

**ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH  
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN  
KHOA KỸ THUẬT MÁY TÍNH**



**BÁO CÁO ĐỒ ÁN  
MÔN: VI XỬ LÝ - VI ĐIỀU KHIỂN  
ĐỀ TÀI: MẠCH ĐIỀU KHIỂN ĐÈN GIAO THÔNG**

GVHD: Trần Hoàng Lộc

MÃ LỚP: CE103.024 - NHÓM 13:  
Châu Bình Thanh\_21521435  
Nguyễn Đình Quốc\_22521213  
Huỳnh Khương Duy\_21520770  
Ngô Đức Huy\_22520554

## MỤC LỤC

<b>PHẦN 1- GIỚI THIỆU ĐỀ TÀI</b>	<b>2</b>
1.1 Giới thiệu	2
1.2 Các thành phần chính của hệ thống giao thông	2
1.3 Nguyên lý hoạt động	2
1.4 Ngôn ngữ sử dụng và phần mềm mô phỏng	3
1.5 Sơ lược	3
1.5.1 Sơ lược về chân của IC 89C51	3
1.5.2 Sơ lược về led 7 đoạn hiển thị thời gian	4
1.5.3 Sơ lược về Numpad 4x4	5
1.5.4 Cài đặt thời gian cho đèn giao thông	5
<b>PHẦN 2- THUẬT TOÁN VÀ CHƯƠNG TRÌNH</b>	<b>6</b>
2.1 Nguyên lý	6
2.2 Thuật toán	7
2.3 Code assembly	14
<b>PHẦN 3- MÔ PHỎNG VÀ THIẾT KẾ MẠCH IN</b>	<b>23</b>
3.1 Linh kiện và thiết bị	23
3.2 Thiết kế	23
3.2.1 Khối điều khiển trung tâm	23
3.2.2 Bàn phím	23
3.3 Thiết kế mạch in	24
<b>PHẦN 4- ĐÁNH GIÁ SƠ BỘ</b>	<b>26</b>

## **PHẦN 1- GIỚI THIỆU ĐỀ TÀI**

### **1.1 Giới thiệu**

Trong thời kỳ hiện đại ngày nay, sự phát triển vượt bậc của công nghệ đã mở ra cánh cửa cho ngành công nghiệp chế tạo linh kiện bán dẫn và vi mạch tổng hợp. Vi điều khiển, với những ưu điểm vượt trội, đã trở thành trung tâm của nhiều lĩnh vực, từ điện tử tiêu dùng đến ô tô và giao thông, từ y tế đến tự động hóa và IoT. Vi điều khiển không chỉ giúp tăng cường hiệu suất mà còn mang lại sự linh hoạt và dễ dàng tùy chỉnh trong quá trình sản xuất và xử lý.

Với đề tài của nhóm “mạch điều khiển đèn giao thông” là một ứng dụng tiêu biểu của vi điều khiển 8051 trong việc điều khiển đèn giao thông. Bằng phương pháp sử dụng 8051, chúng ta có thể tận dụng các tính năng của vi điều khiển để tăng cường hiệu suất, tiết kiệm năng lượng và giảm thiểu sai sót, giúp tránh được những tai nạn không đáng có. Điển hình cho điều này là việc sử dụng đèn 7 đoạn để hiển thị thời gian, kết hợp với 3 đèn LED màu đỏ, vàng và xanh lá, cùng với việc sử dụng numpad để cài đặt thời gian cho các đèn, tạo ra một hệ thống điều khiển đèn giao thông hoàn hảo, đáp ứng các tiêu chí chất lượng và hiệu suất.

### **1.2 Các thành phần chính của hệ thống giao thông**

- Mạch điều khiển trung tâm vi điều khiển AT89C51.
- Mạch dao động, reset
- Mạch hiển thị thời gian, trạng thái đèn.
- Phím nhấn điều khiển, chỉnh thời gian

### **1.3 Nguyên lý hoạt động**

-Nguyên lý hoạt động của mạch điều khiển đèn giao thông dựa trên việc điều chỉnh thời gian chiếu sáng của các đèn đỏ, vàng và xanh tại các ngã tư. Mỗi đèn sẽ chuyển sang một màu khác nhau sau một khoảng thời gian nhất định, và điều này được kiểm soát bằng cách đếm thời gian và cập nhật trạng thái của các đèn qua một chuỗi các hàm và ngắt.

-Khi bắt đầu, mạch sẽ được cấu hình với các giá trị mặc định hoặc được cài đặt bởi người sử dụng thông qua bàn phím. Sau đó, mỗi giây, một hàm ngắt sẽ được gọi để giảm thời gian của các đèn và kiểm tra xem thời gian còn lại có đạt đến 0 chưa. Nếu có, màu sắc của đèn sẽ được thay đổi.

-Ngoài ra, người dùng có thể tương tác với mạch thông qua bàn phím để điều chỉnh thời gian chiếu sáng của các đèn theo ý muốn. Mỗi khi nhập liệu từ bàn phím, mạch sẽ phản ứng bằng cách thay đổi cấu hình và thời gian của các đèn tương ứng với mã phím được nhập vào.

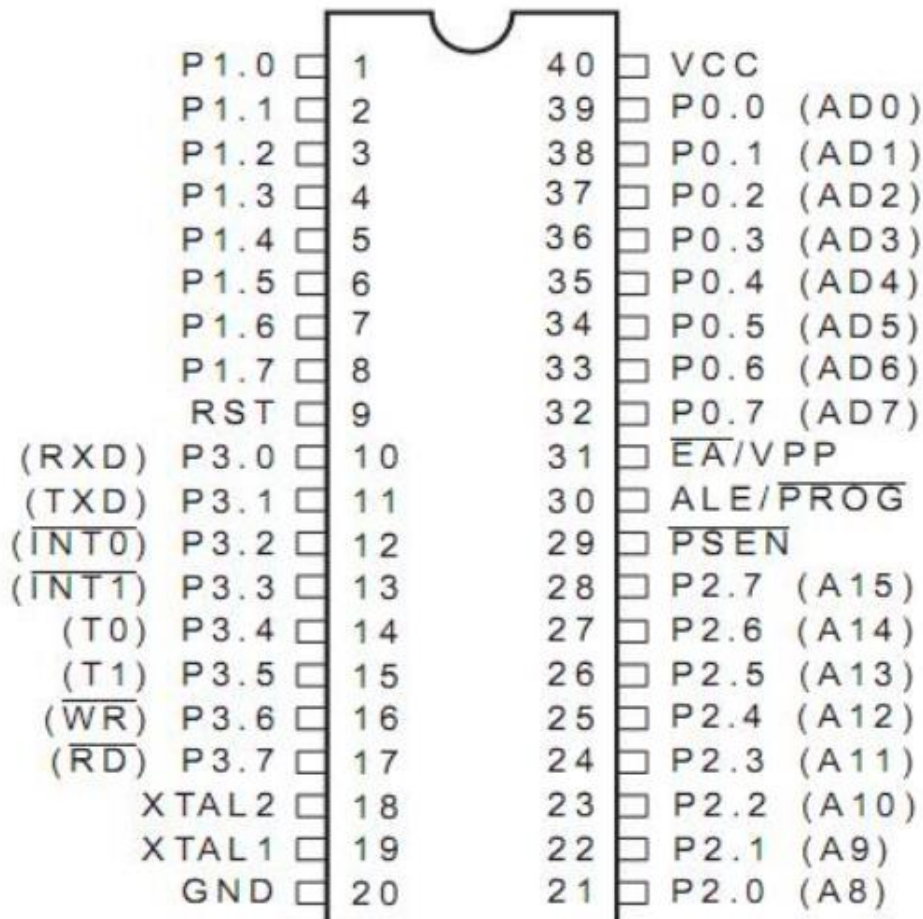
## 1.4 Ngôn ngữ sử dụng và phần mềm mô phỏng

