BÀI 2: CÁC LỆNH CƠ BẢN CỦA 89C51

❖ **MỤC ĐÍCH**

Giúp sinh viên khảo sát các vấn đề sau:

- Sử dụng phần mềm Proteus để mô phỏng mạch điện.
- Thực hiện mô phỏng một số lệnh của 89C51.

❖ **THIẾT BỊ SỬ DỤNG**

- Máy vi tính.
- Phần mềm Proteus

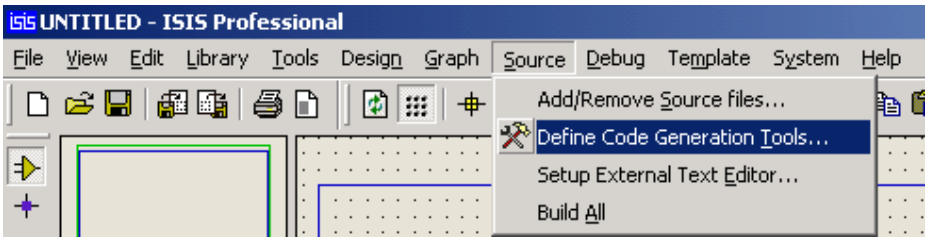
1. Cơ sở lý thuyết

Mô phỏng 89C51

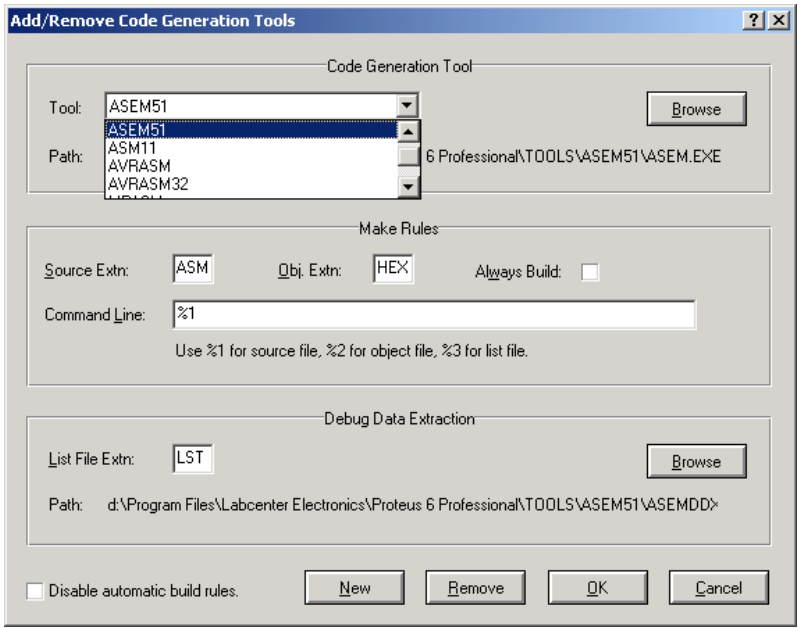
Để thực hiện quá trình mô phỏng 89C51 trong Proteus, ta cần thực hiện các bước sau:

- **Bước 1:** Vẽ mạch nguyên lý.
- **Bước 2:** Định nghĩa chương trình dịch

Chọn menu **Source > Define Code Generation Tools**



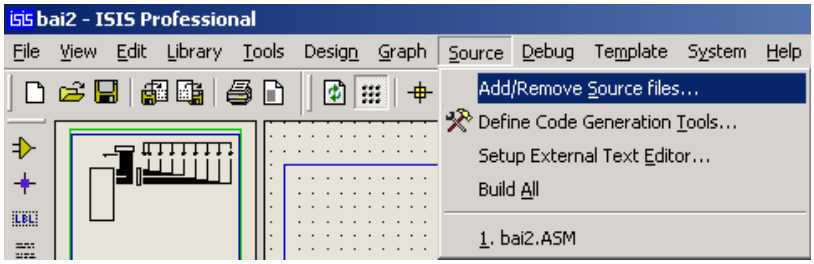
Sau đó thực hiện chọn chương trình dịch mong muốn. Ở đây ta thực hiện mô phỏng cho 89C51 nên chọn chương trình ASEM51.



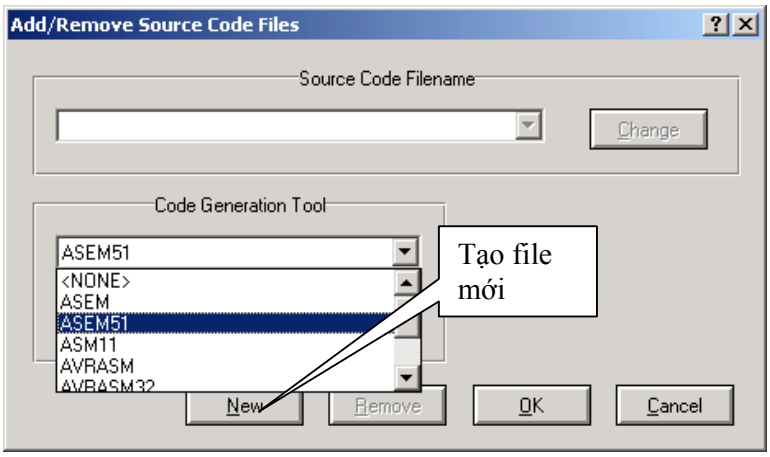
Phần **Tools**: chọn **ASEM51**, phần **Command Line**: gõ vào **%1**.

