi trên trở vào khoảng 2,8V. Dòng trên buzzer khoảng 1mA, từ đó R\_coi = 2,8KΩ

Dùng trở 4.7KΩ để hạn dòng khi kết nối mạch với vi điều khiển, còi sẽ báo khi ngõ ra chân I/O của vi điều khiển có mức logic “1”. Diot nhằm chặn dòng ngược đảm bảo khối cảnh báo hoạt động tốt

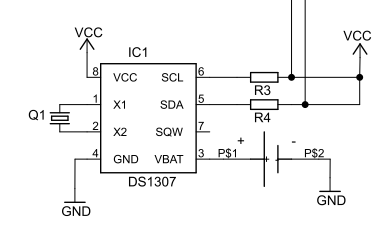


*Hình 3.4 : Khối cảnh báo buzzer*

#### **3.2.2.3 Khối thời gian thực DS1307**

Kết nối chân của DS1307:

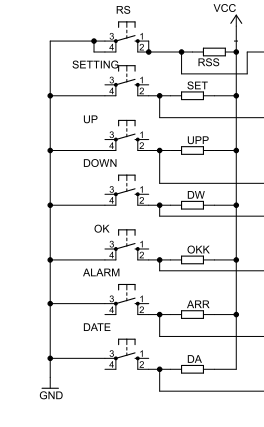
- **X1** và **X2**: là 2 ngõ kết nối với 1 thạch anh 32.768KHz làm nguồn tạo dao động cho chip.  
       - **VBAT**: cực dương của một nguồn pin 3V nuôi chip.  
       - **GND**: chân mass chung cho cả pin 3V và Vcc.  
       - **Vcc**: nguồn cho giao diện I2C, thường là 5V và dùng chung với vi điều khiển. Chú ý là nếu Vcc không được cấp nguồn nhưng VBAT được cấp thì DS1307 vẫn đang hoạt động (nhưng không ghi và đọc được).  
       - **SQW/OUT**: một ngõ phụ tạo xung vuông (Square Wave / Output Driver), tần số của xung được tạo có thể được lập trình. Như vậy chân này hầu như không liên quan đến chức năng của DS1307 là đồng hồ thời gian thực, chúng ta sẽ bỏ trống chân này khi nối mạch.  
       - **SCL và SDA**là 2 đường giao xung nhịp và dữ liệu của giao diện I2C mà chúng ta đã tìm hiểu trong bài TWI của AVR. Được nối vào port RC3 và RC4.

*hình 3.5 DS1307*

#### **3.2.2.4 Khối nút nhấn**

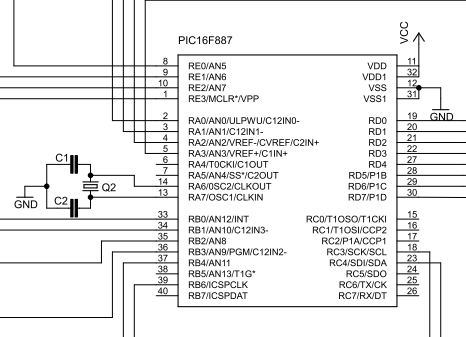
- Ở mạch này dùng 7 nút nhấn bao gồm: Reset, setting, up, down, alarm, date.

- Mỗi nút nhấn được nối với *trở 10k*

* Hình 3.6 các chân nút nhấn*

- Từ vi điều khiển, các chân từ RB0 đến RB6 lần lượt được nối qua trở đến các nút nhấn, chân còn lại của nút thì nối GND.

#### **3.2.2.5 Khối xử lý trung tâm**

* Kết nối của các chân trong Vi điều khiển:
* Chân 11, 32 nối nguồn cung cấp 5V là Adapter
* Chân 12, 31 nối GND
* Chân 13, 14 nối bộ dao động thạch anh 20MHZ
* Chân 1 kết nối reset
* Kết nối các Port trong vi điều khiển:
* Port A từ chân RA0 đến RA3 nối vào chân B của transistor BC548 điều khiển led 7 đoạn theo kết nối 8bits
* Port C chân RC3, RC4 nối với chân SCL và SDA của DS1307
* Port B chân RB0 đến RB6 lần lượt được nối với các nút nhấn setting, up, down, ok, alarm, date.
* Port D chân RD0 đến RD7 được nối với led 7 đoạn.
* Port E chân RE0 và RE1 được nối với 2 led đơn hiển thị số giây. Trong khi đó thì RE2 được nối với chân B của transistor 2SC1815

*Hình 3.7 Khối xử lý trung tâm*

## **3.3 Phần mềm cho vi điều khiển**

### **3.3.1 Tính toán các thông số**

#### **3.3.1.1 Tính chu kỳ quét led 7 đoạn**

Để mắt người có thể thấy được 4 led 7 đoạn sáng liên tục, cần có hiện tượng lưu ảnh ở mắt. Hiện tượng lưu ảnh này xảy ra khi khung hình trên mỗi led vượt hơn 25 khung hình/s. Vậy tổng chu kỳ quét là 100 khung hình/s. Tức tần số quét là 100Hz, ta lấy 1/100= 0.01s =10ms. Mỗi led cần delay một khoảng thời gian nhỏ hơn hoặc bằng 10ms/4=2.5ms.

#### **3.3.1.2 Tính thông số ngắt timer1**

Timer1 có 16 bit, gồm các số từ 0 đến 65535. Vậy để ngắt 100ms, ta tính như sau:

100.000 <65536

Tck . Prescale

Tck = 1 =0.2us

20Mhz / 4

=>Prescale >7.6. Ta chọn Prescale bằng 8

Số xung đếm: 100.000 = 62500 (xung)

0.2 . 8

X 🡪 65536 =62500 = 3036. Vậy ta chọn thời gian bắt đầu đếm là 3036.

### **3.3.2 Các giải thuật**

#### **3.3.2.1 Giải mã 4 led 7 đoạn**

Các biến phut\_ds1307,gio\_ds1307,ngay\_ds1307,thang\_ds1307,nam\_ds1307 lấy được từ IC DS1307 sẽ được chuyển thành mã hex để hiển thị phút, giờ lên led 7 đoạn. Nếu gia\_tri\_mod là một hoặc hai, thì hiển thị thêm một dấu chấm. Nếu gia\_Tri\_mod là ba hoặc sáu thì hiển thị thêm hai dấu chấm.

*Hình 3.8.. Lưu đồ giải mã ngày tháng năm*

Begin

NGAY\_THANG=1?

Giải mã biến ngày tháng, gán lại vào mảng Data

Nam=1?