Ngôn ngữ ASSEMBLY cho vi điều khiển 8051

Phần mềm PROTEUS

## 1.5 Sơ lược

### 1.5.1 Sơ lược về chân của IC 89C51

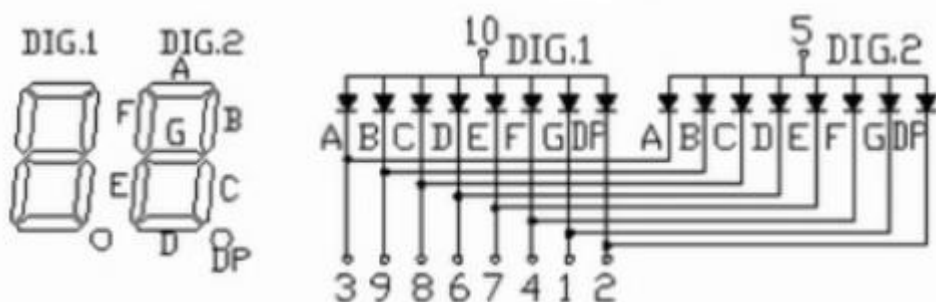


Hình trên cho ta sơ đồ chân của chip 8051, mô tả chức năng các chân như sau:

- Chân 1 đến 8: được gọi là Cổng 1 (Port 1), Tám chân này có duy nhất 1 chức năng là xuất và nhập. Cổng 1 có thể xuất và nhập theo bit hoặc byte. Ta đánh tên cho mỗi chân của Port 1 là P1.X (X = 0 đến 7)
- Chân 9: là chân vào reset của 8051 Khi tín hiệu này được đưa lên mức cao trong ít nhất là 2 chu kỳ máy, các thanh ghi trong bộ vi điều khiển được tải những giá trị thích hợp để khởi động hệ thống. Hay nói cách khác là vi điều khiển sẽ bị reset nếu chân này được kích hoạt mức cao
- Chân 10 đến 17: được gọi là Cổng 3 (Port 3) Tám chân này ngoài chức năng là xuất và nhập như các chân ở cổng 1 (chân 1 đến 8) thì mỗi chân này còn có chức năng riêng

- Chân 18 và 19 (XTAL1 & XTAL2) Hai chân này được sử dụng để nối với bộ dao động ngoài
- Chân 20: được nối vào chân 0V của nguồn cấp
- Chân 21 đến chân 28: được gọi là cổng 2 (Port 2)
- Chân 29 (PSEN): Chân PSEN là chân điều khiển đọc chương trình ở bộ nhớ ngoài, nó được nối với chân OE của ROM ngoài để cho phép đọc các byte mã lệnh trên ROM ngoài. PSEN ở mức thấp trong thời gian đọc mã lệnh. Khi thực hiện chương trình trong ROM nội thì PSEN được duy trì ở mức cao.
- Chân 30 (ALE): Chân ALE cho phép tách các đường dữ liệu và các đường địa chỉ tại Port 0 và Port 2.
- Chân 31 (EA): Tín hiệu chân EA cho phép chọn bộ nhớ chương trình là bộ nhớ trong hay ngoài vi điều khiển. Nếu chân EA được nối ở mức cao (nối nguồn Vcc), thì vi điều khiển thi hành chương trình trong ROM nội. Nếu chân EA ở mức thấp (được nối GND) thì vi điều khiển thi hành chương trình từ bộ nhớ ngoài.
- Chân 32 đến 39: được gọi là cổng 0 (Port 0) Cổng 0 gồm 8 chân cũng có 2 công dụng, ngoài chức năng xuất nhập, cổng 0 còn là bus đa hợp dữ liệu và địa chỉ, chức năng này sẽ được sử dụng khi 8051 giao tiếp với các thiết bị ngoài có kiến trúc Bus như các vi mạch nhớ... Vì cổng P0 là một máng mở khác so với các cổng P1, P2 và P3 nên các chân ở cổng 0 phải được nối với điện trở kéo khi sử dụng các chân này như chân vào/ra. Điện trở này tùy thuộc vào đặc tính ngõ vào của thành phần ghép nối với chân của port 0. Thường ta dùng điện trở kéo khoảng 4K7 đến 10K.
- Chân 40: chân nguồn của vi điều khiển, được nối vào chân Vcc của nguồn.

### 1.5.2 Sơ lược về led 7 đoạn hiển thị thời gian

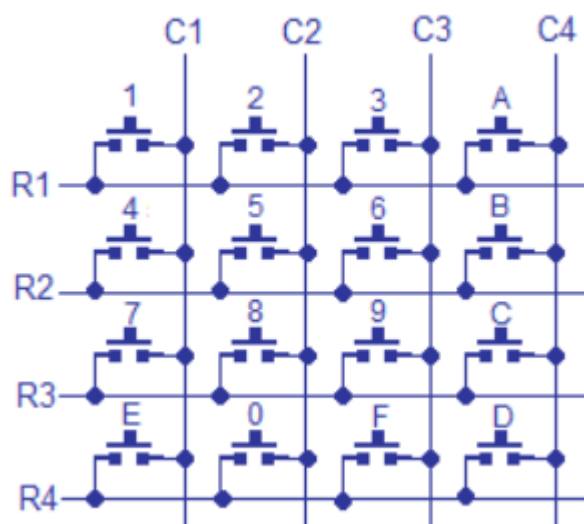


Mỗi đèn led 7 đoạn có chân đưa ra khỏi hộp hình vuông. Mỗi một chân sẽ được gán cho một chữ cái từ a đến g tương ứng với mỗi led. Những chân khác được nối lại với nhau thành một chân chung.

Như vậy bằng cách phân cực thuận (forward biasing) các chân của led theo một thứ tự cụ thể, một số đoạn sẽ sáng và một số đoạn khác không sáng cho phép hiển thị ký tự mong muốn. Điều này cho phép chúng ta hiển thị các số thập phân từ 0 đến 9 trên cùng một led 7 đoạn.

Chân chung được sử dụng để phân loại led 7 đoạn. Vì đèn led có 2 chân, 1 chân là anode và 1 chân là cathode nên có 2 loại led 7 đoạn là cathode chung (CC) và anode chung (CA). Loại CA (common anode): Tất cả các chân anode được nối với nhau với logic là 1. Mỗi phân đoạn được chiếu sáng bằng cách sử dụng điện trở tín hiệu logic 0 (hay low) vào các cực cathode (từ a đến g) .

### 1.5.3 Sơ lược về Numpad 4x4



Bàn phím hex có **8 đường** là R1, R2, R3, R4, C1, C2, C3 và C4. R1 đến R4 đại diện cho bốn hàng và C1 đến C4 đại diện cho bốn cột. Chương trình xác định phím nào được nhấn bằng phương pháp được gọi là quét cột. Trong phương pháp này, một hàng cụ thể được giữ ở mức thấp (các hàng khác được giữ ở mức cao) và các cột được kiểm tra ở mức thấp. Nếu một cột cụ thể được tìm thấy ở mức thấp thì điều đó có nghĩa là phím được kết nối giữa cột đó và hàng tương ứng (hàng được giữ ở mức thấp) được nhấn. Ví dụ: nếu hàng R1 ban đầu được giữ ở mức thấp và cột C1 được tìm thấy ở mức thấp trong quá trình quét, điều đó có nghĩa là phím 1 được nhấn.

### 1.5.4 Cài đặt thời gian cho đèn giao thông

Chỉ cần thiết lập thời gian cho 2 đèn là đèn xanh và đèn vàng, còn thời gian đèn đỏ = đèn xanh + đèn vàng. Điều này giúp đơn giản hóa thiết lập, đảm bảo tính nhất quán và an toàn. VD như xanh 20s, vàng 5s => đỏ 25s.

