

TRƯỜNG ĐẠI HỌC KỸ THUẬT CÔNG NGHIỆP
KHOA ĐIỆN TỬ
BỘ MÔN: CÔNG NGHỆ THÔNG TIN

BÀI TẬP LỚN

MÔN HỌC

HỆ THỐNG NHÚNG

Sinh viên: Nguyễn Thị Thu Hiền

Lớp: K58KMT.01

Giáo viên hướng dẫn: ThS.Tăng Cẩm Nhung

Thái Nguyên, 2024

BÀI TẬP LỚN

MÔN HỌC: HỆ THỐNG NHÚNG

BỘ MÔN: CÔNG NGHỆ THÔNG TIN

Sinh viên: Nguyễn Thị Thu Hiền MSSV: K225480106015

Lớp: K58KMT.01 Ngành: Kỹ thuật máy tính

Giáo viên hướng dẫn: ThS. Tăng Cẩm Nhung

Ngày giao đề: 13/5/2024 Ngày hoàn thành: 18/6/2024

Tên đề tài: Báo chuông tự động theo lịch

Yêu cầu:

- Tự động đọc thời gian từ IC thời gian thực và so sánh với lịch hoặc thời gian biểu đã lưu trữ trong bộ nhớ để báo chuông đúng thời điểm.

GIÁO VIÊN HƯỚNG
DẪN

(ký và ghi rõ họ tên)

ThS.Tăng Cẩm Nhung

NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

Thái Nguyên, ngày ... tháng ... năm 2024

GIÁO VIÊN HƯỚNG DẪN

(Ký, ghi rõ họ tên)

NHẬN XÉT CỦA GIÁO VIÊN CHẤM THI

Thái Nguyên, ngày ... tháng ... năm 2024

GIÁO VIÊN CHẤM THI

(Ký, ghi rõ họ tên)

LỜI CẢM ƠN

Lời đầu tiên em xin tỏ lòng biết ơn sâu sắc đến cô giáo Tăng Cẩm Nhung đã tạo điều kiện học tập tốt nhất cho chúng em. Trong suốt học kì vừa qua cô đã rất tận tâm , tận tình giảng dạy, hướng dẫn chi tiết cho chúng em . Mặc dù chúng em hay mắc lỗi khiến cô không vui.

Trong suốt quá trình học tập và tìm hiểu bộ môn Hệ Thống Nhúng, em cũng đã cố gắng tìm đọc tài liệu, thực hành đầy đủ các buổi , làm bài tập đầy đủ mà cô giáo giao. Do kiến thức thực tế chưa nhiều nên bài tập lớn của em khó tránh khỏi những sai sót. Em rất mong nhận được sự đóng góp ý kiến của Quý thầy cô và các bạn để bài tập của em được hoàn thiện hơn!

Em xin chân thành cảm ơn cô!

MỤC LỤC

LỜI MỞ ĐẦU	10
CHƯƠNG 1. TỔNG QUAN	11
1.1. Đặt vấn đề	11
1.2 Mục tiêu đề tài.....	11
1.3 Giới hạn.....	11
1.4. Xây dựng phương án.....	12
1.5. Bố cục	12
CHƯƠNG 2. THIẾT KẾ NGUYÊN LÝ	14
2.1. Tổng quan về phần cứng	14
2.1.1. Vi điều khiển PIC16F877A	14
2.1.2. Màn hình LCD 20x4 (LM044L)	16
2.1.3. Keypad 4x3	19
2.1.4. IC thời gian thực DS1307	21
2.1.5. Sơ đồ khôi hệ thống và chức năng từng khôi	23
2.1.6. Sơ đồ nguyên lý hệ thống.....	24
2.2. Giới thiệu về chuẩn giao tiếp I2C	24
2.2.1. Cách thức hoạt động.....	25
2.2.2. Các chế độ hoạt động của chuẩn giao tiếp I2C	28
CHƯƠNG 3. THIẾT KẾ KĨ THUẬT	30
3.1. Thiết kế mạch mô phỏng theo sơ đồ khôi	30
3.1.1. Sơ đồ kết nối linh kiện với vi điều khiển PIC16F877A	30
3.1.2. Sơ đồ mạch mô phỏng hệ thống (Proteus)	34
3.2. Lưu đồ giải thuật	35
CHƯƠNG 4. THI CÔNG SẢN PHẨM	36
4.1. Giới thiệu.....	36
4.2. Quy trình sử dụng hệ thống trên mạch mô phỏng.....	37
CHƯƠNG 5. KIỂM NGHIỆM, ĐÁNH GIÁ KẾT QUẢ	41
5.1. Kết quả đạt được	41
5.2. Nhận xét và đánh giá.....	41
CHƯƠNG 6. TỔNG KẾT	42

PHỤ LỤC	42
TÀI LIỆU THAM KHẢO	52

DANH MỤC HÌNH VẼ

Hình	Trang
Hình 2.1 PIC16F877A.....	13
Hình 2.2 Sơ đồ chân của PIC16F877A.....	14
Hình 2.3 Hình ảnh thực tế của LCD 20x4.....	15
Hình 2.4 Hình ảnh thực tế Keypad 4x3.....	18
Hình 2.5 Cấu trúc Keypad 4x3.....	18
Hình 2.6 Sơ đồ chân Keypad 4x3.....	19
Hình 2.7 Hình ảnh thực tế module DS1307.....	20
Hình 2.8 Linh kiện mô phỏng DS1307 trong Proteus.....	21
Hình 2.9 Sơ đồ khái niệm hệ thống.....	22
Hình 2.10 Sơ đồ nguyên lý hệ thống.....	23
Hình 2.11. Cách thức hoạt động của I2C (1)	24
Hình 2.12. Cách thức hoạt động của I2C (2)	25
Hình 2.13. Cách thức hoạt động của I2C (3)	25
Hình 2.14. Cách thức hoạt động của I2C (4)	26
Hình 2.15. Cách thức hoạt động của I2C (5)	26
Hình 2.16. Cách thức hoạt động của I2C (6)	27
Hình 2.17. Cách thức hoạt động của I2C (7)	27
Hình 2.18. Cách thức hoạt động của I2C (8)	29
Hình 3.1 Sơ đồ kết nối xung thạch anh với vi điều khiển.....	30
Hình 3.2 Sơ đồ kết nối chân Keypad 4x3 với vi điều khiển.....	31
Hình 3.3 Sơ đồ kết nối LCD 20x4, DS1307 với vi điều khiển.....	32
Hình 3.4 Sơ đồ kết nối Speaker với vi điều khiển.....	33
Hình 3.5 Sơ đồ mạch mô phỏng Proteus cho hệ thống.....	33
Hình 3.6 Lưu đồ giải thuật.....	34
Hình 4.1 Mạch in 3D cho hệ thống trên Proteus.....	36
Hình 4.2 Mạch in PDF cho hệ thống trên Proteus.....	37
Hình 4.3 Quy trình sử dụng (bước 1)	37
Hình 4.4 Quy trình sử dụng (bước 2.1.1)	38
Hình 4.5 Quy trình sử dụng (bước 2.1.2)	38
Hình 4.6 Quy trình sử dụng (bước 2.2.1)	39
Hình 4.7 Quy trình sử dụng (bước 2.2.2)	39
Hình 4.8 Quy trình sử dụng (bước 2.2.3)	40

Hình 4.9 Quy trình sử dụng (bước 3)40

LỜI MỞ ĐẦU

Hiện nay, do sự phát triển theo hướng công nghệ của xã hội, từ lâu chúng ta đã không còn nghe thấy tiếng trống, chuông báo một cách thủ công, mà hoàn toàn dùng thiết bị công nghệ để theo dõi, thay đổi theo nhu cầu thời gian biến và tự động kêu thông báo đúng thời gian đã thiết lập.

Trong các giảng đường đại học hiện nay, chuông báo là một thiết bị không thể thiếu trong một ngày học tập, chúng được thiết lập để kêu đúng khoảng thời gian đã thiết lập ra, hoàn toàn thay thế phương pháp thủ công trước đây. Em là sinh viên trường kỹ thuật đang được học tập môn học Hệ Thống Nhúng, em sẽ làm rõ các nguyên lý hoạt động, xây dựng hệ thống chuông báo tự động và thiết kế cải tiến mới hơn sao cho linh hoạt và hiệu quả hơn.

Em xin được trình bày đề tài “**Báo chuông tự động theo lịch**” làm bài tập lớn cho môn học Hệ Thống Nhúng.

CHƯƠNG 1. TỔNG QUAN

1.1. Đặt vấn đề

Hiện nay với sự phát triển của vi điều khiển. Các hệ thống cần thiết đều được hoạt động một cách tự động. Đơn giản như hệ thống chuông hẹn giờ, hệ thống báo tự động, báo giờ học , báo giờ công sở...

Vấn đề báo giờ là thực sự cấp thiết ở bất cứ trường học,công ty,môi trường làm việc nào, giúp người quản lý điều chỉnh được thời gian chung. Sẽ mất thời gian khi ta phải canh thời gian cho từng khung giờ . Chính vì thế em thiết kế mạch “**Báo chuông tự động theo lịch**” cho các môi trường làm việc,học tập... nhằm mong muốn giải quyết vấn đề đó.

1.2. Mục tiêu đề tài

Vì đây là vấn đề cấp thiết , thực tế trên môi trường giảng đường hiện nay hoặc bất cứ môi trường làm việc có quy tắc nào đó, đều được ứng dụng rất nhiều.

Nhằm mục tiêu cung cấp kiến thức lập trình vi xử lý,vi điều khiển vừa học và rèn luyện kỹ năng học hỏi thêm từ các tài liệu liên quan đến môn học, dưới sự hướng dẫn của giảng viên.

Thiết kế hệ thống chuông báo theo lịch để đảm bảo:

- Báo hiệu cho người dùng đúng như lịch trình được thiết lập.
- Có khả năng lập trình và điều chỉnh lịch báo chuông theo lịch học hoặc thời gian biểu nhất định khác.
- Hỗ trợ thông báo khẩn cấp hoặc thay đổi lịch trình đột xuất.

1.3. Giới hạn

Khi thực hiện đề tài cần đảm bảo những giới hạn sao cho thực thi đề tài một cách hiệu quả nhất. Có thể phân loại những nhóm giới hạn như sau:

❖ Phạm vi nghiên cứu

- **Phạm vi triển khai:** Hệ thống sẽ được triển khai thử nghiệm trong một phòng làm việc thay vì toàn bộ khu vực. Điều này giúp tập trung vào việc hoàn thiện hệ thống trước khi mở rộng quy mô.
- **Khung thời gian:** Đề tài sẽ được thực hiện trong một khoảng thời gian xác định phù hợp với tiến độ học tập và nghiên cứu của nhóm.

❖ Tài chính , ngân sách

- **Ngân sách:** Hạn chế về ngân sách sẽ ảnh hưởng đến việc lựa chọn các thiết bị và linh kiện. Nhóm nghiên cứu cần phải lựa chọn các giải pháp kỹ thuật và thiết bị phù hợp với nguồn lực tài chính sẵn có.
- **Chi phí bảo trì và vận hành:** Cần xác định các chi phí bảo trì và vận hành hệ thống sau khi triển khai, đảm bảo rằng các chi phí này không quá cao so với ngân sách của nơi làm việc.

- ❖ Giới hạn về kĩ thuật
 - **Khả năng của thiết bị:** Các thiết bị sử dụng (Arduino, PIC16F877A, Keypad,...) có giới hạn về khả năng xử lý và tính năng. Điều này cần được xem xét khi thiết kế hệ thống.
 - **Tích hợp với hệ thống hiện có:** Nếu môi trường làm việc đã có sẵn hệ thống chuông hoặc các hệ thống quản lý khác, việc tích hợp và tương thích có thể gặp khó khăn.
- ❖ Nguồn nhân lực
 - **Kiến thức và kỹ năng:** Nhóm nghiên cứu có thể bị giới hạn bởi kiến thức và kỹ năng hiện có trong lĩnh vực lập trình, thiết kế mạch điện, và âm thanh.
- ❖ Cơ sở pháp lý quy định
 - **Quy định của nhà trường:** Cân tuân thủ các quy định và chính sách của nhà trường về việc lắp đặt và sử dụng các thiết bị điện tử trong môi trường nghiên cứu.
 - **Quy định về âm thanh:** Đảm bảo rằng hệ thống chuông báo không vi phạm các quy định về mức độ âm thanh trong khu vực học tập, không gây ảnh hưởng đến các khu vực khác hoặc hoạt động khác.
- ❖ Môi trường sử dụng
 - **Điều kiện vật lý:** Các điều kiện vật lý của môi trường nghiên cứu (như kích thước phòng, chất lượng âm thanh trong phòng, vị trí lắp đặt thiết bị) sẽ ảnh hưởng đến thiết kế và triển khai hệ thống.
 - **Tác động môi trường:** Đảm bảo hệ thống hoạt động ổn định trong các điều kiện môi trường khác nhau (nhiệt độ, độ ẩm, nhiễu điện từ,...).

1.4. Xây dựng phương án

Yếu tố khách thể của đề tài “**Báo chuông tự động theo lịch**” là môi trường làm việc có quy tắc về thời gian, em sẽ đưa ra các cơ sở lý thuyết như sau:

- ❖ **Cơ sở xây dựng lý thuyết**

Thời gian thực sẽ luôn hiển thị, nếu người dùng muốn hẹn giờ thì sẽ thay đổi các bước cài đặt trong hệ thống. Khi thời gian thực trùng với thời gian đã được thiết lập thì chuông sẽ kêu thông báo.
- ❖ **Đối tượng nghiên cứu**
 - Hệ thống trung tâm: Vi điều khiển PIC16F877A
 - Linh kiện thiết kế hệ thống: Keypad, Màn hình hiển thị LCD 20x4, IC thời gian thực DS1307 và các linh kiện liên quan khác.
 - Hệ thống chuông báo: Speaker
 - Phần mềm lập trình: PIC-C compiler
 - Phần mềm thiết kế mạch mô phỏng, mạch PCB: Proteus 8
 - Các linh kiện thực cần dùng để thiết kế mạch thật, phần cứng cho đề tài trên.

1.5. Bố cục

Chương 1: Tổng quan

- Chương này trình bày tổng quan, lý do chọn đề tài, mục tiêu, nội dung nghiên cứu, các giới hạn và bối cảnh.

Chương 2: Thiết kế nguyên lý

- Giới thiệu các linh kiện, thiết bị sử dụng thiết kế hệ thống, sơ đồ khái niệm và sơ đồ nguyên lý cho hệ thống.

