

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH  
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN  
KHOA VẬT LÝ - VẬT LÝ KỸ THUẬT  
BỘ MÔN VẬT LÝ ĐIỆN TỬ

ĐỀ TÀI THỰC TẬP VI ĐIỀU KHIỂN

THIẾT KẾ LỊCH VẠN NIÊN

SVTH: Võ Tân Minh Khôi  
Ngạc Bảo Nam  
Phùng Thị Hương  
CBHD: HVCH. Hà Minh Khuê

TP. HỒ CHÍ MINH - 2022

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH  
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN  
KHOA VẬT LÝ - VẬT LÝ KỸ THUẬT  
BỘ MÔN VẬT LÝ ĐIỆN TỬ

ĐỀ TÀI THỰC TẬP VI ĐIỀU KHIỂN

THIẾT KẾ LỊCH VẠN NIÊN

SVTH: Võ Tân Minh Khôi  
Ngạc Bảo Nam  
Phùng Thị Hương  
CBHD: HVCH. Hà Minh Khuê

TP. HỒ CHÍ MINH - 2022

# Tóm tắt

Đề tài **Thiết kế lịch vạn niên** của nhóm 2 sử dụng Vi điều khiển họ AVR 8bits để làm bộ xử lý chính và hiển thị lên LED 7 đoạn. Khác với các đề tài sử dụng LED 7 đoạn trước đây là sử dụng phương pháp quét LED. Phương pháp này có một mặt hạn chế là sẽ sử dụng nhiều chân của vi điều khiển hơn và sẽ tốn nhiều diện tích hơn khi thiết kế mạch in vì phải lắp thêm các linh kiện điện tử thụ động khác chẳng hạn như điện trở và transistor. Chính vì thế, nhóm 2 quyết định sẽ sử dụng IC MAX7219 để điều khiển số LED 7 đoạn đó mà không cần phải lắp thêm trở hoặc transistor cho mỗi LED 7 đoạn. IC này sẽ giao tiếp với vi điều khiển bằng giao thức SPI.

Để đọc thời gian thực, nhóm sử dụng IC DS1307 giao tiếp với vi điều khiển bằng giao thức I2C. Bên cạnh việc hiển thị giờ, phút, ngày, tháng, năm dương lịch và ngày, tháng, năm âm lịch nhóm còn thiết kế thêm tính năng tăng giảm độ sáng hiển thị, hẹn giờ báo thức cho lịch vạn niên.

## Lời cảm ơn

Đề tài **Thiết kế lịch vạn niên** đã được hoàn thành. Trong quá trình nghiên cứu và hoàn thiện, nhóm em đã nhận được sự hướng dẫn và giúp đỡ nhiệt tình từ thầy cô trong bộ môn.

Em xin gửi lời cảm ơn chân thành tới nhà trường, Khoa Vật Lý - Vật lý kỹ thuật. Bộ môn Vật Lý Điện Tử đã tạo điều kiện và giúp đỡ cho chúng em được học tập, tiếp thu, và nghiên cứu một cách tốt nhất về môn: “ Vì điều khiển và ứng dụng”.

Em xin cảm ơn anh Hà Minh Khuê đã nhiệt tình hướng dẫn giúp đỡ nhóm em trong quá trình thực hiện đề tài nghiên cứu.

Do trình độ nghiên cứu còn hạn chế và những nguyên nhân khác nên dù cố gắng xong bài báo cáo đề tài nhóm em cũng không tránh khỏi những hạn chế, thiếu sót. Vì vậy, nhóm em rất mong nhận được sự quan tâm đóng góp ý kiến từ anh và mọi người.

Những ý kiến đóng góp của anh và mọi người sẽ giúp chúng em nhận ra được những hạn chế và qua đó sẽ có thêm những nguồn tư liệu mới trên con đường học tập cũng như nghiên cứu sau này. Nhóm em xin chân thành cảm ơn.

# Mục lục

Tóm tắt	i
Lời cảm ơn	ii
Mục lục	iv
Danh sách hình vẽ	v
<b>1 Tổng quan</b>	<b>1</b>
1.1 Giới thiệu . . . . .	1
1.2 Mục tiêu . . . . .	1
1.3 Phương pháp nghiên cứu . . . . .	1
<b>2 Nội dung nghiên cứu</b>	<b>2</b>
2.1 Sơ đồ khái hệ thống . . . . .	2
2.1.1 Nguyên lý hoạt động . . . . .	2
2.1.2 Các chức năng của hệ thống . . . . .	3
2.1.3 Thuật toán tính lịch âm . . . . .	3
2.2 Thiết kế phần cứng . . . . .	4
2.2.1 Atmega32 . . . . .	4
2.2.2 IC DS1307 . . . . .	5
2.2.3 IC MAX7219 và LED 7 đoạn Cathode chung . . . . .	6
2.2.4 IC 40106 . . . . .	6
<b>3 Phương pháp thực hiện</b>	<b>9</b>

<b>4 Dánh giá và thử nghiệm</b>	<b>13</b>
4.1 Chế độ hiển thị 1 . . . . .	13
4.2 Chế độ hiển thị 2 . . . . .	14
4.3 Chế độ hiển thị 3 . . . . .	14
4.4 Chế độ hiển thị 4 . . . . .	14
<b>5 Kết luận và hướng phát triển</b>	<b>16</b>
5.1 Dánh giá hệ thống . . . . .	16
5.2 Các vấn đề còn tồn đọng . . . . .	16
5.3 Hướng phát triển của đề tài . . . . .	17
<b>Phụ lục A</b>	<b>19</b>

# Danh sách hình vẽ

2.1	Sơ đồ khối hệ thống . . . . .	2
2.2	Sơ đồ chân Atmega32 [1] . . . . .	4
2.3	IC DS1307 [2] . . . . .	5
2.4	IC MAX7219 [3] . . . . .	6
2.5	Sơ đồ nối dây MAX7219 và LED 7 đoạn . . . . .	6
2.6	Tình trạng nhiễu do phần cứng . . . . .	7
2.7	Mạch thực tế phiên bản 1 . . . . .	7
2.8	IC 40106 [4] . . . . .	8
2.9	Dạng sóng nút nhấn sau khi debounce . . . . .	8
3.1	Quá trình cắm testboard . . . . .	9
3.2	Schematic của hệ thống . . . . .	10
3.3	Mặt Bottom layout PCB . . . . .	10
3.4	Mặt Top 3D . . . . .	11
3.5	Mặt Bottom 3D . . . . .	11
3.6	Hình ảnh thực tế mặt Top . . . . .	12
3.7	Hình ảnh thực tế mặt Bottom . . . . .	12
4.1	Chế độ 1 . . . . .	13
4.2	Chế độ 2 . . . . .	14
4.3	Chế độ 3 . . . . .	15
4.4	Chế độ 4 . . . . .	15

# **Chương 1**

## **Tổng quan**

### **1.1 Giới thiệu**

Ngày nay cùng với sự phát triển của khoa học công nghệ, ngành điện tử đã ứng dụng rất nhiều trong công nghiệp cũng được ứng dụng rộng rãi trong các thiết bị, sản phẩm phục vụ nhu cầu sinh hoạt hằng ngày như máy giặt, đồng hồ hẹn giờ.... Đã giúp đời sống của chúng ta ngày càng hiện đại và tiện nghi hơn.

Dề tài “ Đồng hồ lịch vạn niên” nhằm đáp ứng nhu cầu ham muốn học hỏi của bản thân, cũng như góp phần nâng cao giá trị mạch điện tử trong đời sống con người.

### **1.2 Mục tiêu**

Ứng dụng kỹ thuật vi điều khiển, áp dụng kiến thức đã học vi điều khiển để tạo ra lịch vạn niên có các chức năng báo thức, hẹn giờ. Đồng thời phải thiết kế mạch in nhỏ gọn, phù hợp làm đồng hồ để bàn.

### **1.3 Phương pháp nghiên cứu**

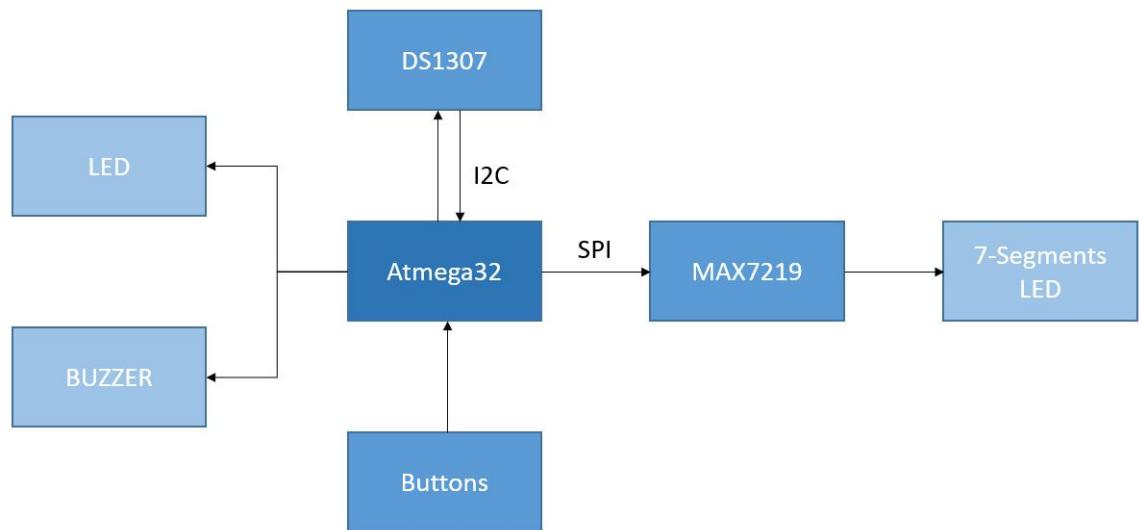
Phương pháp nghiên cứu dựa trên nguyên lý hoạt động của vi điều khiển, các giao thức như SPI, I2C, lý thuyết về Schmitt-trigger

Tìm hiểu nguyên lý hoạt động của Atmega32, DS1307, MAX7210, IC40106.

## Chương 2

### Nội dung nghiên cứu

#### 2.1 Sơ đồ khái niệm



Hình 2.1: Sơ đồ khái niệm

##### 2.1.1 Nguyên lý hoạt động

Khối xử lý Atmega32 nhận dữ liệu từ DS1307 bằng giao thức I2C, sau đó dữ liệu thời gian sẽ được hiển thị lên LED 7 đoạn thông qua IC MAX7219. IC này đóng vai trò giải mã sang LED 7 đoạn, giao tiếp với Atmega32 bằng giao thức SPI.