## **PHẦN 2-THUẬT TOÁN VÀ CHƯƠNG TRÌNH**

### **2.1 Nguyên lý**

#### **1. Khởi tạo và thiết lập giá trị mặc định (Default Setup)**

- Bắt đầu (Start)
- Thiết lập các thanh ghi và giá trị mặc định
- Kết thúc (Ret)

#### **2. Vòng lặp hiển thị thời gian và trạng thái đèn (Main Loop)**

- Bắt đầu (Loop)
- Hiển thị số hàng đơn vị và hàng chục của LED 2
- Hiển thị số hàng đơn vị và hàng chục của LED 1
- Kiểm tra trạng thái ngắt TR1
- Gọi hàm xử lý bàn phím nếu TR1 không được bật
- Kết thúc (Jmp Config)

#### **3. Chuyển đổi số thập phân sang BCD (BCD Conversion)**

- Bắt đầu (InputBlock)
- Chia giá trị bởi 10 để lấy hàng chục
- Lưu kết quả hàng chục vào BCDoutput1
- Lấy phần dư để lưu vào BCDoutput2
- Kết thúc (Ret)

#### **4. Xử lý ngắt Timer1 (Timer1 Interrupt Handling)**

- Bắt đầu (Timer1\_ISR)
- Thiết lập lại Timer1
- Giảm thời gian đếm (i1)
- Gọi hàm kiểm tra thời gian của đèn giao thông (Check1 và Check2)
- Kết thúc (Reti)

#### **5. Xử lý thay đổi màu đèn giao thông (Traffic Light Color Change Handling)**

- Bắt đầu (Check1/Check2)
- Kiểm tra thời gian còn lại của đèn
- Gọi hàm thay đổi màu đèn (Color1/Color2) nếu thời gian hết
- Kết thúc (Ret)
- Bắt đầu (Color1/Color2)
- Chuyển đổi trạng thái màu của đèn giao thông
- Cập nhật thời gian mới cho màu hiện tại
- Kết thúc (Ret)

#### **6. Xử lý bàn phím (Keypad Handling)**

- Bắt đầu (Keypad\_handling)
- Quét và phát hiện phím nhấn
- Xác định vị trí hàng và cột của phím nhấn
- Thực hiện hành động tương ứng với phím nhấn

-Kết thúc (Ret)

7. Chuyển đổi trạng thái và thời gian của đèn giao thông (Traffic Light State and Time Transition)

-Bắt đầu (SetGreen1/SetGreen2)

-Chuyển đổi trạng thái đèn sang màu xanh

-Thiết lập thời gian mới cho đèn xanh

-Kết thúc (Ret)

## **2.2 Thuật toán**

### **Quét led**

1.Chuẩn bị giá trị thời gian để hiển thị:

-Lấy giá trị time1 và time2 (được đặt trong thanh ghi R5 lần lượt) để hiển thị trên LED.

-Gọi hàm InputBlock để chuyển đổi giá trị thời gian từ thập phân sang BCD (Binary-Coded Decimal).

2.Hiển thị giá trị đơn vị và chục của LED:

-Sử dụng kết quả từ hàm InputBlock để hiển thị giá trị đơn vị và chục của thời gian trên LED.

3.Kiểm tra trạng thái của TR1:

-Nếu TR1 được bật (TR1 = 1), quay lại bước đầu tiên của vòng lặp (Loop).

-Nếu TR1 không được bật, xử lý bàn phím (Keypad\_handling).

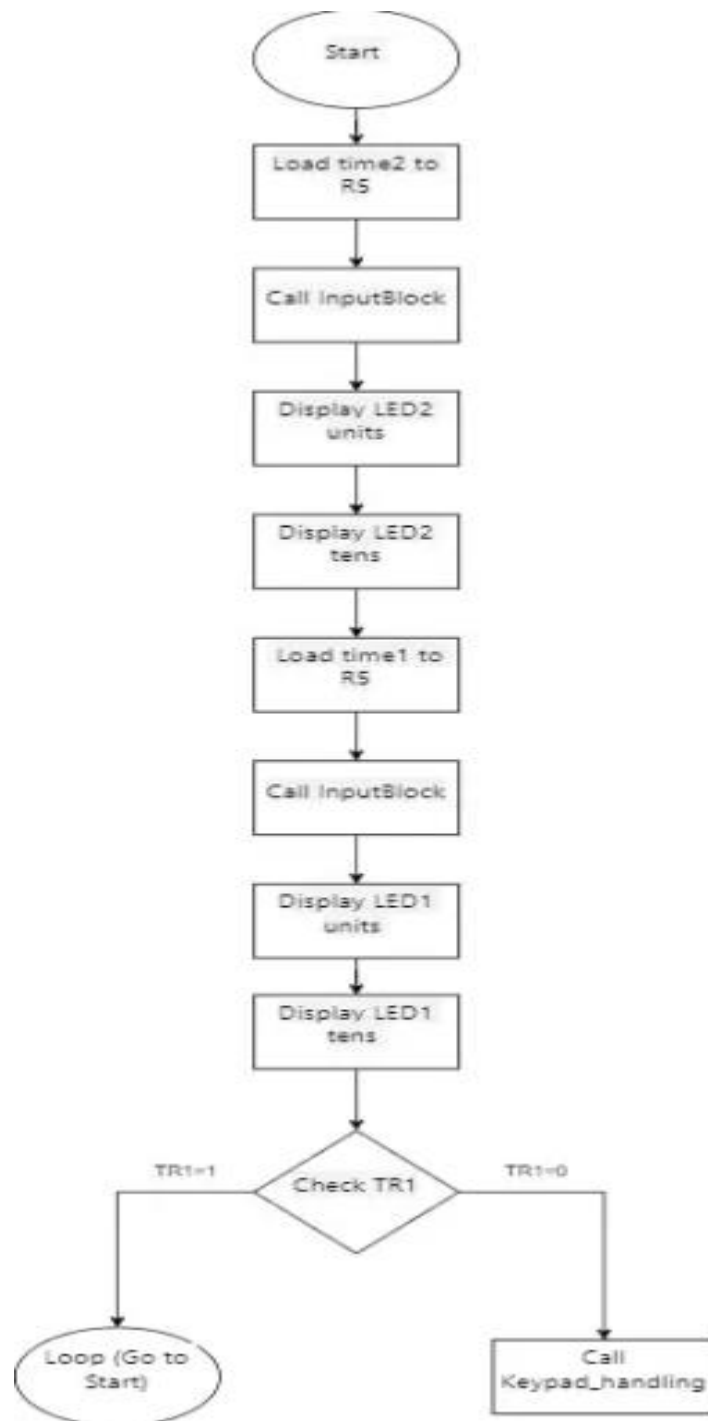
4.Hiển thị đơn vị và chục cho từng LED:

-Hiển thị giá trị BCD của thời gian (time1 và time2) lên từng LED.