- **Bước 3:** Định nghĩa file chương trình cho 89C51.

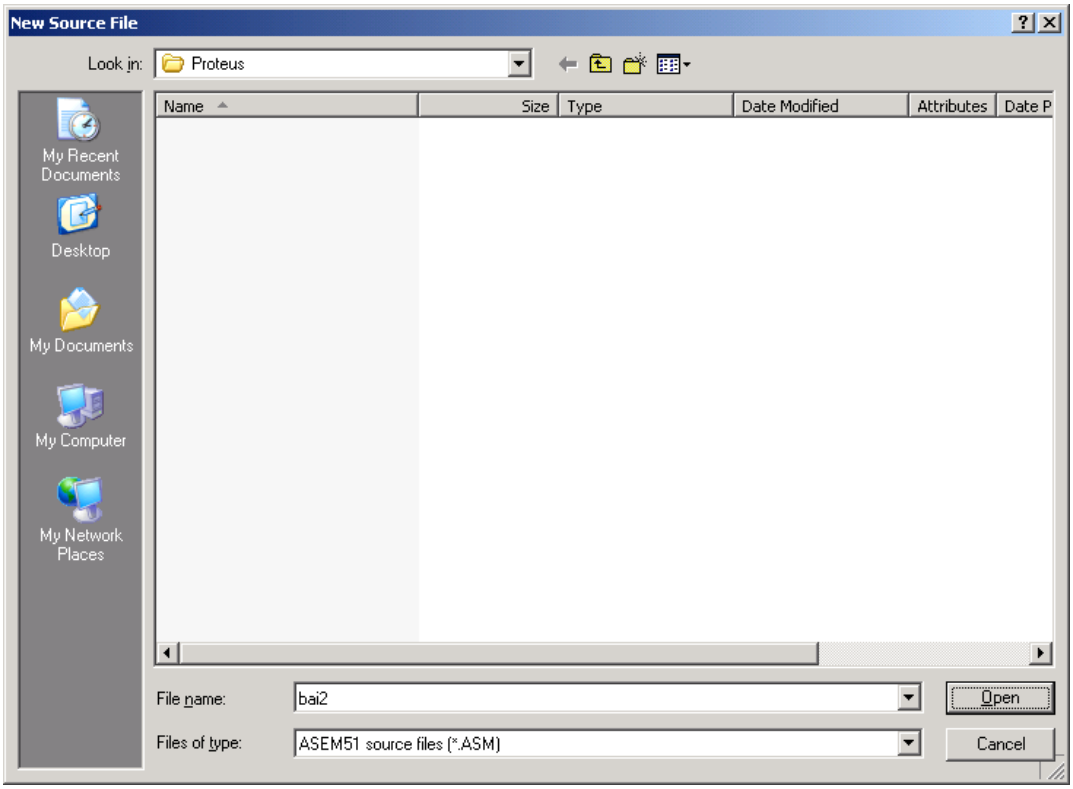
Chọn menu **Source > Add/Remove Source File**



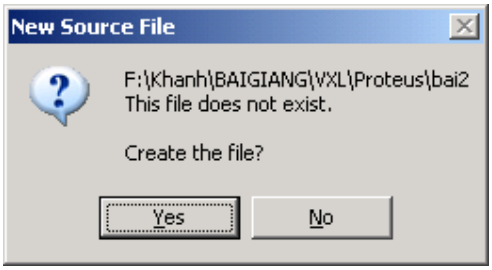
Chọn phần **Code Generation Tool** là **ASEM51**.



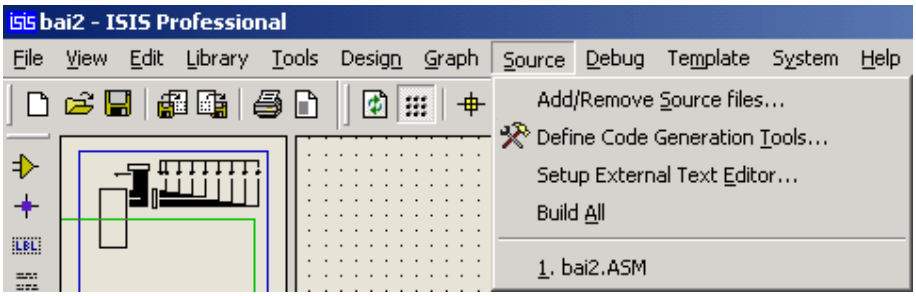
Do chưa có chương trình cho 89C51, ta nhấn vào nút New để tạo file. Trong phần **File name**, ta gõ vào tên chương trình (giả sử gõ vào bai2).