Giải mã biến năm, gán lại vào mảng Data

END

Đ

Đ

S

S

#### **3.3.2.2 Quét 4 led 7 đoạn**

Ban đầu kích transistor NPN dẫn ở led 7 đoạn thứ nhất, bằng cách đưa chân lên mức 1, xuất dữ liệu cần hiển thị ra, delay khoảng 1 ms. Sau đó tắt dữ liệu đi, đưa chân về mức 0 để tắt transistor không dẫn. Tiếp tục cho 3 led 7 đoạn còn lại, lặp lại nhiều lần, dựa vào hiện tượng lưu ảnh của mắt ta sẽ thấy được 4 led 7 đoạn sáng đồng đều. Vào tốc độ 25 khung hình, mắt ta sẽ có hiện tượng lưu ảnh, ở đây sử dụng 4 led 7 đoạn, vậy lặp lại chu kỳ 100 lần.

*Hình 3.9. Lưu đồ quét led 7 đoạn*

Begin

I=0

I<100?

QUET=0x01

Q=0

Q<4?

OUTPUT\_A(QUET)

OUTPUT\_D(DATA[Q])

DELAY\_MS(1)

OUTPUT\_D(0XFF)

QUET=QUET<<1

Q=Q+1

END

I=I+1

S

S

Đ

Đ

*Hình 3.10 Lưu đồ giải mã 4 led 7 đoạn*

Begin

Giải mã dữ liệu thu được từ DS1307

Phân tích thành các thành phần chục, đơn vị

Gán dữ liệu vừa phân tích vào mảng Data

Gia\_tri\_mod=1?

Gia\_tri\_mod=2?

Gia\_tri\_mod=3?

Gia\_tri\_mod=6?

END

Làm sáng dấu chấm ở hàng đơn vị của phút

Làm sáng dấu chấm ở hàng đơn vị của giờ

Làm sáng dấu chấm ở hàng đơn vị và chục của phút

Làm sáng dấu chấm ở hàng đơn vị và chục của giờ

Đ

Đ

Đ

Đ

S

S

S

S

#### **3.3.2.3 Lấy thời gian thực/ghi thời gian IC DS1307**

*Hình 3.11. Lưu đồ lấy thời gian từ IC DSDS1307*

Begin

I2C\_START

I2C\_WRITE(0xD0)

I2C\_WRITE(0x01)

I2C\_START

I2C\_WRITE(0xD1)

Lấy thời gian lưu vào biến

I2C\_STOP()

END

*Hình 3.12. Lưu đồ ghi thời gian vào IC DS1307*

Begin

I2C\_START

I2C\_WRITE(0xD0)

I2C\_WRITE(0x00)

I2C\_STOP()

END

I2C\_WRITE(GIAY\_DS1307)

I2C\_WRITE(PHUT\_DS1307)

I2C\_WRITE(GIO\_DS1307)

I2C\_WRITE(TEM\_1)

I2C\_WRITE(NGAY\_DS1307)

I2C\_WRITE(THANG\_DS1307)

I2C\_WRITE(NAM\_DS1307)

#### **3.3.2.4 Điều chỉnh thời gian thực sử dụng nút bấm**

Sử dụng nút bấm Setting để vào chỉnh thời gian, nhấn Up hoặc Down thì tăng, giảm thời gian đồng thời ghi vào các thanh ghi dữ liệu thời gian trên DS1307.

Begin

Button Setting?

Button Up?

Button Down?

Ghi thời gian mới vào DS1307

Giai\_ma\_gan\_cho\_4led\_quet

Hienthi

END

Đ

Đ

Đ

S

S

S

*Hình 3.13. Lưu đồ nút bấm*

#### **3.3.2.5 Báo thức sử dụng buzzer**

*Hình 3.14. Lưu đồ báo thức sử dụng buzzer*

Begin

Buzzer tắt

Button Alarm?

Button OK?

Hẹn giờ báo thức

Alarm=1?

Ring\_Alarm

END

S

Đ

S

Đ

Đ

S

#### **3.3.2.6 Điều chỉnh số BCD tăng/giảm**

Begin

Tách lấy byte thấp của dữ liệu đầu vào X

Y=Y+0x06

If Y=0x0A?

If Y=0x0F?

Y=Y-0x06

END

Đ

Đ

S

S

Return Y

*Hình 3.15. Lưu đồ điều chỉnh số BCD*

#### **3.3.2.7 Chương trình chính**

Begin

Khai báo cấu hình Port, các biến,ngắt timer

BDN<10?

hienthi

KT nút nhấn

Gọi CT con nút nhấn tương ứng

Gọi CT con lấy thời gian và giải mã

Alarm=1?