Chương 3: Thiết kế kỹ thuật

- Tính toán thiết kế, đưa ra sơ đồ mạch mô phỏng của hệ thống.
- Thiết kế hệ thống, lưu đồ, đưa ra giải thuật và chương trình.

Chương 5: Thi công, kiểm nghiệm kết quả

- Đưa ra kết quả đạt được sau một thời gian nghiên cứu, một số hình ảnh của hệ thống, đưa ra những nhận xét, đánh giá toàn bộ hệ thống.

Chương 6: Tổng kết

- Trình bày những kết luận về hệ thống những phần làm rồi và chưa làm, đồng thời nêu ra hướng phát triển cho hệ thống.

CHƯƠNG 2. THIẾT KẾ NGUYÊN LÝ

2.1. Tổng quan về phần cứng

2.1.1. Vi điều khiển PIC16F877A



Hình 2.1 PIC16F877A

PIC16F877A là một Vi điều khiển PIC 40 chân và được sử dụng hầu hết trong các dự án và ứng dụng nhúng. Nó có năm cổng bắt đầu từ cổng A đến cổng E. Nó có ba bộ định thời trong đó có 2 bộ định thời 8 bit và 1 bộ định thời là 16 Bit. Nó hỗ trợ nhiều giao thức giao tiếp như giao thức nối tiếp, giao thức song song, giao thức I2C. PIC16F877A hỗ trợ cả ngắt chân phần cứng và ngắt bộ định thời.

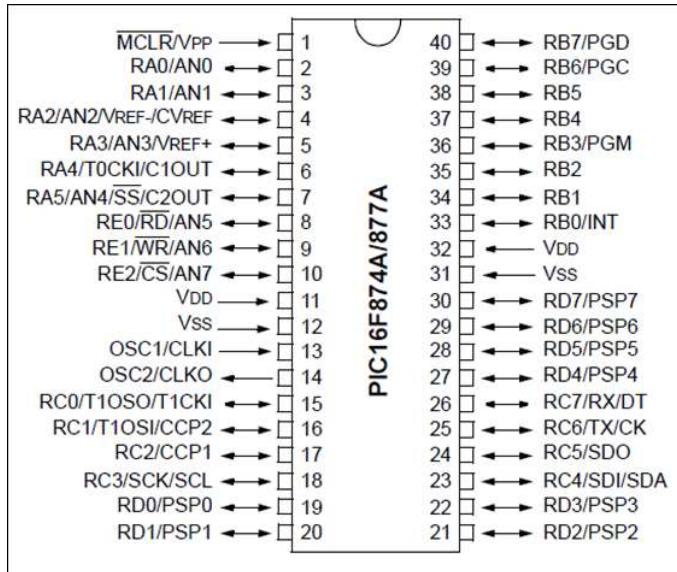
❖ Thông số kỹ thuật

CPU: PIC 8 bit

- Điện áp hoạt động: 5VDC.
- Điện áp vào giới hạn: 4 – 5.5 VDC.
- Điện áp khuyên dùng: 5 VDC.
- Dòng ra tối đa : 50 mA.
- Dòng ra tối đa trên mỗi chân I/O: 25 mA.
- Số chân digital: 33 (2 PWM).
- Số chân analog: 8
- Giao tiếp UART: 1 bộ UART.
- Giao tiếp SPI: 1 bộ (chân 7,33,34) dùng với thư viện SPI.
- Giao tiếp I2C: 0 bộ.
- Ngắt ngoài: 3 chân.
- Bộ nhớ Flash: 14.3 KB, 2 KB sử dụng cho Bootloader.
- SRAM: 368 bytes

- EEPROM: 256 bytes
- Thạch anh: 20 MHz.

❖ Sơ đồ chân của PIC16F877A



Hình 2.2. Sơ đồ chân của PIC16F877A

❖ Một số chức năng chuyên biệt của các chân

- Chân MCLR (Master Clear): Chức năng của chân này là thực hiện reset hoặc khởi động lại vi mạch.
- Chân OSC1/CLKIN và OSC2/CLKOUT: Chân này được sử dụng để kết nối với các thành phần dao động ngoài hoặc để cung cấp xung clock cho vi mạch.
- Chân VDD và VSS: Chân này được sử dụng để cung cấp nguồn điện (VDD) và kết nối với mặt đất (VSS).
- Chân RA4/T0CKI: Chức năng của chân này có thể được cấu hình là chân đầu vào ngoại vi hoặc là chân đầu vào cho bộ đếm thời gian Timer0.
- Chân RA0/AN0 đến RA5/AN5: Các chân này có thể được cấu hình là các chân đầu vào analog để sử dụng với bộ chuyển đổi Analog-to-Digital Converter (ADC) hoặc là các chân đầu vào kỹ thuật số.
- Chân RB0/RB7: Các chân này thường được sử dụng làm các chân đầu ra kỹ thuật số hoặc làm chân đầu vào ngoại vi.
- Chân RB6/RB7: Chức năng của chân này có thể được cấu hình là chân đầu ra kỹ thuật số hoặc làm chân đầu vào cho giao tiếp SPI.

- Chân RB4/RB5: Chức năng của chân này có thể được cấu hình là chân đầu ra kỹ thuật số hoặc làm chân đầu vào cho giao tiếp I2C.

❖ **Bộ nhớ**

- Flash Memory: Dùng để lưu trữ chương trình máy, với dung lượng 14.3 KB.
- SRAM (Static Random Access Memory): Bộ nhớ tạm thời dùng để lưu trữ dữ liệu khi vi xử lý đang hoạt động, với dung lượng 368 bytes.
- EEPROM (Electrically Erasable Programmable Read-Only Memory): Dùng để lưu trữ dữ liệu có thể lưu trữ lâu dài và có thể được xóa và ghi lại nhiều lần mà không cần nguồn điện, với dung lượng 256 bytes.

2.1.2. Màn hình LCD 20x4 (LM044L)

❖ **Giới thiệu sơ lược về màn hình LCD 20x4**

LCD 20x4 là loại màn hình tinh thể lỏng nhỏ dùng để hiển thị chữ hoặc số trong bảng mã ASCII. Mỗi ô của Text LCD bao gồm các chấm tinh thể lỏng, các chấm này kết hợp với nhau theo trình tự “ẩn” hoặc “hiện” sẽ tạo nên các kí tự cần hiển thị và mỗi ô chỉ hiển thị được một kí tự duy nhất.

LCD 20x4 nghĩa là loại LCD có 4 dòng và mỗi dòng chỉ hiển thị được 20 kí tự. Đây là loại màn hình được sử dụng rất phổ biến trong các loại mạch điện.



Hình 2.3 Hình ảnh thực tế của LCD 20x4

❖ **Thông số kĩ thuật**

- Điện áp: 5V
- Ngõ giao tiếp: 14 chân
- Màu sắc: xanh lá hoặc xanh dương

❖ **Chức năng chân**

- **VCC** (Positive Supply Voltage): Chân này cung cấp điện dương cho màn hình LCD.
- **GND** (Ground): Chân này được kết nối với đất, điện âm của nguồn cấp.
- **V0** (Contrast Voltage): Chân này điều chỉnh độ tương phản của màn hình. Khi áp đầu vào tăng lên, độ tương phản cũng tăng.

- **RS (Register Select):** Chân này xác định liệu dữ liệu đang được truyền đến màn hình là dữ liệu để hiển thị hoặc là lệnh điều khiển.
- **RW (Read/Write):** Chân này xác định hoạt động đọc hoặc ghi dữ liệu vào màn hình LCD. Trong hầu hết các ứng dụng, chúng ta thường sử dụng chế độ ghi (write), nên chân này thường được kết nối với đất để chỉ định hoạt động ghi.
- **E (Enable):** Chân này xác định kích hoạt hoạt động của màn hình LCD. Mỗi khi tín hiệu trên chân này chuyển từ mức thấp lên mức cao, màn hình sẽ đọc dữ liệu từ các chân dữ liệu (D0-D7).
- **D0-D7 (Data Lines):** Đây là các chân dữ liệu 8 bit, chúng được sử dụng để truyền dữ liệu đến màn hình LCD. Trong một số trường hợp, chúng có thể được sử dụng trong chế độ 4 bit, nghĩa là chỉ 4 chân (D4-D7) được sử dụng, giảm bớt số lượng chân cần thiết cho việc kết nối.
- **A (Anode):** Đây là chân anot của đèn nền nếu màn hình LCD có đèn nền.
- **K (Kathode):** Đây là chân catot của đèn nền nếu màn hình LCD có đèn nền.

❖ **Các thanh ghi**

- **Thanh ghi lệnh (Instruction Register):** Thanh ghi này được sử dụng để gửi các lệnh điều khiển đến màn hình LCD. Các lệnh bao gồm việc di chuyển con trỏ hiển thị, xóa màn hình, và thiết lập chế độ hiển thị.
- **Thanh ghi dữ liệu (Data Register):** Thanh ghi này được sử dụng để gửi dữ liệu cần hiển thị trên màn hình LCD, bao gồm các ký tự, số và các ký tự đặc biệt.
- **Thanh ghi trạng thái (Status Register):** Thanh ghi này thường không được truy cập trực tiếp từ bên ngoài, nhưng nó chứa thông tin về trạng thái hiện tại của màn hình LCD, chẳng hạn như trạng thái của các hoạt động đọc/ghi, trạng thái lệnh hiển thị hiện tại, và các cờ báo hiệu như kết thúc đợi hoặc sẵn sàng cho lệnh tiếp theo.

❖ **Cờ báo bận (Busy Flag)**

- Cờ này thường được đọc từ thanh ghi trạng thái (Status Register) của màn hình LCD. Khi BF được set, nghĩa là màn hình LCD đang bận xử lý một lệnh hoặc dữ liệu trước đó và không thể chấp nhận lệnh mới.

❖ **Tập lệnh của LCD**

Dưới đây là một số tập lệnh phổ biến được sử dụng để điều khiển một màn hình LCD 20x4 thông thường:

- Hiển thị ký tự: Gửi ký tự để hiển thị trên màn hình LCD.

RS = 1 (chế độ dữ liệu)

RW = 0 (ghi)

D7-D0 = mã ASCII của ký tự cần hiển thị

E = 1 (Xác nhận)

- Chọn hàng và cột: Di chuyển con trỏ đến một hàng và cột cụ thể trên màn hình LCD.

RS = 0 (chế độ lệnh)

RW = 0 (ghi)

D7-D0 chứa mã lệnh để di chuyển con trỏ chuột đến hàng và cột cụ thể trên màn hình

E = 1 (Xác nhận)

- Xóa màn hình: Xóa nội dung của màn hình LCD.

RS = 0 (chế độ lệnh)

RW = 0 (ghi)

D7-D0 = 0x01

E = 1 (Xác nhận)

- Di chuyển con trỏ: Di chuyển con trỏ hiển thị mà không thay đổi nội dung.

RS = 0 (chế độ lệnh)

RW = 0 (ghi)

D7-D0 = 0x14 (Di chuyển sang trái) hoặc 0x10 (Di chuyển sang phải)

E = 1 (Xác nhận)

- Bật/tắt con trỏ: Bật hoặc tắt con trỏ hiển thị.

RS = 0 (chế độ lệnh)

RW = 0 (ghi)

D7-D0 = 0x0C (Bật con trỏ, tắt nháy nháy) hoặc 0x0E (Bật con trỏ, bật nháy nháy)

E = 1 (Xác nhận)

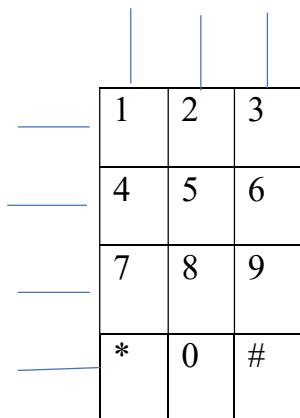
2.1.3. Keypad 4x3



Hình 2.4 Hình ảnh thực tế keypad 4x3

Keypad 4x3 là một thiết bị nhập liệu phổ biến trong các hệ thống điện tử, có cấu trúc gồm 4 hàng và 3 cột, tạo ra tổng cộng 12 nút bấm. Nó thường được sử dụng trong các ứng dụng như máy tính tiền, hệ thống bảo mật, điều khiển truy cập, và các hệ thống nhúng khác.

❖ Cấu trúc Keypad 4x3

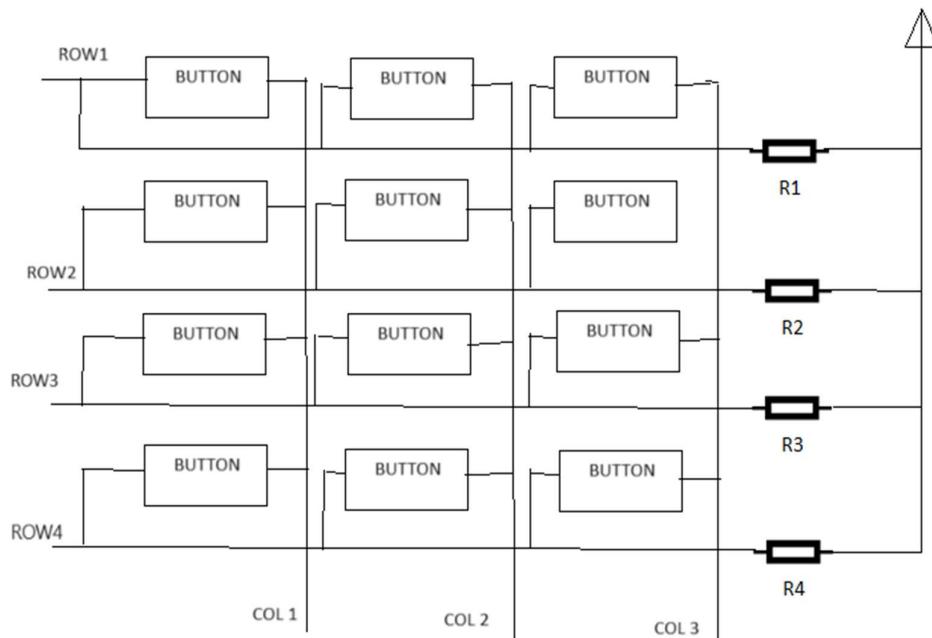


Hình 2.5 Cấu trúc Keypad 4x3

- Các Hàng (Rows): Thường được đánh số từ 1 , 4 , 7 , *
- Các Cột (Columns): Thường được đánh số từ 1, 2, 3.

