BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC NHA TRANG

KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO THỰC TẬP CƠ SỞ

THIẾT KẾ ĐỒNG HỒ THỜI GIAN THỰC SỬ DỤNG KIT ARDUINO VÀ MODULE DS1307

Giảng viên hướng dẫn: ThS. Đoàn Vũ Thịnh

Sinh viên thực hiện: **Nguyễn Duy Tín - Nguyễn Võ Tiền**

Mã số sinh viên: **61131272 - 60137144**

KHÁNH HÒA-2021

TRƯỜNG ĐẠI HỌC NHA TRANG

KHOA CÔNG NGHỆ THÔNG TIN

BỘ MÔN HỆ THỐNG THÔNG TIN

BÁO CÁO THỰC TẬP CƠ SỞ

THIẾT KẾ ĐỒNG HỒ THỜI GIAN THỰC SỬ DỤNG KIT ARDUINO VÀ MODULE DS1307

Giảng viên hướng dẫn: ThS. Đoàn Vũ Thịnh

Sinh viên thực hiện: Nguyễn Duy Tín - Nguyễn Võ Tiền

Mã số sinh viên: 61131272- 60137144

Khánh Hòa, tháng 01/2021

TRƯỜNG ĐẠI HỌC NHA TRANG
Khoa: Công nghệ Thông tin

PHIẾU THEO DÕI TIẾN ĐỘ VÀ ĐÁNH GIÁ BÁO CÁO THỰC TẬP CƠ SỞ

Tên đề tài: Thiết kế đồng hồ thời gian thực sử dụng Kit Arduino và module DS1307

Giảng viên hướng dẫn: ThS. Đoàn Vũ Thịnh

Sinh viên được hướng dẫn: Nguyễn Duy Tín **MSSV:** 61131272

Nguyễn Võ Tiền 60137144

Khóa: 60 **Ngành:** Công nghệ Thông tin

Lần	Ngày	Nội dung	Nhận xét của GVHD
1	7/12/2020	Nhận đề tài hướng dẫn và định hướng giải quyết vấn đề. Sau đó, tiến hành cài đặt phần mềm Proteus và tiến hành mô phỏng chức năng của các linh kiện, mô đun chức năng.	Sinh viên và GVHD trao đổi nội dung của đề tài. Sinh viên có nhiều phương án để triển khai, tuy nhiên cần kiểm thử các phương pháp trước khi đưa ra phương án cuối cùng thông qua phần mềm mô phỏng và linh kiện điện tử.
2	14/12/2020	Sinh viên trình bày ý tưởng và giải pháp của mình. Sinh viên mô phỏng phần mềm trên Proteus thể hiện chức năng hiện giờ, cài đặt báo thức và âm báo	Sinh viên có ý tưởng rõ ràng về nội dung thực hiện. Phần mô phỏng có kết quả tốt thể hiện được các yêu cầu đặt ra, tuy nhiên về mặt thực tế sản phẩm có thể gặp lỗi, đề nghị tiến hành kiêm thử trên mạch thực.
3	21/12/2020	Sinh viên trình bày kết quả mô phỏng và hoàn tất quá trình này. Về mặt phần cứng Sinh viên cần chuẩn bị thêm kiến thức cần thiết về điện tử, điện tử số.	Kết thúc quá trình mô phỏng sinh viên đã đáp ứng đầy đủ các yêu cầu đặt ra. Về phần cứng Sinh viên lắp ghép các mô đun Arduino, LCD 20x4, RTC DS1307 cho kết quả đạt yêu cầu.
4	4/01/2021	Sinh viên nộp bản thảo của báo cáo thực tập lần thứ 1 và tiến hành chỉnh sửa.	Báo cáo chỉ trình bày chung chung chưa đi vào cụ thể phân tích các yêu cầu của bài toán, hình ảnh, bảng biểu chưa trình bày rõ ràng. Cần

			hiệu chỉnh theo yêu cầu của GVHD.
5	11/01/2021	Sinh viên nộp bản thảo lần 2 và demo sản phẩm với mạch thực tế	Báo cáo lần này đã khắc phục được các lỗi của lần trước, tuy nhiên phần phương pháp và kết quả chưa nổi bật, chưa có sự liên kết giữa các phần. Phần cứng chưa hoàn thiện và còn nhiều sai sót cần khắc phục
6	18/1/2021	Sinh viên nộp báo cáo lần cuối sau khi đã hoàn thiện phần cứng.	Báo cáo lần này đã khắc phục được các sai sót lần trước. Về mạch điện đã được hoàn thành các yêu cầu trước đó. Sản phẩm đạt yêu cầu

Nhận xét chung (sau khi sinh viên hoàn thành ĐA/KL):

Sinh viên thực hiện tốt các yêu cầu của GVHD, trong quá trình thực hiện đề tài có sự liên hệ chặt chẽ với GV. Ngoài ra, cũng cần ghi nhận sự nỗ lực hết mình của các bạn, với lượng kiến thức còn hạn chế về điện tử và điện tử số nhưng cơ bản đề tài đã hoàn thành khối lượng công việc đã đặt ra trước đó.

Về nội dung báo cáo đã thỏa mãn các yêu cầu của đề tài như trong đề cương. Về phần mạch thực và kết quả thực hiện, đề tài có kết quả mô phỏng tốt và cũng thể hiện được kết quả đó trên sản phẩm thực tế.

Về hình thức của báo cáo và sản phẩm, báo cáo trình bày rõ ràng các mục tiêu, phương pháp, kết quả và thảo luận cho sản phẩm. Còn về sản phẩm chưa thực hiện việc ghép các mô đun vào mạch điện tổng thể (chưa thiết kế được mạch in) cho sản phẩm hoàn thiện.

Điểm hình thức: 9/10 Điểm nội dung: 9/10 **Điểm tổng kết: 9/10**

Đồng ý cho sinh viên: Được bảo vệ: ☑ Không được bảo vệ: ☐

Khánh Hòa, ngày 20 tháng 01 năm 2021

Cán bộ hướng dẫn

(Ký và ghi rõ họ tên)

MỤC LỤC

DANH MỤC HÌNH

LỜI NÓI ĐẦU

Đồng hồ kỹ thuật số hiện đang kinh doanh trên thị trường có thể theo dõi được ngày, tháng, năm dương lịch, âm lịch và giờ giấc giúp chúng ta sắp xếp được thời gian biểu của mình một cách hợp lý. Để có thể tạo ra một đồng hồ điện tử cần vận dụng các kiến thức về lập trình thiết bị nhúng, cụ thể là Kit Arduino, kiến thức về điện tử số và điện tử tương tự để tính toán các thành phần điện tử cơ bản như LCD, mô đun thời gian thực DS1307.

Đề tài thực hiện trình tự các bước từ thiết kế sơ đồ nguyên lý, mô phỏng bằng phần mềm Proteus 8.9, lập trình điều khiển thông qua Arduino IDE và tiến hành lắp ghép các thành phần trên board test. Về cơ bản sản phẩm thực hiện được các yêu cầu sau: Hiển thị thời gian hệ thống, thiết lập thời gian báo thức và đi kèm với âm thanh phát ra khi báo thức được xác lập.

Tuy đã có nhiều cố gắng nhưng đề tài vẫn chưa thể hoàn thiện như mong đợi. Đó là thiết kế mạch in để ghép nối các thành phần cũng như đóng gói được sản phẩm. Do đó, trong tương lai nhóm sẽ hoàn thiện sản phẩm của mình tốt hơn với hình dạng của sản phẩm đẹp như sản phẩm thương mại và còn có thể tạo ra nhiều âm thanh báo thức.

Toàn bộ mã nguồn của chương trình được tải lên theo địa chỉ:

https://github.com/thinhdoanvu/ThuctapCoSo2020/tree/main/NguyenDuyTin

CHƯƠNG 1: TỔNG QUAN

Đồng hồ kỹ thuật số hiện đang kinh doanh trên thị trường có thể theo dõi được ngày, tháng, năm dương lịch, âm lịch và giờ giấc giúp chúng ta sắp xếp được thời gian biểu của mình một cách hợp lý. Hình 1.1 là một ví dụ như thế. Thành phần cơ bản bao gồm bộ hiển thị LCD cho biết thời gian, chip thời gian thực tự động cập nhật thời gian theo chu kỳ, các nút nhấn để thiết lập thời gian cũng như báo thức. Đồng hồ có thể hoạt động khi không có nguồn cung cấp trong thời gian dài. Khi được cấp nguồn trở lại, đồng hồ sẽ tiếp tục hiển thị thời gian hiện hành mà không cần phải lập trình hay cấu hình lần nữa. Để có thể tạo ra một đồng hồ điện tử như hình 1.1 cần vận dụng các kiến thức về lập trình thiết bị nhúng, cụ thể là Kit Arduino, kiến thức về điện tử số và điện tử tương tự để tính toán các thành phần điện tử cơ bản và thực hiện các thao tác ghép nối các thành phần điện tử với nhau. Ngoài ra, một thành phần không thể thiếu chính là module thời gian thực DS1307được dùng để đọc thời gian hiện hành và hiển thị lên màn hình LCD.