Nếu chưa có file bai2.ASM, Proteus sẽ xuất hiện thông báo yêu cầu tạo file, nhấn **Yes** để tạo:

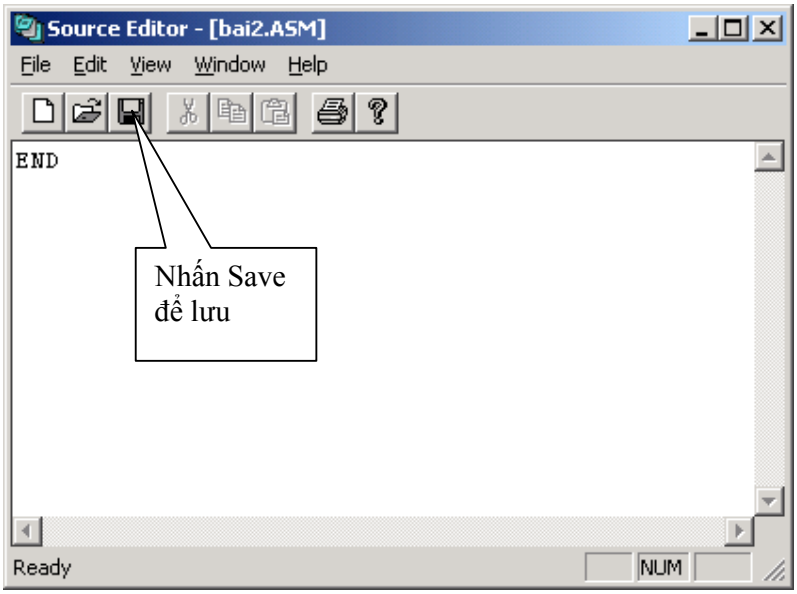


Sao khi tạo file thành công, trên menu Source sẽ xuất hiện thêm file bai2.ASM.

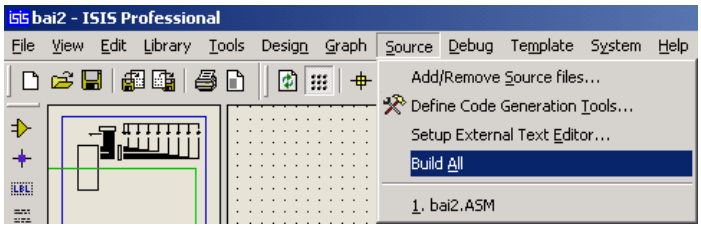


- **Bước 4:** Định nghĩa file thực thi cho 89C51

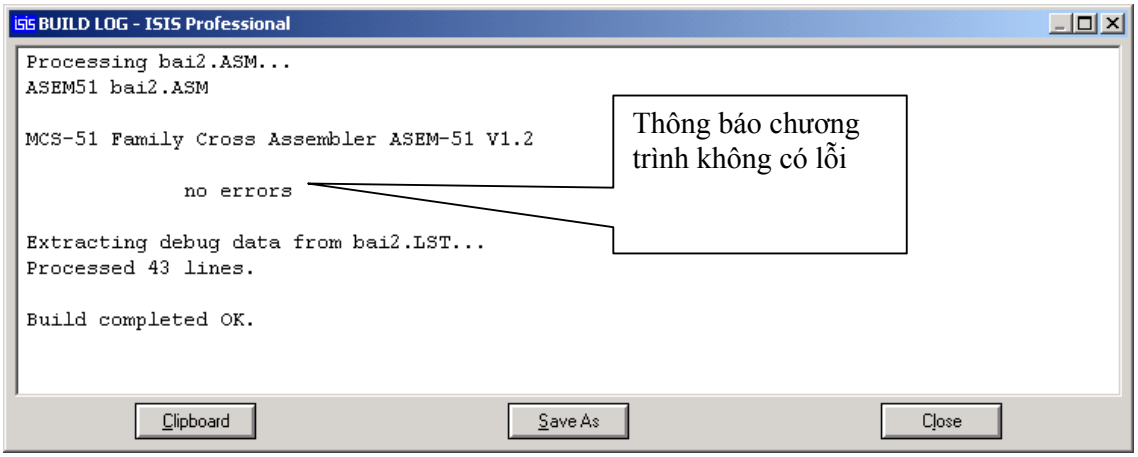
Chọn file bai2.ASM để soạn thảo chương trình nguồn, nhập vào **END** và nhấn nút **Save**.