Sử dụng thư viện toán học của lập trình C để thực hiện các phép toán chuyển từ dương lịch sang âm lịch.

### 2.1.2 Các chức năng của hệ thống

Sử dụng 3 nút nhấn để thao tác điều chỉnh hệ thống.

#### Nút nhấn 1

Được kết nối với INT0 của Atmega. Bao gồm 2 chức năng:

1. Chuyển qua-lại giữa các chế độ: Hiển thị Ngày-tháng-Giờ Phút; Hiển thị Ngày-tháng-năm dương lịch; Hiển thị ngày-tháng-năm âm lịch; Hẹn giờ báo thức.
2. Lưu giá trị sau khi chỉnh sửa, hẹn giờ.

#### Nút nhấn 2

Được kết nối với INT1 của Atmega. Dùng để vào chế độ điều chỉnh, khi ấn nút này sẽ bắt đầu điều chỉnh ngày-tháng dương lịch, giờ-phút, báo thức. Ngày-tháng-năm âm lịch sẽ được tính toán dựa theo ngày dương.

#### Nút nhấn 3

Được kết nối với INT2 của Atmega. Khi ở chế độ điều chỉnh, sẽ sử dụng nút này để tăng giá trị của đối tượng được điều chỉnh.

Khi còi báo thức kêu, sử dụng nút này để tắt.

### 2.1.3 Thuật toán tính lịch âm

Âm lịch Việt Nam là một loại lịch thiên văn. Nó được tính toán dựa trên sự chuyển động của mặt trời, trái đất và mặt trăng. Ngày tháng âm lịch được tính dựa theo các nguyên tắc sau [5]

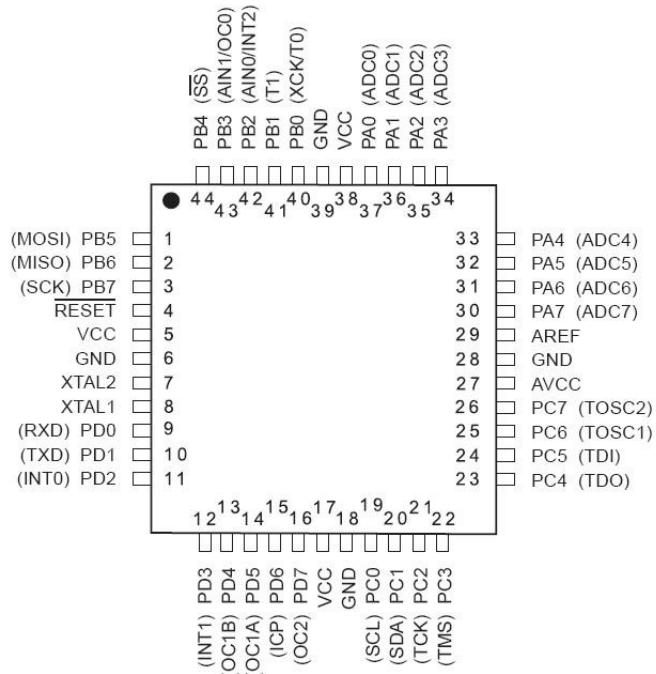
1. Ngày đầu tiên của tháng âm lịch là ngày chứa điểm Sóc.
2. Một năm bình thường có 12 tháng âm lịch, một năm nhuận có 13 tháng âm lịch.
3. Đông chí luôn rơi vào tháng 11 âm lịch.
4. Trong một năm nhuận, nếu có 1 tháng không có Trung khí thì tháng đó là tháng nhuận. Nếu nhiều tháng trong năm nhuận đều không có Trung khí thì chỉ tháng đầu tiên sau Đông chí là tháng nhuận.
5. Việc tính toán dựa trên kinh tuyến  $105^{\circ}$  đông.

Do không đủ kiến thức về thiên văn học, nhóm quyết định sẽ chỉ sử dụng lại các thuật toán nhằm tính lịch âm thay vì đi sâu vào cốt lõi. Sau khi tìm hiểu, các bước tính toán lịch âm bao gồm:

1. Đổi ngày dd/mm/yyyy ra số ngày Julius jd.
2. Đổi số ngày Julius jd ra ngày dd/mm/yyyy.
3. Tính ngày Sóc.
4. Tính tọa độ mặt trời.
5. Tìm ngày bắt đầu tháng 11 âm lịch.
6. Xác định tháng nhuận.
7. Xác định tháng nhuận.
8. Đổi ngày dương dd/mm/yyyy ra ngày âm.

## 2.2 Thiết kế phần cứng

### 2.2.1 Atmega32



Hình 2.2: Sơ đồ chân Atmega32 [1]

Sử dụng vi điều khiển **Atmega32** là vi điều khiển 8 bit CMOS công suất tiêu thụ thấp dựa trên RISC AVR. Bằng cách thực hiện các lệnh mạnh trong một chu kỳ đồng hồ.

Atmega32 đạt được tốc độ xấp xỉ 1MIPS trên 1MHz cho phép người thiết kế tối ưu công suất tiêu thụ với tốc độ xử lý.

- Cấu trúc RISC (Reduced Instruction Set Computing).
- 32 x 8 thanh ghi dùng chung.
- Tốc độ 16 MIPS với thạch anh 16 Mhz
- 32 KB bộ nhớ Flash.
- 2 KB SRAM.
- Hai bộ đếm /định thời 8 bit.
- Một bộ đếm/ định thời 16 bit.
- Bộ đếm thời gian thực với bộ giao động riêng.
- Giao tiếp I2C, SPI, ...
- Có bộ giao động RC bên trong.
- Ngắt trong và ngắt ngoài.

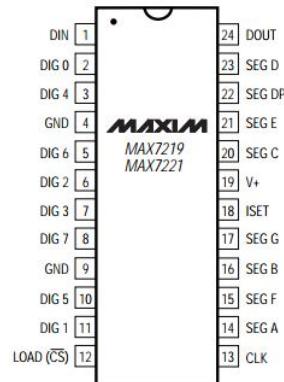
### 2.2.2 IC DS1307



Hình 2.3: IC DS1307 [2]

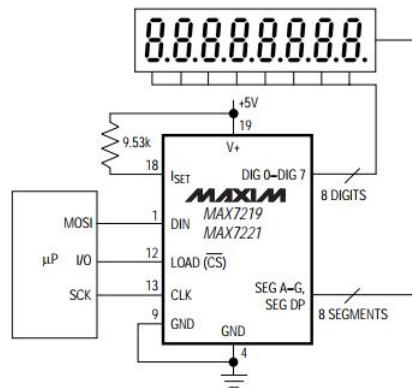
Khôi thời gian thực sử dụng IC DS1307. Nguồn pin CMOS 3.3V được nối vào chân 3 là nguồn dùng để đảm bảo IC vẫn hoạt động khi ngắt nguồn cung cấp chính. Chân 1 và 2 của IC nối với thạch anh 32,768 Hz để tạo dao động. Hai chân 5 và 6 là SCL và SCA nối với 2 chân I2C của Vi điều khiển, các chân này lần lượt là chân CLock và chân Data, ở mỗi chân có điện trở kéo lên 4,7KOhm.

### 2.2.3 IC MAX7219 và LED 7 đoạn Cathode chung



Hình 2.4: IC MAX7219 [3]

MAX7219 là IC dùng để giải mã từ DEC sang mã hiển thị LED 7 đoạn và dùng để quét LED. Giao tiếp với Atmega bằng giao thức SPI. Sử dụng 3 đường truyền chính lần lượt là CLK, SS, MOSI.

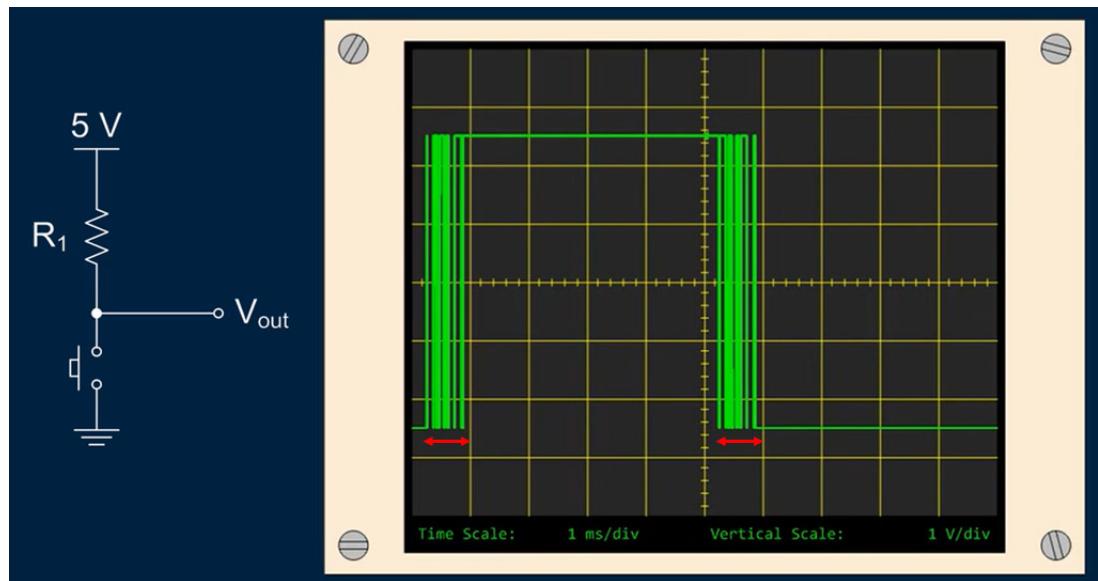


Hình 2.5: Sơ đồ nối dây MAX7219 và LED 7 đoạn

LED 7 đoạn sử dụng là loại 7 đoạn Cathode chung. Chân số 18 **ISET** có thể được nối với nguồn và sử dụng 1 biến trở hạn dòng 50KOhm, hoặc có thể thay vào đó là 1 biến trở 50K để điều chỉnh độ sáng của LED 7 đoạn.