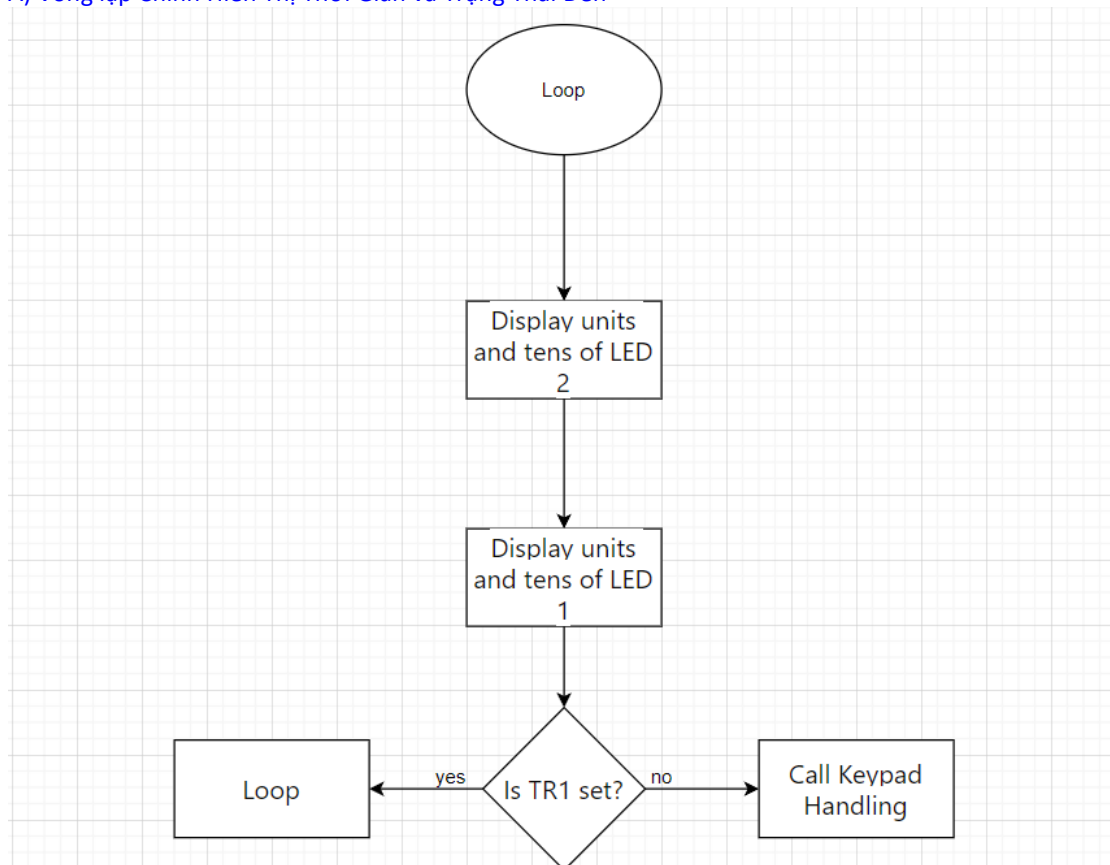
### **Lưu đồ thuật toán**

### **Quét led**

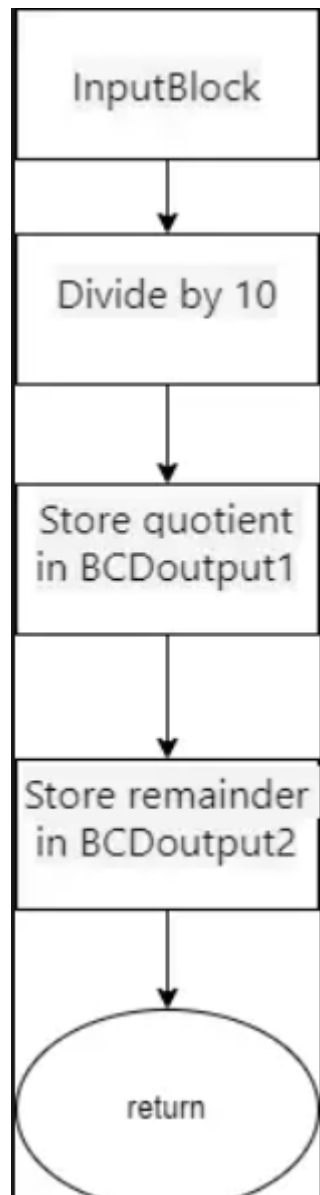




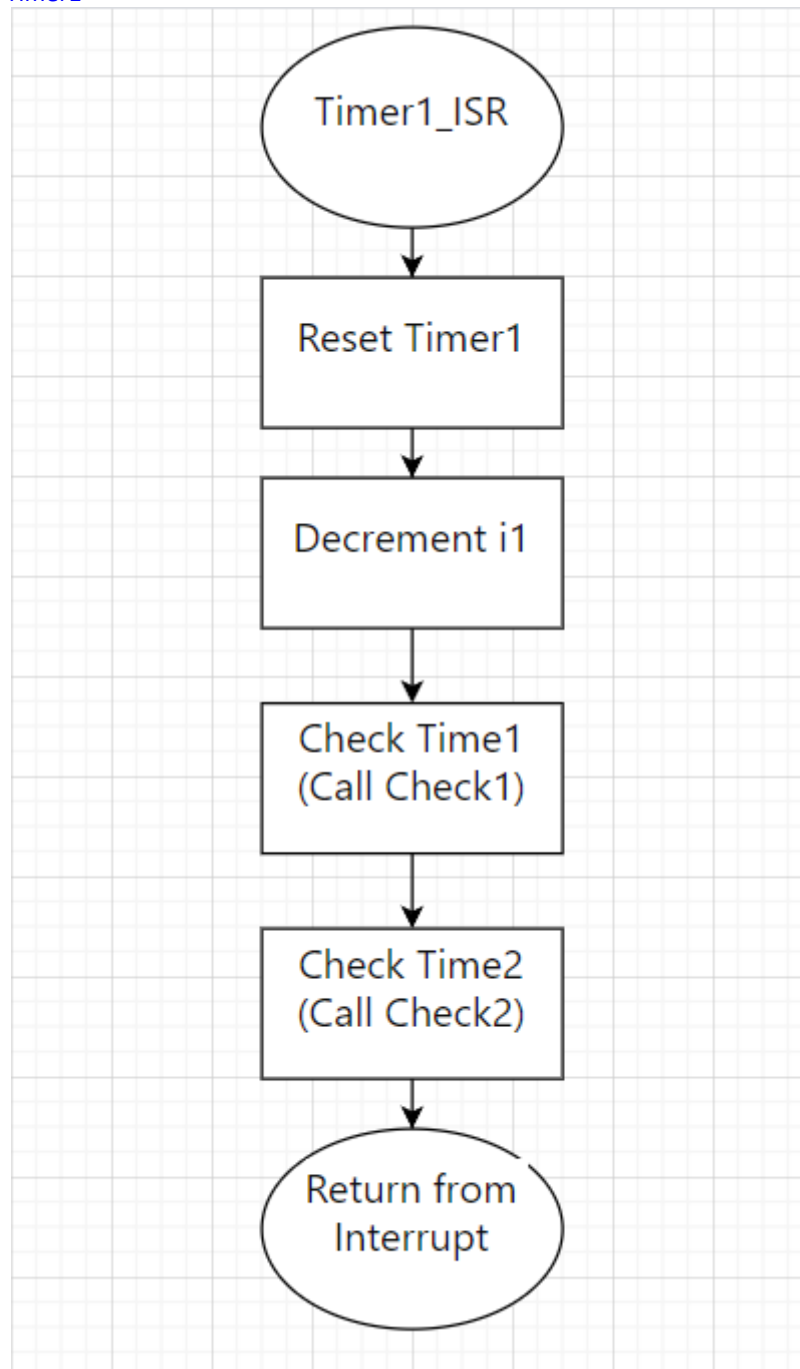
A) Vòng lặp Chính Hiển Thị Thời Gian và Trạng Thái Đèn