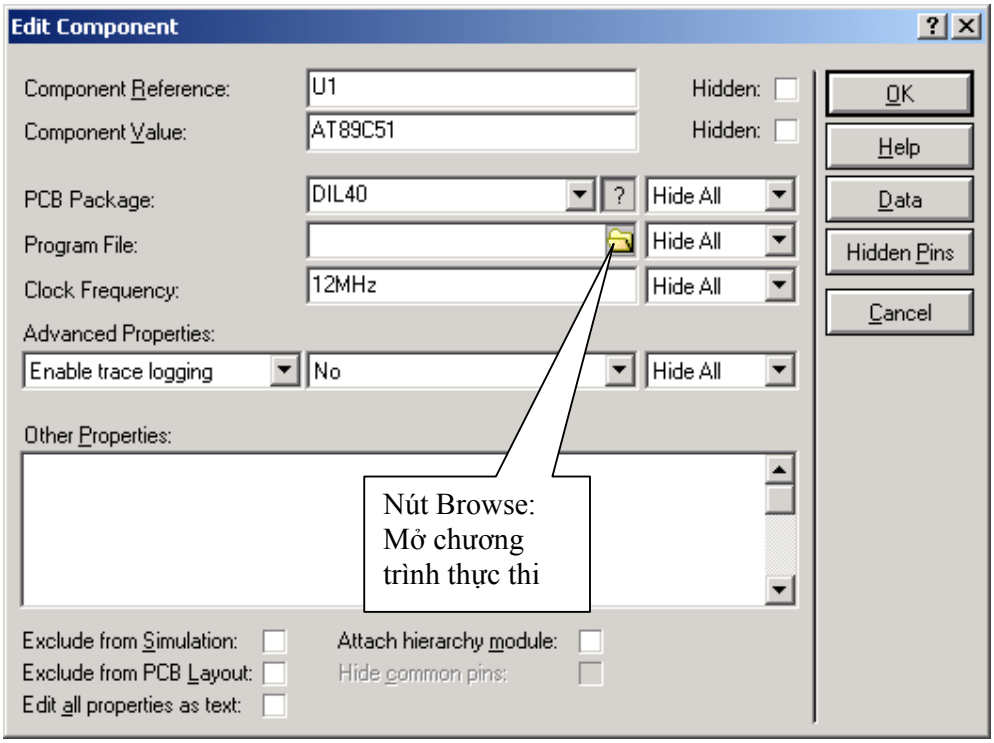
Sau khi lưu file nguồn, ta thực hiện dịch chương trình nguồn.



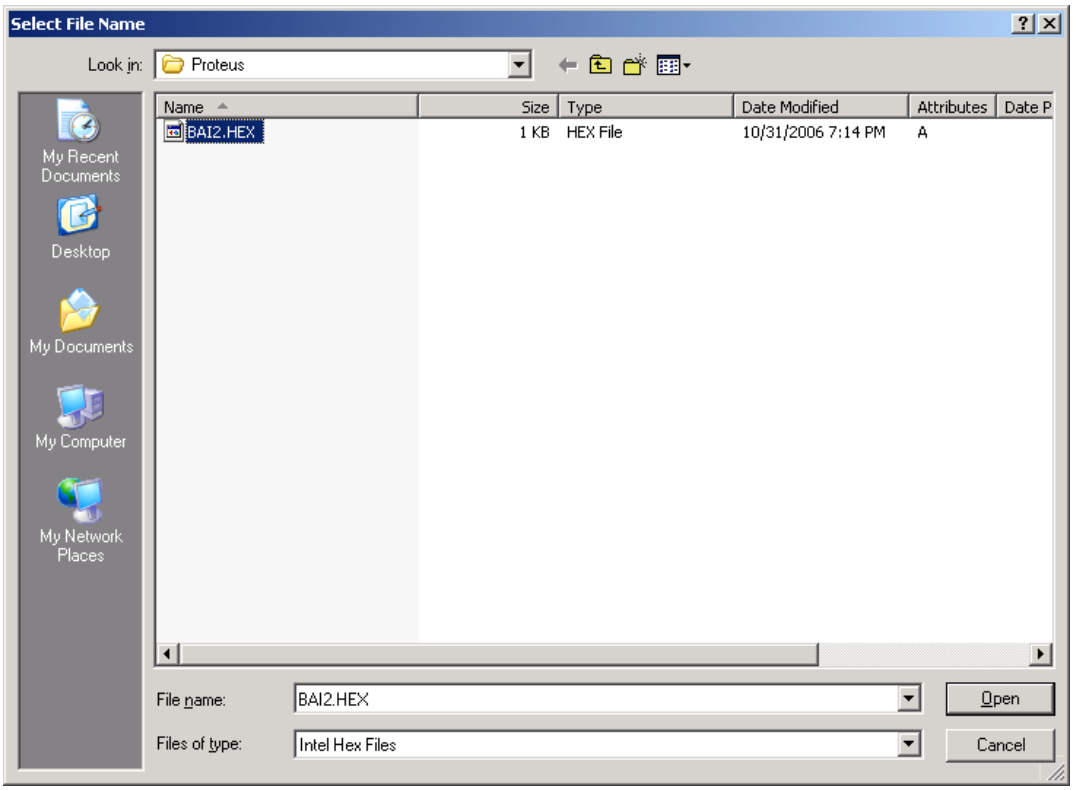
Khi biên dịch, nếu có lỗi, chương trình dịch sẽ thông báo lỗi, nếu không thì sẽ tạo ra file bai2.HEX.