Hình 1.1. Đồng hồ kỹ thuật số

(Nguồn : https://www.officeworks.com.au/shop/officeworks/p/phillips-aj3400-lcd-alarm-clock-fm-black-pmaj3400)

1.1. LCD20x4

Hình 1.2. LCD 20x4

(Nguồn: https://www.reichelt.com/de/en/developer-boards-display-20-x-4-characters-blue-debo-lcd-20x4-bl-p192144.html)

LCD (Liquid Crystal Display) được sử dụng trong rất nhiều các ứng dụng của vi điều khiển. LCD có rất nhiều ưu điểm so với các dạng hiển thị khác. Nó có khả năng hiển thị kí tự đa dạng, trực quan (chữ, số và kí tự đồ họa), dễ dàng đưa vào mạch ứng dụng theo nhiều giao thức giao tiếp khác nhau, tốn rất ít tài nguyên hệ thống và giá thành rẻ.

Thông số kỹ thuật:

- Điện áp hoạt động: 5V
- Hiển thị tối đa 20 tự trên 4 dòng
- Chữ trắng nền xanh dương
- Kích thước: 98 x 60 x 13.5 mm
- Khoảng cách giữa hai chân là 0.1 inch phù hợp để kết nối với Breadboard.

Hình 1.3. Các chân của LCD 20x4

• Chân (1): (Vss) Chân nối đất cho LCD, khi thiết kế mạch ta nối chân này với GND của mạch điều khiển

• Chân (2): VDD Là chân cấp nguồn cho LCD, khi thiết kế mạch ta nối chân này với VCC = 5V của mạch điều khiển

• Chân (3): V0 là chân điều chỉnh độ tương phản của LCD.

• Chân (4): RS Là chân chọn thanh ghi (Register select). Nối chân RS với logic “0” (GND) hoặc logic “1” (VCC) để chọn thanh ghi.

- Logic “0”: LCD hoạt động ở chế độ “ghi” - write dữ liệu hoặc nối với bộ đệm địa chỉ của LCD ở chế độ “đọc” - read
- Logic “1”: Truy xuất đến mã lệnh tương ứng của LCD

• Chân (5): R/W là chân chọn chế độ đọc/ghi (Read/Write). Nối chân R/W với logic “0” để LCD hoạt động ở chế độ ghi, hoặc nối với logic “1” ở chế độ đọc.

• Chân (6): E Là chân cho phép (Enable). Các lệnh/dữ liệu chỉ được chấp nhận khi có 1 xung cho phép của chân E (mức tín hiệu chuyển trạng thái từ 1/0 xuống/lên 0/1).

- Ở chế độ ghi: Dữ liệu ở bus sẽ được LCD chuyển vào thanh ghi bên trong khi phát hiện một xung (sườn âm) trên tín hiệu chân E.

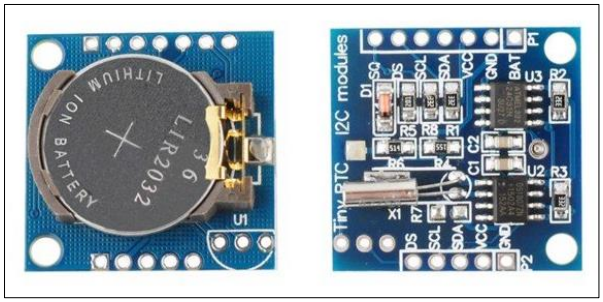
➢ Ở chế độ đọc: Dữ liệu sẽ được LCD xuất ra DB0-DB7 khi phát hiện cạnh lên (sườn dương) ở chân E và được LCD lưu trữ ở bus đến khi nào chân E xuống mức thấp.

• Chân (7 - 14): D0 - D7 - 8 đường của bus dữ liệu dùng để trao đổi thông tin với vi điều khiển. Có 2 chế độ sử dụng 8 đường bus này: (1) Chế độ 8 bit: Dữ liệu được truyền trên cả 8 đường, với bit MSB là bit DB7. (2) Chế độ 4 bit: Dữ liệu được truyền trên 4 đường từ DB4 tới DB7, bit MSB là DB7

• Chân (15): Nguồn dương cho đèn nền

• Chân (16): GND cho đèn nền

1.2. Mô đun thời gian thực IC DS1307



Hình 1.4. Đồng hồ thời gian thực IC DS1307

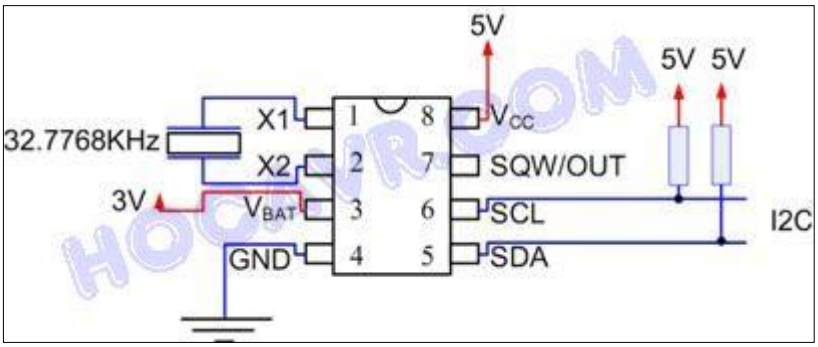
(Nguồn https://hshop.vn/products/mach-thoi-gian-thuc-rtc-ds1307)

Đồng hồ thời gian thực DS1307 (RTC) là đồng hồ được mã hóa BCD ở đầu ra, với 56 byte NVSRAM. Địa chỉ và dữ liệu là được chuyển nối tiếp qua chuẩn I2C. Đồng hồ xuất tín hiệu giây, phút, giờ, thông tin ngày, tháng, và năm lên trên bus dữ liệu. Trong đó, số tháng có ngày 31 hay 30 hay năm nhuận được tự động tính toán mà không cần phải lập trình. Đồng hồ có thể hiển thị ở 2 chế độ 24 hoặc 12 giờ thông báo AM/PM. DS1307 có một mạch cảm biến nguồn tích hợp phát hiện sự cố mất điện và tự động chuyển sang nguồn dự phòng.

Thông số kỹ thuật:

- IC chính: RTC DS1307 + EEPROM AT24C32 (4KB data)
- Nguồn cung cấp: 5VDC.
- Giao tiếp: I2C
- Lưu trữ và cung cấp các thông tin thời gian thực: ngày, tháng, năm, giờ, phút, giây...
- Có nguồn dự phòng để duy trì thời gian trong trường hợp mất nguồn.

- Có ngõ ra tần số 1Hz.
- Kích thước: 27 x 28 x 8.4mm



Hình 1.5. Sơ đồ chân IC DS1307

(Nguồn: http://www.hocavr.com/2018/06/ong-ho-thoi-gian-thuc-ds1307.html)

• Chân (1,2) X1-X2: Là 2 ngõ kết nối với bộ dao động thạch anh với tần số 32.768KHz làm nguồn tạo dao động cho chip. Mạch dao động bên trong cần được nối với tụ có dung lượng 12,5pF. X1 là đầu vào cho bộ dao động và có thể được kết nối tùy chọn với tần số 32,768kHz bên ngoài hoặc sử dụng bộ dao động bên trong. Chân X2 được để hở nếu bộ dao động bên ngoài là kết nối với X1.

• Chân (3) VBAT: nguồn 3V, điốt mắc nối tiếp giữa GND và chân VBAT để tránh trường hợp mắc ngược cực điện áp. Pin lithium (nguồn nội) có dung lượng 48mAh sẽ sao lưu thời gian của DS1307 trong hơn 10 năm khi không có điện ở nhiệt độ 25°C

• Chân (4) GROUND: chân đất chung cho cả nguồn 3V và Vcc

• Chân (5) SDA: Đầu vào/đầu ra dữ liệu nối tiếp của chuẩn I2C. Các Chân SDA là các cực máng hở nên cần có điện trở treo bên ngoài. Điện áp rơi trên điện trở kéo có thể là lên đến 5.5V bất kể điện áp trên VCC.

• Chân (6) SCL: SCL là đầu vào xung nhịp cho chuẩn giao tiếp I2C và được sử dụng để đồng bộ hóa chuyển động dữ liệu trên giao diện nối tiếp. SCL cũng cần có điện trở treo giống như SDA.

• Chân (7) SQW/OUT: bộ tạo xung vuông khi bit SQWE được đặt thành 1, chân SQW/OUT xuất ra một trong bốn tần số sóng vuông (1Hz, 4kHz, 8kHz, 32kHz). SQW/OUT cũng giống như SDA và SCL khi cần tới điện trở treo. SQW/OUT hoạt động với VCC hoặc VBAT. Khi không sử dụng chân này thường để hở.

• Chân (8) VCC: nguồn cấp, thường là 5V.