❖ Cách hoạt động

- Nguyên lý: Keypad hoạt động dựa trên nguyên lý ma trận. Khi một nút bấm được nhấn, nó sẽ kết nối một hàng và một cột lại với nhau. Vi điều khiển sẽ quét hàng và cột để xác định nút nào đã được nhấn.
- Quét Keypad: Vi điều khiển sẽ đặt một mức logic thấp (LOW) lần lượt trên từng cột và đọc trạng thái của các hàng. Nếu hàng nào đó ở mức thấp (LOW) khi cột tương ứng được quét, điều đó có nghĩa là nút ở giao điểm của hàng và cột đó đã được nhấn.



Hình 2.6 Sơ đồ chân Keypad 4x3

2.1.4. IC thời gian thực DS1307

Module thời gian thực RTC DS1307 rất nhỏ gọn và đầy đủ tính năng, rất lý tưởng cho việc thêm vào dự án của bạn hoặc thử nghiệm ngay một cách dễ dàng. Giao thức I2C cho phép đếm, giây, phút, giờ, ngày trong tuần, ngày trong tháng và năm. Sẵn pin backup, cho phép duy trì thời gian ngay cả khi mất điện.

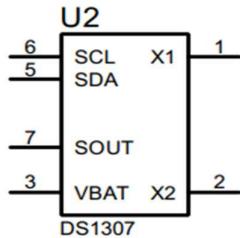


Hình 2.7 Hình ảnh thực tế module DS1307

❖ Thông số kỹ thuật

- Điện áp làm việc: 3.3V đến 5V
- Bao gồm 1 IC thời gian thực DS1307
- Các thành phần cần thiết như thạch anh 32768kHz, điện trở pull-up và tụ lọc nguồn đều được tích hợp trên board
- LED báo nguồn
- Có sẵn pin dự phòng duy trì thời gian khi mất điện
- 5-pin bao gồm giao thức I2C sẵn sàng giao tiếp: INT (QWO), SCL, SDA, VCC và GND
- Dễ dàng Thêm một đồng hồ thời gian thực để dự án của bạn
- Nhỏ gọn và dễ dàng để lắp thêm vào bo mạch hoặc test board

❖ Cách thức hoạt động của DS1307



Hình 2.8 Linh kiện mô phỏng DS1307 trong Proteus

- **Giao tiếp I2C:**

DS1307 sử dụng giao thức I2C để giao tiếp với vi điều khiển. Nó có hai chân cho giao tiếp I2C: SDA (Data) và SCL (Clock).

Địa chỉ I2C của DS1307 là 0x68.

- **Cấu trúc thanh ghi:**

DS1307 có các thanh ghi từ 00h đến 06h lưu trữ các giá trị thời gian và ngày tháng.

Cụ thể:

00h: Giây

01h: Phút

02h: Giờ

03h: Ngày trong tuần (1 đến 7)

04h: Ngày

05h: Tháng

06h: Năm

- **Định dạng dữ liệu:**

Thời gian và ngày tháng được lưu trữ ở định dạng BCD (Binary-Coded Decimal). Ví dụ, giá trị 0x25 trong thanh ghi phút tương ứng với 25 phút.

- **Đọc dữ liệu từ DS1307:**

- Để đọc dữ liệu từ DS1307, bạn cần gửi địa chỉ thiết bị (0xD0 để ghi và 0xD1 để đọc) và địa chỉ thanh ghi muốn đọc. Sau đó, DS1307 sẽ gửi dữ liệu từ thanh ghi đó.

- Ghi dữ liệu vào DS1307:

- Để ghi dữ liệu vào DS1307, bạn cần gửi địa chỉ thiết bị (0xD0), địa chỉ thanh ghi, và dữ liệu muốn ghi vào thanh ghi đó.

- Lập trình thời gian thực:

Khi lập trình thời gian cho DS1307, bạn cần chuyển đổi thời gian từ định dạng thập phân sang BCD trước khi ghi vào các thanh ghi của DS1307.

Ví dụ: Để đặt thời gian là 14:25:36, bạn cần chuyển đổi và ghi các giá trị như sau:

Giây: 36 -> 0x36

Phút: 25 -> 0x25

Giờ: 14 -> 0x14

2.1.5. Sơ đồ khái niệm hệ thống và chức năng từng khái niệm



Hình 2.9 Sơ đồ khái niệm hệ thống

❖ Các khái niệm sử dụng trong hệ thống

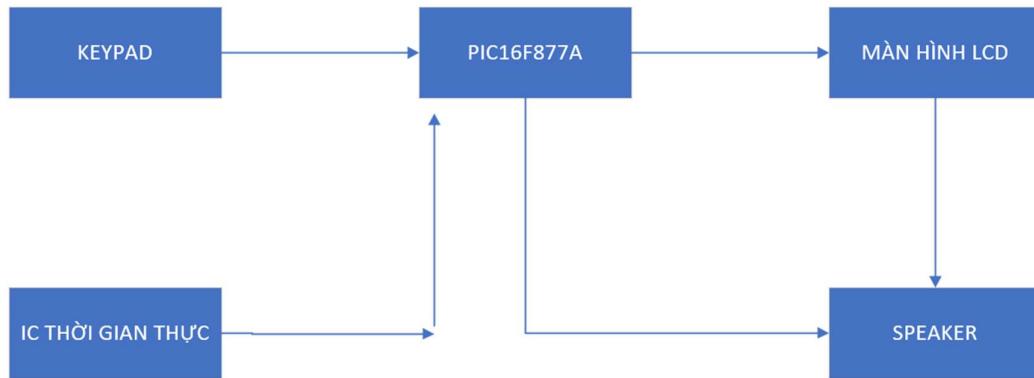
Khối nguồn: cung cấp cho toàn mạch, trong đề tài này sử dụng 2 nguồn: 12VDC cấp cho khái niệm xử lý trung tâm, 5VDC cấp cho khái niệm thu thập, khái niệm hiển thị, khái niệm truyền và nhận.

Khái niệm xử lý trung tâm: điều khiển mọi sự hoạt động của hệ thống theo chương trình đã nạp sẵn. Nhận dữ liệu từ khái niệm thu thập, xử lý rồi gửi tín hiệu đến khái niệm hiển thị và khái niệm truyền và nhận nếu có yêu cầu.

Khái niệm hiển thị: là màn hình LCD. Khái niệm xử lý trung tâm sẽ gửi dữ liệu để hiển thị ra khái niệm hiển thị.

Khối thu thập: thu thập dữ liệu từ các cảm biến nhiệt độ, độ ẩm đến khối xử lý trung tâm.

2.1.6. Sơ đồ nguyên lý hệ thống



Hình 2.10 Sơ đồ nguyên lý hệ thống

❖ Nguyên lý làm việc

IC thời gian thực sẽ đưa dữ liệu đồng hồ thực vào bộ phận xử lý trung tâm PIC16F877A , sau đó sẽ xuất dữ liệu ra màn hình LCD.

Sau đó keypad được dùng để cho người sử dụng nhập và thiết lập dữ liệu thời gian, lịch thực để bộ xử lý trung tâm sẽ in lên màn hình LCD thao tác thiết lập và kết quả thu được .

Bước cuối cùng, sau khi thời gian đã được cài đặt hẹn giờ thì màn hình LCD sẽ hiển thị thời gian thực trùng với thời gian đã hẹn. Bộ xử lý trung tâm sẽ gửi tín hiệu đến Speaker để phát âm thanh chuông thông báo .

2.2. Giới thiệu về chuẩn giao tiếp I2C

❖ Giới thiệu

I2C là tên viết tắt của cụm từ tiếng anh “Inter-Integrated Circuit”. Nó là một giao thức giao tiếp được phát triển bởi Philips Semiconductors để truyền dữ liệu giữa một bộ xử lý trung tâm với nhiều IC trên cùng một board mạch chỉ sử dụng hai đường truyền tín hiệu.

Do tính đơn giản của nó nên loại giao thức này được sử dụng rộng rãi cho giao tiếp giữa vi điều khiển và mảng cảm biến, các thiết bị hiển thị, thiết bị IoT, EEPROMs, v.v ...

Đây là một loại giao thức giao tiếp nối tiếp đồng bộ. Nó có nghĩa là các bit dữ liệu được truyền từng bit một theo các khoảng thời gian đều đặn được thiết lập bởi một tín hiệu đồng hồ tham chiếu.

Điểm mạnh của I2C chính là hiệu suất và sự đơn giản của nó: 1 khói điều khiển trung tâm có thể điều khiển cả một mạng thiết bị mà chỉ cần hai ngõ ra điều khiển.

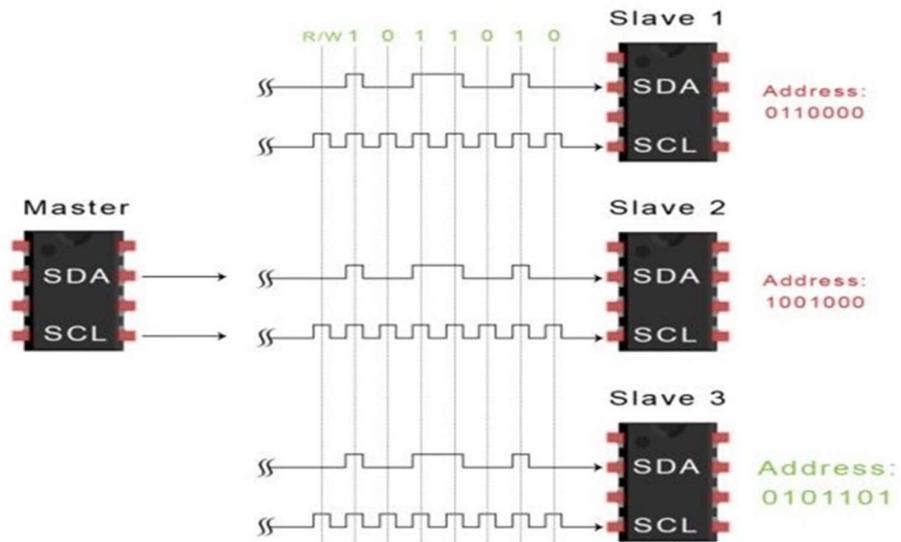
Chuẩn I2C có 2 đường tín hiệu là SDA (serial data) có chức năng truyền dữ liệu và tín hiệu SCL (serial clock) truyền tải xung clock để dịch chuyển dữ liệu. Mỗi thiết bị có 1 địa chỉ được cài sẵn hoặc 1 địa chỉ thiết bị duy nhất để thiết bị chủ (Master) có thể giao tiếp, việc tạo ra xung clock đó là do thiết bị chủ (Master). Còn thiết bị nhận xung clock là tớ (Slave), 2 chân SDA và SCL luôn hoạt động ở chế độ mở, vì vậy để sử dụng được cần phải có trở kéo bởi các thiết bị trên bus I2C hoạt động ở mức thấp. Giá trị thường được sử dụng cho các điện trở là từ 2K cho tốc độ vào khoảng 400 kbps, và 10K cho tốc độ thấp hơn khoảng 100 kbps.

2.2.1. Cách thức hoạt động

❖ Gửi dữ liệu đến thiết bị Slave

Trình tự hoạt động sau đây diễn ra khi một thiết bị Master gửi dữ liệu đến một thiết bị Slave cụ thể thông qua bus I2C:

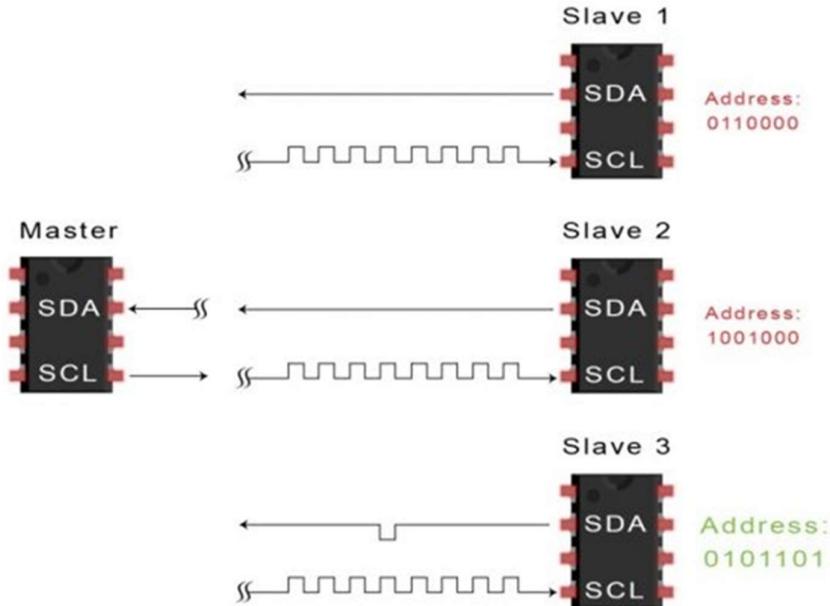
- Thiết bị Master gửi điều kiện bắt đầu đến tất cả các thiết bị Slave
- Thiết bị Master gửi 7 bit địa chỉ của thiết bị Slave mà thiết bị Master muốn giao tiếp cùng với bit Read/write.



Hình 2.11 Cách thức hoạt động của I2C (1)

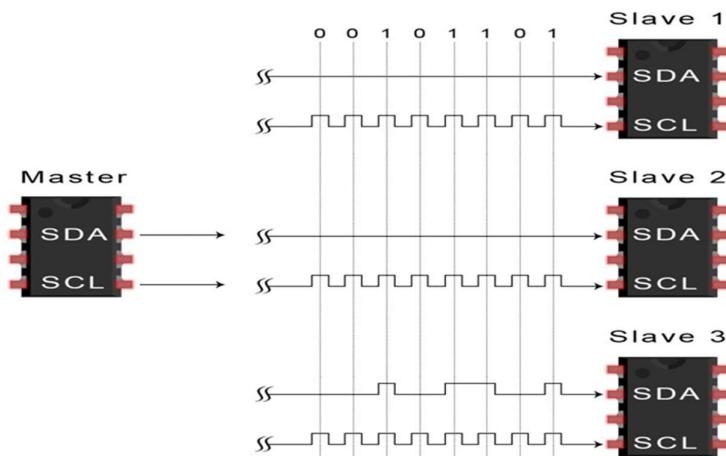
- Mỗi thiết bị Slave so sánh địa chỉ được gửi từ thiết bị Master đến địa chỉ riêng của nó. Nếu địa chỉ trùng khớp, thiết bị Slave gửi về một bit ACK bằng cách kéo đường SDA xuống thấp và bit ACK / NACK được thiết lập là '0'. Nếu địa

chỉ từ thiết bị Master không khớp với địa chỉ riêng của thiết bị Slave thì đường SDA ở mức cao và bit ACK / NACK sẽ ở mức ‘1’.