Ring\_alarm

END

Đ

S

Đ

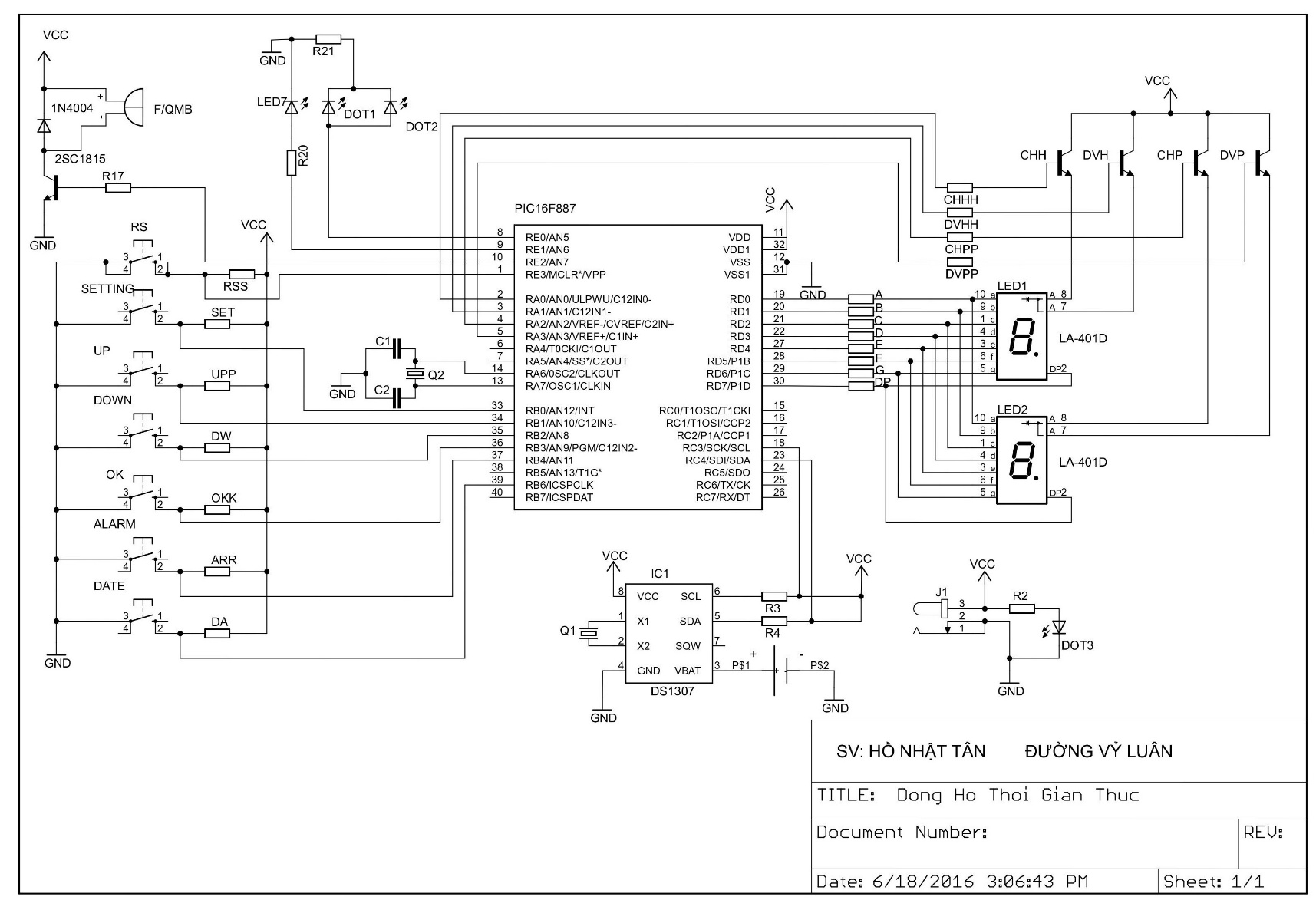
S

S

*Hình 3.16. Lưu đồ chương trình chính*

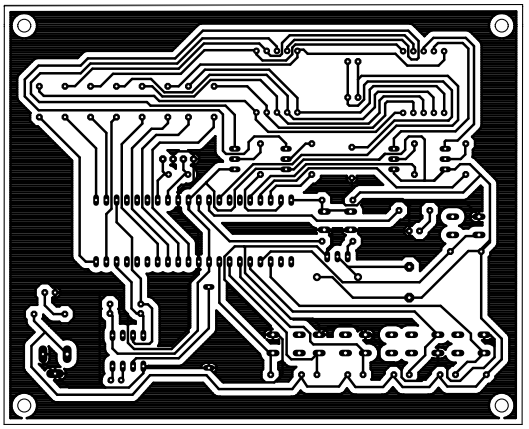
# **CHƯƠNG 4: KẾT QUẢ**

## **4.1 Sơ đồ nguyên lý**

****

*Hình 4.1 . Sơ đồ nguyên lý*

## **4.2 Sơ đồ mạch in**

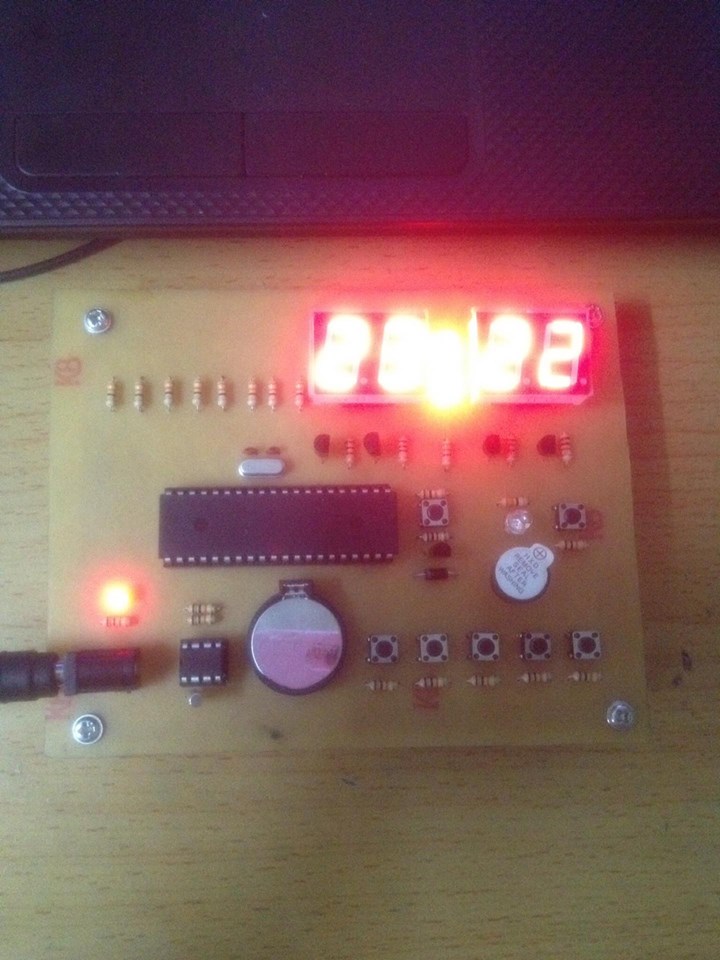


*Hình 4.2 sơ đồ mạch in hoàn chỉnh*

## **4.3 Mô hình sau khi thi công**



*Hình 4.3. Mạch thực tế khi chưa hoạt động*



## 

*Hình 4.4. Mạch thực tế khi hoạt động*

## **4.3 Vận hành**

Ban đầu, khi cấp nguồn cho mạch, vi điều khiển Pic16F887 sẽ chép đè thời gian hiện tại vào IC DS1307, sau đó sẽ đếm tiếp tục. Vi điều khiển lấy thông từ IC DS1307, gọi các chương trình con để giải mã mã BCD. Nhiệm vụ của giải mã là nếu các biến lớn hơn 10 thì cộng dồn cho 6 (nếu là đếm lên), hoặc trừ cho 6 (nếu là đếm xuống), sau đó hiển thị lên 4 led 7 đoạn thông qua phương pháp quét. Đối với phương pháp quét, mỗi một led 7 đoạn có chu kỳ quét là 25 khung hình, ở mức khung hình này thì mắt sẽ có hiện tượng lưu ảnh, vậy sử dụng bốn led 7 đoạn thì tổng chu kỳ là 100. Có chức năng báo thức, sau khi thiết lập thời gian hẹn giờ, đèn Alarm sẽ sáng, báo hiệu báo thức được kích hoạt. Khi đến giờ hẹn sẽ có tiếng báo bằng buzzer, ta nhấn Reset để tắt đi. Có 6 nút ( không kể Reset) là nút Setting, Up, Down, Alarm, Ok, Date. Chức năng của từng nút như sau:

+ Nút Setting: Nhấn một lần, nếu đang ở trạng thái bình thường (không phải đang hiển thị ngày tháng năm) thì gia\_tri\_mod tăng một đơn vị, đồng thời dấu chấm sẽ xuất hiện ở phần phút, báo hiệu cho phép chỉnh phút. Tiếp tục cho giờ, khi đang ở giờ nếu nhấn nữa sẽ quay về trạng thái bình thường. Nếu đang ở trạng thái ngày tháng năm, gia\_tri\_mod\_date tăng 1 đơn vị, tương tự như trên.

