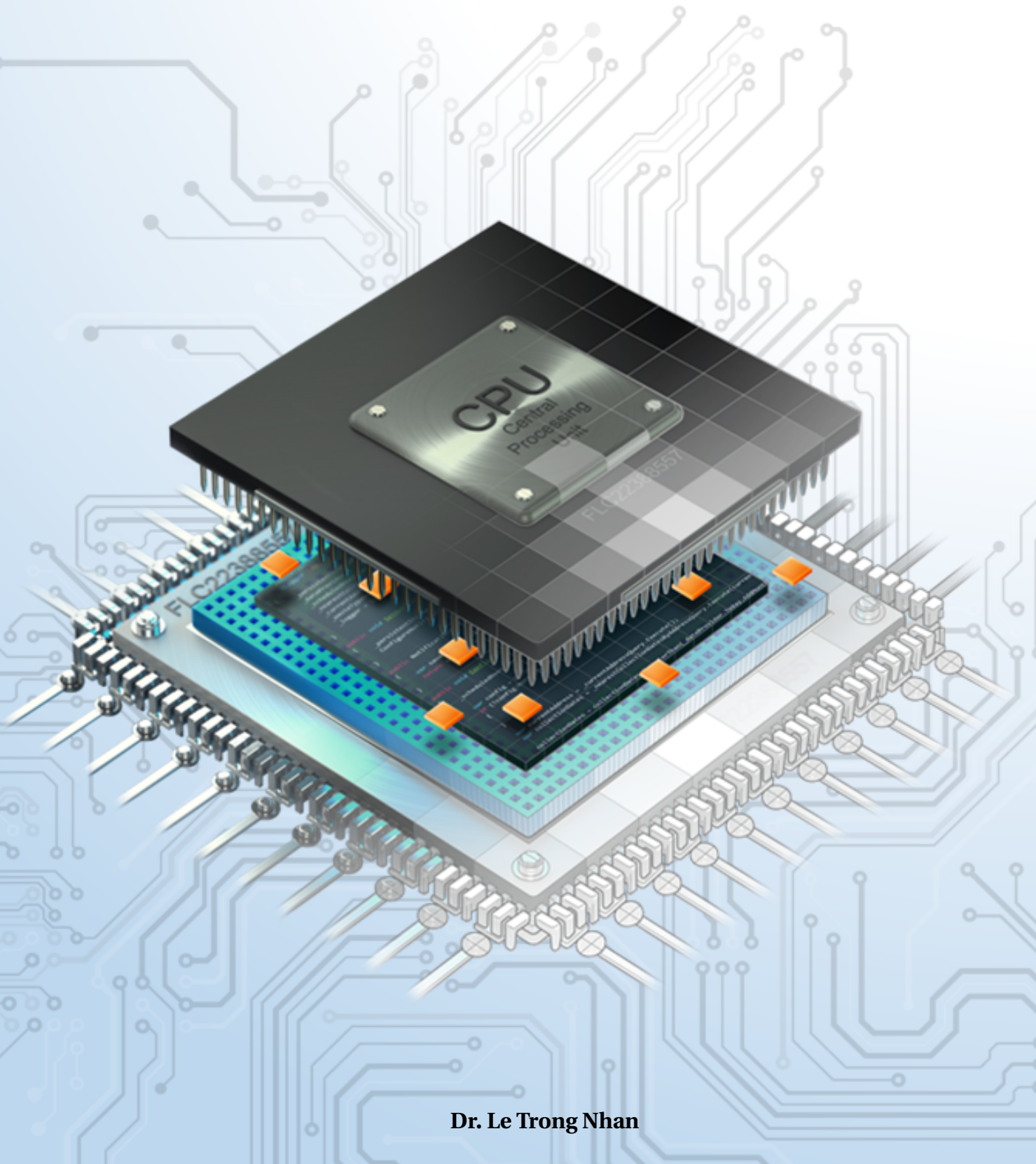




HO CHI MINH CITY UNIVERSITY OF TECHNOLOGY
COMPUTER ENGINEERING

Microcontroller



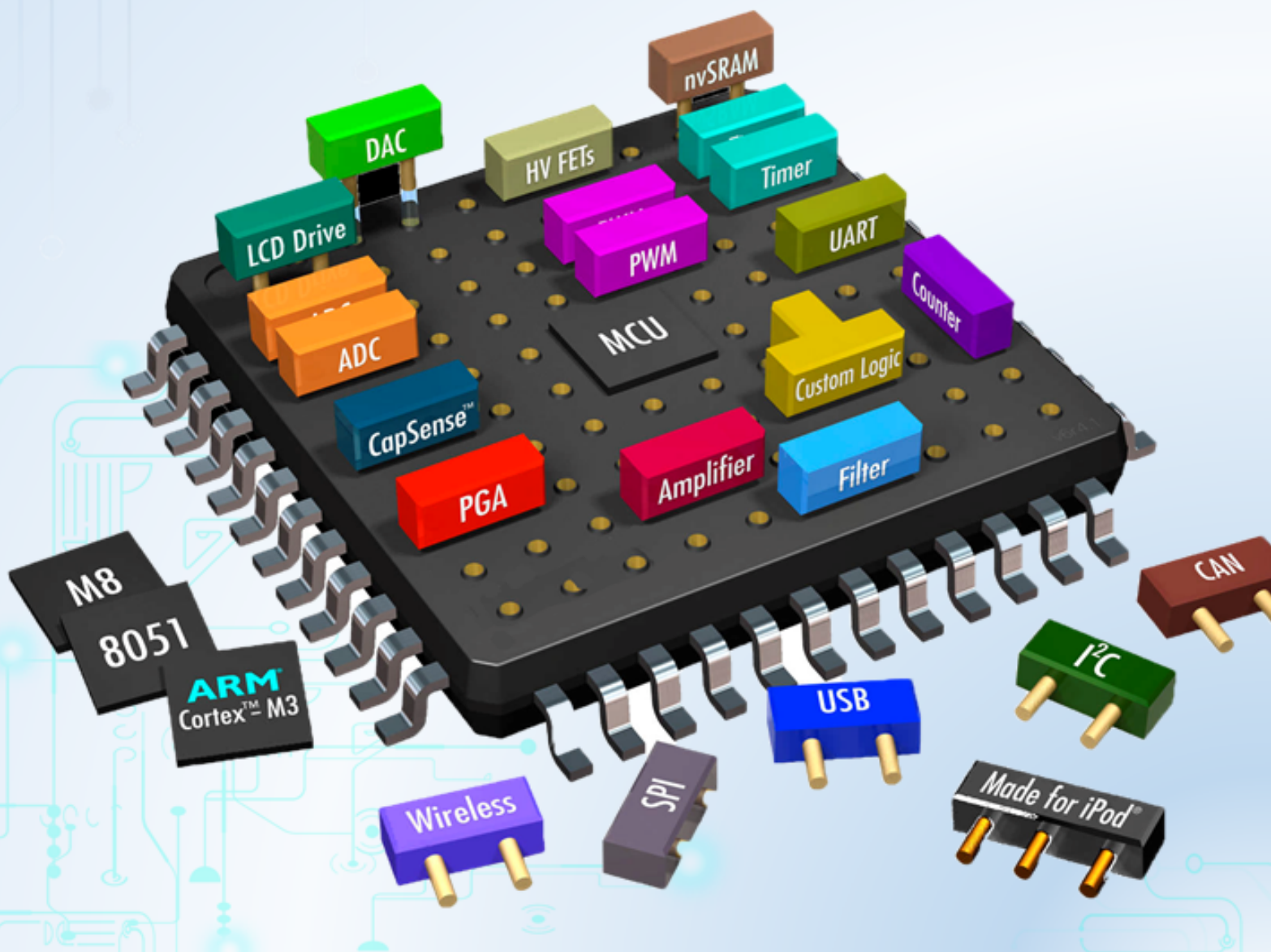
Dr. Le Trong Nhan

Mục lục

Chapter 1. Bài Tập Giữa Kỳ	7
1 Giới thiệu	8
2 Nộp bài	8
3 Report	9
3.1 Mô phỏng trên Proteus	9
3.2 Thiết kế máy trạng thái	9
3.3 Lập trình trên STM32Cube	9
3.4 Module Timer	10
3.4.1 File timer.h	10
3.4.2 File timer.c	10
4 Video demo	11

CHƯƠNG 1

Bài Tập Giữa Kỳ



1 Giới thiệu

Trong yêu cầu của bài giữa kì, một đồng hồ analog với 12 bóng đèn hiển thị, trưng cho 12 số trên đồng hồ. Bên cạnh đó, sẽ có 3 nút nhấn: nút MENU, INC và DEC (để tăng và giảm thông tin)

Để tiện lợi trong quá trình demo, đồng hồ sẽ có 12 giờ. Tuy nhiên, mỗi phút sẽ chỉ có 12 giây, và mỗi giờ cũng sẽ có 12 phút.

Đồng hồ có 3 chế độ hoạt động như sau:

- Mode 0: Đây là chế độ mặt định mỗi khi bật nguồn hoặc khởi động lại hệ thống. Thông tin giờ phút và giây sẽ được hiển thị trên màn hình. Tại một thời điểm, chỉ có tối đa 3 đèn được hiển thị. Nếu 2 thông tin (hoặc 3 thông tin) trùng nhau, thì chỉ 1 đèn sẽ được hiển thị. Khi đang ở chế độ này, thông tin về giờ phút giây sẽ được cập nhật theo đúng quy luật của đồng hồ.
- Mode 1: Khi đang ở Mode 0 và nhấn vào nút MENU, đồng hồ sẽ chuyển sang **chế độ này để chỉnh giờ**. Thông tin về giờ phút giây sẽ ngừng cập nhật để người dùng chỉnh giờ. Chỉ một thông tin về giờ hiện tại sẽ được hiển thị. Khi nhấn nút INC và DEC, thông tin giờ sẽ được cộng thêm, hoặc trừ đi. Đồng thời, đèn hiển thị cũng sẽ được cập nhật trên mặt của đồng hồ. Lưu ý rằng, khi đang ở vị trí số 1, và nhấn nút trừ (DEC), thì thông tin giờ sẽ đếm vòng sang 12. Mỗi lần nhấn nút INC hoặc DEC, thông tin mới về giờ sẽ được lưu lại ngay lập tức.
- Mode 2: Khi đang ở Mode 1 và nhấn vào nút MENU, đồng hồ sẽ chuyển sang **chế độ chỉnh phút**. Chỉ một thông tin về phút hiện tại sẽ được hiển thị trên màn hình. Việc chỉnh phút cũng được thực hiện tương tự như chỉnh giờ.

Khi đang ở Mode 1 hoặc Mode 2, người dùng không tương tác vào bất cứ nút nào (MENU, INC hay DEC) trong vòng 5 giây, hệ thống tự động chuyển về Mode 0.

Hệ thống sẽ được hiện thực trên phần mềm mô phỏng Proteus. Vi điều khiển STM32103C6 sẽ được sử dụng. Sinh viên không bắt buộc phải sử dụng ngắt timer. Nút nhấn chỉ tích cực mỗi khi nhấn xuống và **không kiểm tra trường hợp nhấn đè**. Ví dụ đang ở trạng thái chỉnh giờ, nút INC được nhấn xuống, giờ ngay lập tức sẽ được tăng lên 1 đơn vị. Tuy nhiên nếu cứ giữ đè, thì giờ vẫn không tăng. Nút nhấn được chống rung bằng cách đọc 2 lần liên tiếp giống nhau, mỗi lần cách nhau 10ms.