Thực hiện gán file thực thi cho 89C51 bằng cách nhấn chuột phải lên 89C51 để chọn (89C51 sẽ chuyển sang màu đỏ) rồi nhấn chuột trái để mở cửa sổ thuộc tính của 89C51.



Nhấn vào nút **Browse** (hình vẽ trên) để mở chương trình thực thi, chọn chương trình là bai2.HEX



Nhấn nút Open để mở file, khi đó trong thuộc tính **Program File** của 89C51 sẽ có tên chương trình là bai2.HEX.

Sau khi gán file thực thi cho 89C51, ta chỉ cần thực hiện sửa chương trình nguồn và biên dịch lại mà không cần gán lại file thực thi.

Các lệnh cơ bản

- **Lệnh MOV**: di chuyển dữ liệu

VD: MOV A,30h ; chuyển nội dung của ô nhớ 30h vào thanh ghi A

MOV A,#30h ; chuyển giá trị 30h vào thanh ghi A

MOV A,R0 ; chuyển nội dung của thanh ghi R0 vào thanh ghi A

MOV A,@R0 ; chuyển nội dung của ô nhớ vào thanh ghi A, địa chỉ của ô nhớ chứa trong thanh ghi R0 (nếu R0 = 30h thì lệnh này tương đương lệnh MOV A,30h)

- **Lệnh INC**: tăng giá trị lên 1
- **Lệnh DEC**: giảm giá trị xuống 1
- **Lệnh SJMP**: lệnh nhảy không điều kiện

- **Lệnh DJNZ**: giảm và nhảy khi giá trị khác 0. Lệnh DJNZ thường dùng để tạo vòng lặp và có dạng sau:

  MOV R7,***#số_lần_lặp***

loop: ……

  ……

  DJNZ R7,loop

- **Lệnh CJNE**: so sánh và nhảy nếu không bằng

VD:

  CJNE A,#10,Khac

  ; Đoạn chương trình xử lý khi nội dung thanh ghi A là 10

  SJMP Tiep

Khac: JC Lonhon

  ; Đoạn chương trình xử lý khi nội dung thanh ghi A < 10

  SJMP Tiep

Lonhon:

  ; Đoạn chương trình xử lý khi nội dung thanh ghi A > 10

Tiep: …

- **Lệnh CALL**: gọi chương trình con
- **Lệnh RET, RETI**: lệnh trả về từ chương trình con hay chương trình phục vụ ngắt
- **Lệnh DIV AB**: chia nội dung thanh ghi A cho thanh ghi B, thương số chứa trong A và số dư chứa trong B.
- **Lệnh MOVC**: chuyển giá trị hằng số vào thanh ghi A, thường dùng cho mục đích tra bảng

VD: Lấy phần tử thứ 2 của bảng MaLed7:

  MOV DPTR,#MaLed7

  MOV A,#2

  MOVC A,@A+DPTR

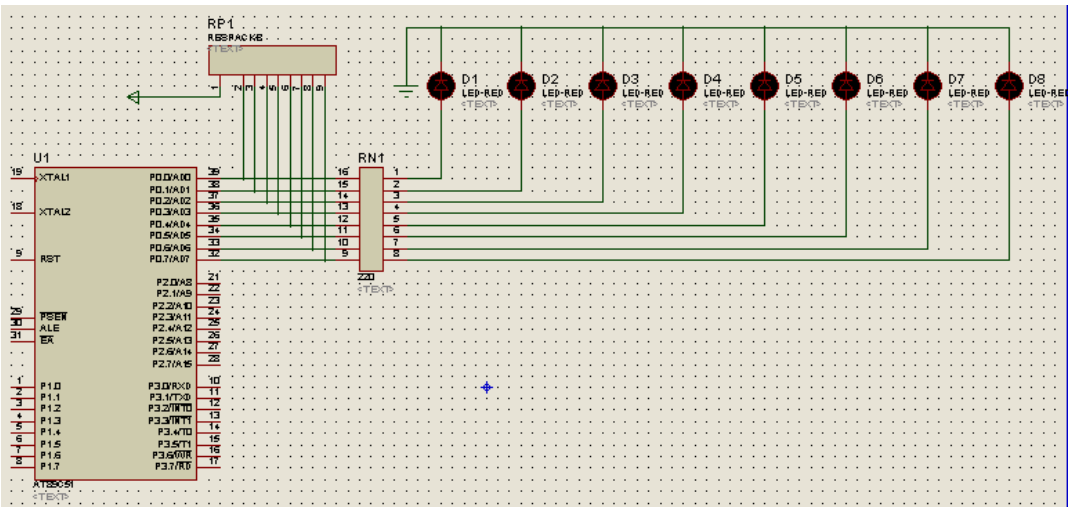
- **Lệnh PUSH**: lưu trữ nội dung thanh ghi vào stack
- **Lệnh POP**: lấy nội dung từ stack.