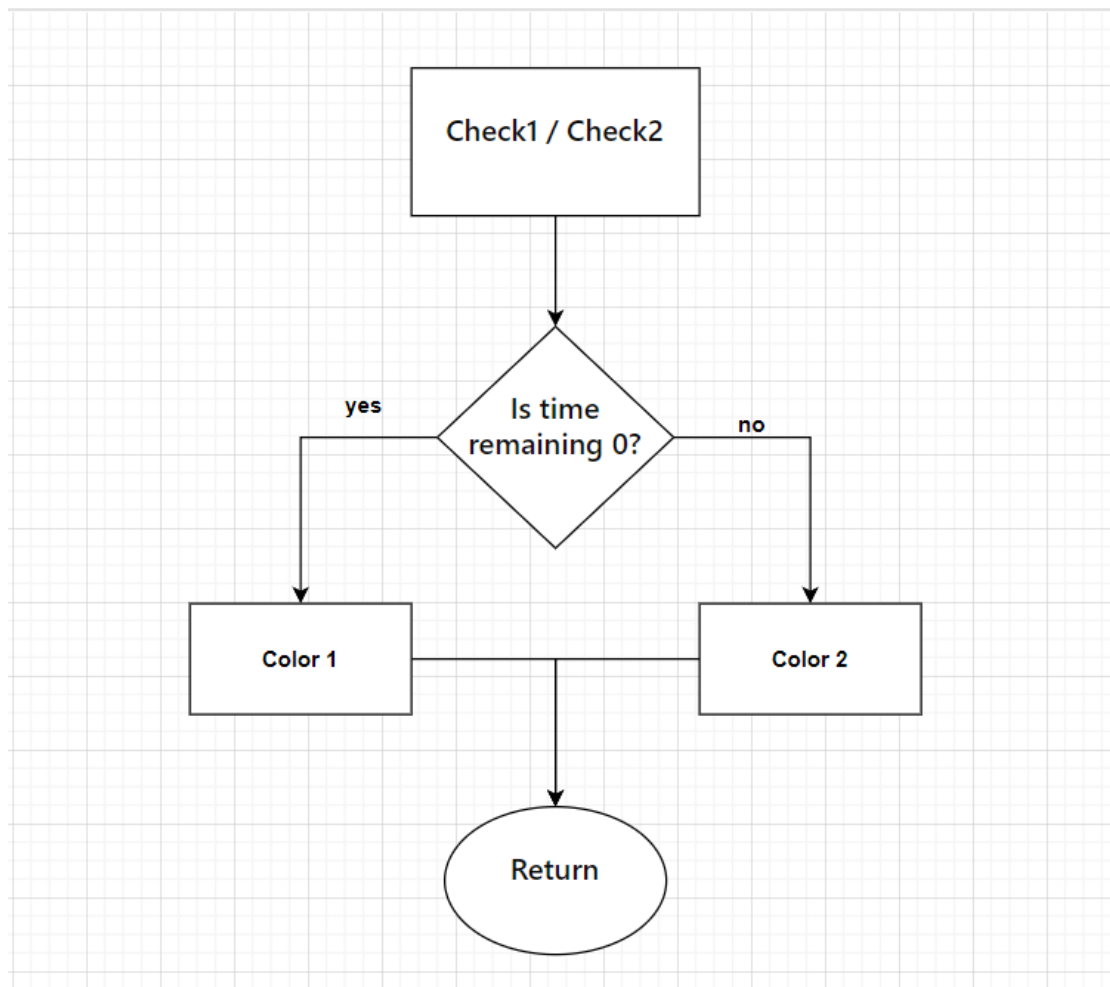
B) Chuyển đổi Số Thập phân sang BCD



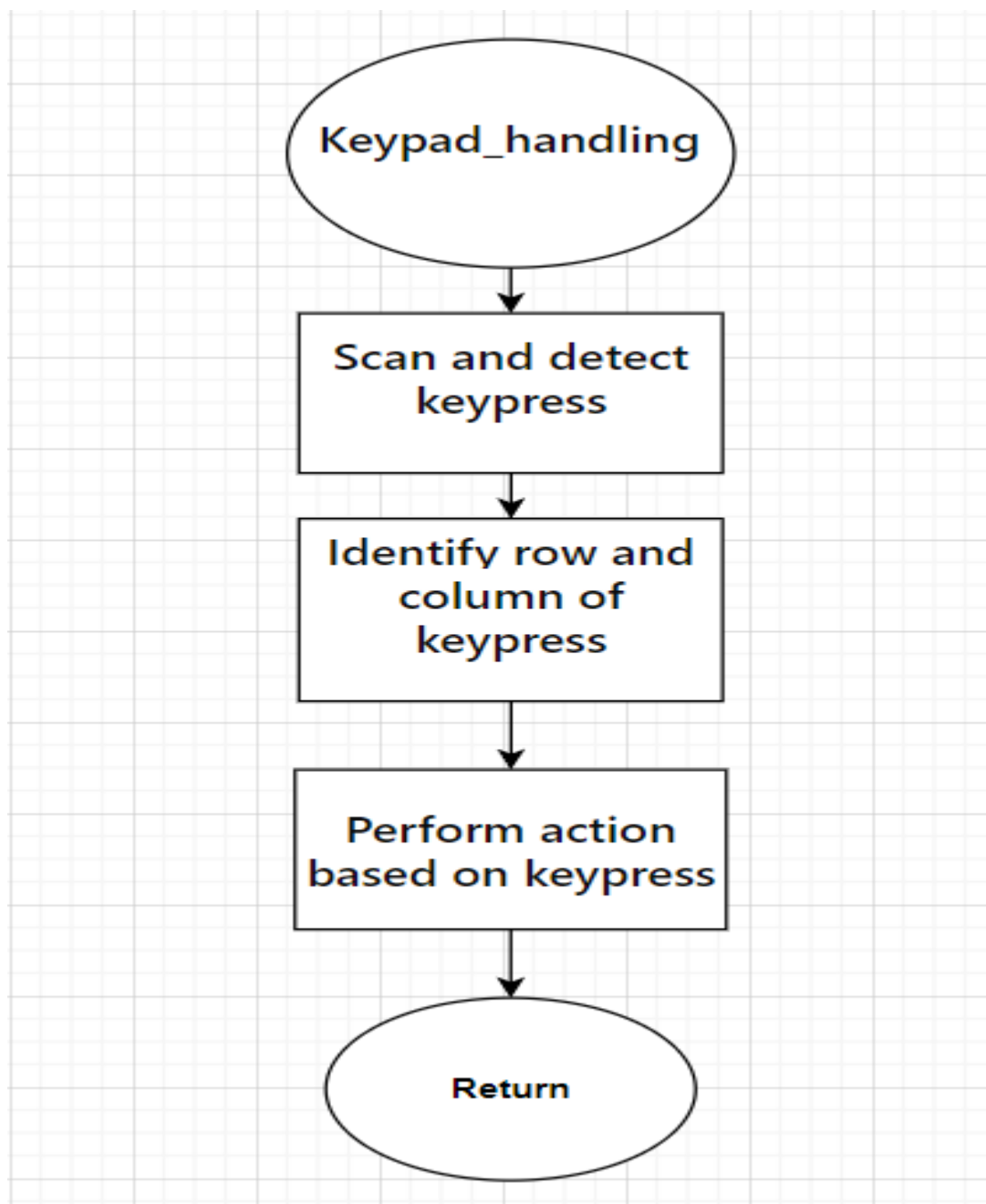
C) Xử lý Ngắt Timer1



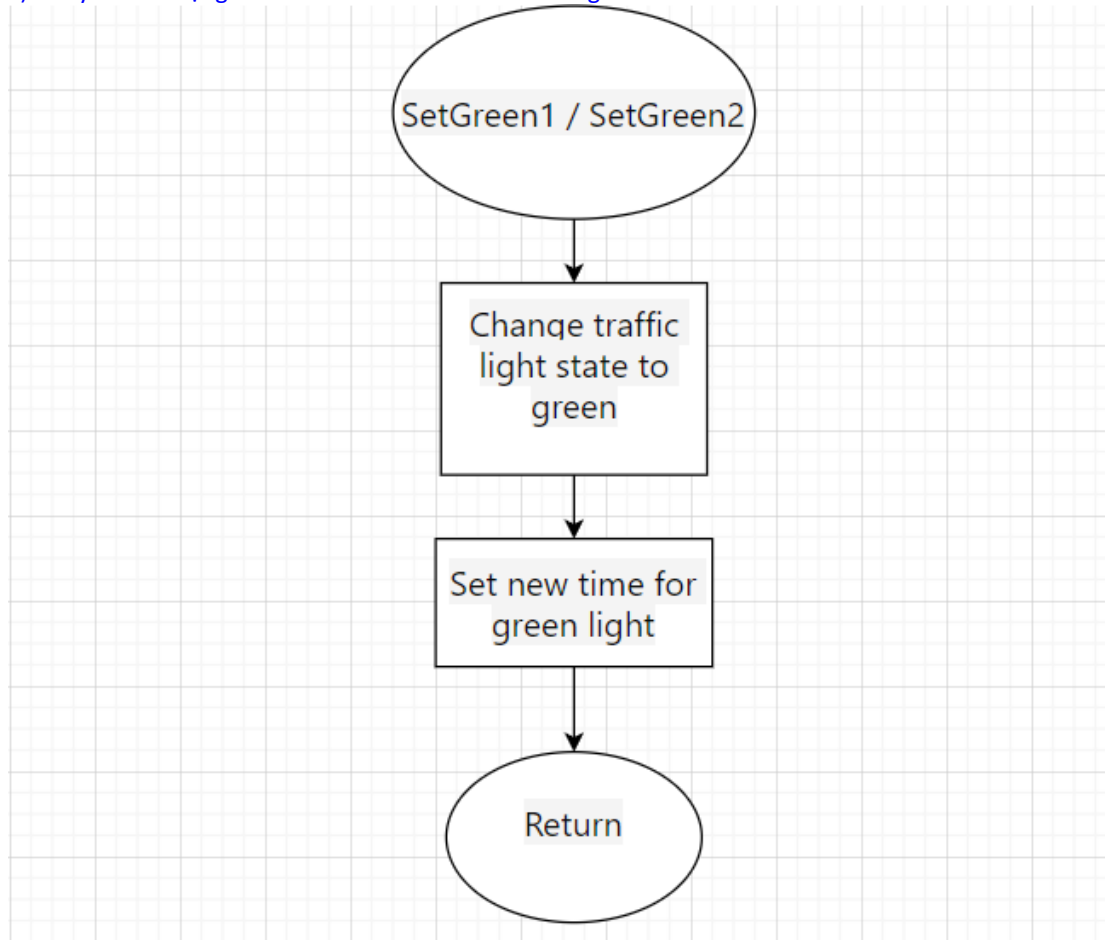
D) Thay Đổi Màu Đèn Giao Thông



#### E) Xử lý Bàn Phím



#### F) Chuyển Đổi Trạng Thái và Thời Gian Đèn Giao Thông



### 3.3 Code assembly

\$NOMOD51

\$INCLUDE (8051.MCU)