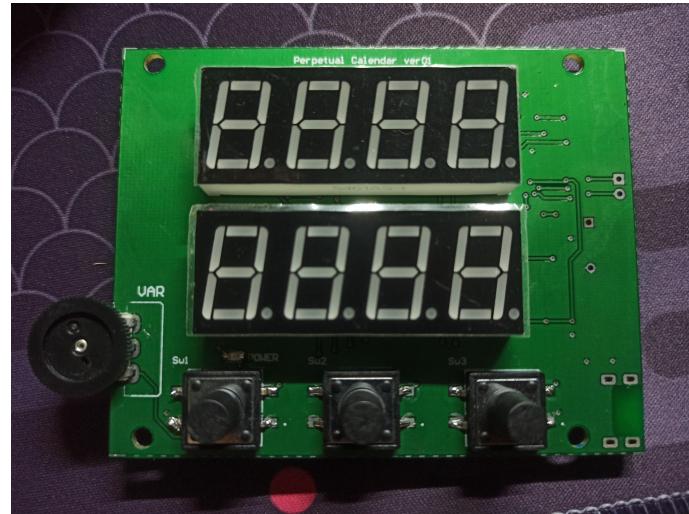
### 2.2.4 IC 40106

Khi làm việc với nút nhấn, chắc chắn một điều rằng sẽ gặp tình trạng nhiễu do tương tác vật lý của nút nhấn không tốt nên sẽ gặp các tín hiệu nhiễu (bouncing).



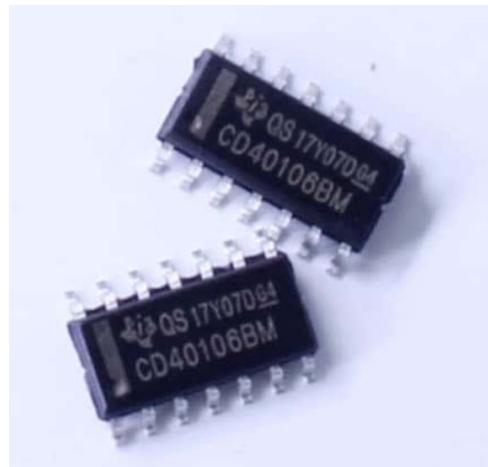
Hình 2.6: Tình trạng nhiễu do phần cứng

Sau khi phiên bản mạch in 1 hoàn thành (Hình 2.7), nút nhấn bị tình trạng bouncing, khi nhấn nút 1 lần có thể bị nhảy 2-3 lần.

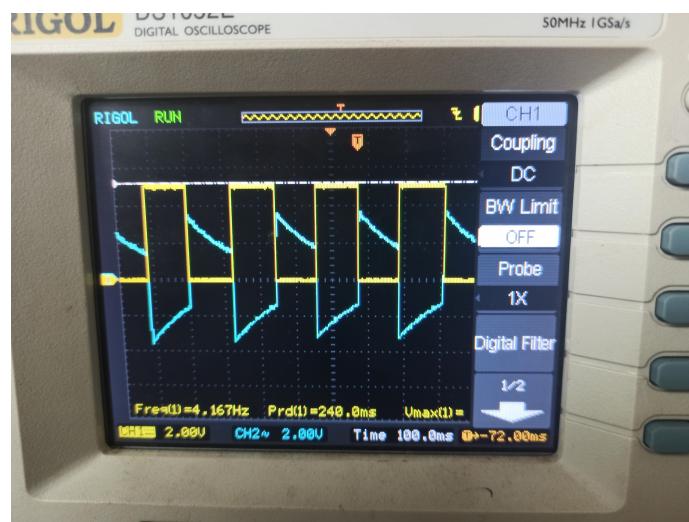


Hình 2.7: Mạch thực tế phiên bản 1

Nhóm quyết định sẽ sử dụng IC Schmitt-trigger 40106 nhằm debounce trường hợp nhiễu này. Bao gồm sáu mạch Schmitt-Trigger riêng biệt, Mỗi mạch hoạt động như cổng NOT với đầu vào Schmitt-Trigger. Mỗi Schmitt-Trigger thay đổi giữa mức cao và mức thấp khi thay đổi tín hiệu tại ngõ vào.



Hình 2.8: IC 40106 [4]

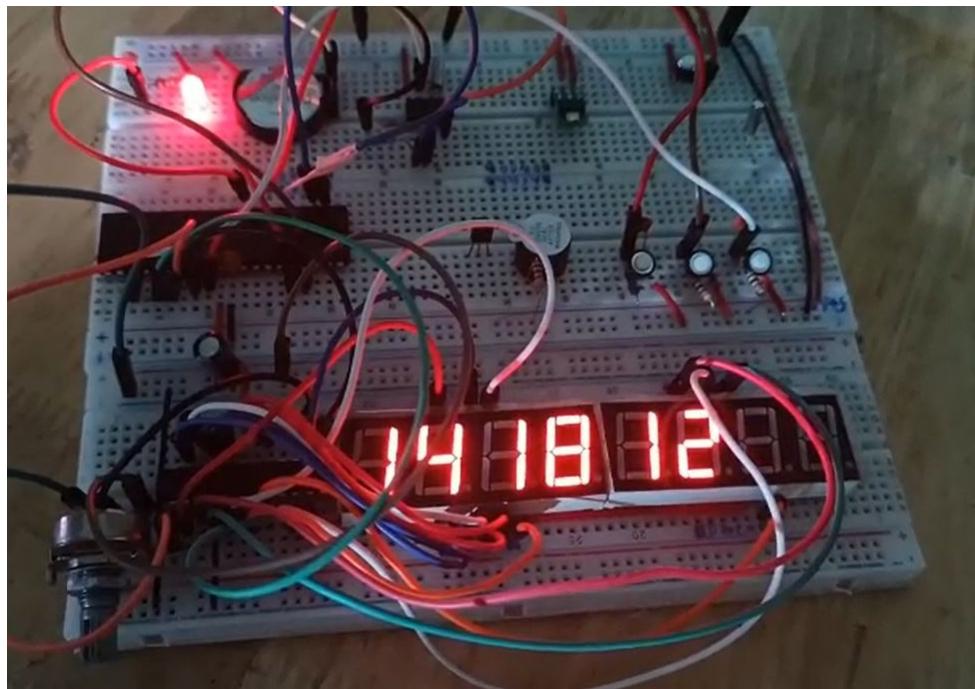


Hình 2.9: Dạng sóng nút nhấn sau khi debounce

Dựa vào hình 2.9, có thể thấy dạng sóng nút nhấn sau khi tác động (đường màu xanh) chưa đúng với lý thuyết, dẫn đến tình trạng bouncing, và sau khi đã qua Schmitt-trigger (đường màu vàng) khá đúng với lý thuyết.

## Chương 3

### Phương pháp thực hiện

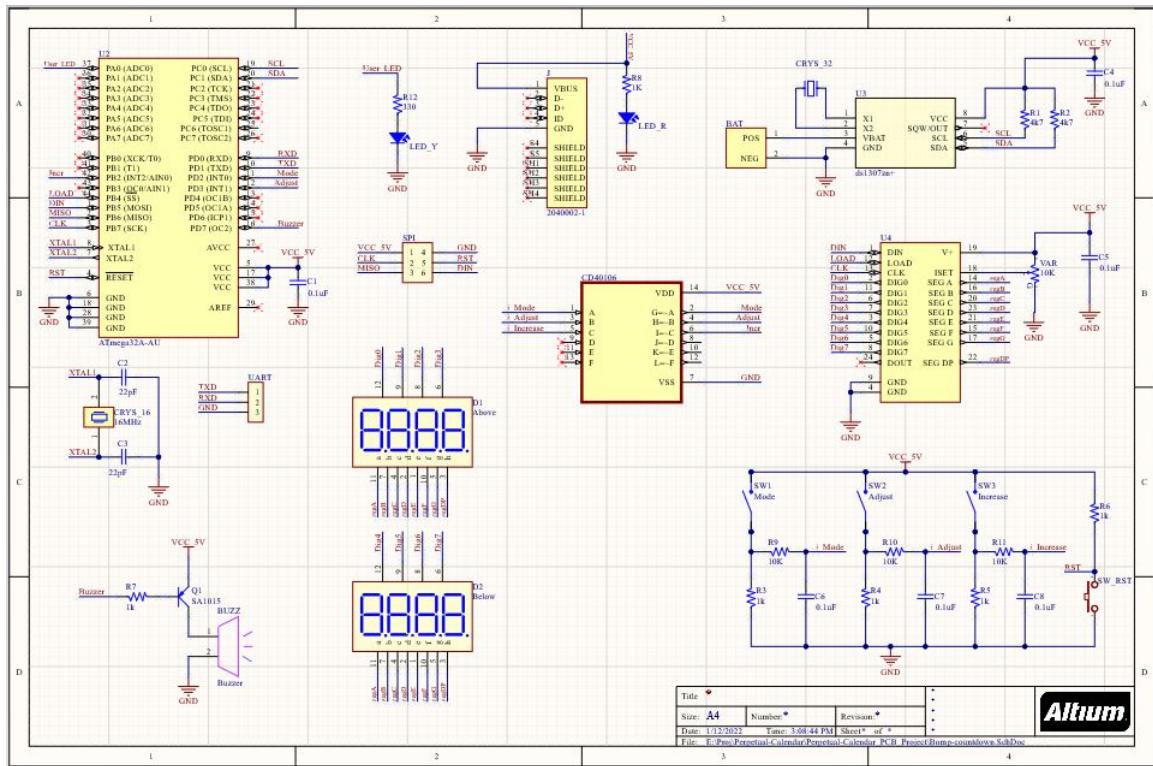


Hình 3.1: Quá trình cắm testboard

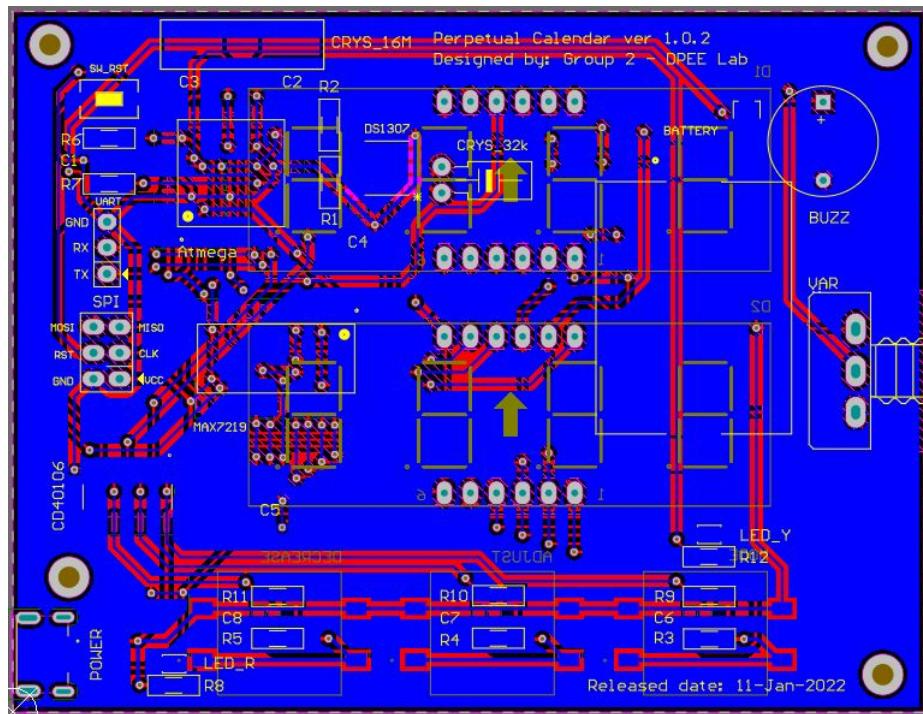
Quá trình cắm testboard của nhóm được thể hiện ở hình 3.1. Sau khi đảm bảo rằng nhiều từ các đường tín hiệu xuất phát từ testboard nên nhóm bắt tay vào và layout mạch, được thể hiện ở các hình 3.2 đến 3.5

Sơ đồ nguyên lý của hệ thống được thể hiện ở hình 3.2, sử dụng phần mềm altium để thực hiện.