2. Tiến trình thực hiện

- Vẽ sơ đồ mạch như hình vẽ:
- Các linh kiện cho như sau:

Keywords	Category	Sub-category	Results
Led	Optoelectronics	LEDs	LED-RED
Resistor	Resistors	Resistor packs	RX8

Resistor	Resistors	Resistor packs	RESPACK-8
8951	All	All	AT89C51



Hiển thị dữ liệu ra Led

- Thực thi chương trình sau và quan sát trạng thái của Led:

  MOV P0,#0Fh ; Sáng 4 Led phải

  END

- Xoá điện trở thanh RP1 rồi thực thi chương trình, quan sát kết quả. Rút ra kết luận về tác dụng của điện trở kéo lên nguồn RP1.
- Thay đổi chương trình để 4 Led bên phải sáng, 2 Led giữa sáng, 2 Led ngoài cùng sáng.

- Thực thi chương trình sau và quan sát trạng thái của Led:

Main:

```
        MOV P0,#0FFh     ; Sáng 8 Led
        CALL Delay
        MOV P0,#0        ; Tắt 8 Led
        CALL Delay
        SJMP main
Delay:
        PUSH 07h
        PUSH 06h
        MOV R6,#255
Delay1:
        MOV R7,#255
        DJNZ R7,$
```

```
        DJNZ R6,Delay1
        POP 06h
        POP 07h
        RET
END
```

- Thay đoạn in đậm bằng đoạn chương trình sau và quan sát trạng thái các Led:

```
Main:
        MOV P0,#01h
        CALL Delay
        MOV P0,#02h
        CALL Delay
        MOV P0,#04h
        CALL Delay
        MOV P0,#08h
        CALL Delay
        MOV P0,#10h
        CALL Delay
        MOV P0,#20h
        CALL Delay
        MOV P0,#40h
        CALL Delay
        MOV P0,#80h
        CALL Delay
        SJMP main
```

- Thay đổi chương trình để Led sáng từ trong ra ngoài.
- Thay thế đoạn in đậm bằng đoạn chương trình sau và quan sát trạng thái các Led:

```
Main:
        MOV R0,#0
        MOV DPTR,#MaLed
Lap:
        MOV A,R0
        MOVC A,@A+DPTR
        MOV P0,A
```

```
        CALL Delay
        INC R0
        CJNE R0,#9,Lap
        SJMP main
MaLed: DB 00h,01h,03h,07h,0Fh,1Fh,3Fh,7Fh,0FFh
```

- Thay đổi chương trình để Led sáng tuỳ ý.

Kiểm tra các lệnh số học

- Thực thi chương trình sau và kiểm tra kết quả:

```
MOV A,#19h
ADD A,#72h
MOV P0,A
END
```

- Thực thi chương trình sau và kiểm tra kết quả:

```
MOV A,#57h
MOV B,#10
 DIV AB
MOV P0,A
MOV A,B
MOV P1,A
END
```

BÀI 3: ĐIỀU KHIỂN LED 7 ĐOẠN

❖ **MỤC ĐÍCH**

Giúp sinh viên khảo sát các vấn đề sau:

- Sử dụng phần mềm Proteus để mô phỏng mạch điện.
- Tìm hiểu các phương pháp hiển thị dữ liệu trên Led 7 đoạn dùng 89C51.

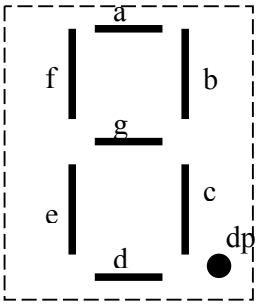
❖ **THIẾT BI SỬ DUNG**

- Máy vi tính.
- Phần mềm Proteus

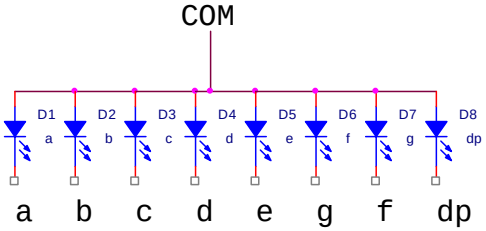
1. Cơ sở lý thuyết

Cấu trúc và mã hiển thị dữ liệu trên Led 7 đoạn

- Dạng Led



- Led Anode chung



Đối với dạng Led anode chung, chân COM phải có mức logic 1 và muốn sáng Led thì tương ứng các chân a – f, dp sẽ ở mức logic 0.