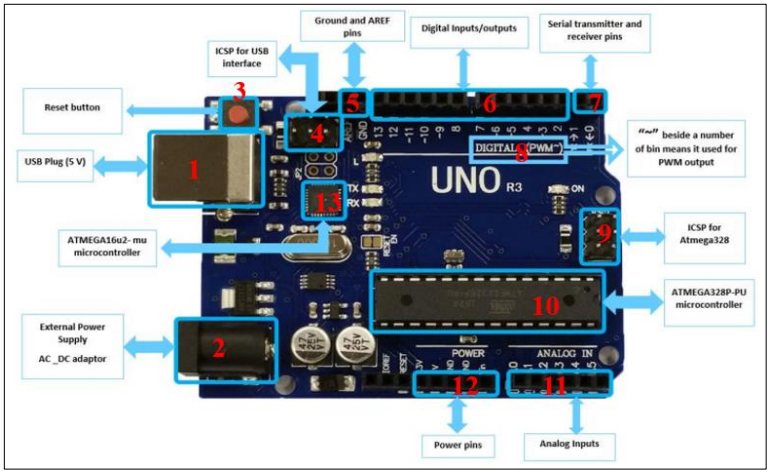
1.3. Arduino UNO R3

Phần cứng bao gồm một board mạch nguồn mở được thiết kế trên nền tảng vi xử lý AVR Atmel 8bit, hoặc ARM Atmel 32-bit. Những Model hiện tại được trang bị gồm 1 cổng giao tiếp USB, 6 chân đầu vào analog, 14 chân I/O kỹ thuật số tương thích với nhiều board mở rộng khác nhau. Đi kèm board mạch là môi trường phát triển tích hợp (IDE) chạy trên các máy tính cá nhân và cho phép người dùng viết các chương trình cho Aduino bằng ngôn ngữ C hoặc C++.

Phần cứng Arduino gốc được sản xuất bởi công ty Italy tên là Smart Projects. Một vài board dẫn xuất từ Arduino cũng được thiết kế bởi công ty của Mỹ tên là SparkFun Electronics. Một số phiên bản Arduino phổ biến như: Arduino Diecimila, Arduino Duemilanove, Arduino UNO, Arduino Leonardo, Arduino Mega, Arduino MEGA 2560 R3, Arduino Nano, Arduino Due, LilyPad Arduino

Đặc tính kỹ thuật:

- Vi điều khiển ATMEGA328P-PU
- Nguồn Cấp: 7-12V
- Dòng Max chân 5V : 500mA
- Dòng Max 3.3V: 50mA
- Dòng Max Chân I/O: 30mA
- 14 Chân Digital I/O (6 chân PWM)
- 6 Chân Analog Inputs
- 32k Flash Memory
- 16Mhz Clock Speed
- SRAM 2 KB
- EEPROM 1 KB



Hình 1.6. Sơ đồ các chân Arduino UNO R3

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

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

(2) Jack Power: Nguồn cấp cho board mạch Arduino. Có 2 loại nguồn có thể sử dụng được là nguồn xoay chiều (AC) tối đa 6V và nguồn một chiều (DC) tối đa 5V.

(3) RESET: Đặt lại trạng thái ngay khi nạp chương trình

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

(5) AREF và GND: GND hay chân Mass hay chân đất dùng để cấp nguồn 0V cho các modue khác có kết nối với Arduino. Trong khi chân AREF được dùng để phối hợp với các chân Analog Input (11) để điều chỉnh dải điện áp đầu vào cho ADC 10 bit.

(6) DIGITAL Input/Output: 12 chân tín hiệu số được dùng làm đầu vào hoặc đầu ra tại mỗi thời điểm.

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

(8) PWM (Pulse Width Modulation): Chân băm xung có ký hiệu là ~ dùng để tạo ra chuỗi xung vuông trong điều khiển tốc độ động cơ hay hiển thị LED.

(9) ICSP: Giao tiếp với Atmega328.

(10) IC lập trình ATMEGA16u2

(11) ANALOG Input: Tín hiệu chuyển đổi ADC được đưa vào các chân này. Trong trường hợp khác các chân này cũng có thể sử dụng để làm đầu vào/ra tín hiệu số. Đặc biệt, Arduino UNO có 2 chân **A4 (SDA)** và **A5 (SCL)** hỗ trợ giao tiếp **I2C/TWI** với các thiết bị khác. I2C là viết tắt của "Inter-Integrated Circuit", một chuẩn giao tiếp được phát minh bởi Philips' semiconductor division (giờ là NXP) nhằm đơn giản hóa việc trao đổi dữ liệu giữa các ICs. Đôi khi nó cũng được gọi là Two Wire Interface (TWI) vì chỉ sử dụng 2 kết nối để truyền tải dữ liệu, 2 kết nối của giao tiếp I2C gồm: SDA (Serial Data Line) và SCL (Serial Clock Line).

(12) POWER: Cung cấp nguồn 5V, GND cho các thành phần mở rộng

1.4. Arduino IDE version 1.8.13

Môi trường phát triển tích hợp (IDE) của Arduino là một ứng dụng cross-platform (đa nền tảng) được viết bằng Java, và từ IDE này sẽ được sử dụng cho ngôn ngữ lập trình xử lý (Processing programming language) và project Wiring. Một chương trình hoặc code viết cho Arduino được gọi là một sketch. Các chương trình Arduino được viết bằng C hoặc C++. Arduino IDE đi kèm với một thư viện phần mềm được gọi là "Wiring", từ project Wiring gốc, có thể giúp các thao tác input/output

được dễ dàng hơn. Người dùng chỉ cần định nghĩa 2 hàm để tạo ra một chương trình vòng thực thi (cyclic executive) có thể chạy được:

- setup(): thực thi mỗi khi khởi động một chương trình
- loop(): hàm này được gọi lặp lại cho đến khi tắt nguồn board mạch

Chương trình Arduino có thể được chia làm 3 phần: cấu trúc (structure), biến số (variable) và hằng số (constant), hàm và thủ tục (function). Một số hàm được liệt kê trong bảng ở **phụ lục**

Hình 1.7. Giao diện làm việc Arduino IDE version 1.8.13

Nút kiểm tra chương trình: Dùng để kiểm tra xem chương trình được viết có lỗi không. Nếu chương trình bị lỗi thì phần mềm Arduino IDE sẽ hiển thị thông tin lỗi ở vùng thông báo thông tin.

Nút nạp chương trình xuống bo Arduino: Dùng để nạp chương trình được viết xuống mạch Arduino. Trong quá trình nạp, chương trình sẽ được kiểm tra lỗi trước sau đó mới thực hiện nạp xuống mạch Arduino.

Hiển thị màn hình giao tiếp với máy tính: Khi nhấp vào biểu tượng cái kính lúp thì phần giao tiếp với máy tính sẽ được mở ra. Phần này sẽ hiển thị các thông số mà người dùng muốn đưa lên màn hình. Muốn đưa lên màn hình phải có lệnh Serial.print() mới có thể đưa thông số cần hiển thị lên màn hình

Vùng lập trình: Vùng này để người lập trình thực hiện việc lập trình cho chương trình của mình.

Vùng thông báo thông tin: Có chức năng thông báo các thông tin lỗi của chương trình hoặc các vấn đề liên quan đến chương trình được lập.

Có vài menu trong phần mềm IDE, tuy nhiên thông dụng nhất vẫn là menu File, ngoài những tính năng như mở một file mới hay lưu một file, phần menu này có một mục đáng chú ý là Example. Phần Example (ví dụ) đưa ra các ví dụ sẵn để người lập trình có thể tham khảo, giảm bớt thời gian lập trình. Hình bên dưới thể hiện việc chọn một ví dụ cho led chớp tắt (blink) để nạp cho mạch Arduino. Ví dụ về led chớp tắt này thường được dùng để kiểm tra bo khi mới mua về

Khi mới kết nối board Arduino với máy tính ta click vào **Tools->board** để chọn loại board sử dụng.

Khi lần đầu gắn mạch Arduino vào máy tính, người sử dụng cần nhấn chọn cổng COM bằng cách vào **Tools → Serial Port** (một số phiên bản dùng từ Port) sau đó nhấn chọn cổng COM, ví dụ như COM5. Những lần sau khi đưa chính board Arduino đó vào máy tính thì không cần chọn cổng COM, nếu đưa board Arduino khác vào máy thì cần phải chọn lại cổng COM, quy trình thực hiện cũng tương tự.

Ngoài ra có thể xuất file .HEX từ menu Sketch , phím tắt Ctrl + Alt + S (đường dẫn file .HEX nằm chung thư mục với chương trình).

Thư viện RTC được cài đặt bổ sung sau khi cài đặt Arduino IDE (https://hacksterio.s3.amazonaws.com/uploads/attachments/867168/rtclib_LSmJ1qsBwz.zip).