+ Nút Up: Tăng phút hoặc giờ

+ Nút Down: Giảm phút hoặc giờ

+ Nút OK: Dùng để xác nhận sau khi thiết lập hẹn giờ, hoặc đang ở trạng thái Setting, đều quay về trạng thái hiển thị bình thường.

+ Nút Alarm: Cho phép tinh chỉnh hẹn giờ, khi nhấn xuất hiện hai dấu chấm ở phút hoặc giờ ( tránh nhầm lần với trạng thái Setting).

+ Nút Date: Dùng để hiển thị ngày tháng và năm.

Video chạy thử sản phẩm: https://youtu.be/uNfF7zkEfwA

# **CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN**

## **5.1. Kết luận**

Hệ thống đạt được những yêu cầu thực tế, đó là xem thông tin giờ phút, hoặc ngày tháng năm, có chức năng hẹn giờ đầy đủ như một chiếc đồng hồ điện tử.

Áp dụng các kiến thức về lập trình CCS và tìm hiểu nguyên lý hoạt động của vi điều khiển, IC DS1307 để đưa ra ý tưởng thiết kế, lập lưu đồ khối và vẽ mạch nguyên lý. Tiến hành viết chương trình cho vi điều khiển và tiến hành kiểm tra trên mạch mô phỏng, sau đó lựa chọn các linh kiện cần thiết để thiết kế. Cuối cùng kiểm tra khả năng hoạt động của mạch thực tế, tìm ra lỗi và khắc phục.

* **Các vấn đề khi chạy thực tế:**
  + IC DS1307 trả thông tin về sai, không đếm thời gian
  + Buzzer không kêu
  + Led 7 đoạn hiển thị thiếu vài thanh sáng
* **Khắc phục:** 
  + Nạp thời gian gốc ban đầu cho IC DS1307
  + Do delay quá thấp nên buzzer không thể kêu, bỏ delay đi
  + Do chân đế tiếp xúc chưa tốt, thay chân đế mới.

## **5.2. Hướng phát triển**

Hệ thống cần trau chuốt hơn về mảng thiết kế để đẹp hơn. Thay led 7 đoạn bằng LCD để có thể hiển thị đầy đủ hơn các thông tin thời gian mà không cần dùng nút bấm để chuyển đổi. Đồng thời mang lại nhiều hiệu ứng đẹp mắt và hiển thị thêm các thông tin như: thời gian hẹn giờ còn lại, icon báo thức, chuyển đổi chế độ 12h (AM hoặc PM) và 24h. Cần lược bỏ các nút bấm cho gọn ràng, tích hợp nhiều chức năng với ít nút bấm nhất, cụ thể khi sử dụng màn hình LCD thì có thể bỏ đi nút Date, tích hợp nút Alarm vào nút Setting. Như vậy chỉ cần 3 nút bấm Setting, Up, Down là có thể hiệu chỉnh thông tin ngày tháng, cũng như đặt báo thức.

# **TÀI LIỆU THAM KHẢO**

**Tiếng Việt**

1. Nguyễn Đình Phú (2015), “Giáo trình Vi điều khiển Pic”
2. Nguyễn Đình Phú (2015), “Thực hành Vi điều khiển Pic”, trang 113-123

**Tiếng Anh**

**[1]** DALLAS SEMICONDUCTOR – DS1307 DATASHEET, 2006

**[2]** CCS C COMPILER MANUAL, October 2015

# **PHỤ LỤC**

## **1. Các lệnh lập trình PIC16F887**

**#INCLUDE**

**Cú pháp: #include<filename>**

Khai báo chỉ định đường dẫn cho trình biên dịch

**#BIT**

**Cú pháp: #bit name=x.y**

Tạo biến 1bit có tên là name đặt ở byte x vị trí y. Thường dung để kiểm tra hoặc gán giá trị cho thanh ghi. Điểm khác so với dùng biến 1 bit từ khai báo int1 là: int1 tốn 1 bit bộ nhớ, còn #BIT thì không tốn thêm bộ nhớ do name chỉ danh định đại diện chi bit chỉ định ở biến x, thay đổi name (0/1) sẽ thay đổi giá trị tương ứng y -> thay đổi giá trị x.

**#BYTE**

**Cú pháp: #byte name=x**

Gán tên biến name cho địa chỉ x. name thường trùng với tên thanh ghi có địa chỉ x

**#DEFINE**

**Cú pháp: #define name text**

Dùng để khai báo một chuỗi hoặc số có tên là name

**#USE:**

**Cú pháp: #use delay(clock=speed)**

Dùng để khai báo tốc độ dao động của thạch anh và có chỉ thị này thì bạn mới sử dụng được các hàm trễ thời gian delay\_ms hay delay\_us

**#use fast\_io(port)**

Port là các cổng vào ra của PIC (từ A -> G)

Có chỉ thị này thì chúng ta có thể điều chỉnh các port chỉ với 1 lệnh như output\_low(),input\_high()...Chú ý: trong hàm main () bạn phải dùng hàm set\_tris\_x() để chỉ rõ chân vào ra thì chỉ thị trên mới có hiệu lực, nếu không chương trình sẽ chạy sai!

**Các hàm delay**

**Delay\_cycles (count)**

count: hằng số từ 0 - 255, là số chu kỳ lệnh. 1 chu kỳ lệnh bằng 4 chu kỳ máy nhé!

ví dụ: delay\_cycles (25); với tần số dao động thạch anh OSC=20MHZ thì hàm delay này 5us=5\*10^-6s

**Delay\_us (time)**

Time là biến số thì giá trị từ 0-255, nếu là hằng số thì 0-65535.

**Delay\_ms (time)**

Tương tự với delay\_us nhưng với đơn vị là ms=10^-3s.