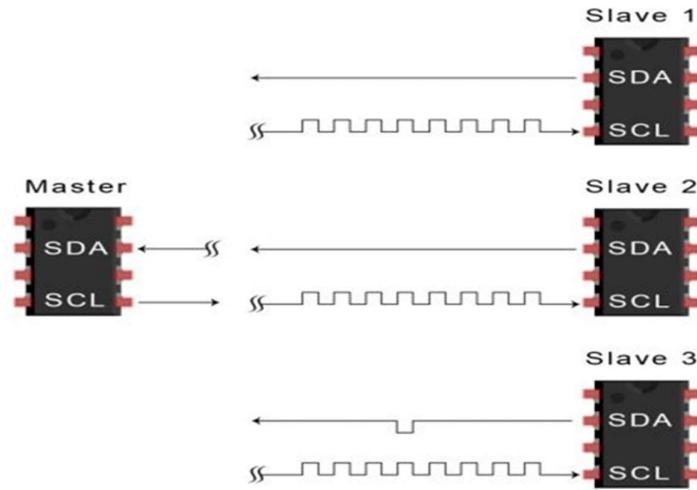
Hình 2.12 Cách thức hoạt động của I2C (2)

- Thiết bị Master gửi hoặc nhận khung dữ liệu. Nếu thiết bị Master muốn gửi dữ liệu đến thiết bị Slave, bit Read / Write là mức điện áp thấp. Nếu thiết bị Master đang nhận dữ liệu từ thiết bị Slave, bit này là mức điện áp cao.



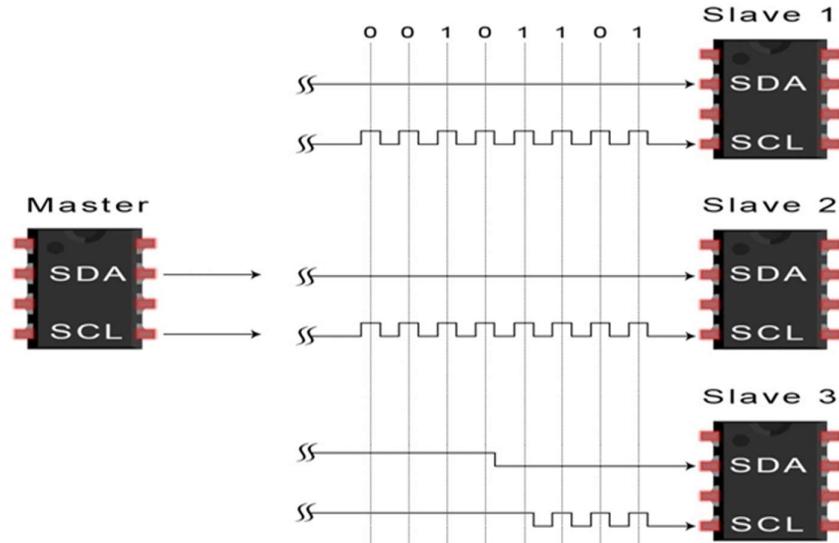
Hình 2.13 Cách thức hoạt động của I2C (3)

- Nếu khung dữ liệu được thiết bị Slave nhận được thành công, nó sẽ thiết lập bit ACK / NACK thành ‘0’, báo hiệu cho thiết bị Master tiếp tục.



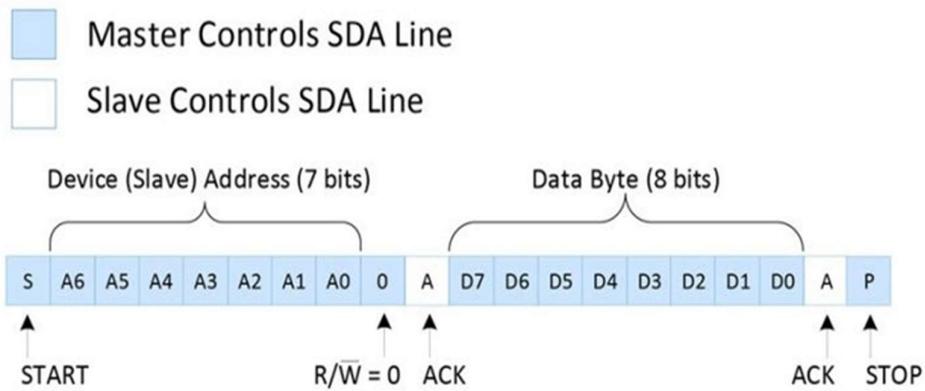
Hình 2.14 Cách thức hoạt động của I2C (4)

- Sau khi tất cả dữ liệu được gửi đến thiết bị Slave, thiết bị Master gửi điều kiện dừng để báo hiệu cho tất cả các thiết bị Slave biết rằng việc truyền dữ liệu đã kết thúc.



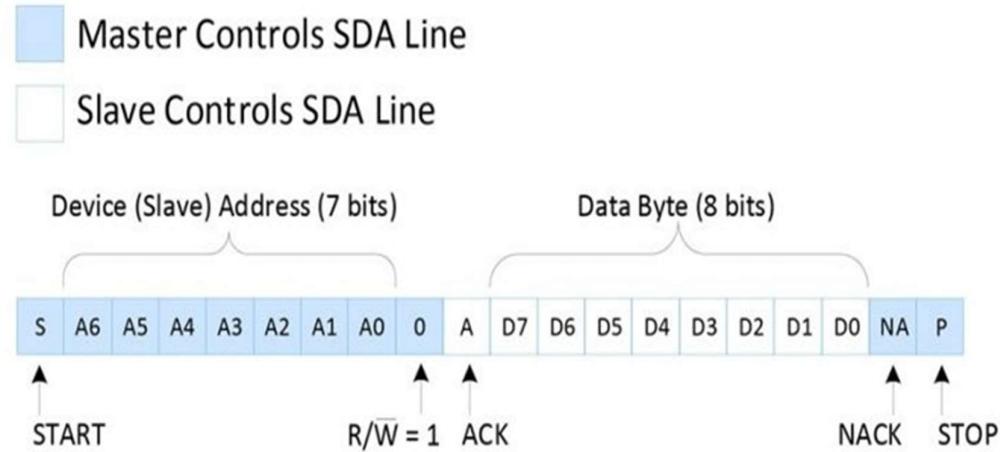
Hình 2.15 Cách thức hoạt động của I2C (5)

Hình dưới đây thể hiện toàn bộ các bit dữ liệu được gửi trên đường SDA và thiết bị điều khiển chúng khi thiết bị Master gửi dữ liệu đến thiết bị Slave.



Hình 2.16 Cách thức hoạt động của I2C (6)

Hình dưới đây thể hiện toàn bộ các bit dữ liệu được gửi trên đường SDA và thiết bị điều khiển chúng khi thiết bị Master gửi dữ liệu đến thiết bị Slave.



Hình 2.17 Cách thức hoạt động của I2C (7)

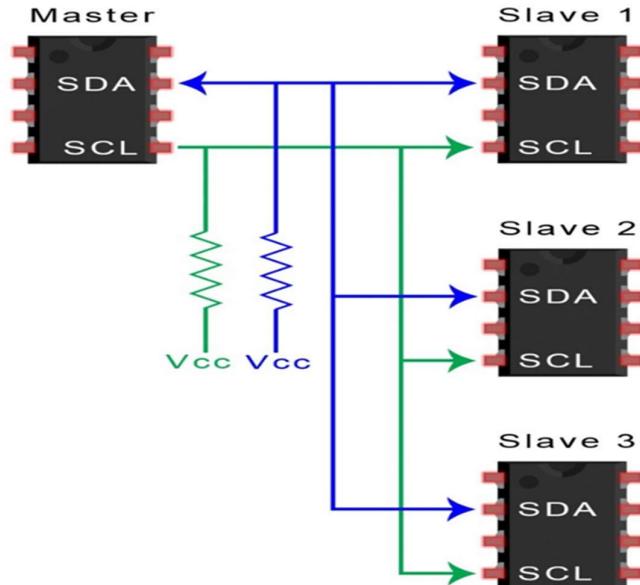
2.2.2. Các chế độ hoạt động của chuẩn giao tiếp I2C

- ❖ Dựa vào tốc độ chia làm 2 loại:
 - Chế độ chuẩn(standard mode) hoạt động ở tốc độ 100 Kbit/s
 - Chế độ tốc độ thấp(low-speed mode) hoạt động ở tốc độ 10 Kbit/s
- ❖ Nếu chia theo quan hệ chủ tớ:

- Một Master một Slave
- Một Master nhiều Slave
- Nhiều Master nhiều Slave

❖ Một Master nhiều Slave

Bởi vì I2C sử dụng địa chỉ, nhiều thiết bị Slave có thể được điều khiển từ một thiết bị Master duy nhất. Với 7 bit địa chỉ tương ứng 128 (27) địa chỉ duy nhất có sẵn. Để kết nối nhiều thiết bị Slave với một thiết bị Master duy nhất, hãy nối dây như hình bên dưới, với các điện trở kéo lên $4.7\text{ k}\Omega$ kết nối các đường SDA và SCL với Vcc:

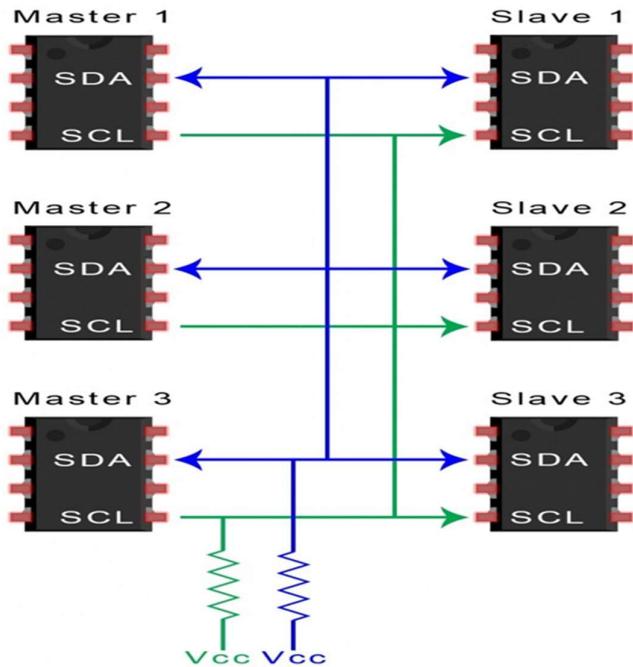


Hình 2.18 Cách thức hoạt động của I2C (8)

❖ Nhiều Master nhiều Slave

- Nhiều thiết bị Master có thể được kết nối với một thiết bị Slave đơn hoặc nhiều thiết bị Slave. Vấn đề với nhiều thiết bị Master trong cùng một hệ thống xuất hiện khi hai thiết bị Master cố gắng gửi hoặc nhận dữ liệu cùng một lúc trên dòng SDA.
- Để giải quyết vấn đề này, mỗi thiết bị Master cần phát hiện xem đường SDA thấp hay cao trước khi truyền một thông điệp. Nếu đường SDA thấp, điều này có nghĩa là một thiết bị Master khác có quyền điều khiển bus và thiết bị Master còn lại phải đợi để gửi tin nhắn. Nếu đường SDA cao thì an toàn để truyền tải thông điệp.

- Để kết nối nhiều thiết bị Master với nhiều thiết bị Slave, hãy sử dụng sơ đồ sau đây, với điện trở kéo lên $4.7\text{ k}\Omega$ kết nối các đường SDA và SCL với Vcc:



Hình 2.19 Cách thức hoạt động của I2C (9)

CHƯƠNG 3. THIẾT KẾ KĨ THUẬT

3.1. Thiết kế mạch mô phỏng theo sơ đồ khôi

3.1.1. Sơ đồ kết nối linh kiện với vi điều khiển PIC16F877A

❖ Khối tạo dao động xung thạch anh

- Linh kiện bao gồm: 1 Crystal , 2 tụ điện CAP

- Kết nối 2 tụ điện vào crystal , chân của crystal sẽ nối vào chân 13(OSC1/CKLIN),14 (OSC2/CKLOUT). Đầu còn lại sẽ kết nối với GND

• Chức năng

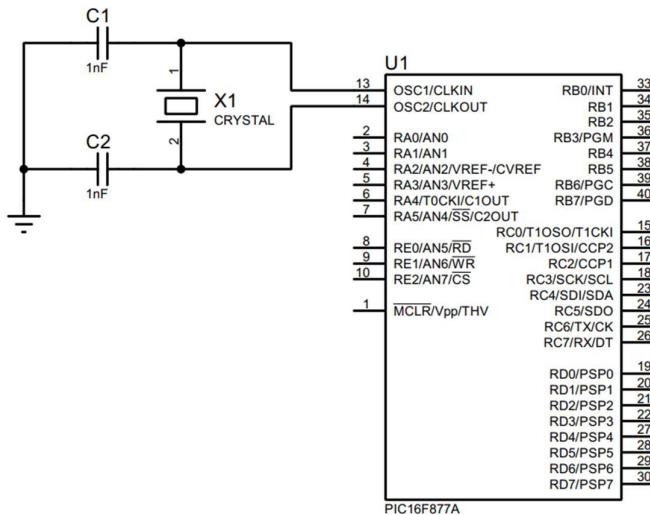
- Xung thạch anh giúp hệ thống ổn định xung nhịp, giúp chương trình chạy một cách nhất quán và chính xác.

- Điều khiển được tốc độ xử lý , tần số cao sẽ thực hiện các câu lệnh nhanh hơn.

- Giúp đồng bộ hóa các thiết bị ngoại vi như đồng bộ I2C, UART ,.. để đảm bảo truyền nhận dữ liệu.

- Tạo ra các tín hiệu thời gian chính xác

- Trong các ứng dụng cần đo lường thời gian chính xác hoặc tạo ra các tín hiệu thời gian chính xác (như báo thức, đếm ngược thời gian), xung thạch anh cung cấp một nguồn thời gian ổn định để vi điều khiển có thể tạo ra các tín hiệu thời gian chính xác và nhát quản.



Hình 3.1 Sơ đồ kết nối xung thạch anh với vi điều khiển

❖ Kết nối Keypad 4x3 với các chân vi điều khiển PIC16F877A

- Có 2 cách để biểu thị một keypad : dùng button , dùng keypad-phone

Đối với đề tài, em sẽ dùng các button để kết nối thành keypad. Cần dùng 12 button , 4 điện trở resistor $10k\Omega$. Các điện trở sẽ được cấp dương VCC , các hàng và cột sẽ kết nối với port của LCD.