1.5. Proteus version 8.9 SP0

Proteus Design Suite là bộ công cụ phần mềm độc quyền được sử dụng chủ yếu cho tự động hóa thiết kế điện tử. Phần mềm được sử dụng chủ yếu bởi các kỹ sư thiết kế điện tử và kỹ thuật viên để tạo sơ đồ và bản in điện tử để sản xuất bảng mạch in.

Thanh Menu: chứa các chức năng chính của chương trình

Thanh công cụ: chứa các công cụ để làm việc như thêm thiết bị, lựa chọn, thêm nhãn v.v.v. Một chức năng quan trọng là chọn thiết bị sử dụng. Trên thanh công cụ, chọn Component mode, Pick Device, tìm kiếm thiết bị trong các thư viện.

Các linh kiện: danh sách các linh kiện được thêm

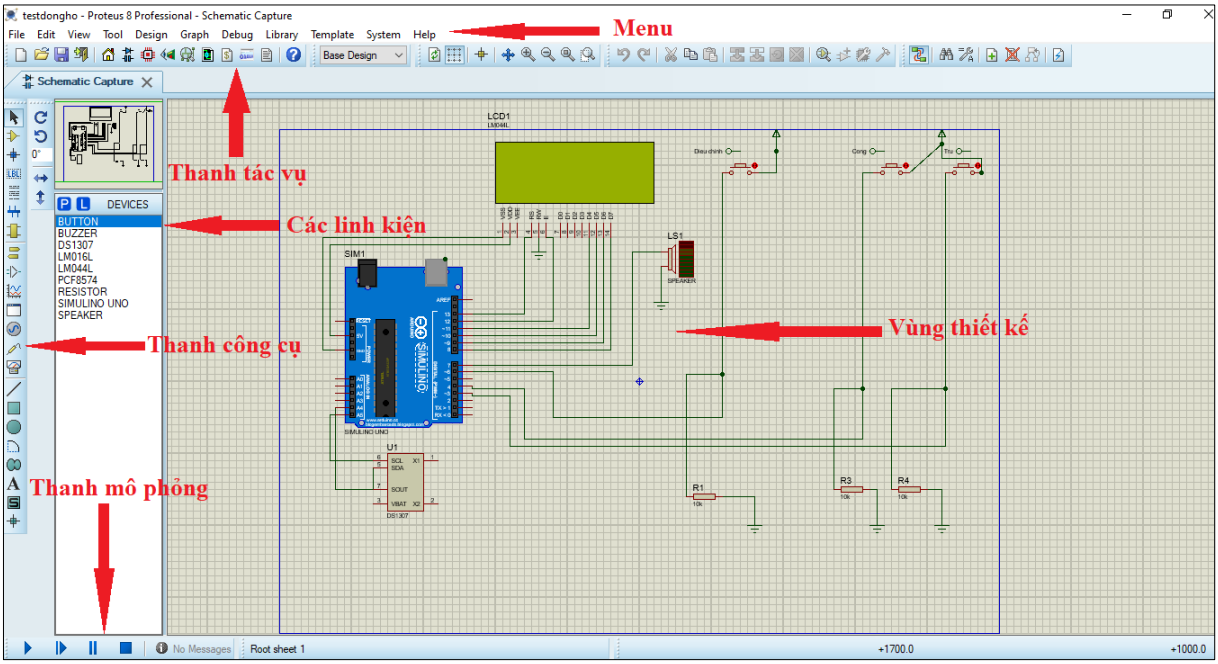
Vùng thiết kế: nơi thiết kế, mô phỏng các linh kiện, sơ đồ mạch

Thanh mô phỏng: mô phỏng hoạt động của các linh kiện trong vùng thiết kế

Để nạp chương trình vào mạch điều khiển mô phỏng, nhấp chuột vào vi xử lý mô phỏng trên mạch, mở thư mục và tìm đến file .HEX mong muốn.

Để sử dụng các thiết bị không có sẵn trong thư viện Proteus, tải các thư viện thiết bị và copy file .IDX và .LIB vào thư mục LIBRARY của thư mục Proteus.

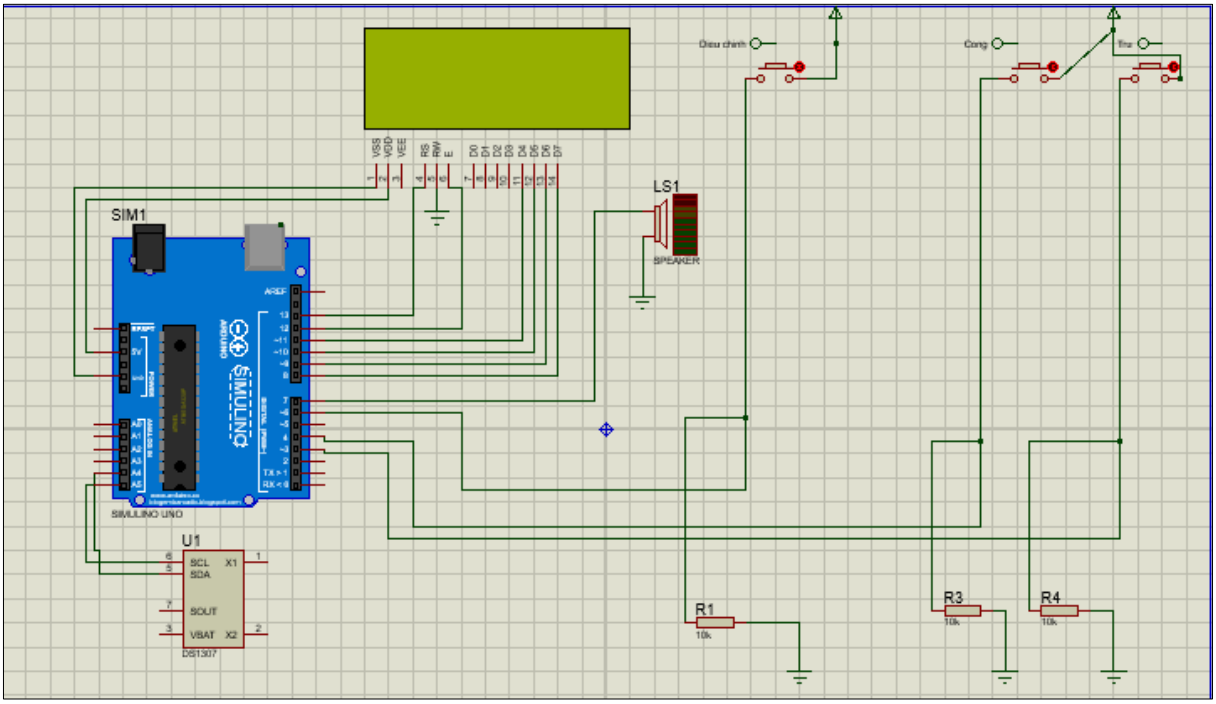
Thư viện Arduino, RTC được cài đặt bổ sung sau khi cài đặt Proteus (https://drive.google.com/file/d/1tVS8_dMfoPrZbJWzv3bjzor4BaNEVeuv/view)



Hình 1.8. Giao diện làm việc Proteus 8.9 SP0

CHƯƠNG 2: PHƯƠNG PHÁP NGHIÊN CỨU

2.1. Sơ đồ nguyên lý mô phỏng bằng Proteus 8.9 SP0



Hình 2.1. Sơ đồ nguyên lý đồng hồ trên Proteus version 8.9

Sử dụng phần mềm mô phỏng Proteus version 8.9 SP0, cùng các thiết bị chính:

- LCD20x4 (LM044) thư viện DISPLAY
- DS1307 thư viện MAXIM
- Arduino UNO thư viện ARDUINO
- Button thư viện ACTIVE
- Speaker thư viện ACTIVE

2.1.1. Giao tiếp giữa Arduino và IC DS1307

Giao tiếp với IC DS1307 theo chuẩn giao tiếp I2C. Trên board Arduino UNO, SDA là chân analog 4, SCL là chân analog 5.

Arduino	**DS1307**
A5	SCL
A4	SDA

2.1.2. Giao tiếp giữa Arduino với LCD20x4

Sau khi đọc dữ liệu từ IC DS1307, Arduino sẽ xuất thông tin thời gian lên LCD. LCD 20x4 có thể được sử dụng ở chế độ 4 bit hoặc chế độ 8 bit tùy thuộc vào yêu cầu của ứng dụng.