Bảng mã cho Led Anode chung (a là MSB, dp là LSB):

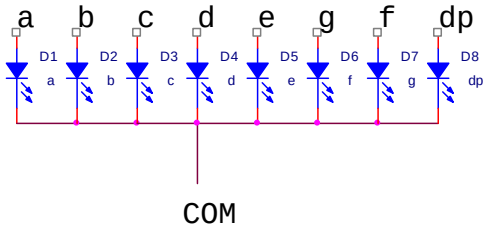
Số	a	b	c	d	e	f	g	dp	Mã hex
0	0	0	0	0	0	0	1	1	03h
1	1	0	0	1	1	1	1	1	9Fh
2	0	0	1	0	0	1	0	1	25h
3	0	0	0	0	1	1	0	1	0Dh
4	1	0	0	1	1	0	0	1	99h
5	0	1	0	0	1	0	0	1	49h

6	0	1	0	0	0	0	0	1	41h
7	0	0	0	1	1	1	1	1	1Fh
8	0	0	0	0	0	0	0	1	01h
9	0	0	0	0	1	0	0	1	09h

Bảng mã cho Led Anode chung (a là LSB, dp là MSB):

Số	**dp**	**g**	**f**	**e**	**d**	**c**	**b**	**a**	**Mã hex**
0	1	1	0	0	0	0	0	0	0C0h
1	1	1	1	1	1	0	0	1	0F9h
2	1	0	1	0	0	1	0	0	0A4h
3	1	0	1	1	0	0	0	0	0B0h
4	1	0	0	1	1	0	0	1	99h
5	1	0	0	1	0	0	1	0	92h
6	1	0	0	0	0	0	1	0	82h
7	1	1	1	1	1	0	0	0	0F8h
8	1	0	0	0	0	0	0	0	80h
9	1	0	0	1	0	0	0	0	90h

- Led Cathode chung



Đối với dạng Led Cathode chung, chân COM phải có mức logic 0 và muốn sáng Led thì tương ứng các chân a – f, dp sẽ ở mức logic 1.

Bảng mã cho Led Cathode chung (a là MSB, dp là LSB):

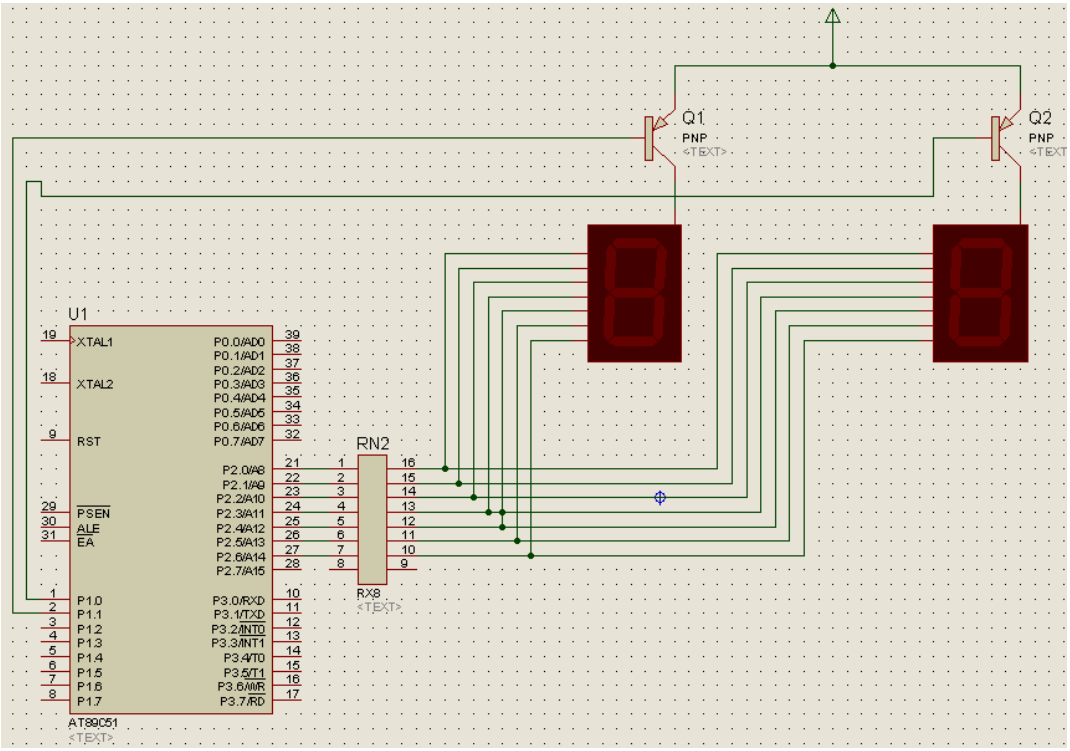
Số	**a**	**b**	**c**	**d**	**e**	**f**	**g**	**dp**	**Mã hex**
0	1	1	1	1	1	1	0	0	0FCh
1	0	1	1	0	0	0	0	0	60h
2	1	1	0	1	1	0	1	0	0DAh
3	1	1	1	1	0	0	1	0	0F2h
4	0	1	1	0	0	1	1	0	66h
5	1	0	1	1	0	1	1	0	0B6h
6	1	0	1	1	1	1	1	0	0BEh
7	1	1	1	0	0	0	0	0	0E0h
8	1	1	1	1	1	1	1	0	0FEh
9	1	1	1	1	0	1	1	0	0F6h

Bảng mã cho Led Anode chung (a là LSB, dp là MSB):