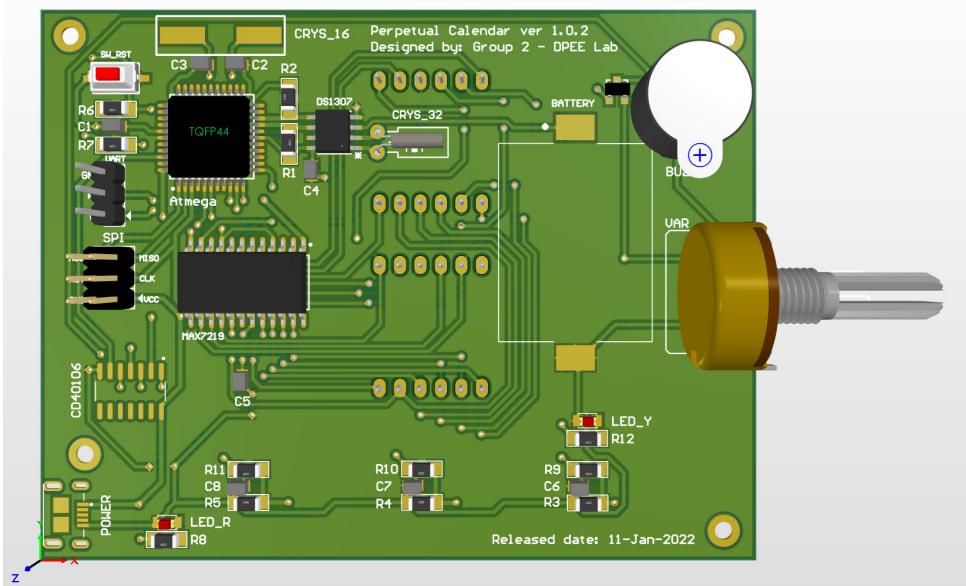
### CHƯƠNG 3. PHƯƠNG PHÁP THỰC HIỆN



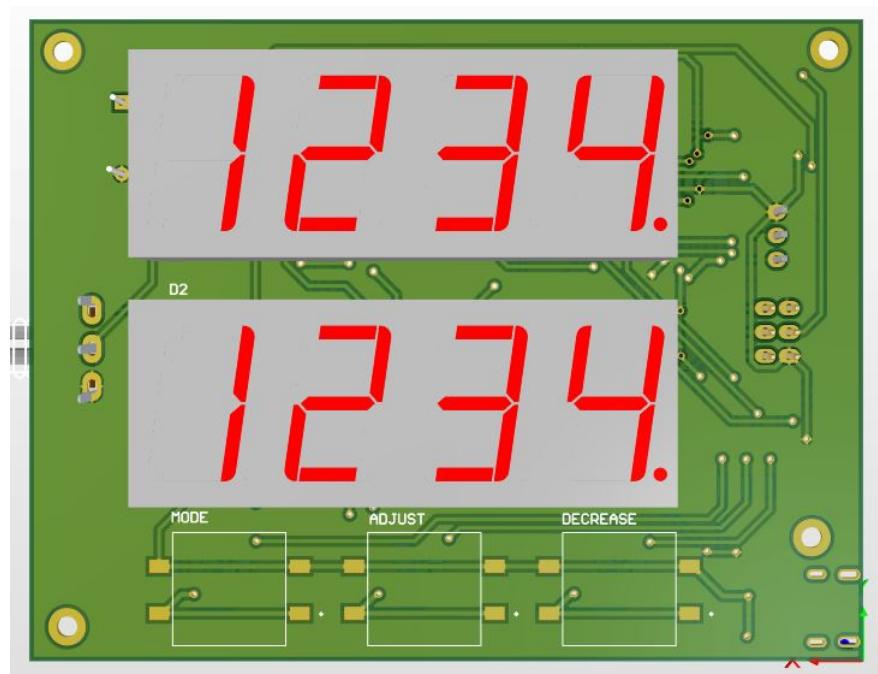
Hình 3.2: Schematic của hệ thống



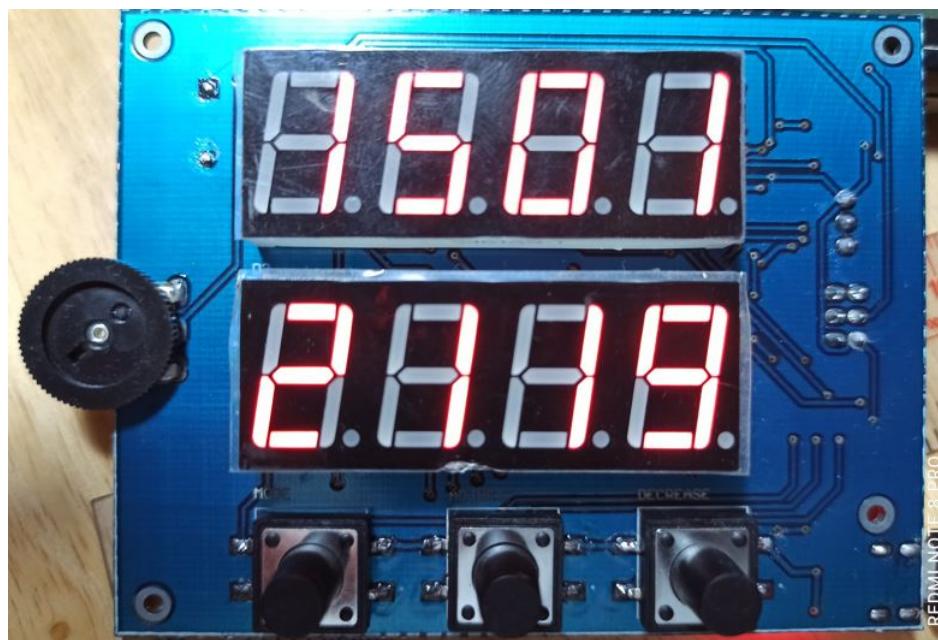
Hình 3.3: Mặt Bottom layout PCB



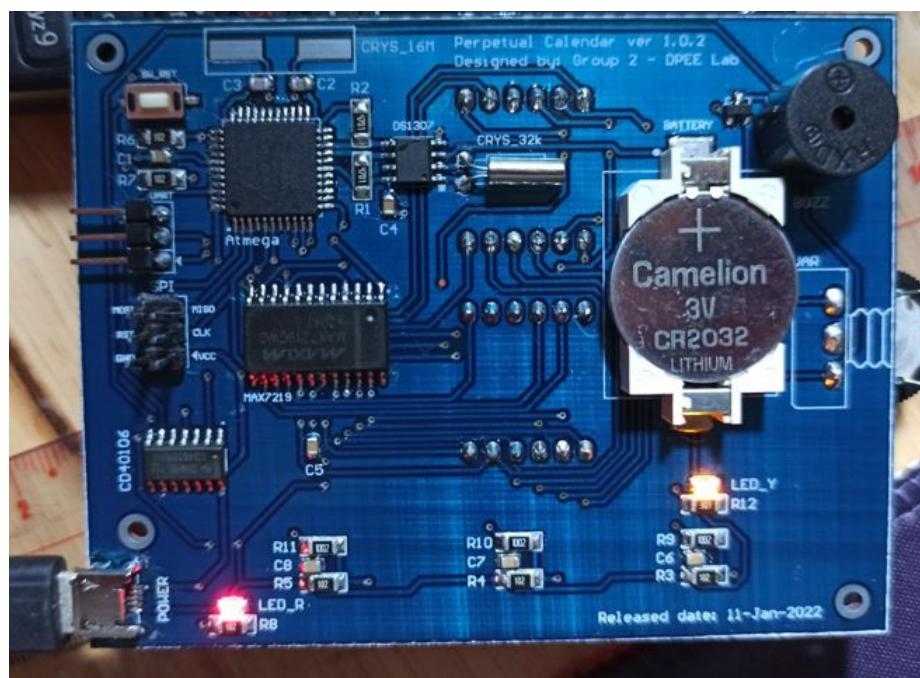
Hình 3.4: Mặt Top 3D



Hình 3.5: Mặt Bottom 3D



Hình 3.6: Hình ảnh thực tế mặt Top



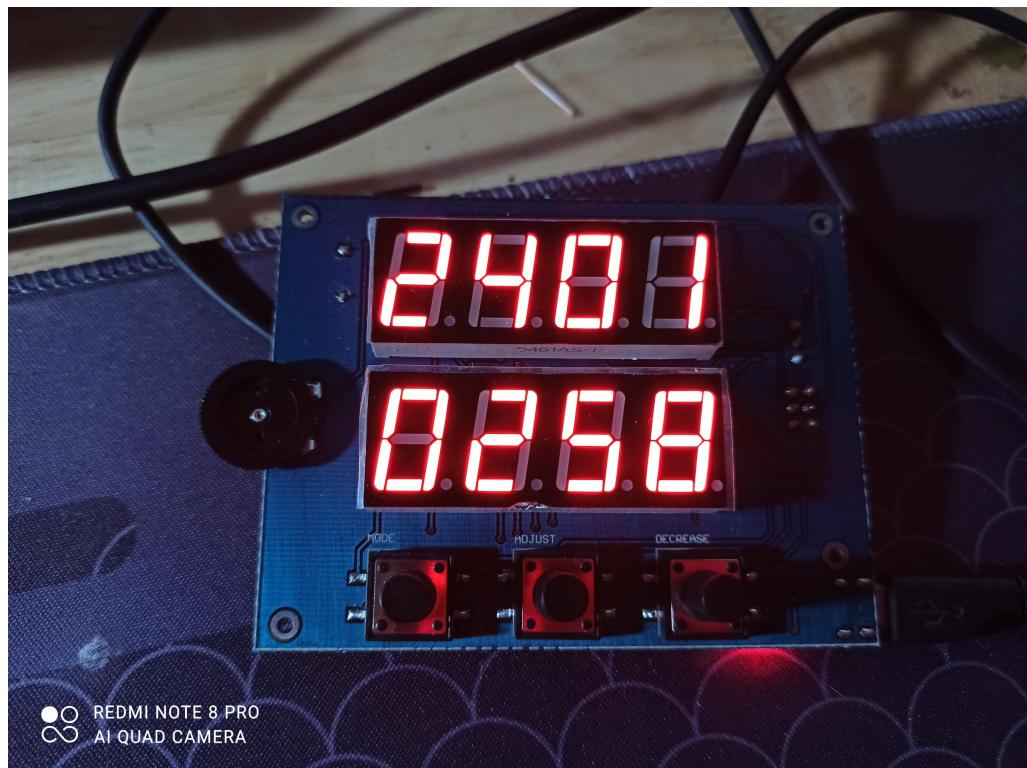
Hình 3.7: Hình ảnh thực tế mặt Bottom

# Chương 4

## Đánh giá và thử nghiệm

### 4.1 Chế độ hiển thị 1

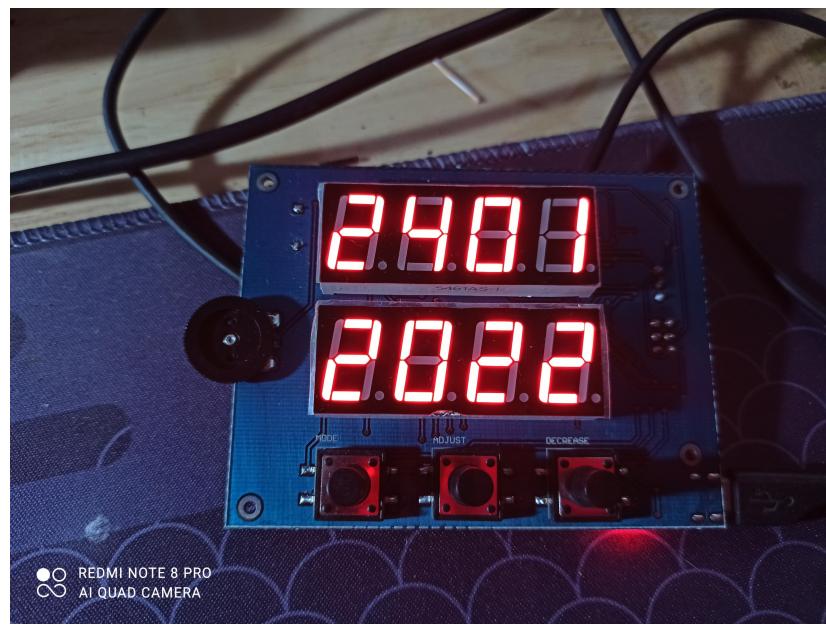
Ở chế độ này sẽ hiển thị lần lượt từ trái qua phải, từ trên xuống dưới là: Ngày, tháng (dương lịch) - giờ, phút.



Hình 4.1: Chế độ 1

## 4.2 Chế độ hiển thị 2

Ở chế độ này sẽ hiển thị lần lượt từ trái qua phải, từ trên xuống dưới là: Ngày, tháng, năm (dương lịch) .



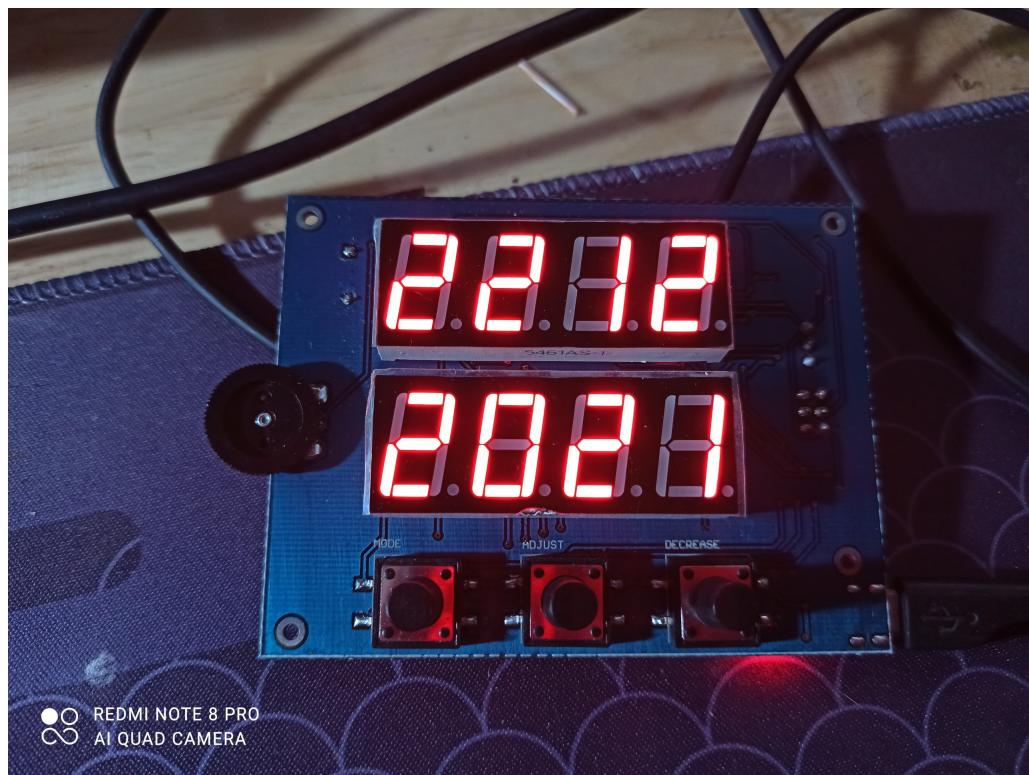
Hình 4.2: Chế độ 2

## 4.3 Chế độ hiển thị 3

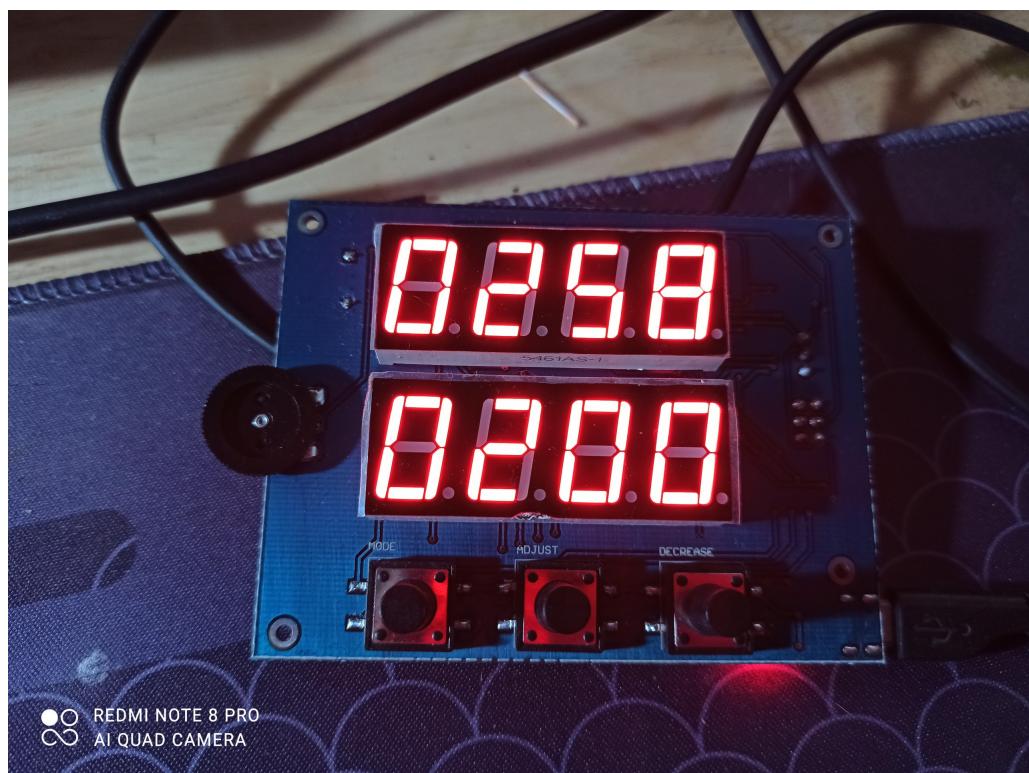