den1	equ	30H	
den2	equ	31H	
red1	equ	32H	; bien thời gian đèn đỏ của block 1
yellow1	equ	33H	; bien thời gian đèn vàng của block 1
green1	equ	34H	; bien thời gian đèn xanh của block 1
red2	equ	35H	; bien thời gian đèn đỏ của block 2
yellow2	equ	36H	; bien thời gian đèn vàng của block 2
green2	equ	37H	; bien thời gian đèn xanh của block 2
BCDinput	equ	38H	; bien đầu vào của hàm BCD
BCDoutput1	equ	39H	; bien đầu ra hàng chục của hàm BCD
BCDoutput2	equ	3AH	; bien đầu ra hàng đơn vị của hàm
i1	equ	3BH	; bien đếm thời gian
time1	equ	3CH	; bien tạm lưu thời gian của block 1
time2	equ	3DH	; bien tạm lưu thời gian của block 2
Keycode	equ	3Eh	; bien lưu giá trị nhập từ keypad
COI	equ	3Fh	; bien lưu vị trí của cột khi nhập từ
keypad			
temp_time	equ	R2	; bien tạm lưu giá trị thời gian nhập vào

```

current_color_dur      equ      40H      ; bien tam luu thoi gian nhap vao tu
keypad
BCD_output             equ      41H
DEFAULT_RED_DUR        equ      24
DEFAULT_YELLOW_DUR     equ      03
DEFAULT_GREEN_DUR      equ      21

col1                   bit      P2.0
col2                   bit      P2.1
col3                   bit      P2.2
col4                   bit      P2.3
rowA                   bit      P2.4
rowB                   bit      P2.5
rowC                   bit      P2.6
rowD                   bit      P2.7

      org 0f000h
Key_RowA:              db      07h, 08h, 09h, 0Ah
Key_RowB:              db      04h, 05h, 06h, 0Bh
Key_RowC:              db      01h, 02h, 03h, 0Ch
Key_RowD:              db      0Dh, 00h, 0Eh, 0Fh

      org 0000H
      ljmp Start
      org 0003h
      ljmp INTERRUPT0
      org 001BH
      ljmp Timer1_ISR

      org 0030H
Start:
      mov     TMOD, #11H
      mov     IE, #89H
      setb    IT0
      call    Default

Config:
      mov     p1, #00001100B      ; Ham nhap thoi gian cho den
      anl     p0, #0Fh           ; bat cac bit D1 D2 cua 2 LED len 1
      mov     A, green1
      add     A, yellow1
      mov     red1, A
      mov     red2, red1
      mov     yellow2, yellow1
      mov     green2, green1
      mov     den1, #11111100B
      mov     den2, #11001111B
      mov     R0, #green1
      mov     R1, #red2
      mov     time1, @R0
      mov     time2, @R1
      call    Count1s
Loop:
      mov     R5, time2
      acall   InputBlock
Block2d:
      ; Hien thi hang don vi cua LED 2

```



```

        mov            A, p0
    orl            A, BCDoutput2
    mov            p0, A
    setb           p0.7
        setb           p3.0
    call            Delay
    clr            p3.0
    clr            p0.7
    call            ClearBlock
Block2c:                                     ; Hien thi hang chuc cua LED 2
    mov            A, p0
    orl            A, BCDoutput1
    mov            p0, A
    setb           p0.6
        setb           p3.0
    call            Delay
        clr            p3.0
    clr            p0.6
    call            ClearBlock

InputBlock1:                               ; Nhap vao thoi gian cho block 1
        mov            R5, time1
        acall           InputBlock
Block1d:                                     ; Hien thi hang don vi cua LED 1
    mov            A, p0
    orl            A, BCDoutput2
    mov            p0, A
    setb           p0.5
        setb           p3.0
    call            Delay
        clr            p3.0
    clr            p0.5
    call            ClearBlock
Block1c:                                     ; Hien thi hang chuc cua LED 1
    mov            A, p0
    orl            A, BCDoutput1
    mov            p0, A
    setb           p0.4
        setb           p3.0
    call            Delay
        clr            p3.0
    clr            p0.4
    call            ClearBlock

    jb             TR1, Loop                 ; Cho phép được sử dụng keypad hay không
    ljmp            Keypad_handling
    jmp             Config

InputBlock:;{
    ; chuyen doi tu co so 10 sang so BCD
    mov            A, R5
    mov            B, #10
    div            AB
    mov            BCDoutput1, A

        mov            BCDoutput2, B

```

```

        ret
    ;}
;-----

Delay:                                     ; Ham delay bang vong lap
    mov                                     R3, #16
LoopD:
    mov                                     R4, #200
    djnz                                     R4, $
    djnz                                     R3, LoopD
    ret

;-----

Count1s:                                   ; Timer1 10ms
    mov                                     i1, #100
    mov                                     TL1, #000H
    mov                                     TH1, #0DCH
    setb                                     TR1
    ret

;-----

Timer1_ISR:                               ; Interrupt Timer1 dem 1s,kiem tra thoi gian cac den
    mov                                     TL1, #00H
    mov                                     TH1, #0dcH
    setb                                     TR1
    djnz                                     i1, End_Count1s
    dec                                     time1
    dec                                     time2
    mov                                     i1, #100
    call                                    Check1
    call                                    Check2
End_Count1s:
    reti

;-----

;-----

Check1:                                    ; ham kiem tra thoi gian con lai cua den giao thong
    mov                                     A, time1
    jnz                                     EndCheck1
    call                                    Color1
EndCheck1:
    ret

;-----

Color1:                                    ; ham chuyen doi mau cua den giao thong
    call                                    Set1
    mov                                     A, den1
    clr                                     ACC.3
    rr                                     A
    jb                                     ACC.7, SetGreen1
    setb                                    ACC.7

```

```

    mov     den1, A
    mov     A, p1
    anl     A, den1
    mov     p1, A
    dec     R0
EndColor1:
    mov     time1, @R0
    ret

```

;-----

```

SetGreen1:                                ; ham doi sang den mau xanh neu truoc do la mau do
    mov     den1, #11111100B
    call    Set1
    mov     A, p1
    anl     A, den1
    mov     p1, A
    mov     R0, #green1
    jmp     EndColor1

```

;-----

```

Check2:
    mov     A, time2
    jnz     EndCheck2
    call    Color2
EndCheck2:
    ret

```

;-----

```

Color2:
    call    Set2
    mov     A, den2
    clr     ACC.6
    rr      A
    jb      ACC.2, SetGreen2
    setb    ACC.2
    mov     den2, A
    mov     A, p1
    anl     A, den2
    mov     p1, A
    dec     R1
EndColor2:
    mov     time2, @R1
    ret

```

;-----

```

SetGreen2:
    mov     den2, #11100111B
    call    Set2
    mov     A, p1
    anl     A, den2
    mov     p1, A
    mov     R1, #green2
    jmp     EndColor2

```

```

;-----
ClearBlock:                                     ; tat man hinh hien thi led
    mov     A, p0
    anl     A, #0F0h
    mov     p0, A
    ret

;-----

Set1:
    setb    p1.0
    setb    p1.1
    setb    p1.2
    ret

;-----

Set2:
    setb    p1.3
    setb    p1.4
    setb    p1.5
    ret

;-----

Keypad_handling:
    clr     rowA                                ; keo cac hang xuong muc thap
    clr     rowB                                ; ==> phat hien su kien nhan phim
    clr     rowC
    clr     rowD
    jnb     col1,scan                            ; kiem tra bat ki cot nao duoc nhan
    jnb     col2,scan
    jnb     col3,scan
    jnb     col4,scan
    ljmp    Loop

scan:
    acall   delay                                ; neu co phim duoc nhan bat dau tim vi tri phim
    acall   scan_keypad                          ; chong doi phim
    mov     A, Keycode

    ; switch(Keycode)
    cjne    A, #0ah, not_0ah
    jmp     rst                                    ; case 0ah (dau chia)

not_0ah:
    cjne    A, #0bh, not_0bh
    call    Default                              ; case 0bh (dau nhan): set ve gia tri mac dinh
    jmp     rst

not_0bh:
    cjne    A, #0ch, not_0ch
    jmp     rst                                    ; case 0bh (dau tru)

not_0ch:
    cjne    A, #0fh, not_0fh
chooseColorKeypad:                             ; case 0fh (dau cong): chon loai den muon chinh thoi gian