Các hàm vào ra

**Output\_low ( pin ) , Output\_high (pin ) :**

Dùng để thiết lập ngõ ra mức 0 (low, 0v) hoặc mức 1 (high, 5v) cho chân PIC, pin chỉ vị trí của chân.

Có thể xuất xung dùng set\_tris\_x() và #use fast\_io

**Output\_bit ( pin , value ):**

Pin: tên chân

Value: giá trị 0 hoặc 1Hàm này cũng xuất ra chân giá trị 0 hoặc 1. Tương tự như hàm trên. Thường dùng khi giá trị ra tùy thuộc giá trị của 1 biến nào đó, hoặc muốn đảo lại giá trị ngõ ra trước đó.

**Input ( pin ):**

Hàm cho giá trị đang có của chân pin là 0 hoặc 1. Giá trị 1bit

**Output\_X ( value ) :**

Hàm xuất giá trị 1 byte (8bit) ra port x, tất cả các chân của port đều là ngõ ra

**Input\_X ( ) :**

Hàm trả về giá trị 8bit (1byte) là giá trị đang có của port tên X (A -> G).

**Set\_tris\_X ( value ) :**

Hàm này dùng để định nghĩa chân IO (vào/ra) cho 1 port là ngõ vào hay ngõ ra. Chỉ dùng được khi đã khai báo tiền xử lý #use\_fast\_io.

Sử dụng #byte để tạo 1 biến chỉ đến port và thao tác trên biến này chính là thao tác trên port đó.

Value là giá trị 8 bit. Mỗi bit đại diện cho 1 chân, bit=0 thì chân đó là ngõ ra còn bit=1 chân đó là ngõ vào.

## **2. Code điều khiển**

#INCLUDE <16F887.H> //include thu vien

#FUSES NOWDT,PUT,NOLVP,HS,NOPROTECT //khai bao cac cau chi can thiet

#USE DELAY(CLOCK=20M) //khai bao thach anh su dung la 20MHZ

#USE I2C(MASTER, SLOW, SDA=PIN\_C4, SCL=PIN\_C3) //khai bao su dung giao tiep i2c, co chan SDA la chan C4, SCL la chan C3, toc do truyen la slow

/\*Dinh nghia cac bien

#DEFINE ADDR\_WR\_13B07 0xD0

#DEFINE ADDR\_RD\_13B07 0xD1

#DEFINE ADDR\_MEM 0x00

#DEFINE SETTING PIN\_B0

#DEFINE UP PIN\_B1

#DEFINE DW PIN\_B2

#DEFINE OK PIN\_B3

#DEFINE ALARM PIN\_B4

#DEFINE DATE PIN\_B5

#DEFINE BUZZER PIN\_E2

#DEFINE LED\_ALARM PIN\_E1

#DEFINE LED PIN\_E0

\*/

UNSIGNED INT8 Ma7D[10]={0xc0,0xf9,0xa4,0xb0,0x99,0x92,0x82,0xf8,0x80,0x90}; //tao bang ma led 7 doan tu so 0 den so 9

/\* khai bao cac bien kieu so nguyen 8 bit

UNSIGNED INT8 GIO\_CHUC,GIO\_DONVI,PHUT\_CHUC,PHUT\_DONVI,NGAY\_DONVI,NGAY\_CHUC,THANG\_DONVI,THANG\_CHUC,

NAM\_DONVI,NAM\_CHUC;

UNSIGNED INT8 GIAY\_DS1307,PHUT\_DS1307,GIO\_DS1307,NGAY\_DS1307,THANG\_DS1307,TEMP1,NAM\_DS1307;

UNSIGNED INT8 PHUT\_HENGIO,GIO\_HENGIO;

UNSIGNED INT8 DATA[4];

UNSIGNED INT8 QUET,BDN;

UNSIGNED INT8 GIA\_TRI\_MOD=0;

UNSIGNED INT8 GIA\_TRI\_MOD\_DATE=0;

UNSIGNED INT8 NGAY\_THANG\_NAM=0;

\*/

SIGNED INT8 ALRM; //khai báo kieu signed vi co su dung so âm

VOID DOC\_THOI\_GIAN\_DS1307()

{

I2C\_START();//cau lenh khoi tao dieu kien start khi i2c o che do Chu

I2C\_WRITE(ADDR\_WR\_13B07);//gui 1 byte tu thiet bi chu co noi dung la dia chi cua thiet bi to kem theo bit R/W=0

I2C\_WRITE(ADDR\_MEM); // thiet lap vi tri con tro thanh ghi trong ic ds1307

//RESTART THE CONDITION

I2C\_STOP();

I2C\_START();

//--get time data

I2C\_WRITE(ADDR\_RD\_13B07); //gui 1 byte tu thiet bi chu co noi dung la dia chi cua thiet bi to kem theo bit R/W=1

GIAY\_DS1307=I2C\_READ(1);//lay du lieu ve gan cho bien, tham so ben trong cho biet can lay them du lieu

PHUT\_DS1307 = I2C\_READ(1);

GIO\_DS1307 = I2C\_READ(1);

TEMP1 = I2C\_READ(1);

NGAY\_DS1307 = I2C\_REAd(1);

THANG\_DS1307 =I2C\_READ(1);

NAM\_DS1307 = I2C\_READ(0);//LAY DU LIEU XONG NGUNG (NOT ACK)

I2C\_STOP(); //ket thuc giao tiep i2c

}

VOID GHI\_THOIGIAN\_DS1307()

{

I2C\_START();

I2C\_WRITE(ADDR\_WR\_13B07);

I2C\_WRITE(ADDR\_MEM); //thiet lap vi tri con tro o dia chi 0x00

I2C\_WRITE(GIAY\_DS1307);

I2C\_WRITE(PHUT\_DS1307);

I2C\_WRITE(GIO\_DS1307);

I2C\_WRITE(TEMP1);

I2C\_WRITE(NGAY\_DS1307);

I2C\_WRITE(THANG\_DS1307);

I2C\_WRITE(NAM\_DS1307);

I2C\_STOP();

}

VOID GHI\_NGAY\_THANG\_NAM\_DS1307()

{

I2C\_START();

I2C\_WRITE(ADDR\_WR\_13B07);

I2C\_WRITE(0X04); //thiet lap vi tri con tro o dia chi 0x04

I2C\_WRITE(NGAY\_DS1307);

I2C\_WRITE(THANG\_DS1307);

I2C\_WRITE(NAM\_DS1307);

I2C\_STOP();

}

UNSIGNED INT8 CONVERT\_TO\_BCD\_UP(UNSIGNED INT8 X)