Ở chế độ này sẽ hiển thị lần lượt từ trái qua phải, từ trên xuống dưới là: Ngày, tháng, năm (âm lịch). Ở chế độ này sẽ không thể chỉnh sửa ngày tháng do lịch âm được tính toán từ lịch dương.(Hình 4.3)

## 4.4 Chế độ hiển thị 4

Ở chế độ này sẽ hiển thị lần lượt từ trái qua phải, từ trên xuống dưới là: Giờ, phút hiện tại - giờ, phút muôn cài báo thức. (Hình 4.4)



Hình 4.3: Chế độ 3



Hình 4.4: Chế độ 4

# Chương 5

## Kết luận và hướng phát triển

### 5.1 Đánh giá hệ thống

Hệ thống đã đáp ứng được các vấn đề đã đặt ra trước khi tiến hành làm:

- Đã hoàn thành được việc hiển thị ngày tháng năm dương lịch và âm lịch đúng với yêu cầu.
- Mạch in đáp ứng được các tiêu chí nhỏ, gọn, thích hợp làm đồng hồ để bàn.
- Đồng thời có thêm chức năng hẹn giờ báo thức.
- Các nút nhấn sau khi thử nghiệm qua schmitt-trigger đã ổn định hơn so với phiên bản thứ nhất.
- Có thể điều chỉnh độ sáng của LED tùy theo môi trường sử dụng.
- Có thể nạp lại code tùy thích do nhóm có layout thêm cổng nạp SPI và có thể debug thông qua cổng UART.