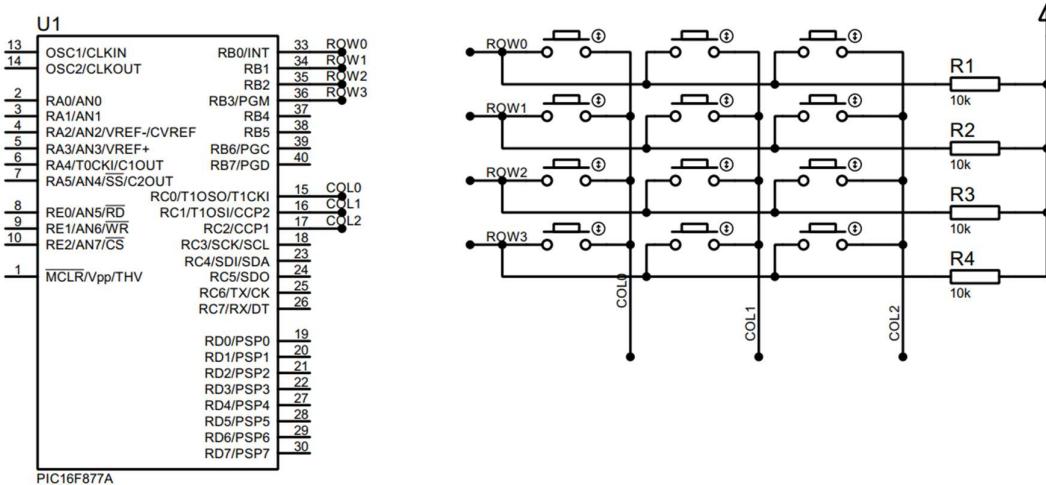
- Chức năng của điện trở: để đảm bảo các đầu vào của vi điều khiển có mức điện áp xác định khi nút bấm chưa được bấm.

+ Kéo mức điện áp lên: khi nút bấm không được nhấn , đầu vào sẽ giữ mức điện áp cao(mức logic 1). Khi được nhấn thì đầu vào sẽ bị kéo xuống mức thấp (logic 0) qua công tắc.

+ Kéo mức điện áp xuống thấp: Khi nút không được nhấn, đầu vào sẽ giữ mức tích cực thấp (mức logic 0). Khi nút bấm được nhấn, đầu vào sẽ giữ mức cao qua công tắc.

+ Chống nhiễu: Điện trở giúp giảm thiểu nhiễu trên các đầu vào của vi điều khiển và nguồn âm. Khi không có điện trở, các chân đầu vào có thể bị nhiễu từ môi trường dẫn đến việc đọc sai giá trị.

+ Đảm bảo hoạt động ổn định: Đảm bảo cho vi điều khiển nhận được các tín hiệu rõ ràng, ổn định từ nút bấm, giúp vi điều khiển tránh hiểu nhầm về trạng thái nút bấm.



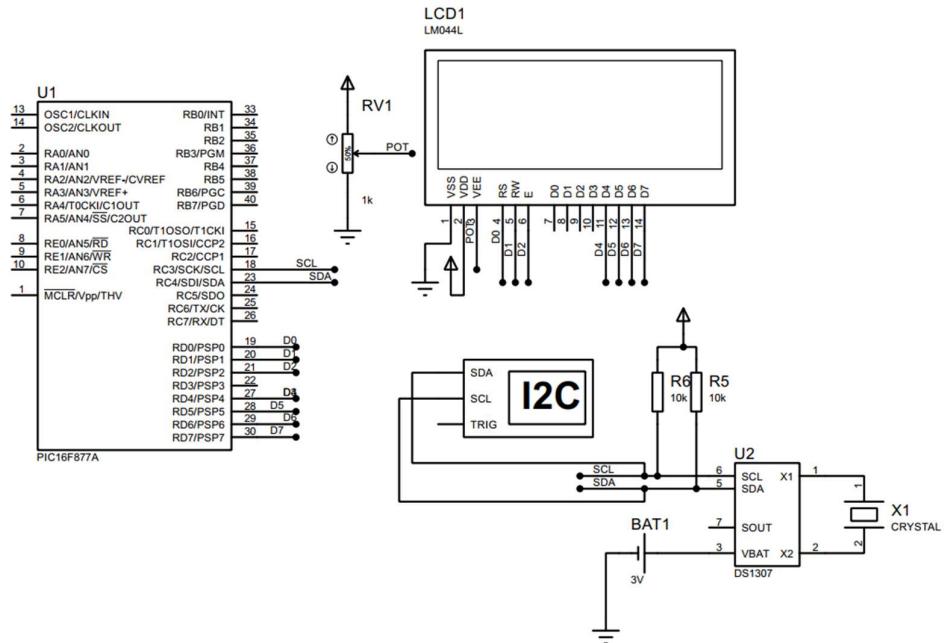
Hình 3.2 Sơ đồ nối chân Keypad 4x3 với vi điều khiển

❖ **Kết nối LCD 20x4 ,DS1307 với vi điều khiển.**

- LCD 20x4 bao gồm 14 chân:
- + Chân VSS nối GND , chân VDD nối với nguồn dương , chân VEE có thể kết nối với biến trở(nếu cần). Vì biến trở có chức năng giúp tăng giảm độ tương phản của màn hình.
- + Các chân RS,RW,E , D4-D7 nối với port VĐK.
- + Sử dụng các chân từ D4-D7 vì giao tiếp 4 bits nhằm tiết kiệm chân I/O. Nếu nối hết các chân D0-D7 là đang sử dụng 8 bits .
- + Các chân RS,RW,E sử dụng để lựa chọn giữa các thanh ghi dữ liệu và thanh ghi lệnh của LCD. Đọc/viết các dữ liệu vào LCD. Chân E để kích hoạt LCD.
- DS1307
- 2 chân SCL,SDA của DS1307 được nối với chân C3,C4 của vi điều khiển. 2 chân x1,x2 được kết nối với Crystal. Chân VBAT được kết nối với một battery 3V, 2 điện trở .
- + Battery 3V được dùng để kết nối với DS1307 bởi thực tế linh kiện thực của đồng hồ sẽ dùng pin 3V, tích trữ năng lượng , phòng trường hợp nguồn điện bị ngắt (mất điện) gây mất độ chính xác của thời gian thực.
- + Dùng 2 điện trở để tránh nhiễu , đảm bảo tín hiệu rõ ràng . Đặc tính của giao tiếp I2C . I2C là giao tiếp bus hai dây (SCL và SDA) được thiết kế để các thiết bị trên bus có thể cùng kết nối với nhau. Mỗi thiết bị trên bus I2C có thể là một master (chủ) hoặc slave (tớ).
- Dây SCL và SDA được chia sẻ giữa tất cả các thiết bị trên bus và hoạt động ở chế độ "open-drain" hoặc "open-collector". Điều này có nghĩa là các thiết bị chỉ có thể kéo dây xuống mức thấp (0V), nhưng không thể đẩy dây lên mức cao (VCC).

Để đạt mức cao (1), dây SCL và SDA cần phải được kéo lên mức điện áp VCC bằng điện trở kéo lên.

+ Xung thạch anh giúp xung nhịp ổn định, đảm bảo tính chính xác của RTC . Thông thường tần số sẽ được duy trì 32.768KHz.

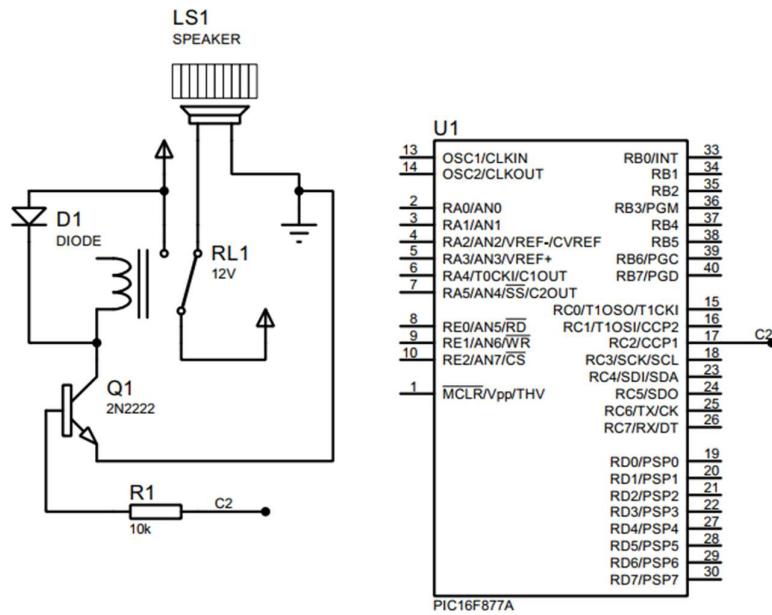


Hình 3.3 Sơ đồ nối LCD 20x4 ,DS1307 với vi điều khiển

❖ Kết nối Speaker với vi điều khiển PIC16F877A

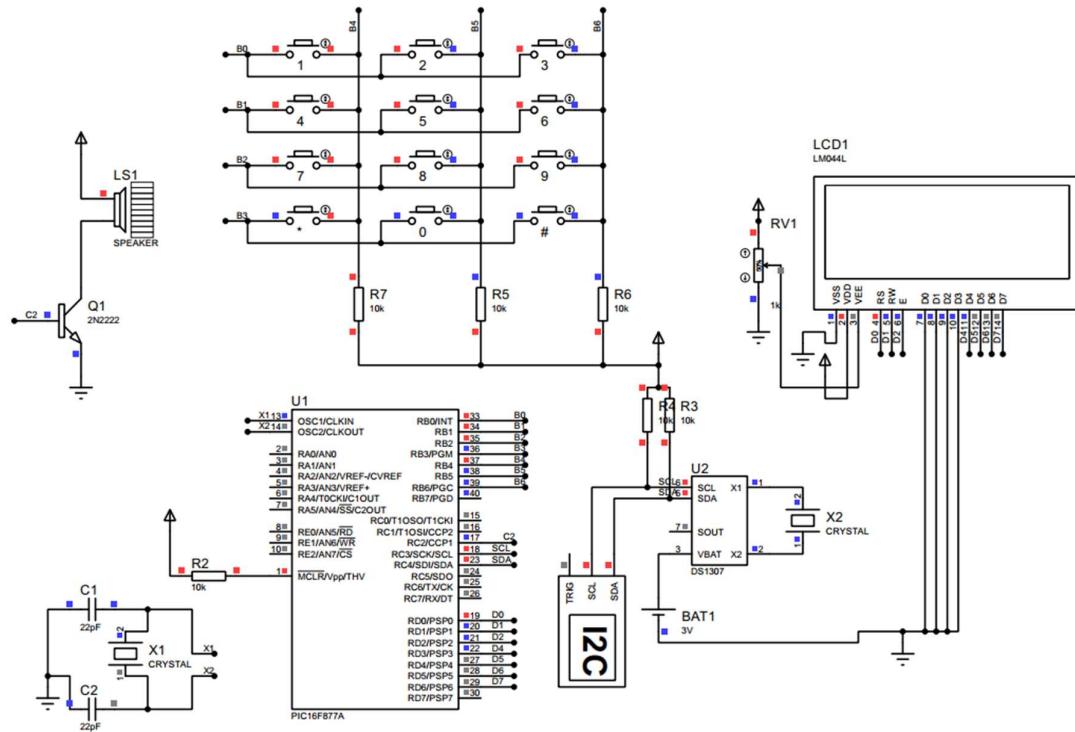
- Speaker

- Đầu âm của loa được nối GND , đầu còn lại nối với đầu Collector của transistor . Chân của transistor nối với chân vi điều khiển thông qua một điện trở, chân Emitter của transistor thì nối GND.
 - + Transistor có tác dụng như một công tắc kiểm soát dòng điện đi qua loa từ nguồn cấp của vi điều khiển.
 - + Điện trở được dùng để hạn chế dòng điện đi đến transistor tránh mức cao quá sẽ gây quá tải cho vi điều khiển, gây hư hỏng , tránh những xung điện áp đột ngột vào transistor.
 - + Ngoài ra(nếu cần) có thể thêm Relay, diode bảo vệ. Relay có tác dụng để cách điện ,điều khiển dòng điện nếu quá lớn , transistor sẽ không thể xử lý trực tiếp được. Diode nối song song với loa để đảm bảo transistor khỏi dao động, không có điện áp ngược từ loa khi ngắn.



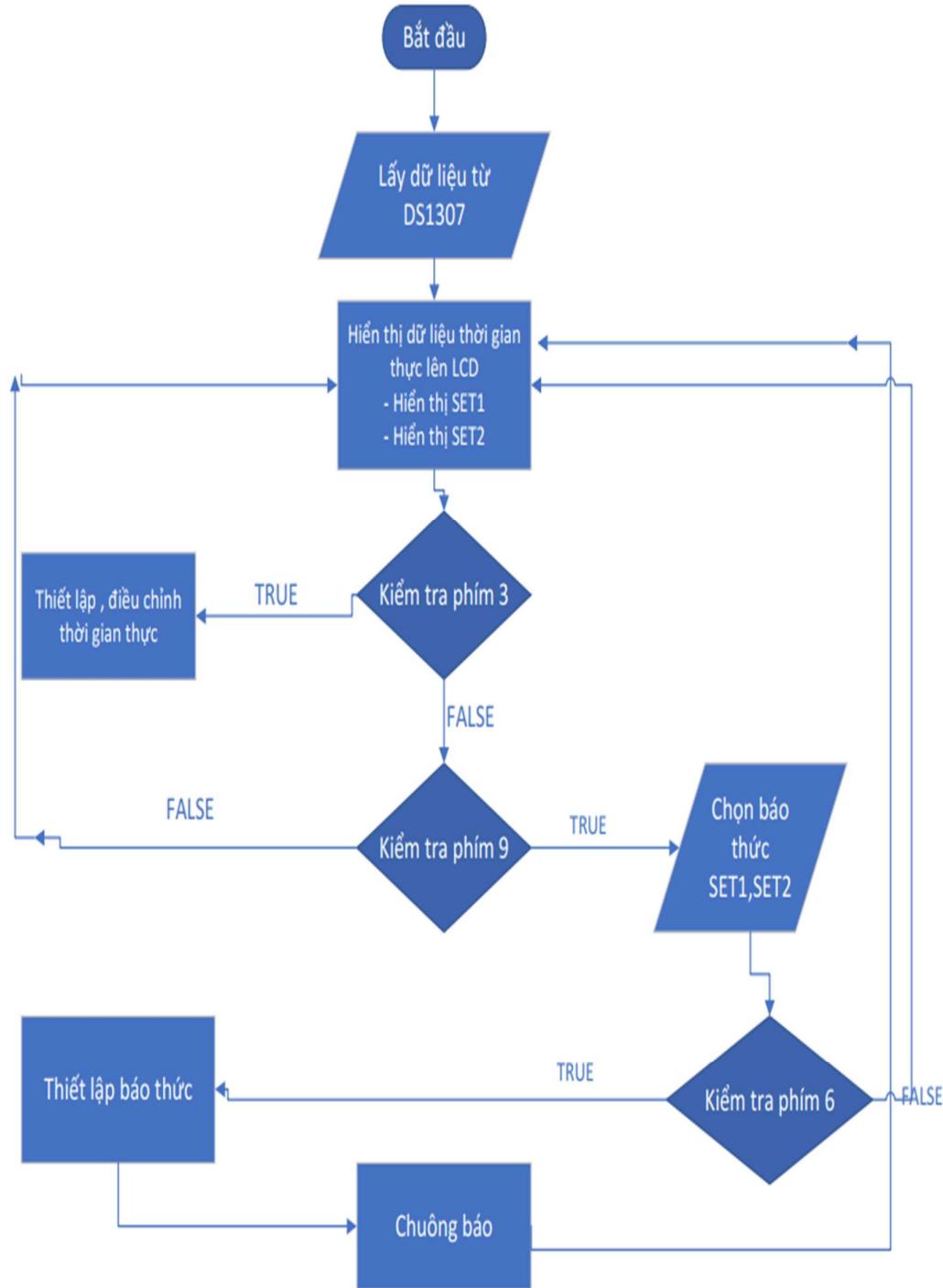
Hình 3.4 Sơ đồ kết nối Speaker với vi điều khiển