{

UNSIGNED INT8 Y; //khai bao 1 bien Y kieu du lieu so nguyen 8 bit

Y=X; //lay noi dung cua X gan cho Y

X=X & 0X0F; //Tach lay 4 bit thap cua X

IF (X==0X0A) Y=Y+0X06; //so sanh 4 bit thap co bang 0x0A khong, neu co thi cong 0x06 de thanh so BCD

RETURN Y; //Tra noi dung bien Y

}

UNSIGNED INT8 CONVERT\_TO\_BCD\_DW(UNSIGNED INT8 X)

{

UNSIGNED INT8 Y;

Y=X;

X=X & 0X0F;

IF (X==0X0F) Y=Y-0X06; //so sanh neu bang 0x0F thi tru 0x06 de thanh so BCD

RETURN Y;

}

VOID GIAI\_MA\_NGAY\_THANG\_NAM(UNSIGNED INT1 NGAY\_THANG,NAM)

{

IF(NGAY\_THANG==1)

{

NGAY\_DONVI=MA7D[NGAY\_DS1307%16];//chuyen ngay\_ds1307 thanh so thap phan roi lay phan du

NGAY\_CHUC=MA7D[NGAY\_DS1307/16];//chuyen ngay\_ds1307 thanh so thap phan roi lay phan nguyen

THANG\_DONVI=MA7D[THANG\_DS1307%16];

THANG\_CHUC=MA7D[THANG\_DS1307/16];

/\*gan cac gia tri vao mang DATA phuc vu cho viec quet du lieu

DATA[0]=NGAY\_CHUC;

DATA[1]=NGAY\_DONVI;

DATA[2]=THANG\_CHUC;

DATA[3]=THANG\_DONVI;

\*/

/\* Lam sang dau cham cua led 7 doan anode chung bang toan tu AND giua du lieu va ma 0x7F

IF (GIA\_TRI\_MOD\_DATE==1) DATA[3] =DATA[3] & 0X7F; //xuat hien dau cham de setting Ngay

ELSE IF(GIA\_TRI\_MOD\_DATE==2) DATA[1]=DATA[1] & 0X7F; //xuat hie dau cham de setting Thang

\*/

}

ELSE IF(NAM==1)

{

NAM\_DONVI=MA7D[NAM\_DS1307%16];

NAM\_CHUC=MA7D[NAM\_DS1307/16];

DATA[0]=MA7D[2];

DATA[1]=MA7D[0];

DATA[2]=NAM\_CHUC;

DATA[3]=NAM\_DONVI;

IF(GIA\_TRI\_MOD\_DATE==3) DATA[3] =DATA[3] & 0X7F;// hien thi them dau cham trong phan Setting de thay doi cai dat Nam

}

}

VOID GIAI\_MA\_GAN\_CHO\_4LED\_QUET()

{

/\* chuyen doi va tach lay cac phan don vi, chuc luu vao cac bien tuong ung

PHUT\_DONVI=MA7D[PHUT\_DS1307%16];

PHUT\_CHUC=MA7D[PHUT\_DS1307/16];

GIO\_DONVI=MA7D[GIO\_DS1307%16];

GIO\_CHUC=MA7D[GIO\_DS1307/16];

\*/

/\* Luu cac bien vao DATA phuc vu hien thi du lieu

DATA[0]=GIO\_CHUC;

DATA[1]=GIO\_DONVI;

DATA[2]=PHUT\_CHUC;

DATA[3]=PHUT\_DONVI;

\*/

/\* Xuat hien cac dau cham lua chon trong Setting

IF (GIA\_TRI\_MOD==1) DATA[3] =DATA[3] & 0X7F;

ELSE IF(GIA\_TRI\_MOD==2) DATA[1]=DATA[1] & 0X7F;

ELSE IF(GIA\_TRI\_MOD==3) {DATA[3] =DATA[3] & 0X7F;DATA[2] =DATA[2] & 0X7F;}

ELSE IF (GIA\_TRI\_MOD==6) {DATA[1]=DATA[1] & 0X7F;DATA[0]=DATA[0] & 0X7F;}

\*/

}

VOID HIENTHI()

{

UNSIGNED INT8 I,Q;

FOR(I=0;I<100;I++) //chu ky quet 100 lan

{

QUET=0x01; //khoi tao gia tri ban dau la 0x01 sau moi lan quet

/\*dua du lieu ra tung led 7 doan

FOR(Q=0;Q<4;Q++)

{

OUTPUT\_A(QUET);//xuat du lieu ra toan bo port A, nham kich hoat transistor dan o hang don vi cua 2 led 7 doan ben phai

OUTPUT\_D(DATA[Q]);//xuat du lieu can hien thi

DELAY\_MS(1);// tao do tre la 1 ms

OUTPUT\_D(0XFF); // tat du lieu

QUET=(QUET<<1); //dich trai 1 don vi, de tiep tuc kich hoat transistor tiep theo

}

\*/

}

}

VOID PHIM\_SETTING()

{

IF(!INPUT(SETTING))

{

IF(NGAY\_THANG\_NAM!=0) GIA\_TRI\_MOD\_DATE++; //neu dang o trang thai hien thi ngay thang nam, thi bien tang len 1 don vi

ELSE GIA\_TRI\_MOD++; //nguoc lai, neu o trang thai hien thi gio phut, thi bien GIA\_TRI\_MOD tang 1 don vi

IF(NGAY\_THANG\_NAM==2) GIA\_TRI\_MOD\_DATE=3; //Chuyen trang thai sang Setting Nam

/\*Moi lan nhan Setting, GIA\_TRI\_MOD tang len, se ung voi tung CASE, CASE 1 va 2 dung de setting phut, gio

//CASE 3 dung de quay lai setting phut

SWITCH(GIA\_TRI\_MOD)

{

CASE 1:

CASE 2:

{

GIAI\_MA\_GAN\_CHO\_4LED\_QUET();

BREAK;

}

CASE 3:GIA\_TRI\_MOD=1;

BREAK;

DEFAULT:BREAK;

}

\*/

/\* CASE 1 va 2 dung de setting ngay, thang, CASE 3 dung de setting nam, CASE 4 thoat ra khong setting nua

SWITCH(GIA\_TRI\_MOD\_DATE)

{

CASE 1:

CASE 2:

{

GIAI\_MA\_NGAY\_THANG\_NAM(1,0);

}

BREAK;

CASE 3: GIAI\_MA\_NGAY\_THANG\_NAM(0,1);

BREAK;

CASE 4: GIA\_TRI\_MOD\_DATE=0;

BREAK;

DEFAULT:BREAK;

}

\*/

WHILE(!INPUT(SETTING)) HIENTHI(); //trong luc nhan nut Setting led 7 doan van tiep tuc hien thi ma khong bi gian doan