- Trong chế độ 4 bit, dữ liệu / lệnh được gửi ở định dạng 4 bit (nibble).
- Chỉ có 4 chân dữ liệu (D4 - D7) của LCD 20x4 và các chốt điều khiển khác RS (Đăng ký chọn), RW (Đọc / ghi), E (Bật) được kết nối được kết nối với vi điều khiển . Do các kết nối như vậy, có thể tiết kiệm bốn chân trên vi điều khiển, có thể được sử dụng cho một ứng dụng khác.
- Cấp nguồn cho LCD:

Arduino	LCD
GND	VSS
5V	VDD

- Thiết lập chế độ LCD:

Arduino	LCD
13	RS
12	E
GND	RW

- 4 chân truyền dữ liệu:

Arduino	LCD
11	D4
10	D5
9	D6
8	D7

2.1.3. Giao tiếp giữa Arduino-Loa speaker

Arduino sẽ điều khiển Loa phát âm thanh khi báo thức

Arduino	Loa
7	NC
GND	GND
5V	VSS

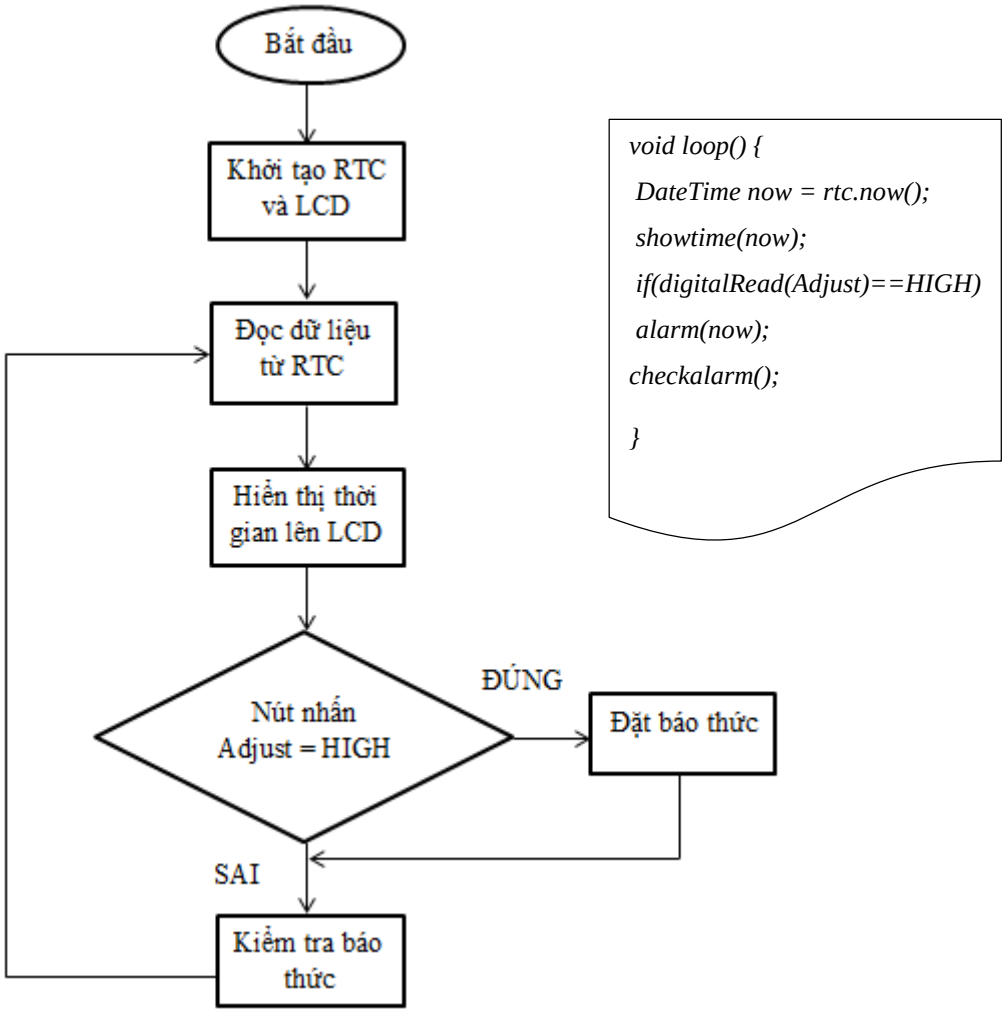
2.1.4. Giao tiếp giữa Arduino – 3 nút nhấn

Dùng để điều khiển các chức năng của chương trình

- 6 – Điều chỉnh (Enter)
- 4 – Tiến (tăng)
- 3 – Lùi (giảm)
- Các đầu cực còn lại của 3 nút nhấn nối với nguồn điện
- Nối các điện trở theo sơ đồ (liên quan đến dòng điện trong vật lý)

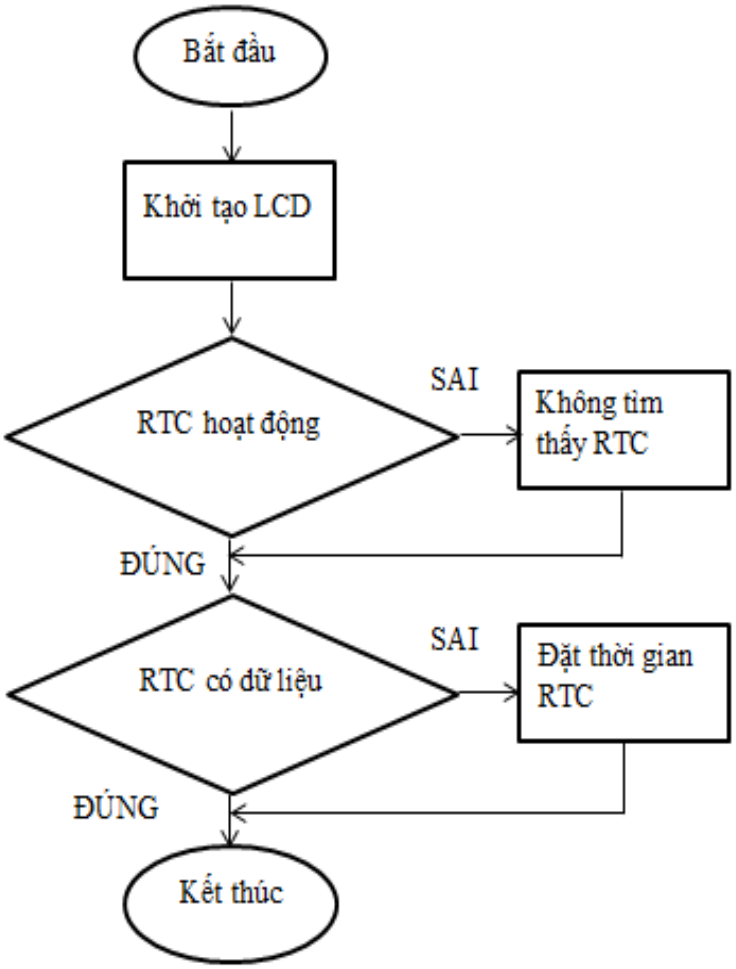
2.2. Thuật toán

2.2.1. Chương trình chính



Hình 2.2. Sơ đồ thuật toán chương trình chính

2.2.2. Khởi tạo RTC và LCD



Hình 2.3. Sơ đồ trình tự khởi tạo RTC và LCD

- Để làm việc với RTC và LCD ta cần 3 thư viện là Wire.h, RTClib.h, LiquidCrystal.h. Khai báo 3 thư viện này ngay từ đầu chương trình

```
#include <Wire.h>
#include "RTClib.h"
#include <LiquidCrystal.h>
// khai báo lcd
LiquidCrystal lcd(pin_RS,pin_EN,pin_d4,pin_d5,pin_d6,pin_d7);
// doi tuong dong ho
RTC_DS1307 rtc;
```

- Trong hàm setup(). Khởi tạo LCD bằng hàm **lcd.begin().** Kiểm tra RTC có hoạt động không bằng hàm **rtc.begin(),** kiểm tra RTC có dữ liệu không bằng hàm **rtc.isrunning().** Đặt thời gian cho RTC là ngày giờ hiện tại của hệ thống bằng hàm **rtc.adjust(DateTime(F(__DATE__), F(__TIME__)))**

```
void setup() {
 //khoi tao cac nut bam
 pinMode(Adjust,INPUT);
 pinMode(Tien,INPUT);
 pinMode(Lui,INPUT);
 // khoi tao lcd
 lcd.begin(20, 4);
 // khoi tao dong ho
  if (! rtc.begin()) {
   lcd.print("Khong tim thay RTC");
   while (1);
  }
 if (! rtc.isrunning()) {
    lcd.clear();
    lcd.print("RTC chua dat thoi gian");
   // Dat ngay gio dong ho theo hien tai cua may tinh
   rtc.adjust(DateTime(F(__DATE__), F(__TIME__)));
   // ta cung co the dat theo y muon theo vd ben duoi
   // January 21, 2014 at 3am
   // rtc.adjust(DateTime(2014, 1, 21, 3, 0, 0));
  }
}
```

2.2.3. Hiển thị thời gian lên LCD

Để hiển thị thời gian trong chương trình, các số có ít hơn 2 chữ số sẽ hiển thị thêm số 0 phía trước, mã lệnh dưới đây hiển thị giờ, tương tự cho phút, giây, ngày, tháng , năm.

```
    // hien thi gio
     lcd.setCursor(0, 1);
     if(now.hour()<=9)
       {
        lcd.print("0");
        lcd.print(now.hour());
```

```
  }
  else {
   lcd.print(now.hour());
  }
```

Tương tự là phần hiển thị cho các biến: thứ, ngày, tháng, năm, phút, giây.