### 5.2 Các vấn đề còn tồn đọng

Hiện tại hệ thống vẫn bị vấn đề overflow, thời gian chạy ổn định theo thực tế vào khoảng 4 - 5 tiếng. Sau khoản thời gian đó, hệ thống không hiển thị được, phải ấn nút reset hoặc cấp lại nguồn.

Theo phán đoán của nhóm, lý do bị như vậy là do vấn đề về tràn bộ nhớ của vi điều khiển. Nhóm vẫn chưa khắc phục được tình trạng này.

### **5.3 Hướng phát triển của đề tài**

Đóng hộp sản phẩm lại để có thể đặt cố định ở một vị trí nào đó.

Tối ưu lại thuật toán hoặc có thể sử dụng vi điều khiển khác có bộ nhớ lớn hơn nhằm hạn chế vấn đề overflow.

Có thể thiết kế lại mạch nhỏ gọn hơn, nhằm mục đích thương mại.

# Tài liệu tham khảo

- [1] Thegioiic, “Atmega32u4-au,” Available at [https://www.thegioiic.com/products/atmega32u4-  
au](https://www.thegioiic.com/products/atmega32u4-au) (2022/01/23).
- [2] ——, “Ds1307z+ ic rtc clock/calendar 56b, 8-soic,” Available at <https://www.thegioiic.com/products/ds1307z-ic-rtc-clock-calendar-56b-8-soic> (2022/01/23).
- [3] ——, “Max7219cwg,” Available at <https://www.thegioiic.com/products/max7219cwg> (2022/01/23).
- [4] banlinhkien, “Cd40106 sop14,” Available at [https://banlinhkien.com/cd40106-  
sop14-p6648503.html](https://banlinhkien.com/cd40106-sop14-p6648503.html) (2022/01/23).
- [5] H. N. Đức, “Thuật toán tính âm lịch,” Available at [https://www.informatik.uni-  
leipzig.de/ duc/amlich/calrules.html](https://www.informatik.uni-leipzig.de/duc/amlich/calrules.html) (2022/01/23).

# Phụ lục A

```
/*
 * lich van nien -ver01
 *
 * Author : Vo Tan Minh Khoi
 Ngac Bao Nam
 Phung Thi Huong
 */

#define F_CPU 8000000UL
#include <avr/io.h>
#include <util/delay.h>
#include <avr/interrupt.h>
#include "myDS1307RTC.h"
#include <stdbool.h>
#include <math.h>

// Define buttons
#define BTN_DDRD DDRD
#define BTN_PORTD PORTD
#define SW 2
#define ADJ 3
```

```
#define BTN_DDRB DDRB
#define BTN_PORTB PORTB
#define INCR 2

#define DDR_LED_0 DDRA
#define PORT_LED_0 PORTA
#define BIT_LED_0 0

#define PORT_BUZZER_0 PORTD
#define DDR_BUZZER_0 DDRD
#define BIT_BUZZER_0 7

//-----MAX7219-----
#define PIN_SCK           PORTB7
#define PIN_MOSI          PORTB5
#define PIN_SS            PORTB4

#define ON                1
#define OFF               0

#define MAX7219_LOAD1    PORTB |= (1<<PIN_SS)
#define MAX7219_LOAD0    PORTB &= ~(1<<PIN_SS)

#define MAX7219_MODE_DECODE 0x09
#define MAX7219_MODE_INTENSITY 0x0A
#define MAX7219_MODE_SCAN_LIMIT 0x0B
#define MAX7219_MODE_POWER 0x0C
#define MAX7219_CHAR_BLANK 0xF

//****Define digits
```

```

#define MAX7219_DIGIT0          0x01
#define MAX7219_DIGIT1          0x02
#define MAX7219_DIGIT2          0x03
#define MAX7219_DIGIT3          0x04
#define MAX7219_DIGIT4          0x05
#define MAX7219_DIGIT5          0x06
#define MAX7219_DIGIT6          0x07
#define MAX7219_DIGIT7          0x08

#define digitsInUse    8

//-----
//Define variables
uint8_t Second = 00, Minute = 10, Hour = 14, Day = 4,
Date = 19, Month = 1, Year = 22, Mode = 0, AP = 1, A_Hour = 0, A_Minute = 0;
double timeZone = 7;
uint8_t lunarDate, lunarMonth;
uint16_t lunarYear, yyyy;
uint8_t tData[7]; //tData[7]: mang du lieu tam thoi
uint16_t Time_count = 0, blink_count=0;
//char digitsInUse = 8;
bool set = false; //set = true: cho phep dieu chinh thoi gian
bool EN_alarm = false; //EN_alarm = true: Set alarm
bool blinkmode = false;
uint8_t count = 0;
char SW_time_date = 0;

//****chuyen doi nhi phan sang thap phan****/
uint8_t BCDToDec(uint8_t BCD){