}

}

VOID PHIM\_UP()

{

IF(!INPUT(UP))

{

/\* Neu nhan Setting 1 lan, se vao che do thiet lap phut

// Neu nhan Setting 2 lan, se vao che do thiet lap gio

// Bao hieu bang cach hien thi dau cham o phut hoac gio

SWITCH(GIA\_TRI\_MOD)

{

CASE 1: PHUT\_DS1307++; //tang 1 don vi

PHUT\_DS1307=CONVERT\_TO\_BCD\_UP(PHUT\_DS1307);//bien PHUT\_DS1307 duoc bien doi thanh so BCD

IF(PHUT\_DS1307==0X60) PHUT\_DS1307=0X00; //Phut khong duoc vuot qua 60, neu vuot qua se quay ve 0

GHI\_THOIGIAN\_DS1307();//goi chuong trinh con ghi du lieu

GIAI\_MA\_GAN\_CHO\_4LED\_QUET();//giai ma thanh cac du lieu don vi, chuc de hien thi led 7 doan

HIENTHI(); //goi chuong trinh quet hien thi led 7 doan

BREAK;

CASE 2: GIO\_DS1307++;

GIO\_DS1307=CONVERT\_TO\_BCD\_UP(GIO\_DS1307);

IF(GIO\_DS1307==0X24) GIO\_DS1307=0X00; //gioi han khong cho GIO\_DS1307 vuot qua 24, neu vuot qua quay ve 00

GHI\_THOIGIAN\_DS1307();

GIAI\_MA\_GAN\_CHO\_4LED\_QUET();

HIENTHI();

BREAK;

\*/

/\* Trong che do hen gio

// Nhan nut Setting de thiet lap phut, gio cho viec hen gio

// Dieu khac biet so voi doan code o tren la khong su dung ham GHI\_THOIGIAN\_DS1307()

CASE 3:

PHUT\_DS1307++;

PHUT\_DS1307=CONVERT\_TO\_BCD\_UP(PHUT\_DS1307);

IF(PHUT\_DS1307==0X60) PHUT\_DS1307=0X00;

PHUT\_HENGIO=PHUT\_DS1307;

GIO\_HENGIO=GIO\_DS1307;

GIAI\_MA\_GAN\_CHO\_4LED\_QUET();

HIENTHI();

BREAK;

CASE 6: GIO\_DS1307++;

GIO\_DS1307=CONVERT\_TO\_BCD\_UP(GIO\_DS1307);

IF(GIO\_DS1307==0X24) GIO\_DS1307=0X00;

GIO\_HENGIO=GIO\_DS1307;

PHUT\_HENGIO=PHUT\_DS1307;

GIAI\_MA\_GAN\_CHO\_4LED\_QUET();

HIENTHI();

BREAK;

DEFAULT:BREAK;

}

\*/

/\* Dung de Setting ngay, thang, nam

//

SWITCH(GIA\_TRI\_MOD\_DATE)

{

CASE 1: THANG\_DS1307++;

THANG\_DS1307=CONVERT\_TO\_BCD\_UP(THANG\_DS1307);//chuyen doi so BCD

IF(THANG\_DS1307==0X13) THANG\_DS1307=0X01;//gioi han Thang khong duoc vuot qua 13

GHI\_NGAY\_THANG\_NAM\_DS1307();//goi chuong trinh ghi du lieu vao DS1307

GIAI\_MA\_NGAY\_THANG\_NAM(1,0);

HIENTHI();

BREAK;

CASE 2: NGAY\_DS1307++;

NGAY\_DS1307=CONVERT\_TO\_BCD\_UP(NGAY\_DS1307);

IF(NGAY\_DS1307==0X32) NGAY\_DS1307=0X01; //gioi han Ngay khong vuot qua 32

GHI\_NGAY\_THANG\_NAM\_DS1307();

GIAI\_MA\_NGAY\_THANG\_NAM(1,0);

HIENTHI();

BREAK;

CASE 3: NAM\_DS1307++;

NAM\_DS1307=CONVERT\_TO\_BCD\_UP(NAM\_DS1307);

IF(NAM\_DS1307==0X99) NAM\_DS1307=0X01;//gioi han Nam khong vuot qua 99

GHI\_NGAY\_THANG\_NAM\_DS1307();

GIAI\_MA\_NGAY\_THANG\_NAM(0,1);

HIENTHI();

BREAK;

DEFAULT:BREAK;

}

\*/

}

WHILE(!INPUT(UP)) HIENTHI();

}

VOID PHIM\_DW()

{

IF(!INPUT(DW))

{

/\*

SWITCH(GIA\_TRI\_MOD)

{

CASE 1: PHUT\_DS1307--; //giam 1 don vi

IF(PHUT\_DS1307==0XFF) PHUT\_DS1307=0X59;//gioi han khong duoc nho hon 0, neu nho hon 0 quay ve 59

PHUT\_DS1307=CONVERT\_TO\_BCD\_DW(PHUT\_DS1307); //bien do so BCD

GHI\_THOIGIAN\_DS1307();

GIAI\_MA\_GAN\_CHO\_4LED\_QUET();

HIENTHI();

BREAK;

CASE 2: GIO\_DS1307--;//giam 1 don vi

IF(GIO\_DS1307==0XFF) GIO\_DS1307=0X23;//gioi han khong cho nho hon 0, neu nho hon 9 quay ve 23

GIO\_DS1307=CONVERT\_TO\_BCD\_DW(GIO\_DS1307);

GHI\_THOIGIAN\_DS1307();

GIAI\_MA\_GAN\_CHO\_4LED\_QUET();

HIENTHI();

BREAK;

\*/

/\*Dung de thiet lap phut,gio cho viec hen gio

// Diem khac biet so voi cac doan code o tren la khong co su dung ham GHI\_THOIGIAN\_DS1307()

CASE 3:

PHUT\_DS1307--;

IF(PHUT\_DS1307==0XFF) PHUT\_DS1307=0X59;

PHUT\_DS1307=CONVERT\_TO\_BCD\_DW(PHUT\_DS1307);

PHUT\_HENGIO=PHUT\_DS1307;

GIO\_HENGIO=GIO\_DS1307;

GIAI\_MA\_GAN\_CHO\_4LED\_QUET();

HIENTHI();

BREAK;

CASE 6:

GIO\_DS1307--;

IF(GIO\_DS1307==0XFF) GIO\_DS1307=0X23;

GIO\_DS1307=CONVERT\_TO\_BCD\_DW(GIO\_DS1307);

GIO\_HENGIO=GIO\_DS1307;