2 Nộp bài

Sinh viên hoàn thiện file report này, với các nội dung được yêu cầu bên dưới, file main.c, tất cả các file .h và .c sinh viên hiện thực thêm sẽ được nén lại với MSSV và nộp lên hệ thống.

Sinh viên quay màn hình phần demo trên Proteus và tải lên Drive của mình (ở chế độ public). Link của video demo sẽ được trình bày ở phần cuối của Report.

3 Report

3.1 Mô phỏng trên Proteus

Thiết kế sơ đồ mạch điện trên Proteus, bao gồm 12 chân đèn LED và 3 nút nhấn. Để đơn giản, sinh viên có thể lược bỏ điện trở hạn dòng cho đèn LED.

Sinh viên chụp hình màn hình mô phỏng Proteus và dán vào phần report này.

3.2 Thiết kế máy trạng thái

Sinh viên thiết kế máy trạng thái cho hệ thống và trình bày ở phần này. Sinh viên được khuyến khích giải thích thêm cho máy trạng thái của mình để tiện lợi cho giảng viên khi đánh giá.

3.3 Lập trình trên STM32Cube

Trong trường hợp không sử dụng ngắt timer, cấu trúc chương trình sẽ như sau:

```
1 int main(void)
2 {
3     HAL_Init();
4     SystemClock_Config();
5
6     MX_GPIO_Init();
7
8     while (1){
9         clock_fsm();
10        timer_run();
11        button_scan();
12        HAL_Delay(10);
13    }
14 }
```

Program 1.1: Cấu trúc chương trình trên main

Sinh viên sửa lại các hàm ở trên cho phù hợp với việc hiện thực của mình, chỉ giữ lại hàm HAL_Delay(10) ở cuối cùng. Trong trường hợp sử dụng ngắt, sinh viên cấu hình là ngắt 10ms, không sử dụng HAL_Delay và timer_run, button_scan sẽ đặt trong hàm ngắt.

Từ phần này, sinh viên trình bày mã nguồn của từng module hiện thực, chẳng hạn như software timer (hàm chính sử dụng trong main là timer_run), hàm đọc giá trị từ 3 nút nhấn (hàm chính sử dụng trong main là button_scan).

Quan trọng nhất là mã nguồn cho phần xử lý máy trạng thái của đồng hồ. Một ví dụ cho việc trình bày mã nguồn mở, bao gồm file timer.h và timer.c

3.4 Module Timer

Đặc tả ngắn gọn về module này

3.4.1 File timer.h

```
1 #ifndef _TIMER_H_
2 #define _TIMER_H_
3
4 extern int timer0_flag;
5
6
7 void setTimer0(int duration);
8 void timer_run();
9
10 #endif
```

Program 1.2: Mã nguồn file .h

3.4.2 File timer.c

```
1 #include "timer.h"
2
3 int timer0_counter = 0;
4 int timer0_flag = 0;
5 int TIMER_CYCLE = 10;
6
7 void setTimer0(int duration){
8     timer0_counter = duration / TIMER_CYCLE;
9     timer0_flag = 0;
10 }
11
12 void timer_run(){
13     if(timer0_counter > 0){
14         timer0_counter --;
15         if(timer0_counter <= 0) timer0_flag = 1;
16     }
17 }
```

Program 1.3: Mã nguồn file .c

4 Video demo

Sinh viên cần quay 1 đoạn demo ngắn trên Proteus để minh họa việc **chỉnh giờ, dừng 5 giây không chỉnh phút** để thoát ra chế độ hoạt động bình thường. Sau đó **nhấn vào MENU để chỉnh giờ, rồi nhấn tiếp MENU để chỉnh phút. Sau khi chỉnh phút xong (không tương tác trong 5s)**, hệ thống sẽ quay lại chế độ hoạt động bình thường.

Đường link cho video demo như sau:

<http://>