```

```
uint8_t L, H;
L=BCD & 0x0F;
H=(BCD>>4)*10;
return (H+L);
}

// chuyen doi thap phan sang nhi phan
uint8_t DecToBCD(uint8_t Dec)
{
    uint8_t L, H;
    L=Dec % 10;
    H=(Dec/10)<<4;
    return (H+L);
}

void Decode(void)
{
    //BCD data converter function from DS1307 to DEC
    Second = BCDToDec(tData[0] & 0x7F);
    Minute = BCDToDec(tData[1]);
    Hour = BCDToDec(tData[2] & 0x3F);
    Day = BCDToDec(tData[3]);
    Date = BCDToDec(tData[4]);
    Month = BCDToDec(tData[5]);
    Year = BCDToDec(tData[6]);
}

//Write to DS1307 time that want to change
void FixTime()
{
    tData[0] = DecToBCD(Second);
```

```
tData[1] = DecToBCD(Minute);
tData[2] = DecToBCD(Hour);
tData[3] = DecToBCD(Day);
tData[4] = DecToBCD(Date);
tData[5] = DecToBCD(Month);
tData[6] = DecToBCD(Year);
TWI_DS1307_wblock(0x00, tData, 7);
_delay_ms(1);
TWI_DS1307_wadr(0x00);
_delay_ms(1);
}

void spiSendByte (char databyte)
{
// Copy data into the SPI data register
SPDR = databyte;
// Wait until transfer is complete
while (!(SPSR & (1 << SPIF)));
}

void MAX7219_writeData(uint8_t data_register, uint8_t data)
{
MAX7219_LOAD0;
// Send the register where the data will be stored
spiSendByte(data_register);
// Send the data to be stored
spiSendByte(data);
MAX7219_LOAD1;
}
```

```
void MAX7219_clearDisplay()
{
    char i = digitsInUse;
    // Loop until 0, but don't run for zero
    do {
        // Set each display in use to blank
        MAX7219_writeData(i, MAX7219_CHAR_BLANK);
    } while (--i);
}

void Display_7seg (void)
{
    if (blinkmode==0)
    {
        /******display time -> hh:mm:ss******/
        if (SW_time_date == 0)
        {
            MAX7219_clearDisplay();

            MAX7219_writeData(MAX7219_DIGIT7, (Minute%10));
            MAX7219_writeData(MAX7219_DIGIT6, (Minute/10));
            MAX7219_writeData(MAX7219_DIGIT5, (Hour%10));
            MAX7219_writeData(MAX7219_DIGIT4, (Hour/10));
            MAX7219_writeData(MAX7219_DIGIT3, (Month%10));
            MAX7219_writeData(MAX7219_DIGIT2, (Month/10));
            MAX7219_writeData(MAX7219_DIGIT1, (Date%10));
            MAX7219_writeData(MAX7219_DIGIT0, (Date/10));

        }
    }
}
```

```
*****display date -> DD:MM:YY*****
else if (SW_time_date == 1)
{
    MAX7219_clearDisplay();

    MAX7219_writeData(MAX7219_DIGIT7,((yyyy%1000)%10));
    MAX7219_writeData(MAX7219_DIGIT6,(((yyyy%1000)/10)%10));
    MAX7219_writeData(MAX7219_DIGIT5,(((yyyy%1000)/100)%10));
    MAX7219_writeData(MAX7219_DIGIT4,(yyyy/1000));
    MAX7219_writeData(MAX7219_DIGIT3,(Month%10));
    MAX7219_writeData(MAX7219_DIGIT2,(Month/10));
    MAX7219_writeData(MAX7219_DIGIT1,(Date%10));
    MAX7219_writeData(MAX7219_DIGIT0,(Date/10));

}

*****display AM LICH *****
else if (SW_time_date==2)
{
    MAX7219_clearDisplay();

    MAX7219_writeData(MAX7219_DIGIT7,((lunarYear%1000)%10));
    MAX7219_writeData(MAX7219_DIGIT6,(((lunarYear%1000)/10)%10));
    MAX7219_writeData(MAX7219_DIGIT5,(((lunarYear%1000)/100)%10));
    MAX7219_writeData(MAX7219_DIGIT4,(lunarYear/1000));
    MAX7219_writeData(MAX7219_DIGIT3,(lunarMonth%10));
    MAX7219_writeData(MAX7219_DIGIT2,(lunarMonth/10));
    MAX7219_writeData(MAX7219_DIGIT1,(lunarDate%10));
    MAX7219_writeData(MAX7219_DIGIT0,(lunarDate/10));

}
```

```
else
{
    MAX7219_clearDisplay();

    MAX7219_writeData(MAX7219_DIGIT7, (A_Minute%10));
    MAX7219_writeData(MAX7219_DIGIT6, (A_Minute/10));
    MAX7219_writeData(MAX7219_DIGIT5, (A_Hour%10));
    MAX7219_writeData(MAX7219_DIGIT4, (A_Hour/10));
    MAX7219_writeData(MAX7219_DIGIT3, (Minute%10));
    MAX7219_writeData(MAX7219_DIGIT2, (Minute/10));
    MAX7219_writeData(MAX7219_DIGIT1, (Hour%10));
    MAX7219_writeData(MAX7219_DIGITO, (Hour/10));
}

}
else
{
    if ((count==1)&&(SW_time_date==0)) //blink date
    {
        MAX7219_clearDisplay();

        MAX7219_writeData(MAX7219_DIGIT7, (Minute%10));
        MAX7219_writeData(MAX7219_DIGIT6, (Minute/10));
        MAX7219_writeData(MAX7219_DIGIT5, (Hour%10));
        MAX7219_writeData(MAX7219_DIGIT4, (Hour/10));
        MAX7219_writeData(MAX7219_DIGIT3, (Month%10));
        MAX7219_writeData(MAX7219_DIGIT2, (Month/10));
        MAX7219_writeData(MAX7219_DIGIT1, MAX7219_CHAR_BLANK);
        MAX7219_writeData(MAX7219_DIGITO, MAX7219_CHAR_BLANK);
    }

    if ((count==2)&&(SW_time_date==0)) //blink month
```

```
{  
    MAX7219_clearDisplay();  
  
    MAX7219_writeData(MAX7219_DIGIT7, (Minute%10));  
    MAX7219_writeData(MAX7219_DIGIT6, (Minute/10));  
    MAX7219_writeData(MAX7219_DIGIT5, (Hour%10));  
    MAX7219_writeData(MAX7219_DIGIT4, (Hour/10));  
    MAX7219_writeData(MAX7219_DIGIT3, MAX7219_CHAR_BLANK);  
    MAX7219_writeData(MAX7219_DIGIT2, MAX7219_CHAR_BLANK);  
    MAX7219_writeData(MAX7219_DIGIT1, (Date%10));  
    MAX7219_writeData(MAX7219_DIGIT0, (Date/10));  
}  
  
if ((count==3)&&(SW_time_date==0)) //blink hour  
{  
    MAX7219_clearDisplay();  
  
    MAX7219_writeData(MAX7219_DIGIT7, (Minute%10));  
    MAX7219_writeData(MAX7219_DIGIT6, (Minute/10));  
    MAX7219_writeData(MAX7219_DIGIT5, MAX7219_CHAR_BLANK);  
    MAX7219_writeData(MAX7219_DIGIT4, MAX7219_CHAR_BLANK);  
    MAX7219_writeData(MAX7219_DIGIT3, (Month%10));  
    MAX7219_writeData(MAX7219_DIGIT2, (Month/10));  
    MAX7219_writeData(MAX7219_DIGIT1, (Date%10));  
    MAX7219_writeData(MAX7219_DIGIT0, (Date/10));  
}  
  
if ((count==4)&&(SW_time_date==0)) //blink min  
{  
    MAX7219_clearDisplay();  
  
    MAX7219_writeData(MAX7219_DIGIT7, MAX7219_CHAR_BLANK);
```

```
MAX7219_writeData(MAX7219_DIGIT6,MAX7219_CHAR_BLANK) ;  
MAX7219_writeData(MAX7219_DIGIT5,(Hour%10));  
MAX7219_writeData(MAX7219_DIGIT4,(Hour/10));  
MAX7219_writeData(MAX7219_DIGIT3,(Month%10));  
MAX7219_writeData(MAX7219_DIGIT2,(Month/10));  
MAX7219_writeData(MAX7219_DIGIT1,(Date%10));  
MAX7219_writeData(MAX7219_DIGIT0,(Date/10));  
}  
  
if ((count==1)&&(SW_time_date==1)) //blink date  
{  
    MAX7219_clearDisplay();  
  
    MAX7219_writeData(MAX7219_DIGIT7,((yyyy%1000)%10));  
    MAX7219_writeData(MAX7219_DIGIT6,(((yyyy%1000)/10)%10));  
    MAX7219_writeData(MAX7219_DIGIT5,(((yyyy%1000)/100)%10));  
    MAX7219_writeData(MAX7219_DIGIT4,(yyyy/1000));  
    MAX7219_writeData(MAX7219_DIGIT3,(Month%10));  
    MAX7219_writeData(MAX7219_DIGIT2,(Month/10));  
    MAX7219_writeData(MAX7219_DIGIT1,MAX7219_CHAR_BLANK);  
    MAX7219_writeData(MAX7219_DIGIT0,MAX7219_CHAR_BLANK);  
}  
  
if ((count==2)&&(SW_time_date==1)) //blink month  
{  
    MAX7219_clearDisplay();  
  
    MAX7219_writeData(MAX7219_DIGIT7,((yyyy%1000)%10));  
    MAX7219_writeData(MAX7219_DIGIT6,(((yyyy%1000)/10)%10));  
    MAX7219_writeData(MAX7219_DIGIT5,(((yyyy%1000)/100)%10));  
    MAX7219_writeData(MAX7219_DIGIT4,(yyyy/1000));  
    MAX7219_writeData(MAX7219_DIGIT3,MAX7219_CHAR_BLANK);
```

```
MAX7219_writeData(MAX7219_DIGIT2,MAX7219_CHAR_BLANK) ;  
MAX7219_writeData(MAX7219_DIGIT1,(Date%10));  
MAX7219_writeData(MAX7219_DIGIT0,(Date/10));  
}  
  
if ((count==3)&&(SW_time_date==1)) //blink year  
{  
MAX7219_clearDisplay();  
  
MAX7219_writeData(MAX7219_DIGIT7,MAX7219_CHAR_BLANK) ;  
MAX7219_writeData(MAX7219_DIGIT6,MAX7219_CHAR_BLANK) ;  
MAX7219_writeData(MAX7219_DIGIT5,MAX7219_CHAR_BLANK) ;  
MAX7219_writeData(MAX7219_DIGIT4,MAX7219_CHAR_BLANK) ;  
MAX7219_writeData(MAX7219_DIGIT3,(Month%10));  
MAX7219_writeData(MAX7219_DIGIT2,(Month/10));  
MAX7219_writeData(MAX7219_DIGIT1,(Date%10));  
MAX7219_writeData(MAX7219_DIGIT0,(Date/10));  
}  
  
if ((count==1)&&(SW_time_date==3)) //blink A_HOUR  
{  
MAX7219_clearDisplay();  
  
MAX7219_writeData(MAX7219_DIGIT7,(A_Minute%10));  
MAX7219_writeData(MAX7219_DIGIT6,(A_Minute/10));  
MAX7219_writeData(MAX7219_DIGIT5,MAX7219_CHAR_BLANK) ;  
MAX7219_writeData(MAX7219_DIGIT4,MAX7219_CHAR_BLANK) ;  
MAX7219_writeData(MAX7219_DIGIT3,(Minute%10));  
MAX7219_writeData(MAX7219_DIGIT2,(Minute/10));  
MAX7219_writeData(MAX7219_DIGIT1,(Hour%10));  
MAX7219_writeData(MAX7219_DIGIT0,(Hour/10));
```

```

}

if ((count==2)&&(SW_time_date==3)) //blink A_MIN
{
    MAX7219_clearDisplay();

    MAX7219_writeData(MAX7219_DIGIT7,MAX7219_CHAR_BLANK);
    MAX7219_writeData(MAX7219_DIGIT6,MAX7219_CHAR_BLANK);
    MAX7219_writeData(MAX7219_DIGIT5,(A_Hour%10));
    MAX7219_writeData(MAX7219_DIGIT4,(A_Hour/10));
    MAX7219_writeData(MAX7219_DIGIT3,(Minute%10));
    MAX7219_writeData(MAX7219_DIGIT2,(Minute/10));
    MAX7219_writeData(MAX7219_DIGIT1,(Hour%10));
    MAX7219_writeData(MAX7219_DIGIT0,(Hour/10));
}

}

//-----DOI DUONG LICH - AM LICH-----

double jdFromDate(uint8_t dd, uint8_t mm, uint16_t yy)
{
    long double a, y, m, jd;
    a = floorf((14 - mm) / 12);
    y = yy+4800-a;
    m = mm+12*a-3;
    jd = dd + floorf((153*m+2)/5) + 365*y + floorf(y/4) - floorf(y/100) + floorf(y/400);
    if (jd < 2299161) {
        jd = dd + floorf((153*m+2)/5) + 365*y + floorf(y/4) - 32083;
    }
}

```

```

}

return jd;
}

double getNewMoonDay(long double k)
{
    long double T, T2, T3, dr, Jd1, M, Mpr, F, C1, deltat, JdNew;
    T = k/1236.85; // Time in Julian centuries from 1900 January 0.5
    T2 = T * T;
    T3 = T2 * T;
    dr = M_PI/180;
    Jd1 = 2415020.75933 + 29.53058868*k + 0.0001178*T2 - 0.000000155*T3;
    Jd1 = Jd1 + 0.00033*sinf((166.56 + 132.87*T - 0.009173*T2)*dr); // Mean new moon
    M = 359.2242 + 29.10535608*k - 0.0000333*T2 - 0.00000347*T3; // Sun's mean anomaly
    Mpr = 306.0253 + 385.81691806*k + 0.0107306*T2 + 0.00001236*T3; // Moon's mean anomaly
    F = 21.2964 + 390.67050646*k - 0.0016528*T2 - 0.00000239*T3; // Moon's argument of perigee
    C1=(0.1734 - 0.000393*T)*sinf(M*dr) + 0.0021*sinf(2*dr*M);
    C1 = C1 - 0.4068*sinf(Mpr*dr) + 0.0161*sinf(dr*2*Mpr);
    C1 = C1 - 0.0004*sinf(dr*3*Mpr);
    C1 = C1 + 0.0104*sinf(dr*2*F) - 0.0051*sinf(dr*(M+Mpr));
    C1 = C1 - 0.0074*sinf(dr*(M-Mpr)) + 0.0004*sinf(dr*(2*F+M));
    C1 = C1 - 0.0004*sinf(dr*(2*F-M)) - 0.0006*sinf(dr*(2*F+Mpr));
    C1 = C1 + 0.0010*sinf(dr*(2*F-Mpr)) + 0.0005*sinf(dr*(2*Mpr+M));
    if (T < -11) {
        deltat= 0.001 + 0.000839*T + 0.0002261*T2 - 0.00000845*T3 - 0.000000081*T*T3;
    } else {
        deltat= -0.000278 + 0.000265*T + 0.000262*T2;
    };
    JdNew = Jd1 + C1 - deltat;
    return floorf(JdNew + 0.5 + timeZone/24);
}