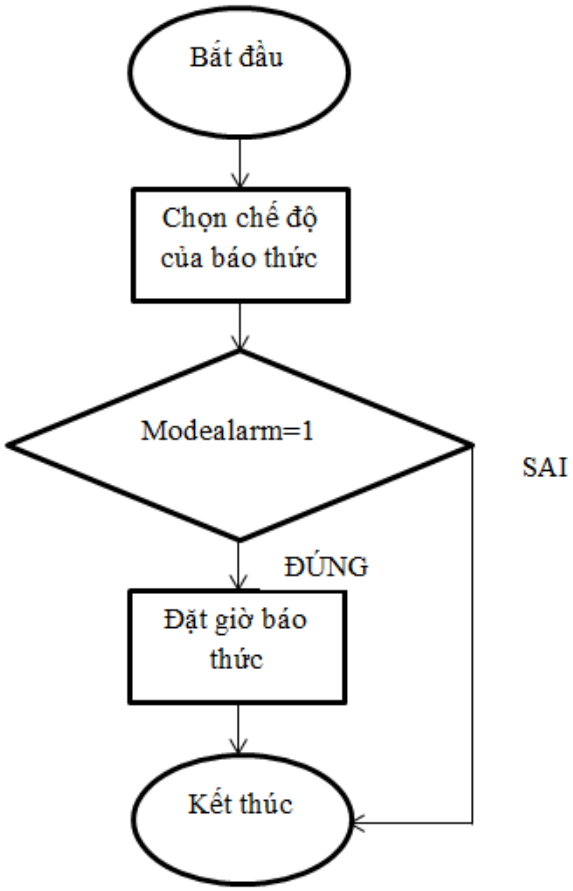
2.2.4. Các hàm phục vụ báo thức

2.2.4.1. Hiển thị giờ phút cho báo thức

Hàm này hiển thị giờ và phút của báo thức, tương tự như hiển thị giờ đồng hồ hiện tại, các số có ít hơn 2 chữ số thì hiển thị số 0 phía trước.

```
void timehour(int gio,int phut){
  lcd.setCursor(0, 1);
 if(gio<=9)
  {
   lcd.print("0");
   lcd.print(gio);
  }
  else {
   lcd.print(gio);
  }
  lcd.print(':');
  if(phut<=9)
  {
    lcd.print("0");
    lcd.print(phut);
  }
  else {
   lcd.print(phut);
  }
}
```

2.2.4.2. Đặt giờ báo thức

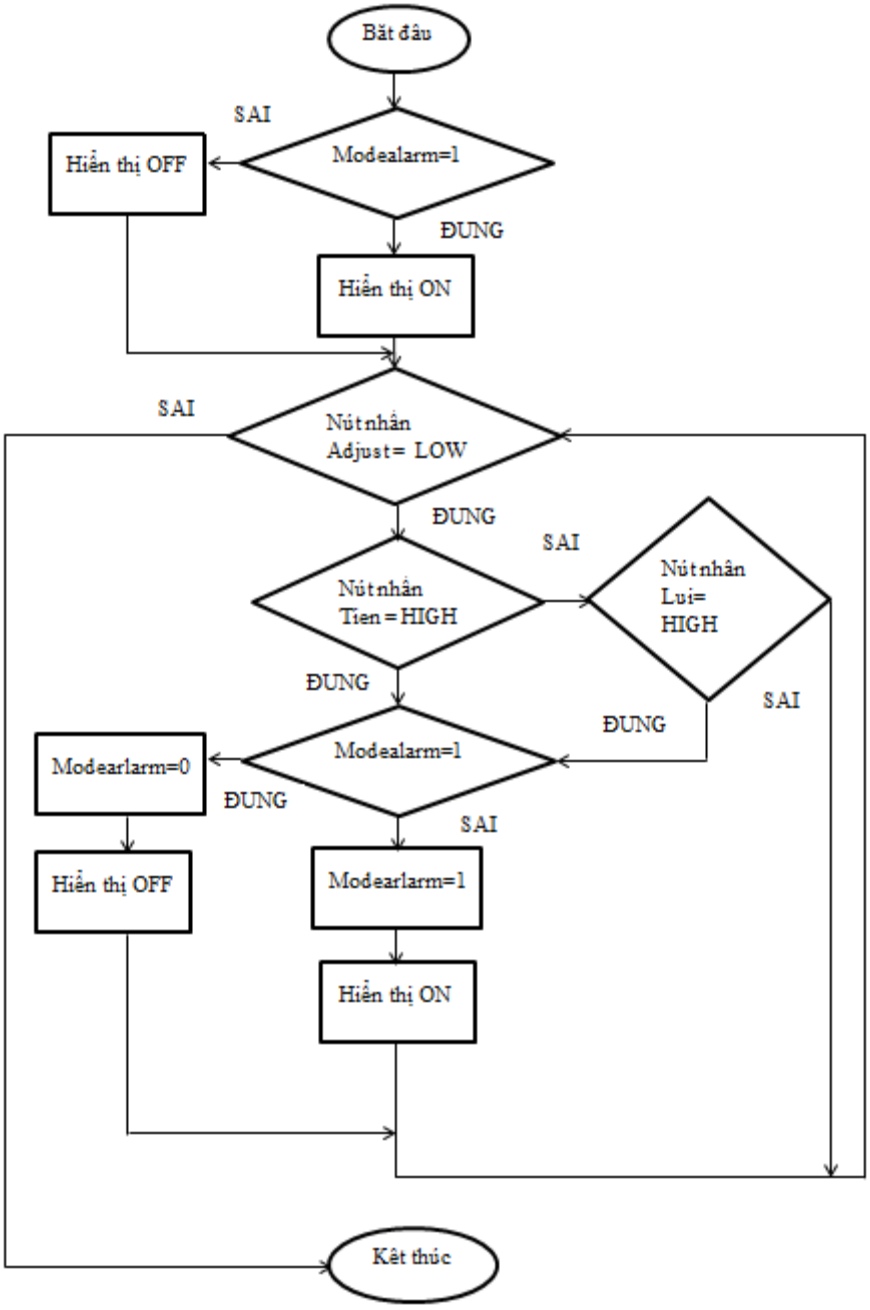


Hình 2.4. Sơ đồ đặt báo thức

Trong hàm này, thực hiện 2 công việc chính, chọn chế độ báo thức Bật/Tắt và đặt giờ báo thức

Chọn chế độ báo thức: một biến toàn cục (Modealarm) lưu trữ chế độ của báo thức 1/0 tương ứng Bật/Tắt và hiển thị lên LCD chế độ hiện tại, tiếp theo đọc dữ liệu nút nhấn Tien hoặc Lui để chọn thay đổi giá trị Modealarm 1/0 và hiển thị lên LCD, kết thúc khi dữ liệu đầu vào nút nhấn Adjust là HIGH (khi nút Adjust được nhấn).