PHUT\_HENGIO=PHUT\_DS1307;

GIAI\_MA\_GAN\_CHO\_4LED\_QUET();

HIENTHI();

BREAK;

DEFAULT:BREAK;

}

\*/

/\*Setting ngay thang nam

SWITCH(GIA\_TRI\_MOD\_DATE)

{

CASE 1: THANG\_DS1307--;

THANG\_DS1307=CONVERT\_TO\_BCD\_DW(THANG\_DS1307);

IF(THANG\_DS1307==0X00) THANG\_DS1307=0X12;

GHI\_NGAY\_THANG\_NAM\_DS1307();

GIAI\_MA\_NGAY\_THANG\_NAM(1,0);

HIENTHI();

BREAK;

CASE 2: NGAY\_DS1307--;

NGAY\_DS1307=CONVERT\_TO\_BCD\_DW(NGAY\_DS1307);

IF(NGAY\_DS1307==0X00) NGAY\_DS1307=0X31;

GHI\_NGAY\_THANG\_NAM\_DS1307();

GIAI\_MA\_NGAY\_THANG\_NAM(1,0);

HIENTHI();

BREAK;

CASE 3: NAM\_DS1307--;

NAM\_DS1307=CONVERT\_TO\_BCD\_DW(NAM\_DS1307);

IF(NAM\_DS1307==0X00) NAM\_DS1307=0X01;

GHI\_NGAY\_THANG\_NAM\_DS1307();

GIAI\_MA\_NGAY\_THANG\_NAM(0,1);

HIENTHI();

BREAK;

DEFAULT:BREAK;

}

\*/

}

WHILE(!INPUT(DW)) HIENTHI();

}

VOID PHIM\_ALARM()

{

IF(!INPUT(ALARM))

{

SETUP\_TIMER\_1(T1\_DISABLED);//tat timer1,neu khong tat timer sau 1s se tu dong cap nhap va hien thi gio, phut

GIA\_TRI\_MOD+=3;//gia tri se tang 3 don vi

IF(GIA\_TRI\_MOD==9) GIA\_TRI\_MOD=0;//tra ve trang thai ban dau

GIAI\_MA\_GAN\_CHO\_4LED\_QUET();

ALRM=-1; //gan gia tri ban dau

WHILE(!INPUT(ALARM)) HIENTHI();

}

}

VOID PHIM\_OK()

{

IF(!INPUT(OK))

{

IF(!INPUT(OK))

//kiem tra neu co hen gio thi led alarm sang, set lai timer1, va tra ve trang thai hien thi ban dau

IF(ALRM==-1) {ALRM=1; GIA\_TRI\_MOD=0; OUTPUT\_BIT(LED\_ALARM,1);SETUP\_TIMER\_1(T1\_INTERNAL | T1\_DIV\_BY\_8);SET\_TIMER1(3036);}

GIA\_TRI\_MOD=0; GIA\_TRI\_MOD\_DATE=0; NGAY\_THANG\_NAM=0;DOC\_THOI\_GIAN\_DS1307(); GIAI\_MA\_GAN\_CHO\_4LED\_QUET();//tra ve trang thai ban dau

WHILE(!INPUT(OK)) HIENTHI();

}

}

VOID PHIM\_DATE()

{

IF(!INPUT(DATE))

{

IF(!INPUT(DATE))

SETUP\_TIMER\_1(T1\_DISABLED);//timer1 bi tat di

NGAY\_THANG\_NAM++; //tang 1 don vi

IF(NGAY\_THANG\_NAM==1) GIAI\_MA\_NGAY\_THANG\_NAM(1,0);//GIAI MA NGAY THANG

ELSE IF(NGAY\_THANG\_NAM==2) GIAI\_MA\_NGAY\_THANG\_NAM(0,1); //GIAI MA NAM

ELSE IF(NGAY\_THANG\_NAM==3) {NGAY\_THANG\_NAM=0;SETUP\_TIMER\_1(T1\_INTERNAL | T1\_DIV\_BY\_8);SET\_TIMER1(3036);}//tra ve trang thai ban dau

WHILE(!INPUT(DATE)) HIENTHI();

}

}

VOID RING\_ALARM()

{

IF ((PHUT\_HENGIO==PHUT\_DS1307) && (GIO\_HENGIO==GIO\_DS1307))

{

OUTPUT\_BIT(BUZZER,1);//dua chan Buzzer len cao

OUTPUT\_BIT(LED\_ALARM,0);//tat led alarm

}

}

#INT\_TIMER1

VOID INTERRUPT\_TIMER1()

{

BDN++; SET\_TIMER1(3036);//BDN tang 1 don vi ; set lai timer

}

VOID MAIN()

{

/\* Set cac port la in hoac out

SET\_TRIS\_D(0);

SET\_TRIS\_B(0XFF);

SET\_TRIS\_A(0);

SET\_TRIS\_E(0);

SET\_TRIS\_C(0);

\*/

OUTPUT\_BIT(LED,0); OUTPUT\_BIT(LED\_ALARM,0); OUTPUT\_BIT(BUZZER,0);//set trang thai ban dau

BDN=0;GIA\_TRI\_MOD=0;ALRM=0;//set trang thai ban dau

SETUP\_TIMER\_1(T1\_INTERNAL | T1\_DIV\_BY\_8);//thiet lap timer 1

SET\_TIMER1(3036);

ENABLE\_INTERRUPTS(GLOBAL);//khai bao su dung ngat toan cuc

ENABLE\_INTERRUPTS(INT\_TIMER1); //khai bao su dung ngat timer1

/\*Lay thoi gian lan dau tien truoc khi vao vong lap

DOC\_THOI\_GIAN\_DS1307();

GIO\_TAM=GIO\_DS1307;

GIAI\_MA\_GAN\_CHO\_4LED\_QUET();

\*/

while(true)

{

IF(BDN<=5) OUTPUT\_HIGH(LED);//Neu BDN <=5 thi den sang ( sang 500ms, tat 500ms)

ELSE OUTPUT\_LOW(LED);

IF(BDN<10)

{

HIENTHI();PHIM\_SETTING();PHIM\_UP();PHIM\_DW(); PHIM\_ALARM();PHIM\_OK();PHIM\_DATE();//hien thi du lieu va quet cac phim bam

}

ELSE

{

BDN=BDN-10; //neu BDN=10 thi reset BDN ve 0

DOC\_THOI\_GIAN\_DS1307(); GIAI\_MA\_GAN\_CHO\_4LED\_QUET();

}

IF(ALRM==1) RING\_ALARM(); //neu co hen gio thi goi ham de kiem tra

}

}