```

---

 }

```

double getSunLongitude(long double jdn)
{
  long double T, T2, dr, M, L0, DL, L;
  T = (jdn - 2451545.5 - timeZone/24) / 36525; // Time in Julian centuries from 2000
  T2 = T*T;
  dr = M_PI/180; // degree to radian
  M = 357.52910 + 35999.05030*T - 0.0001559*T2 - 0.00000048*T*T2; // mean anomaly,
  L0 = 280.46645 + 36000.76983*T + 0.0003032*T2; // mean longitude, degree
  DL = (1.914600 - 0.004817*T - 0.000014*T2)*sinf(dr*M);
  DL = DL + (0.019993 - 0.000101*T)*sinf(dr*2*M) + 0.000290*sinf(dr*3*M);
  L = L0 + DL; // true longitude, degree
  L = L*dr;
  L = L - M_PI*2*(floorf(L/(M_PI*2))); // Normalize to (0, 2*PI)
  return floorf(L / M_PI * 6);
}

```

```

double getLunarMonth11(uint16_t yy)
{
  long double k, off, nm, sunLong;
  off = jdFromDate(31, 12, yy) - 2415021;
  k = floorf(off / 29.530588853);
  nm = getNewMoonDay(k);
  sunLong = getSunLongitude(nm); // sun longitude at local midnight
  if (sunLong >= 9) {
    nm = getNewMoonDay(k-1);
  }
  return nm;
}

```

```

double getLeapMonthOffset(long double a11)
{
    long double k, last, arc, i;
    k = floorf((a11 - 2415021.076998695) / 29.530588853 + 0.5);
    last = 0;
    i = 1; // We start with the month following lunar month 11
    arc = getSunLongitude(getNewMoonDay(k+i));
    do {
        last = arc;
        i++;
        arc = getSunLongitude(getNewMoonDay(k+i));
    } while (arc != last && i < 14);
    return i-1;
}

double convertSolar2Lunar(uint8_t dd, uint8_t mm, uint16_t yy)
{
    long double k, dayNumber, monthStart, a11, b11, diff, leapMonthDiff;
    dayNumber = jdFromDate(dd, mm, yy);
    k = floorf((dayNumber - 2415021.076998695) / 29.530588853);
    monthStart = getNewMoonDay(k+1);
    if (monthStart > dayNumber) {
        monthStart = getNewMoonDay(k);
    }
    a11 = getLunarMonth11(yy);
    b11 = a11;
    if (a11 >= monthStart) {
        lunarYear = yy;
        a11 = getLunarMonth11(yy-1);
    }
}

```

```

} else {
lunarYear = yy+1;
b11 = getLunarMonth11(yy+1);
}

lunarDate = dayNumber-monthStart+1;
diff = floorf((monthStart - a11)/29);
lunarMonth = diff+11;
if (b11 - a11 > 365) {
leapMonthDiff = getLeapMonthOffset(a11);
if (diff >= leapMonthDiff) {
lunarMonth = diff + 10;
}
}

if (lunarMonth > 12) {
lunarMonth = lunarMonth - 12;
}

if (lunarMonth >= 11 && diff < 4) {
lunarYear -= 1;
}

return 0;
}

//-----
void Init_Timer0(void){
//Initialize Timer0 to 1s - overflow interrupt-----
TCCR0=(1<<CS02)|(0<<CS01)|(1<<CS00); //prescaler, clk/1024

TIMSK=(1<<TOIE0);

sei();
}

```

```
void Init_buttons(void)
{
//-----Initialize button-----
//-----Set input for setting buttons-----
BTN_DD RD &= ~((1<<SW) | (1<<ADJ));
BTN_DD RB &= ~(1<<INCR);
}

void Init_IO(void)
{
DDR_LED_0 |=(1<<BIT_LED_0);
DDR_BUZZER_0 |= (1<<BIT_BUZZER_0);
PORT_BUZZER_0 |= (1<<BIT_BUZZER_0);
}

void Init_interrupt(void)
{
MCUCR = (1<<ISC11)|(1<<ISC10)|(1<<ISC01)|(1<<ISC00);
MCUCSR = (1<<ISC2);
GICR =
(1<<INT2)|(1<<INT1)|(1<<INT0);
sei();
}

void Init_start_cond()
{
PORT_LED_0 |= (1<<BIT_LED_0);
_delay_ms(500);
PORT_LED_0 &= ~(1<<BIT_LED_0);
}
```

```
_delay_ms(500);
PORT_LED_0 |= (1<<BIT_LED_0);
_delay_ms(500);
PORT_LED_0 &= ~(1<<BIT_LED_0);
_delay_ms(500);
PORT_LED_0 |= (1<<BIT_LED_0);
_delay_ms(500);
PORT_LED_0 &= ~(1<<BIT_LED_0);
_delay_ms(500);
}
}

void set_Alarm(void)
{
PORT_BUZZER_0 &= ~(1<<BIT_BUZZER_0);
_delay_ms(70);
PORT_BUZZER_0 |= (1<<BIT_BUZZER_0);
_delay_ms(50);
PORT_BUZZER_0 &= ~(1<<BIT_BUZZER_0);
_delay_ms(70);
PORT_BUZZER_0 |= (1<<BIT_BUZZER_0);
_delay_ms(1000);
}

//Main program
int main(void)
{
//MAX7219 init
// SCK MOSI CS/LOAD/SS
DDRB |= (1 << PIN_SCK) | (1 << PIN_MOSI) | (1 << PIN_SS);
// SPI Enable, Master mode
```

```
SPCR |= (1 << SPE) | (1 << MSTR)| (1<<SPR1);
// Scan limit runs from 0.

MAX7219_writeData(MAX7219_MODE_SCAN_LIMIT, 0x07);
MAX7219_writeData(MAX7219_MODE_INTENSITY, 0x0F);
MAX7219_writeData(MAX7219_MODE_POWER, ON);
MAX7219_writeData(MAX7219_MODE_DECODE, 0xFF);

//FixTime();

Init_buttons();
Init_IO();
Init_Timer0();
Init_interrupt();
sei();
TWI_Init();
TWI_DS1307_rblock(tData,7);
Decode(); //BCD data converter function from DS1307 to DEC
_delay_ms(1);
Init_start_cond(); /**BLINK 3 SECOND BEFORE LOOPING**/

while(1)
{
    yyyy=Year+2000;
    convertSolar2Lunar(Date, Month, yyyy);
    Display_7seg();
    if (Hour == A_Hour && Minute == A_Minute && EN_alarm == true)
    {
        Display_7seg();
        set_Alarm();
    }
}

return 0;
```

```
}

ISR(TIMER0_OVF_vect){
    Time_count++; //thoi gian doc ds1307
    blink_count++; //thoi gian blink
    if(Time_count>=10){ //1s Exactly

        if(set == false )
        {
            //Read DS1307
            TWI_DS1307_wadr(0x00);
            _delay_ms(1);
            TWI_DS1307_rblock(tData,7);

            //Print result on 7Seg led
            if(BCDToDec(tData[0]) !=Second)
            {
                Decode();
                Display_7seg();
            }
        }

        Time_count=0;
    }

    if (blink_count>=15) //blink 500ms
    {
        if(set == true ){
            blinkmode^=1;
            Display_7seg();
        }
        blink_count=0;
    }
}
```

```
}

}

//SW mode button
ISR(INT0_vect){

    if(set==false){
        SW_time_date++;
        if(SW_time_date > 3){
            SW_time_date = 0;
        }
    }

    if(SW_time_date==0 && set==true) {
        SW_time_date = 0;
        count=0;
        blinkmode=0;
        FixTime();
        set=false;
    }

    if(SW_time_date==1 && set==true) {
        SW_time_date = 1;
        count=0;
        blinkmode=0;
        FixTime();
        set=false;
    }

    if(SW_time_date==3 && set==true) {
        SW_time_date = 3;
        count=0;
```

```
blinkmode=0;
EN_alarm=true;
set=false;
}

}

//Set time button
ISR(INT1_vect){

if (SW_time_date==0)
{
set = true;
count++;
if(count > 4) {
count = 0;
blinkmode=0;
set=false;
}
}

if (SW_time_date==1)
{
set = true;
count++;
if(count > 3) {
count = 0;
blinkmode=0;
set=false;
}
}
```

```
if (SW_time_date==3)
{
    set = true;
    count++;
    if(count > 2) {
        count = 0;
        blinkmode=0;
        EN_alarm=false;
        set=false;
    }
}
}

//increase button
ISR(INT2_vect){
    if (EN_alarm == true && set==false)
    {
        EN_alarm=false;
        BTN_PORTE |= (1<<BIT_BUZZER_0);
    }
    if((set == true) && (SW_time_date==0)){ //increase dd, mm, h, min
        if(count == 1) {
            Date++;
            if(Month == 4 || Month == 6 || Month == 9 || Month == 11)
            {
                if(Date > 30)
                    Date=1;
            }
            else if(Month == 1 || Month == 3 || Month == 5 || Month == 7 || Month == 8 || Month == 10)
            {
                if(Date > 31)
                    Date=1;
            }
        }
    }
}
```

```
if(Date >31)
Date=1;
}

else if(yyyy/4 == 0 && yyyy/400 == 0)
{
if(Date > 29)
Date=1;
}
else
{
if(Date > 28)
Date=1;
}
}

if(count == 2)
{
Month++;
if(Month > 12) Month = 1;
}
if(count == 3)
{
Hour++;
if(Hour > 23) Hour = 0;
}
if(count == 4)
{
Minute++;
if(Minute > 59) Minute = 0;
}
```

```
}

if((set == true) && (SW_time_date==1)) //increase dd, mm, yyyy
{
    if(count == 1)
    {
        Date++;
        if(Month == 4 || Month == 6 || Month == 9 || Month == 11)
        {
            if(Date > 30)
                Date=1;
        }
        else if(Month == 1 || Month == 3 || Month == 5 || Month == 7 || Month == 8 || Month == 10 || Month == 12)
        {
            if(Date >31)
                Date=1;
        }
        else if(yyyy/4 == 0 && yyyy/400 == 0)
        {
            if(Date > 29)
                Date=1;
        }
        else
        {
            if(Date > 28)
                Date=1;
        }
    }
    if(count == 2)
```

```
{  
Month++;  
if(Month > 12) Month = 1;  
}  
if(count == 3)  
{  
Year++;  
if(Year > 99) Year = 0;  
}  
}  
  
if((set == true) && (SW_time_date==3)) //increase alarm  
{  
  
if(count == 1) {  
A_Hour++;  
if(A_Hour > 23) A_Hour = 0;  
}  
if(count == 2) {  
A_Minute++;  
if(A_Minute > 59) A_Minute = 0;  
}  
}  
}
```