3.1.2. Sơ đồ mạch mô phỏng hệ thống (Proteus)



Hình 3.5 Sơ đồ mạch mô phỏng Proteus cho hệ thống

3.2. Lưu đồ giải thuật hệ thống



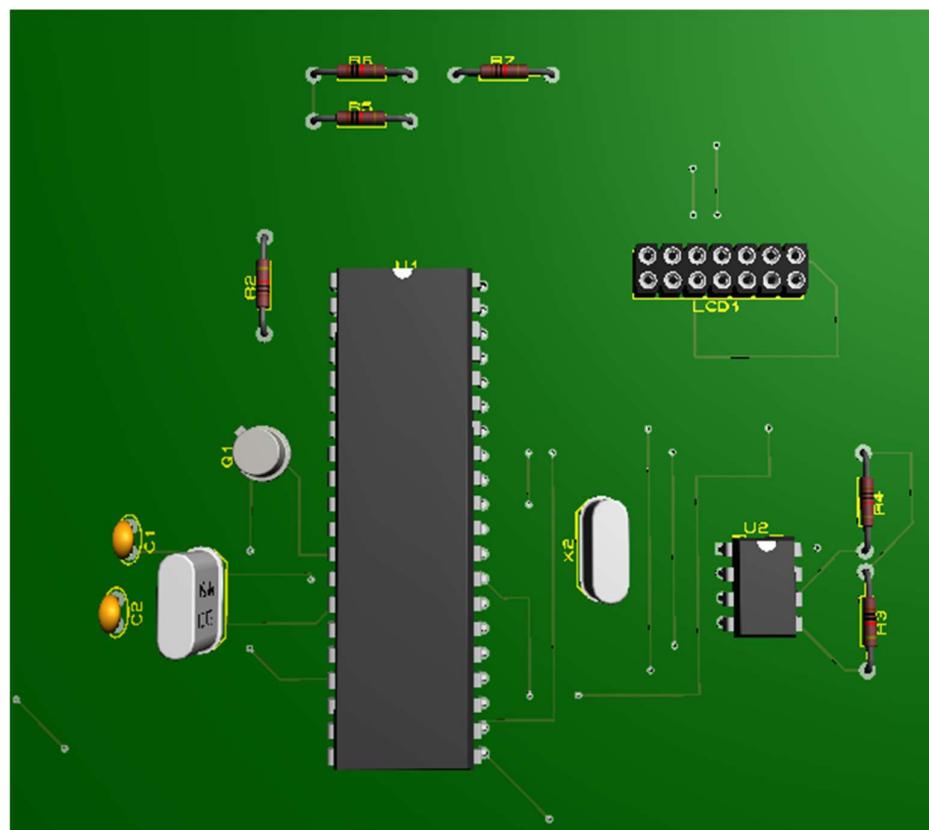
Hình 3.6 Lưu đồ giải thuật hệ thống

CHƯƠNG 4. THI CÔNG SẢN PHẨM

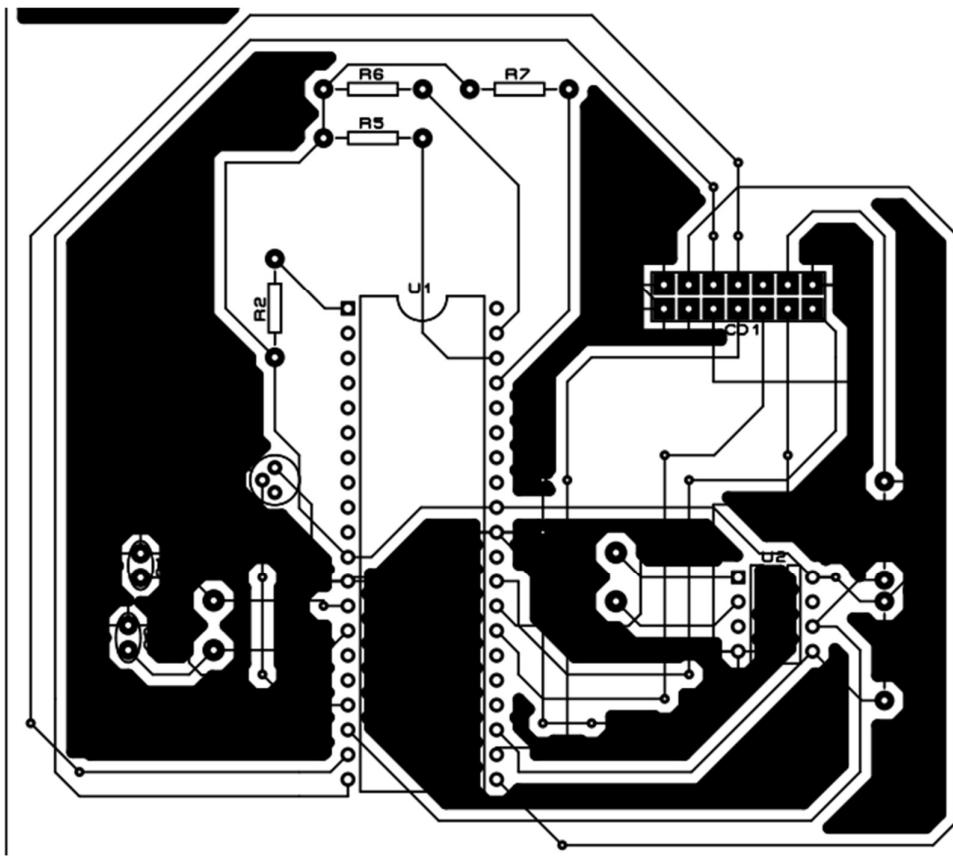
4.1. Giới thiệu

Quá trình thi công để được một hệ thống hoàn chỉnh cần làm theo các bước sau:

- ❖ Vẽ sơ đồ mạch in



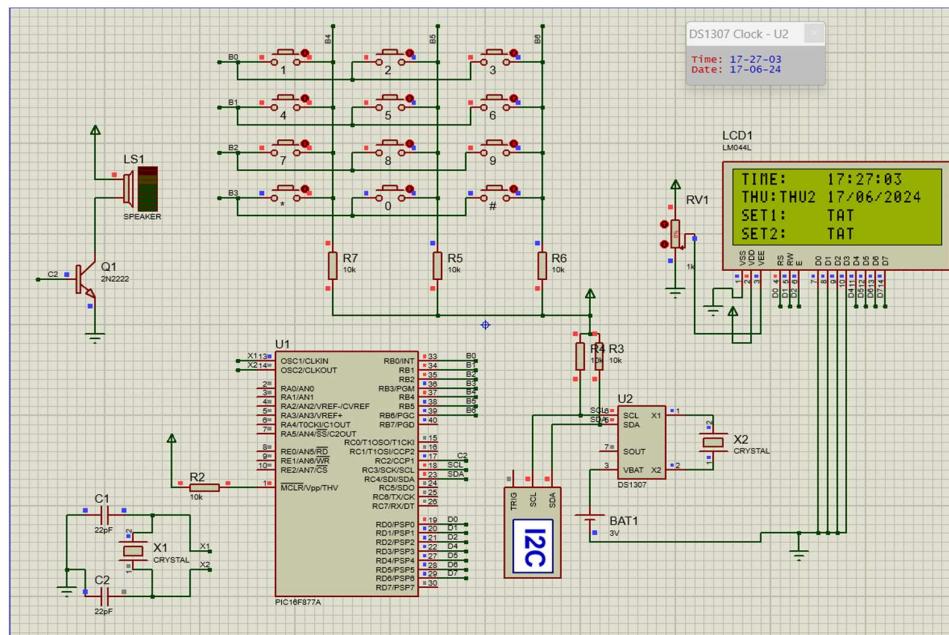
Hình 4.1 Mạch in 3D hệ thống trên Proteus



Hình 4.2 Mạch in PDF hệ thống trên proteus

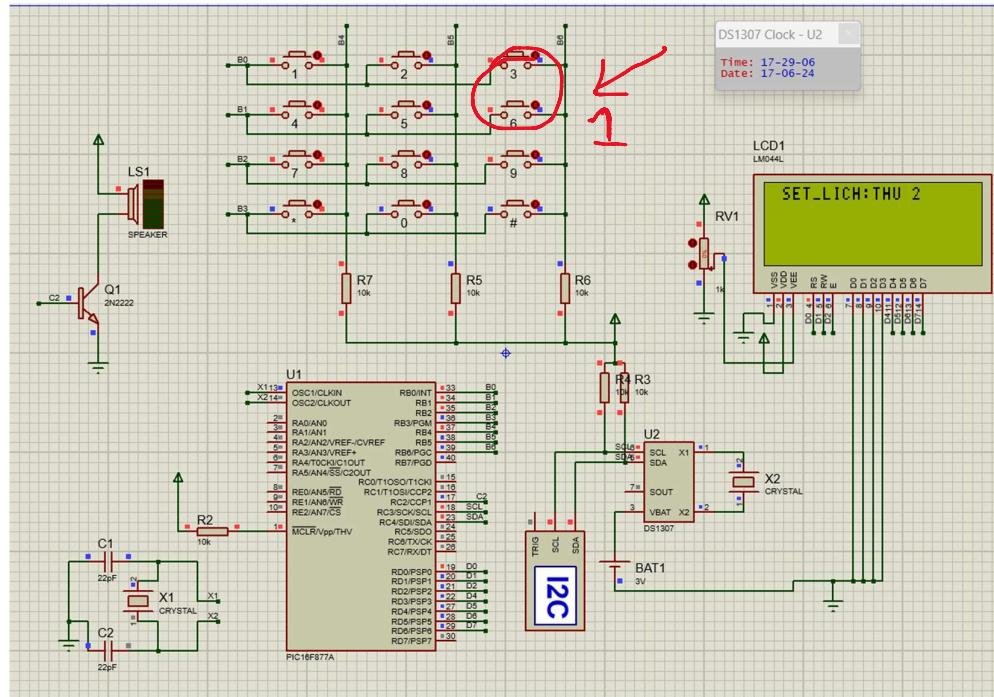
4.2. Quy trình sử dụng hệ thống trên mạch mô phỏng

❖ Bước 1: Màn hình khi khởi động chương trình



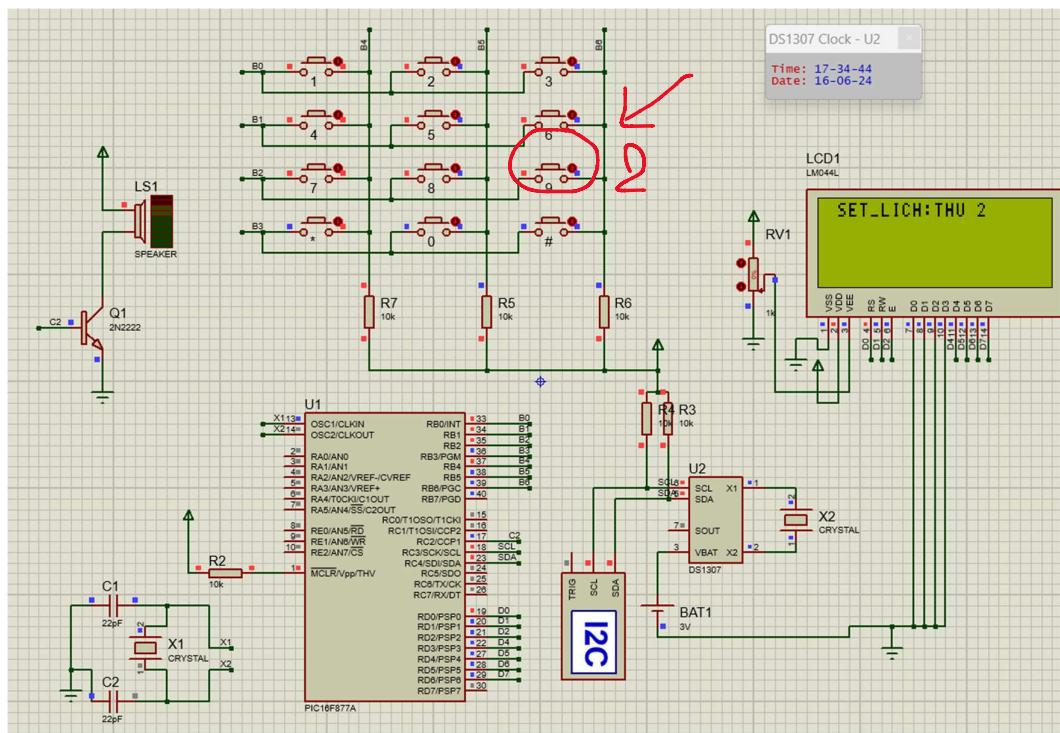
Hình 4.3 Quy trình sử dụng (bước 1)

- ❖ Bước 2:
- Đối với điều chỉnh thời gian thực (Bước 2.1.1-2.1.2)
- Bấm phím 3 để điều chỉnh thời gian, lịch thực.