Khởi tạo chương trình với Adjust, Tien, Lui là LOW; Modealarm bằng 0

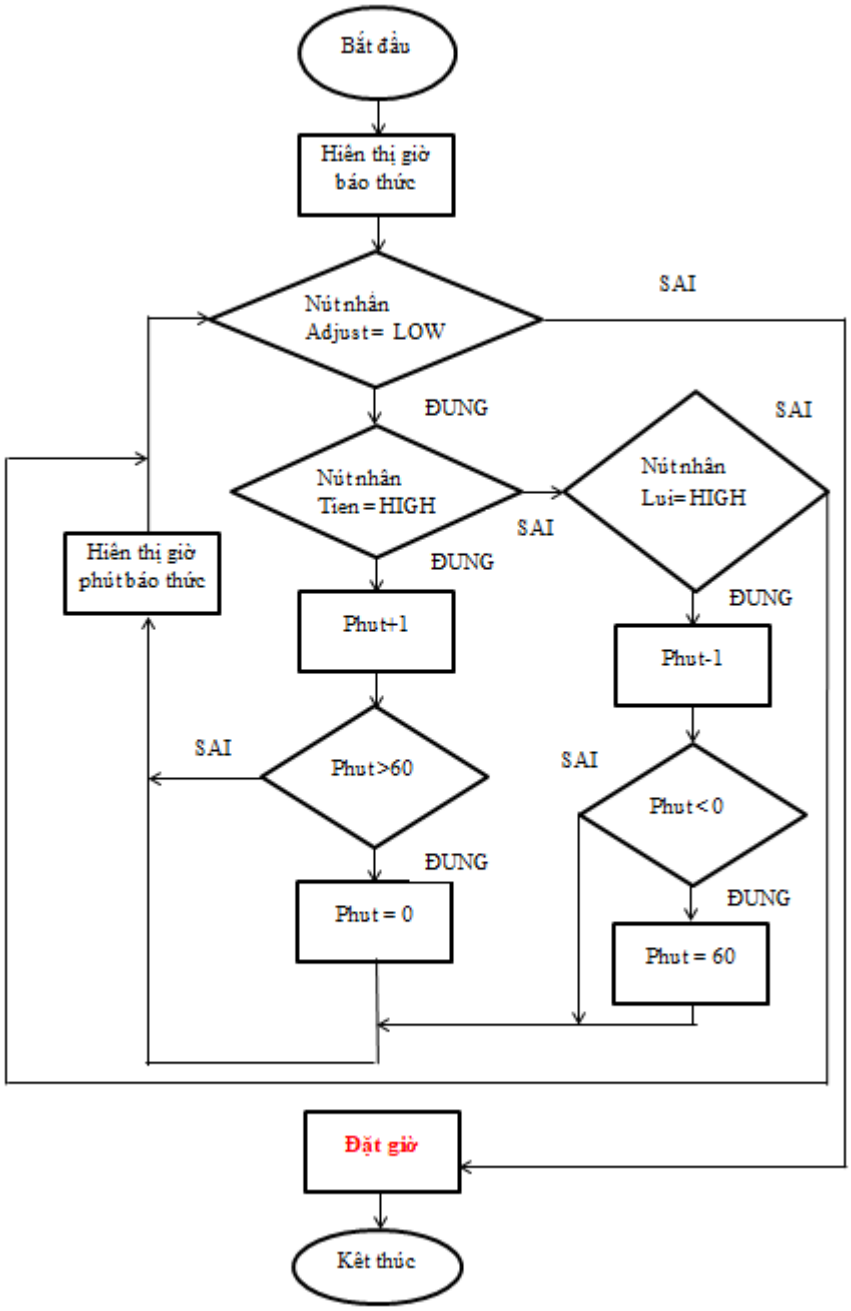


Hình 2.5. Sơ đồ thuật toán chọn chế độ báo thức

```
lcd.setCursor(0,0);
    lcd.print("Alarm: ON/OFF ??");
    if(modealarm==0)
    {
     lcd.setCursor(0,1);
     lcd.print("Mode: OFF");
    }else {lcd.setCursor(0,1); lcd.print("Mode: ON");}
```

```
while(digitalRead(Adjust)==LOW){
 if(digitalRead(Tien)==HIGH)
  {
    delay(350);
    if(modealarm==0)
    {
      modealarm=1;
      lcd.setCursor(0,1);
      lcd.print("Mode: ON ");
    }else if(modealarm==1) {
      modealarm=0;
      lcd.setCursor(0,1);
      lcd.print("Mode: OFF");
    }
  }
  else if(digitalRead(Lui)==HIGH)
  {
   delay(350);
    if(modealarm==0)
    {
      modealarm=1;
      lcd.setCursor(0,1);
      lcd.print("Mode: ON ");
    }else if(modealarm==1) {
      modealarm=0;
      lcd.setCursor(0,1);
      lcd.print("Mode: OFF");
    }
  }
}
```

Đặt giờ báo thức: hai biến toàn cục lưu trữ giờ (gio), phút (phut) của báo thức. Đầu tiên là đặt phút của báo thức, đọc dữ liệu từ 2 nút nhấn Tien và Lui để tăng/giảm biến phut 1 đơn vị, vì giới hạn của phút là [0,60], khi phút đến giá trị 60/0 nếu tiếp sẽ tăng/giảm quay ngược giá trị lại 61=0, -1=60. Đặt phút báo thức kết thúc khi dữ liệu nút Adjust=HIGH (nút nhấn Adjust được nhấn), và chuyển đến **đặt giờ**, tương tự như đặt phút.



Hình 2.6. Sơ đồ thuật toán cho đặt phút báo thức

```
lcd.setCursor(0,0);
 lcd.print("Set time alarm:");
 //hien thi gio bao thuc
 timehour(gio,phut);
  //hien thi con tro
  lcd.cursor();
  lcd.setCursor(4,1);
  //dat thoi gian bao thuc
  //dat phut
  delay(350);
  while(digitalRead(Adjust)==LOW){
    if(digitalRead(Tien)==HIGH)
     {
       delay(350);
       phut=phut+1;
       if(phut>60) phut=0;
       timehour(gio,phut);
       lcd.setCursor(4,1);
     }
     else if(digitalRead(Lui)==HIGH)
     {
      delay(350);
     phut=phut-1;
     if(phut<0) phut=60;
     timehour(gio,phut);
     lcd.setCursor(4,1);}
    }
```

Để giúp nhận biết đang đặt giờ phút báo thức tới đâu, hàm **lcd.Cursor()** và **lcd.noCursor()** bật/tắt nhấp nháy vị trí con trỏ hiện tại.

2.2.4.3. Kiểm tra giờ báo thức

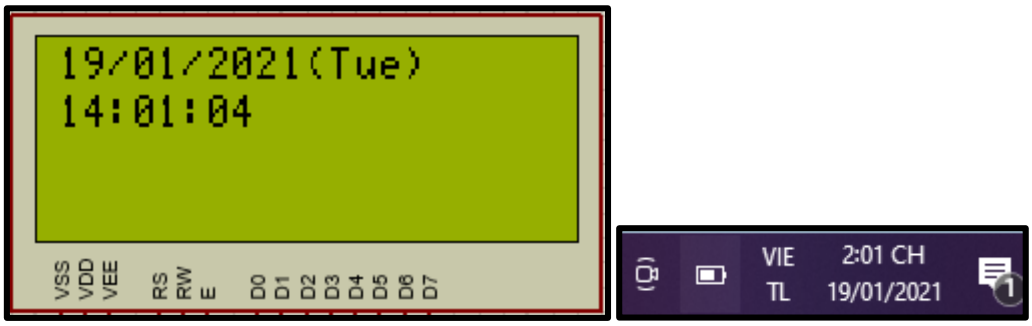
Để kiểm tra đúng giờ báo thức, hàm sẽ đọc thời gian hiện tại của RTC, một vòng lặp while kiểm tra giờ báo thức và giờ của RTC cùng với chế độ của báo thức, nếu thỏa mãn sẽ thông báo lên màn hình và phát tiếng kêu báo thức bằng hàm **tone**(<chân loa>,<tần số>,<thời gian>). Sau mỗi vòng lặp while, đọc lại thời gian hiện tại của RTC.

```
void checkalarm(){
  DateTime now=rtc.now();
  while(gio==now.hour()&& phut==now.minute()&& modealarm==1)
  {
   lcd.clear();
   lcd.print("*******Alarm********");
   timehour(gio,phut);
      tone(buzzer,500,500);
      delay(1000);
      now=rtc.now();
   lcd.clear();
  }
}
```

CHƯƠNG 3: KẾT QUẢ THỰC HIỆN

3.1. Kết quả mô phỏng trên Proteus 8.9 SP0

3.1.1. Hiển thị ngày giờ



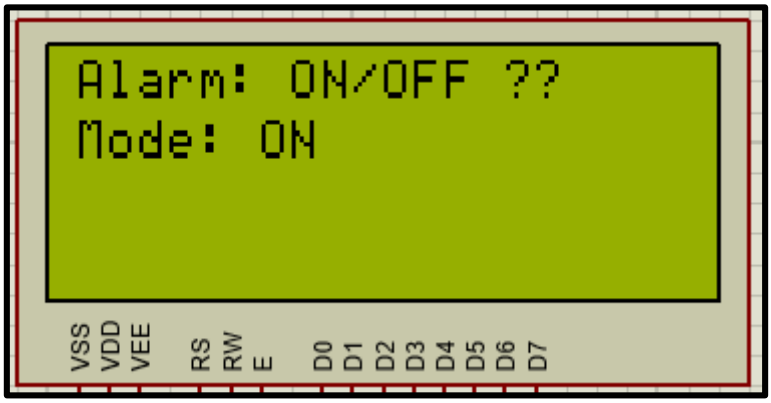
Hình 3.1 Hiển thị ngày giờ mô phỏng trên Proteus 8.9 SP0

Mặc định của hiển thị, hoặc khi không chuyển sang các chế độ khác. Theo định dạng ngày/tháng /năm/thứ, giờ/phút/giây, định dạng 24 giờ. Đặt giờ RTC theo giờ hiện tại của hệ thống.

3.1.2. Đặt báo thức

Bước 1: Nhấn Dieu chinh (Adjust) để chuyển sang chế độ đặt báo thức

Bước 2: Bật/ tắt báo thức ON/OFF sử dụng 2 nút nhấn Tien/Lui để chọn ON/OFF nhấn Dieu chinh (Adjust) để xác nhận.