```

```

    mov     A, p1
    cjne    A, #00100100B, greenKeypad
    mov     p1, #00010010B
    mov     current_color_dur, #yellow1
    jmp     EndchooseColorKeypad
greenKeypad:
    RL      A
    mov     p1, A
    inc     current_color_dur
EndchooseColorKeypad:
    mov     time1, #0
    mov     time2, #0
    jmp     rst

not_0fh:
    cjne    A, #0dh, not_0dh
    mov     COL, #0
    jmp     Config ; case 0dh (nut ON): ket thuc viec nhap tu keypad

not_0dh:
    cjne    A, #0Eh, not_0eh
SetTime:
    mov     A, current_color_dur
    cjne    A, #33H, TimeGreen
    mov     yellow1, temp_time
    jmp     endSetTime
TimeGreen:
    mov     green1, temp_time
endSetTime:
    jmp     rst

not_0eh:
    acall    store_time ; case default: nhap tu cac phim tu 0->9 de tinh toan
rst:
    mov     COL, #0 ; reset trang thai sau khi hien thi
    jmp     Keypad_handling

; kiem tra va xac dinh vi tri Cot cua phim khi thuc hien quet phim
check_col:
    jb      col1, check_col2
    mov     COL, #1 ; co phim o Cot 1 duoc nhan
    ret
check_col2:
    jb      col2, check_col3
    mov     COL, #2 ; Cot 2
    ret
check_col3:
    jb      col3, check_col4
    mov     COL, #3 ; Cot 3
    ret
check_col4:
    jb      col4, finish
    mov     COL, #4 ; Cot 4
finish:
    ret

; quet phim

```

```

scan_keypad:
    clr                rowA                ; quet hang A
    setb               rowB
    setb               rowC
    setb               rowD
    acall               check_col           ; kiem tra co phim nao cua hang A duoc nhan hay khong
    mov                 A,COL               ; tra ve vi tri Cot cua phim duoc nhan
    jz                  to_rowB
    ; nhay sang quet hang tiep theo neu khong co phim nao cua hang A duoc nhan
    mov                 DPTR, #Key_RowA
    ; luu cac phim cua hang A neu co phim trong hang duoc nhan
    sjmp                asign_keycode
to_rowB:
    clr                rowB
    setb               rowA
    setb               rowC
    setb               rowD
    acall               check_col
    mov                 A,COL
    jz                  to_rowC
    mov                 DPTR, #Key_RowB
    sjmp                asign_keycode
to_rowC:
    clr                rowC
    setb               rowB
    setb               rowA
    setb               rowD
    acall               check_col
    mov                 A,COL
    jz                  to_rowD
    mov                 DPTR, #Key_RowC
    sjmp                asign_keycode
to_rowD:
    clr                rowD
    setb               rowB
    setb               rowC
    setb               rowA
    acall               check_col
    mov                 A,COL
    jz                  ok
    mov                 DPTR, #Key_RowD
asign_keycode:
    ; gan gia tri phim
    setb               C
    anl                 C, col1
    anl                 C, col2
    anl                 C, col3
    anl                 C, col4
    jnc                 asign_keycode
    add                 A, #-1              ; tru vi tri cot di 1
    movc                A, @A + DPTR        ; gan gia tri phim tai o nho A + DPTR cho thanh ghi A
    mov                 Keycode, A          ; luu lai gia tri phim
ok:
    ret

INTERRUPT0:
    clr                TR1
    lcall               start_keypad

```

```

    reti

start_keypad:                                ; ham cho phep su dung keypad
    clr                                     p0.7
    clr                                     p0.6
    clr                                     p0.5
    clr                                     p0.4
    mov                                     p1, #00010010B
    mov                                     time1, #0
    mov                                     time2, #0
    mov                                     current_color_dur, #yellow1
    ret

store_time::{
    mov                                     R2, A
    mov                                     A, time1
    jz                                     store
    cjne                                    A, #10, next0
next0:
    jc                                     next1
    mov                                     B, #10
    div                                    AB
    mov                                     A, B
next1:
    mov                                     B, #10
    mul                                    AB
store:
    add                                    A, R2
    mov                                     time1, A
    mov                                     time2, A
    mov                                     R2, A
    ret
;}

Default:                                    ; ham dat gia tri mac dinh
    mov                                     red1, #DEFAULT_RED_DUR
    mov                                     yellow1, #DEFAULT_YELLOW_DUR
    mov                                     green1, #DEFAULT_GREEN_DUR
    ret

END

```

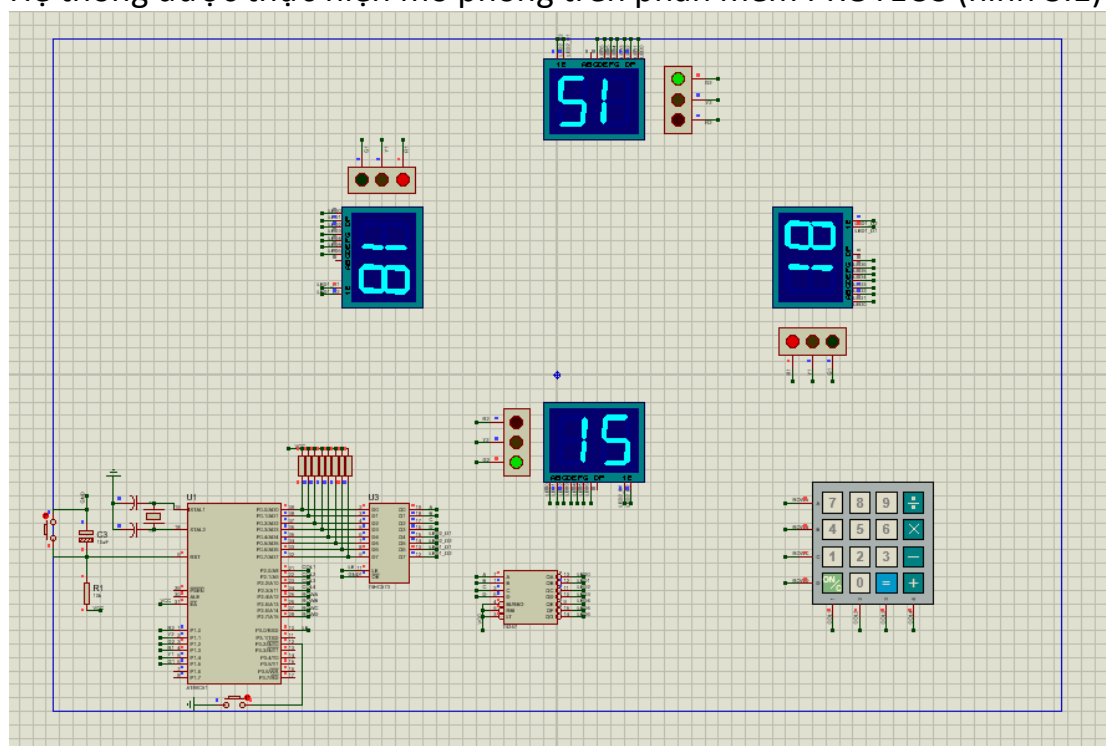
## PHẦN 3- MÔ PHỎNG VÀ THIẾT KẾ MẠCH IN PCB

### 3.1 Linh kiện và thiết bị

- + Led 7 đoạn đôi, loại 0.56 inch x 4
- + 3 đèn giao thông, loại cathode chung x 4
- + Phím ma trận 4x4 x 1
- + IC giải mã LED 7 đoạn 74LS247 x 1
- + AT89S51 x 1
- + Điện trở (1k $\Omega$ ) x 12
- + Button x 2
- + Mạch nạp ISP x 1
- + IC chốt 74HC573 x 1
- + Tụ hoá 30 pF x 2
- + Tụ hoá 10  $\mu$ F x 1
- + Thạch anh ngoại tần số 11.0592MHz x 1
- + Jack nguồn tròn 5.5 mm  $\times$  2.1 mm

### 3.2 Thiết kế và mô phỏng trên Proteus

Hệ thống được thực hiện mô phỏng trên phần mềm PROTEUS (hình 3.1)



Hình 3.1: Sơ đồ mạch mô phỏng

#### 3.1.1 Khối điều khiển trung tâm

Sử dụng vi điều khiển 8051 làm nhiệm vụ điều khiển các khối chức năng khác.