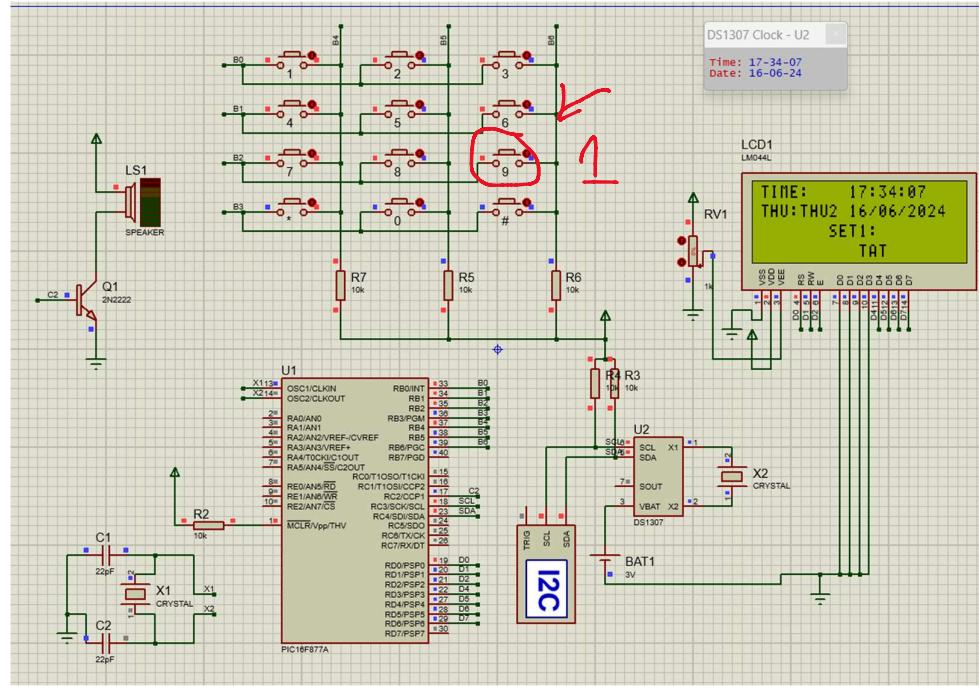
Hình 4.4 Quy trình sử dụng (bước 2.1.1)

- Bấm phím 6 để điều chỉnh tăng giảm thời gian, lịch.(nếu cần điều chỉnh) và ấn phím 3 để lưu cài đặt.



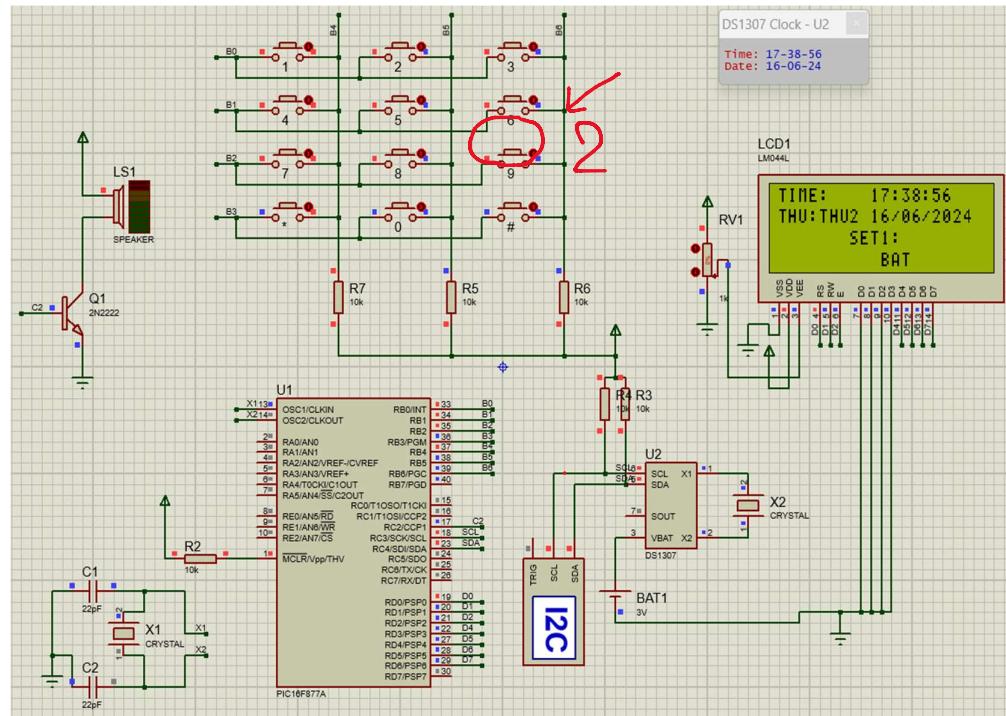
Hình 4.5 Quy trình sử dụng (bước 2.1.2)

- Đối với thiết lập báo thức (Bước 2.2.1-2.2.3)
 - Bấm phím 9 để lựa chọn báo thức cần hẹn giờ.

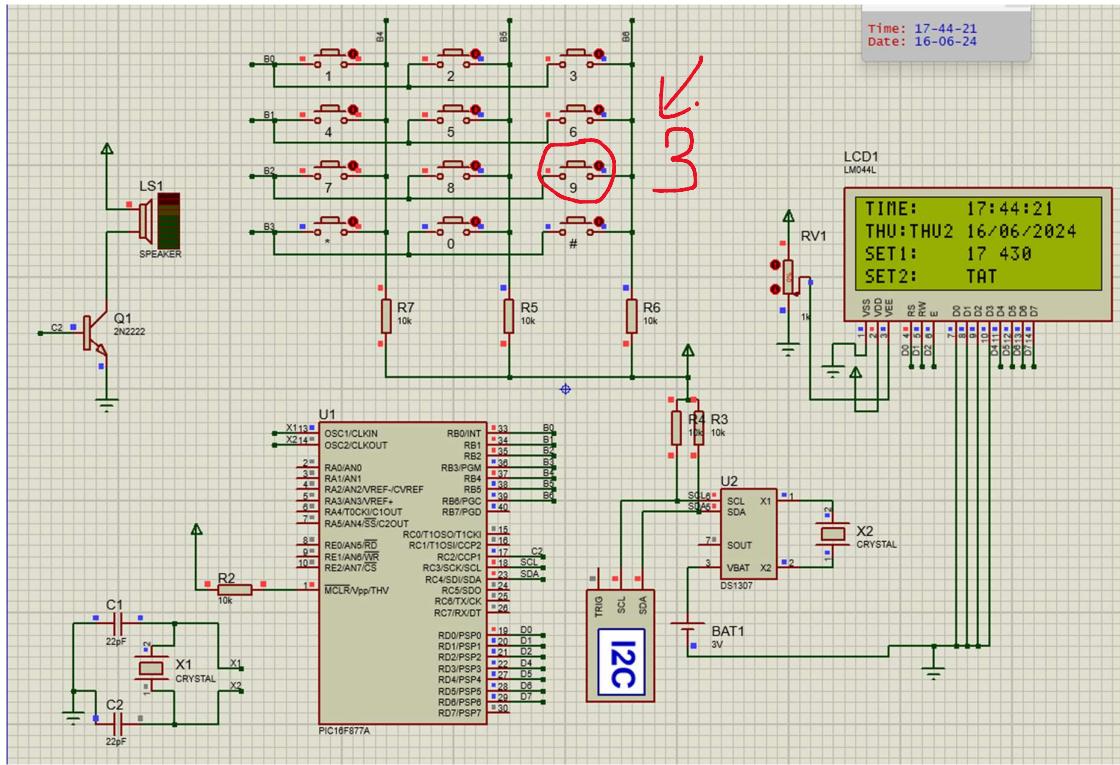


Hình 4.6 Quy trình sử dụng (bước 2.2.1)

- Bấm phím 6 để điều chỉnh tăng giảm thời gian. Sau khi điều chỉnh xong thì ấn phím 9 để xác nhận và về màn hình hiển thị sau khi thiết lập.

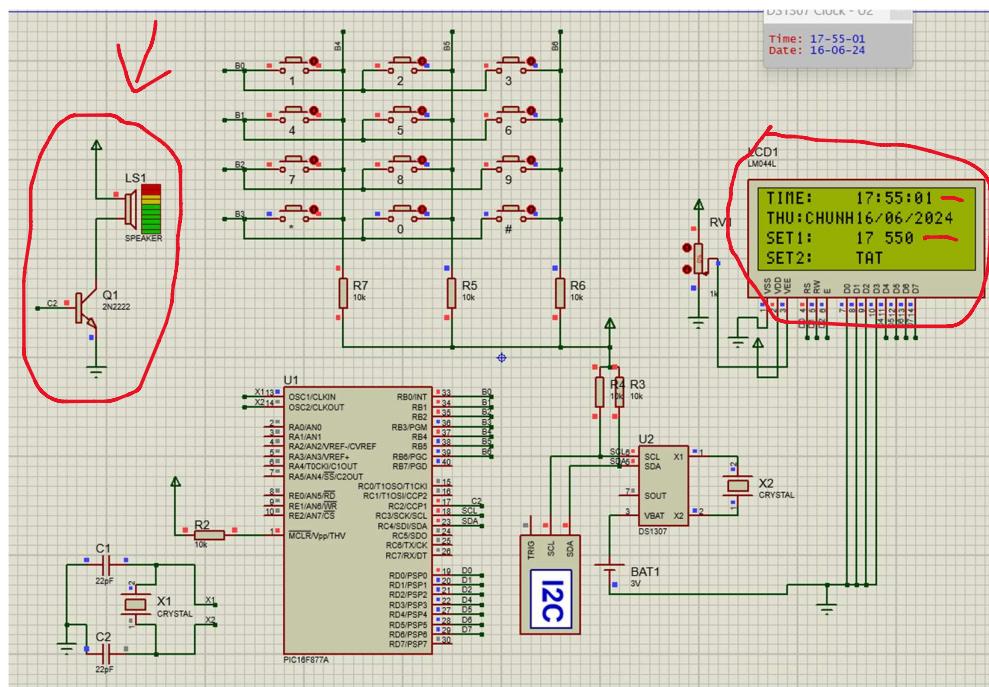


Hình 4.7 Quy trình sử dụng (bước 2.2.2)



Hình 4.8 Quy trình sử dụng (bước 2.2.3)

- ❖ Bước 3: Sau khi thiết lập báo thức xong , thời điểm thời gian thực trùng với thời gian thiết lập thì chuông sẽ reo.



Hình 4.9 Quy trình sử dụng (bước 3)

CHƯƠNG 5. KIỂM NGHIỆM, ĐÁNH GIÁ KẾT QUẢ

5.1. Kết quả đạt được

Sau khi tìm hiểu nghiên cứu các tài liệu chuyên ngành, tìm hiểu qua mạng Internet, cũng như được sự hướng dẫn của cô giáo Tăng Cẩm Nhung. Em cũng hoàn thành được bài tập lớn với đề tài “**Báo chuông theo lịch**”.

Sau đề tài này, em đã nghiên cứu và tích lũy được thêm nhiều hiểu biết, kiến thức mới như, tăng khả năng vận dụng lý thuyết vào thực tế, hiểu biết hơn các tính năng của PIC16F877A, DS1307, Keypad, Speaker...

5.2. Nhận xét và đánh giá

Sau thời gian nghiên cứu và thực hiện với đề tài “Báo chuông theo lịch” đã tương đối hoàn thiện, tuy chưa thể thực thi hơn bằng phần cứng chạy mạch thật nhưng hệ thống đã đáp ứng được những tính năng, nội dung và mục tiêu như sau:

- Giao tiếp và truyền dữ liệu giữa Vi điều khiển PIC16F877A với LCD thông qua chuẩn giao tiếp I2C
- Hiển thị được đồng hồ thời gian thực lên LCD
- Thiết lập và điều chỉnh được thời gian, báo thức và có chuông kêu .

➤ Các hạn chế:

- Độ chính xác của đồng hồ đôi khi chưa thể chạy chuẩn xác vì chương trình có nhiều vòng lặp chưa tối giản, linh kiện chưa tối giản nên đòi hỏi CPU để chạy mô phỏng khá cao, dẫn đến nhiều .
- Hiện đề tài đang sử dụng keypad được thiết kế bằng các nút bấm, có mong muốn thay thế bằng keypad-phone 4x3 .
- Các thao tác không linh hoạt, chưa có quản lý từ xa .

CHƯƠNG 6. TỔNG KẾT

Dự thảo đề tài trong tương lai , em sẽ cố gắng khắc phục những hạn chế như đã nêu trên. Cung cấp thêm chức năng cho hệ thống, phát triển thiết kế:

- ❖ Thực thi được mô hình phần cứng bằng mạch thật
- ❖ Thêm chức năng điều khiển qua APP hoặc điều khiển từ xa
- ❖ Cải thiện hệ thống tối ưu , gọn gàng và hiệu quả.

Thông qua việc thực hiện đề tài “ **Báo chuông theo lịch**”, em đã tính lũy rất nhiều kiến thức thực tế về bộ môn Hệ Thống Nhúng, cô giáo Tăng Cẩm Nhung đã tạo cho em niềm say mê học tập, tìm tòi kiến thức mới. Ngoài những kiến thức trên giảng đường, em cũng học thêm được những kinh nghiệm, kỹ năng thiết thực giúp ích cho chúng em thực hiện tốt hơn những nhiệm vụ, bài tập của mình như là kỹ năng mềm, kỹ năng làm việc nhóm.

Trong quá trình thực hiện đề tài này, dù rất cố gắng nhưng do vốn kiến thức hạn hẹp nên không thể tránh khỏi những sai sót. Em rất mong nhận được những đóng góp, phê bình, chia sẻ của thầy cô để các đề tài tiếp theo sẽ hoàn thiện hơn.

Em xin chân thành cảm ơn cô!