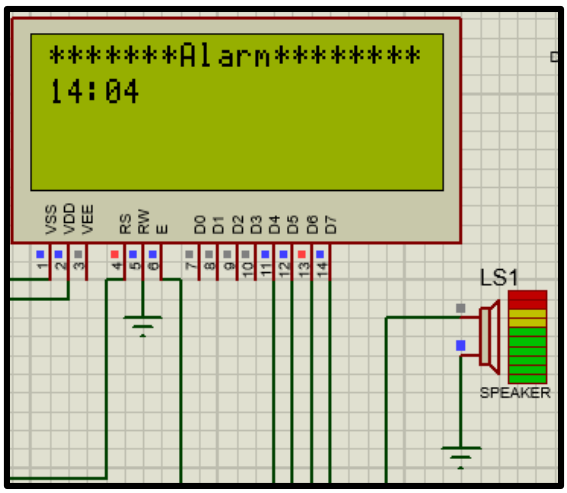


Hình 3.2 Hiển thị chế độ báo thức mô phỏng trên Proteus 8.9 SP0

Bước 3: Nếu chế độ ON, đặt giờ báo thức, con trỏ bắt đầu từ điều chỉnh phút, sử dụng 2 nút nhấn Tien/Lui để tăng hoặc giảm số phút, nhấn Dieu chinh (Adjust) để xác nhận và chuyển sang điều chỉnh tương tự cho giờ.

Hình 3.3 Đặt giờ báo thức mô phỏng trên Proteus 8.9 SP0

Khi đúng giờ Báo thức sẽ phát tiếng kêu trong cả phút và hiển thị giờ đã đặt báo thức lên màn hình



Hình 3.4 Báo thức khi đúng giờ mô phỏng trên Proteus 8.9 SP0

3.2. Kết quả mạch thực tế

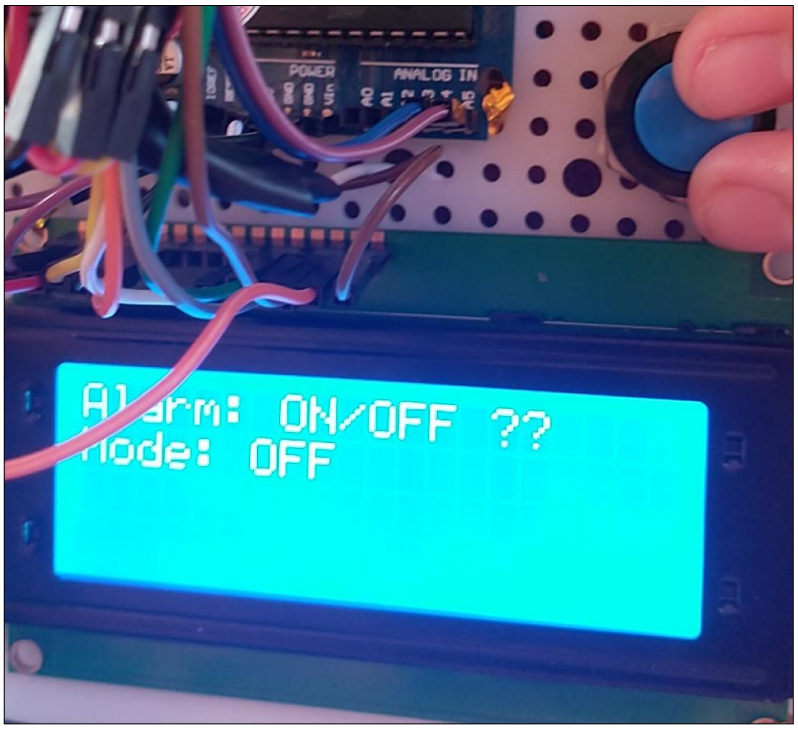
3.2.1. Hiển thị ngày giờ

Chương trình nạp từ máy tính vào mạch Adruino, đặt thời gian đồng hồ theo thời gian máy tính hiện tại, và hiển thị lên LCD.

Hình 3.5. Kết quả hiển thị ngày giờ trên mạch thực tế

3.2.2. Đặt báo thức

Nhấn nút Xanh (Adjust) chuyển sang chế độ đặt báo thức



Hình 3.6. Chế độ đặt báo thức

Sử dụng hai nút Vàng/Đen (Tien/Lui) để chọn ON/OFF

Hình 3.7. Chọn chế độ báo thức

Sử dụng hai nút Vàng/Đen (Tien/Lui) để chọn Tăng/Giảm đặt phút báo thức, nhấn nút Xanh (Adjust) để xác nhận, tương tự đặt giờ báo thức.

Hình 3.8. Đặt giờ phút báo thức

Khi đúng giờ báo thức, sẽ hiển thị thông báo lên màn hình và phát tiếng kêu trong cả phút.

Hình 3.9. Báo thức khi đúng giờ đã đặt

THẢO LUẬN

Sản phẩm cơ bản đáp ứng các yêu cầu đề ra, tuy nhiên chưa hoàn thiện về mặt hình thức trang trí. Trong tương lai, sẽ phát triển sản phẩm hoàn thiện hơn, có các chức năng khác như đặt lại ngày giờ, xem lịch, bấm thời gian... Toàn bộ quy trình thiết kế, thi công sản phẩm được hoàn thiện trong thời gian 4 tuần của đợt thực tập cơ sở.

TÀI LIỆU THAM KHẢO

[1]. Lập trình Arduino (bài giảng), Mai Cường Thọ, Đại Học Nha Trang, 2020

[2]. ElectroPeak (2019), Interfacing DS1307 RTC Module with Arduino & Make a Reminder [online], viewed 01/01/2021, from:<https://create.arduino.cc/projecthub/electropeak/interfacing-ds1307-rtc-module-with-arduino-make-a-reminder-08cb61>

[3]. Nguyễn Đức Khôi (2018), Các hiệu ứng cơ bản của LCD 16x2, viewed 01/01/2021.

PHỤ LỤC

Các thành phần cơ bản trong chương trình Arduino

Cấu trúc	Giá trị	Hàm và thủ tục
• setup() • loop() **Cấu trúc điều khiển** • if • if...else • switch / case • for • while • break • continue • return • goto **Cú pháp mở rộng** • ; (dấu chấm phẩy) • {} (dấu ngoặc nhọn) • // (single line comment) • /* */ (multi-line comment) • #define • #include **Toán tử số học** • = (phép gán) • + (phép cộng) • - (phép trừ) • * (phép nhân) • / (phép chia) • % (phép chia lấy dư) **Toán tử so sánh** • == (so sánh bằng) • != (khác bằng) • > (lớn hơn) • < (bé hơn) • >= (lớn hơn hoặc bằng) • <= (bé hơn hoặc bằng) **Toán tử logic** • && (và) • || (hoặc) • ! (phủ định)	**Hằng số** • HIGH | LOW • INPUT | INPUT_INPUT_PULLUP | OUTPUT • LED_BUILTIN • true | false • hằng số nguyên (interger constans) • hằng số thực (floating point constans) **Kiểu dữ liệu** • void • boolean • char • unsigned char • byte • int • unsigned int • word • long • unsigned long • short • float • double • array • string (chuỗi kí tự biểu diễn bằng array) • String (object) **Chuyển đổi kiểu dữ liệu** • char() • byte() • int() • word() • long() • float() **Phạm vi của biến và phân loại** • Phạm vi hiệu lực của biến • static – biến tĩnh • const – biến hằng • volatile **Hàm hỗ trợ** • sizeof()	**Nhập xuất Digital (Digital I/O)** • pinMode() • digitalWrite() • digitalRead() **Nhập xuất Analog (Analog I/O)** • analogReference() • analogRead() • analogWrite() • PWM –PPM **Hàm thời gian** • millis() • micros() • delay() • delayMicroseconds() **Hàm toán học** • min() • max() • abs() • map() • pow() • sqrt() • sq() • isnan() • constrain() • exp(x) • frexp(x, int *exp) • ldexp(x,int exp) • log(x) • log10(x) • modf(x, *i) • ceil(x) • floor(x) • atoi(a[]) **Hàm lượng giác** • cos() • sin() • tan() • asin(x) • acos(x) • atan(x) • atan2(x,y) • cosh(x) • sinh(x) • tanh(x)

• ^ (loại trừ) **Phép toán hợp nhất** • ++ (cộng thêm 1 đơn vị) • --(trừ đi 1 đơn vị) • += (phép rút gọn của phép cộng) • -= (phép rút gọn của phép trừ) • *= (phép rút gọn của phép nhân) • /= (phép rút gọn của phép chia)		**Sinh số ngẫu nhiên** • randomSeed() • random() **Nhập xuất nâng cao (Advanced I/O)** • tone() • noTone() • shiftOut() • shiftIn() • pulseIn() **Xử lý chuỗi** • isAscii() • isWhitespace() • isAlpha() • isAlphaNumeric() • isControl() • isDigit() • isGraph() • isLowerCase() • isPrintable() • isPunct() • isSpace() • isUpperCase() • isHexadecimalDigit() • tolower() • toupper() **Bits và Bytes** • lowByte() • highByte() • bitRead() • bitWrite() • bitSet() • bitClear() • bit() **Ngắt (interrupt)** • attachInterrupt() • detachInterrupt() • interrupts() • noInterrupts() **Giao tiếp** • Serial