#### 3.1.2 Bàn phím

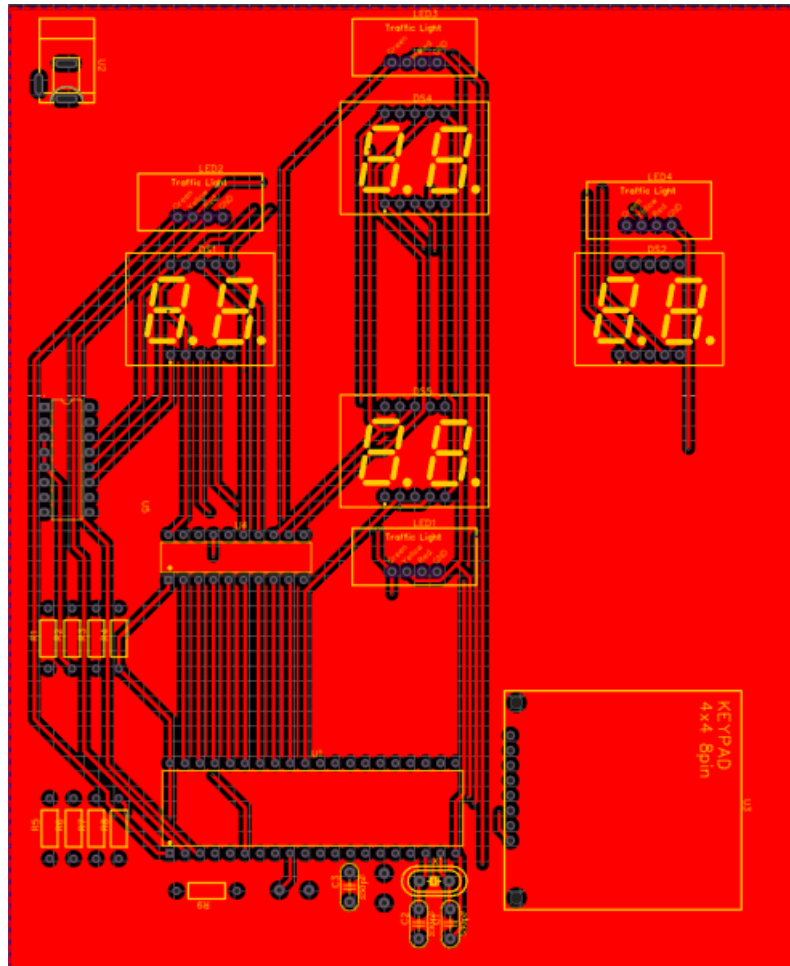
Bàn phím 4x4 gồm:



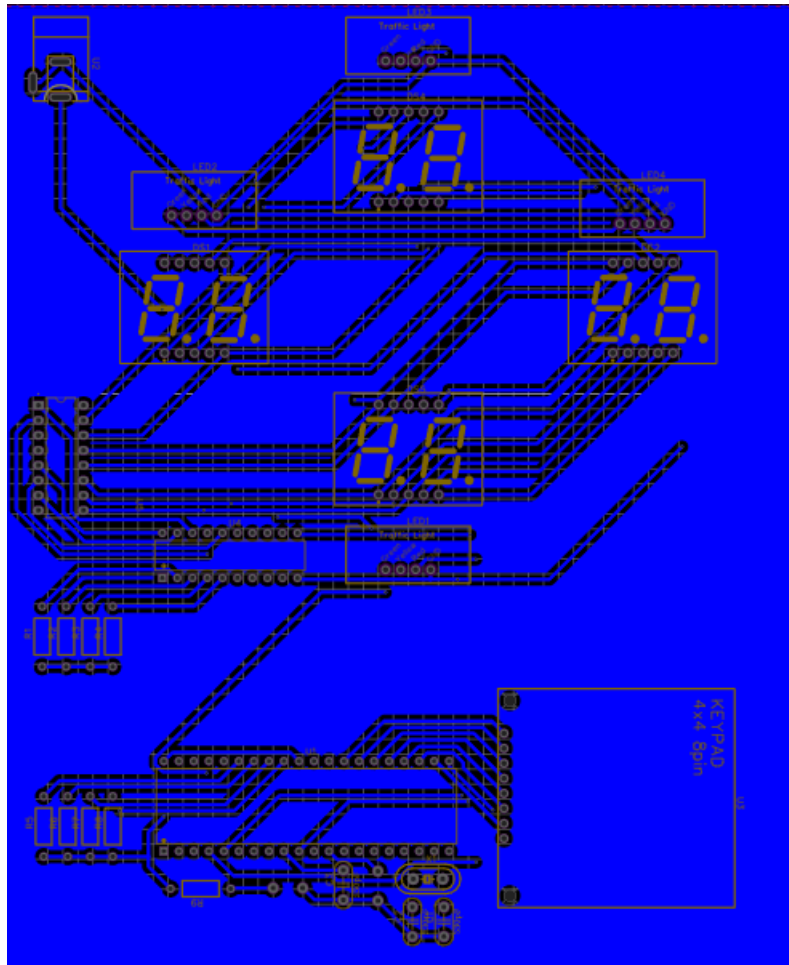
- 16 nút bấm nối với nhau thành 4 hàng ( ROW A đến ROW D) và 4 cột ( COL 1 đến COL 4)
- Sử dụng tín hiệu từ PORT 2 để điều khiển.

### 3.3 Thiết kế mạch in

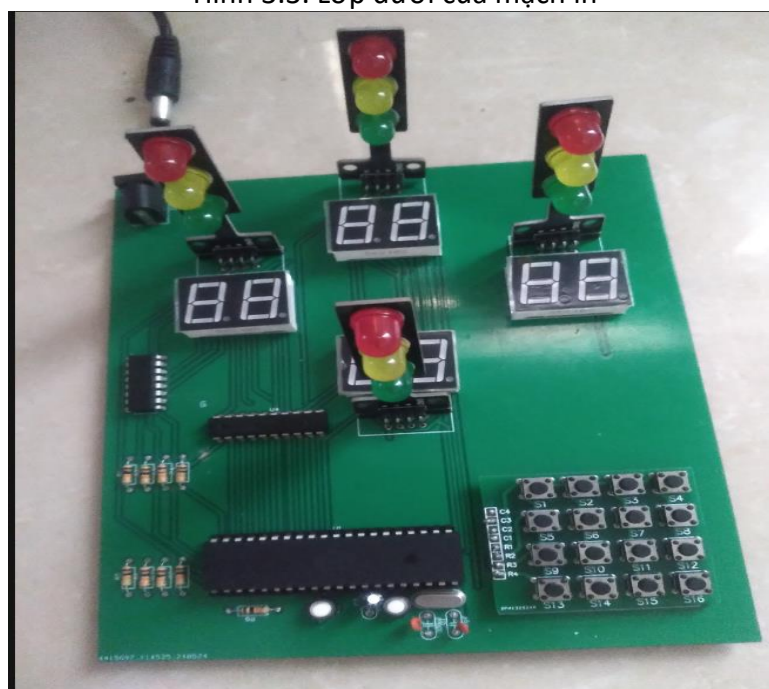
- Mạch in được thiết kế với kích thước 16.3 cm × 13.4 cm, đi dây đồng ở cả hai lớp trên và dưới, lớp phủ đồng được nối với dây GND của vi điều khiển AT89C51.



Hình 3.2: Lớp trên của mạch in



Hình 3.3. Lớp dưới của mạch in



Hình 3.4: Mạch thực tế sau khi lắp linh kiện

#### **PHẦN 4- ĐÁNH GIÁ SƠ BỘ**

- Mạch đèn giao thông sử dụng được hầu hết các chức năng của dòng vi điều khiển 8051 (Timer, ngắt ngoài, ngắt timer,...) để hiện thực các chức năng cơ bản của đèn giao thông như hiển thị LED 7 đoạn, đếm giờ và hiển thị đèn giao thông theo thời gian cũng như giao tiếp với phím ma trận  $4 \times 4$ . Tuy nhiên trong quá trình làm cũng không tránh khỏi những sai sót trong tính toán và cách đi dây còn chưa chuẩn nên không đủ dòng để cấp cho đèn giao thông sáng mạnh.

- [Link Github đồ án](#)