PHỤ LỤC

- ❖ Code chương trình viết trên phần mềm PIC-C Compiler

```
//LCD module connections
#define LCD_RS_PIN PIN_D0
#define LCD_RW_PIN PIN_D1
#define LCD_ENABLE_PIN PIN_D2
#define LCD_DATA4_PIN_D3
#define LCD_DATA5_PIN_D4
#define LCD_DATA6_PIN_D5
#define LCD_DATA7_PIN_D6
//Khai bao chan keypad
#define ROW1 PIN_B0
#define ROW2 PIN_B1
#define ROW3 PIN_B2
#define ROW4 PIN_B4
#define COL1 PIN_B5
#define COL2 PIN_B6
```

```

#define COL3 PIN_B7
//khai bao thu vien
#include <16F877A.h>
#fuses HS,NOWDT,NOPROTECT,NOLVP
#use delay(clock = 8000000)
#include <lcd.c>
#use fast_io(B)
#use I2C(master, I2C1, FAST = 100000)
#use pwm (PWM1, FREQUENCY = 2400Hz, DUTY = 50, PWM_OFF)
//khai bapo kieu du lieu
short alarm1_status, alarm2_status; //kieu nguyen
char time[] = "TIME: : : "; // kieu ky tu
char calendar[] = " / /20 ";
char alarm1[] = "SET1: : :00 ";
char alarm2[] = "SET2: : :00 ";
unsigned int8 second, second10, minute, minute10,
           hour, hour10, date, date10, month, month10,
           year, year10, day, alarm1_minute, alarm1_hour,
           alarm2_minute, alarm2_hour;
//các khai báo thời gian , ngày tháng của tung set1 set2
char keypad[4][3] = {
    {'1', '2', '3'},
    {'4', '5', '6'},
    {'7', '8', '9'},
    {'*', '0', '#'}
};
    char get_key() {
char key = '\0';

output_low(ROW1);
output_low(ROW2);
output_low(ROW3);
output_low(ROW4);

for (int row = 0; row < 4; row++) {
    output_low(ROW1);
    output_low(ROW2);
    output_low(ROW3);
    output_low(ROW4);

switch(row) {
    case 0: output_high(ROW1); break;
    case 1: output_high(ROW2); break;
    case 2: output_high(ROW3); break;
    case 3: output_high(ROW4); break;
}

```

```

        if (input(COL1)) {
            while(input(COL1));
            key = keypad[row][0];
            break;
        }
        if (input(COL2)) {
            while(input(COL2));
            key = keypad[row][1];
            break;
        }
        if (input(COL3)) {
            while(input(COL3));
            key = keypad[row][2];
            break;
        }
    }

    delay_ms(5);
    return key;
}
void ds1307_display(){ //ham hien thi dong ho thoi gian
    thuc
    second10 = (second & 0x70) >> 4;
    second = second & 0x0F;
    minute10 = (minute & 0x70) >> 4;
    minute = minute & 0x0F;
    hour10 = (hour & 0x30) >> 4;
    hour = hour & 0x0F;
    date10 = (date & 0x30) >> 4;
    date = date & 0x0F;
    month10 = (month & 0x10) >> 4;
    month = month & 0x0F;
    year10 = (year & 0xF0) >> 4;
    year = year & 0x0F;
    time[16] = second + 48;
    time[15] = second10 + 48;
    time[13] = minute + 48;
    time[12] = minute10 + 48;
    time[10] = hour + 48;
    time[9] = hour10 + 48;
    calendar[9] = year + 48;
    calendar[8] = year10 + 48;
    calendar[4] = month + 48;
    calendar[3] = month10 + 48;
    calendar[1] = date + 48;
}

```

```

calendar[0] = date10 + 48;
lcd_gotoxy(1, 1); // Go to column 1 row 1
printf(lcd_putc, time); // Display time
lcd_gotoxy(1, 2); // Go to column 1 row 2
switch(day){
    case 1: lcd_putc("THU:CHUNHAT "); break;
    case 2: lcd_putc("THU:THU2 "); break;
    case 3: lcd_putc("THU:THU3 "); break; //dat ra cac cach chon
    case 4: lcd_putc("THU:THU4 "); break;
    case 5: lcd_putc("THU:THU5 "); break;
    case 6: lcd_putc("THU:THU6 "); break;
    case 7: lcd_putc("THU:THU7 "); break;}
lcd_gotoxy(10, 2); // Go to column 10 row 2
printf(lcd_putc, calendar); // Display calendar
}
void alarm_display() // hien thi cac set1 set2
lcd_gotoxy(21, 1); // Go to column 1 row 3
if(alarm1_status){
    alarm1[13] = alarm1_minute % 10 + 48;
    alarm1[12] = alarm1_minute/10 + 48;
    alarm1[10] = alarm1_hour%10 + 48;
    alarm1[9] = alarm1_hour/10 + 48;
    printf(lcd_putc, alarm1);}
else lcd_putc("SET1: TAT ");
lcd_gotoxy(21, 2);
if(alarm2_status){
    alarm2[13] = alarm2_minute % 10 + 48;
    alarm2[12] = alarm2_minute/10 + 48;
    alarm2[10] = alarm2_hour%10 + 48;
    alarm2[9] = alarm2_hour/10 + 48;
    lcd_gotoxy(21, 2); // Go to column 1 row 4
    printf(lcd_putc, alarm2);}
else lcd_putc("SET2: TAT ");
}
void ds1307_write(unsigned int8 address, data_){ //viet thoi gian
i2c_start(); // Start I2C
i2c_write(0xD0); // DS1307 address
i2c_write(address); // Send register address
i2c_write(data_); // Write data to the selected register
i2c_stop(); // Stop I2C
}
void ds1307_read(){ //doc thoi gian
i2c_start(); // Start I2C
i2c_write(0xD0); // DS1307 address
i2c_write(0); // Send register address
i2c_start(); // Restart I2C
}

```

```

i2c_write(0xD1);           // Initialize data read
second = i2c_read(1);      // Read seconds from register 0
minute = i2c_read(1);      // Read minutes from register 1
hour = i2c_read(1);        // Read hour from register 2
day = i2c_read(1);         // Read day from register 3
date = i2c_read(1);        // Read date from register 4
month = i2c_read(1);       // Read month from register 5
year = i2c_read(0);        // Read year from register 6
i2c_stop();                // Stop I2C
}
void alarm_check(){          //kiem tra bao thuc thuan
if((alarm1_minute == ((minute & 0x0F) + (minute >> 4) * 10)) &&
(alarm1_hour == ((hour & 0x0F) + (hour >> 4) * 10)) && (second == 0) &&
alarm1_status)
{
    output_high(PIN_C2);
}
if((alarm2_minute == ((minute & 0x0F) + (minute >> 4) * 10)) &&
(alarm2_hour == ((hour & 0x0F) + (hour >> 4) * 10)) && (second == 0) &&
alarm2_status)
{
    OUTPUT_HIGH(PIN_C2);
    pwm_on();
}
}
void main(){
port_b_pullups(TRUE);      // Enable PORTB pull-ups
output_b(0);
set_tris_b(7);             // Configure RB0 & RB1 as inputs
lcd_init();                 // Initialize LCD module
lcd_putc('\f');
while(TRUE){
set_tris_d(0xF0);
char key = get_key();
if(input(PIN_B0) == 0){
lcd_putc('\f');
minute = minute + minute10 * 10;
hour = hour + hour10 * 10;
date = date + date10 * 10;
month = month + month10 * 10;
year = year + year10 * 10;
lcd_gotoxy(7, 1);           // Go to column 7 row 1
lcd_putc("GIO:");
lcd_gotoxy(11, 1);          // Go to column 10 row 2
lcd_putc(" ");
delay_ms(200);
}
}
}

```

```

while(TRUE){
    if(input(PIN_B1) == 0)
        hour++;
    if(hour > 23)
        hour = 0;
    lcd_gotoxy(11, 1);           // Go to column 10 row 2
    printf(lcd_putc,"%02u", hour);
    if(input(PIN_B0) == 0)
    {
        lcd_putc('f');
        break;
    }
    delay_ms(200);

    lcd_gotoxy(7, 1);           // Go to column 1 row 1
    lcd_putc("PHUT:");
    delay_ms(200);
    while(TRUE){
        if(input(PIN_B1) == 0)
            minute++;
        if(minute > 59)
            minute = 0;
        lcd_gotoxy(12, 1);           // Go to column 10 row 2
        printf(lcd_putc,"%02u", minute);
        if(input(PIN_B0) == 0){
            lcd_putc('f');
            break;
        }
        delay_ms(200);

        lcd_gotoxy(2, 1);           // Go to column 9 row 1
        lcd_putc("SET_LICH:");
        delay_ms(200);
        while(TRUE){
            if(input(PIN_B1) == 0)
                day++;
            if(day > 7)
                day = 1;
            lcd_gotoxy(11, 1);           // Go to column 6 row 2
            switch(day){
                case 1: lcd_putc("CHUNHAT"); break;
                case 2: lcd_putc("THU 2 "); break;
                case 3: lcd_putc("THU 3 "); break;
                case 4: lcd_putc("THU 4 "); break;
                case 5: lcd_putc("THU 5 "); break;
                case 6: lcd_putc("THU 6 "); break;
                case 7: lcd_putc("THU 7 "); break;
            }
            if(input(PIN_B0) == 0)

```

```

{
lcd_putc('f');
break;
delay_ms(200);
}
lcd_gotoxy(4, 1);           // Go to column 9 row 1
lcd_putc("NGAY:");
delay_ms(200);
while(TRUE){
if(input(PIN_B1) == 0)
date++;
if(date > 31)
date = 1;
lcd_gotoxy(9, 1);           // Go to column 10 row 2
printf(lcd_putc,"%02u", date);
if(input(PIN_B0) == 0){
lcd_putc('f');
break;
delay_ms(200);
}
lcd_gotoxy(8, 1);           // Go to column 8 row 1
lcd_putc("THANG:");
delay_ms(200);
while(TRUE){
if(input(PIN_B1) == 0)
month++;
if(month > 12)
month = 1;
lcd_gotoxy(10, 2);          // Go to column 10 row 2
printf(lcd_putc,"%02u", month);
if(input(PIN_B0) == 0)
break;
delay_ms(200);
}
lcd_gotoxy(8, 1);
lcd_putc(" NAM: ");
lcd_gotoxy(9, 2);
lcd_putc("20 ");
delay_ms(200);
while(TRUE){
if(input(PIN_B1) == 0)
year++;
if(year > 99)
year = 0;
lcd_gotoxy(11, 2);
printf(lcd_putc,"%02u", year);
}

```

```

if(input(PIN_B0) == 0){
    // Convert decimal to BCD
    minute = ((minute/10) << 4) + (minute % 10);
    hour = ((hour/10) << 4) + (hour % 10);
    date = ((date/10) << 4) + (date % 10);
    month = ((month/10) << 4) + (month % 10);
    year = ((year/10) << 4) + (year % 10);
    // End conversion
    ds1307_write(1, minute);
    ds1307_write(2, hour);
    ds1307_write(3, day);
    ds1307_write(4, date);
    ds1307_write(5, month);
    ds1307_write(6, year);
    ds1307_write(0, 0);
    delay_ms(200);
    break;
}
delay_ms(200);
}
}

if(input(PIN_B2) == 0){
    lcd_gotoxy(21, 1);
    lcd_putc("    SET1:    ");
    lcd_gotoxy(21, 2);
    lcd_putc("        ");
    delay_ms(200);
    while(TRUE){
        ds1307_read();
        ds1307_display();
        if(input(PIN_B1) == 0)
            alarm1_status++;
        if(alarm1_status){
            lcd_gotoxy(30, 2);
            lcd_putc(" BAT ");
        }
        else{
            lcd_gotoxy(30, 2);
            lcd_putc(" TAT ");
        }
        if(input(PIN_B2) == 0)
            break;
        delay_ms(200);
    }
    if(alarm1_status){
        lcd_gotoxy(21, 1);
        lcd_putc("    SET1 PHUT: ");
        lcd_gotoxy(21, 2);
        lcd_putc("        ");
    }
}

```

```

delay_ms(200);
while(TRUE){
    ds1307_read();
    ds1307_display();
    if(input(PIN_B1) == 0)
        alarm1_minute++;
    if(alarm1_minute > 59)
        alarm1_minute = 0;
    lcd_gotoxy(30, 2);
    printf(lcd_putc,"%02u", alarm1_minute);
    if(input(PIN_B2) == 0)
        break;
    delay_ms(200);
}
lcd_gotoxy(21, 1);
lcd_putc("  SET1 GIO: ");
lcd_gotoxy(21, 2);
lcd_putc("          ");
delay_ms(200);
while(TRUE){
    ds1307_read();
    ds1307_display();
    if(input(PIN_B1) == 0)
        alarm1_hour++;
    if(alarm1_hour > 23)
        alarm1_hour = 0;
    lcd_gotoxy(30, 2);
    printf(lcd_putc,"%02u", alarm1_hour);
    if(input(PIN_B2) == 0)
        break;
    delay_ms(200);
}
lcd_gotoxy(21, 1);
lcd_putc("  SET2: ");
lcd_gotoxy(21, 2);
lcd_putc("          ");
delay_ms(200);
while(TRUE){
    ds1307_read();
    ds1307_display();
    if(input(PIN_B1) == 0)
        alarm2_status++;
    if(alarm2_status){
        lcd_gotoxy(30, 2);
        lcd_putc(" BAT ");
    }
    else{

```

```

lcd_gotoxy(30, 2);
lcd_putc(" TAT ");
if(input(PIN_B2) == 0)
break;
delay_ms(200);
if(alarm2_status){
lcd_gotoxy(21, 1);
lcd_putc(" SET2 PHUT:");
lcd_gotoxy(21, 2);
lcd_putc(" ");
delay_ms(200);
while(TRUE){
ds1307_read();
ds1307_display();
if(input(PIN_B1) == 0)
alarm2_minute++;
if(alarm2_minute > 59)
alarm2_minute = 0;
lcd_gotoxy(30, 2);
printf(lcd_putc,"%02u", alarm2_minute);
if(input(PIN_B2) == 0)
break;
delay_ms(200);
}
lcd_gotoxy(21, 1);
lcd_putc(" SET2 GIO: ");
lcd_gotoxy(21, 2);
lcd_putc(" ");
delay_ms(200);
while(TRUE){
ds1307_read();
ds1307_display();
if(input(PIN_B1) == 0)
alarm2_hour++;
if(alarm2_hour > 23)
alarm2_hour = 0;
lcd_gotoxy(30, 2);
printf(lcd_putc,"%02u", alarm2_hour);
if(input(PIN_B2) == 0)
break;
delay_ms(200);
}
delay_ms(200);
}
if(input(PIN_B1) == 0){
output_low(PIN_B3);

```

```

        pwm_off();
    }
    ds1307_read();           // Read data from DS1307 RTCC
    alarm_check();           // Check if there is an alarm
    ds1307_display();        // Diaplay time and calendar
    alarm_display();          // Display alarms
    delay_ms(50);
}
}

```

- ❖ Mã QR và link GitHub của đề tài.
- <https://github.com/Thuhien2004/HETHONGNHUNG.git>



TÀI LIỆU THAM KHẢO

- <https://luanvan.co/luan-van/thiet-ke-mach-chuong-bao-tiet-hoc-tu-dong-cho-cac-truong-hoc-44885/>
- <https://nshopvn.com/product/module-thoi-gian-thuc-rtc-ds1307/>
- <https://dientutuonglai.com/chuan-giao-tiep-i2c-la-gi.html>
- Bài giảng Hệ Thống Nhúng – Bộ Môn Kỹ Thuật Máy Tính – ĐH Kỹ
Thuật Công Nghiệp Thái Nguyên.
- Giáo trình tổng hợp kiến thức PIC cộng đồng PICVIET