Số	dp	g	f	e	d	c	b	a	Mã hex
0	0	0	1	1	1	1	1	1	3Fh
1	0	0	0	0	0	1	1	0	06h
2	0	1	0	1	1	0	1	1	5Bh
3	0	1	0	0	1	1	1	1	4Fh
4	0	1	1	0	0	1	1	0	66h
5	0	1	1	0	1	1	0	1	6Dh
6	0	1	1	1	1	1	0	1	7Dh
7	0	0	0	0	0	1	1	1	07h
8	0	1	1	1	1	1	1	1	7Fh
9	0	1	1	0	1	1	1	1	6Fh

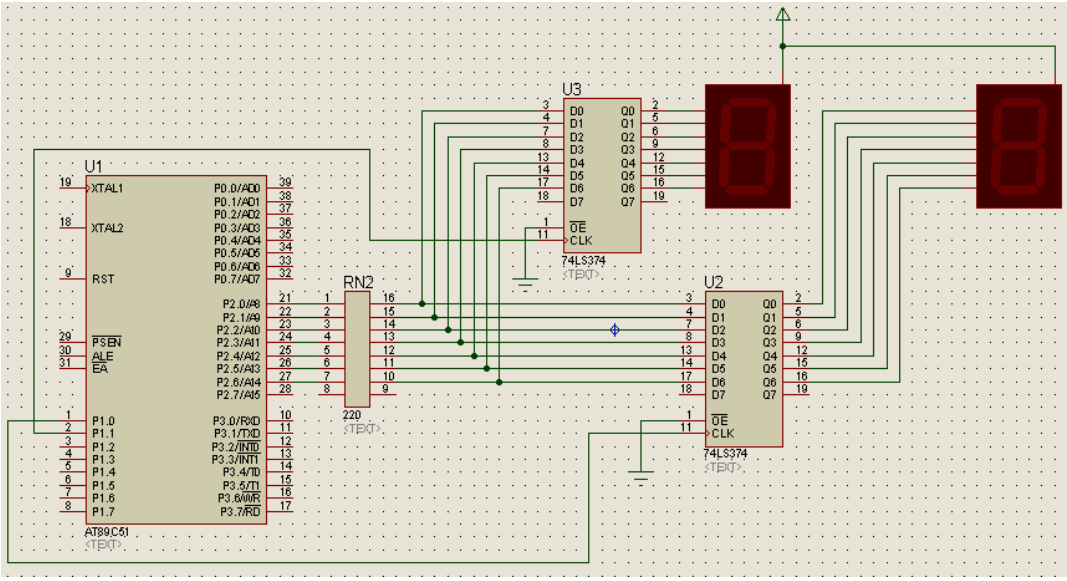
Dùng phương pháp quét

Khi kết nối chung các đường dữ liệu của Led 7 đoạn (hình vẽ), ta không thể cho các Led này sáng đồng thời (do ảnh hưởng lẫn nhau giữa các Led) mà phải thực hiện phương pháp quét, nghĩa là tại mỗi thời điểm chỉ sáng một Led và tắt các Led còn lại. Do hiện tượng lưu ảnh của mắt, ta sẽ thấy các Led sáng đồng thời.



Dùng phương pháp chốt

Khi thực hiện tách riêng các đường dữ liệu của Led, ta có thể cho phép các Led sáng đồng thời mà sẽ không có hiện tượng ảnh hưởng giữa các Led. IC chốt cho phép lưu trữ dữ liệu cho các Led có thể sử dụng là 74LS373, 74LS374.



2. Tiến trình thực hiện

Dùng phương pháp quét

Sử dụng mạch như hình vẽ phần trên với các linh kiện:

Keywords	Category	Sub-category	Results	Value
7seg	All	All	7SEG-COM-ANODE	
8951	All	All	AT89C51	
Pnp	Transistor	Generic	PNP	
Resistor	Resistors	Resistor Packs	RX8	220

- Thực hiện đoạn chương trình sau để hiển thị số 26 ra 2 Led 7 đoạn:

```
main:
        MOV     P2,#82h     ; Mã của số 6
        CLR     P1.0        ; Hiện số
        CALL    Delay
        SETB    P1.0
        MOV     P2,#0A4H    ; Mã của số 2
        CLR     P1.1
        CALL    Delay
        SETB    P1.1
```

```
        SJMP    main
Delay:
        PUSH    07H
        MOV     R7,#100
        DJNZ    R7,$
        POP     07H
        RET
END
```

- Sửa đoạn chương trình trên để hiển thị số 15, 37 ra 2 Led 7 đoạn.
- Bỏ các lệnh SETB và nhận xét tác dụng của các lệnh này.

Dùng phương pháp chốt

Sử dụng mạch như hình vẽ phần trên với các linh kiện:

Keywords	Category	Sub-category	Results	Value
7seg	All	All	7SEG-COM-ANODE	
8951	All	All	AT89C51	
374	74 TTL Series	All	74LS374	
Resistor	Resistors	Resistor Packs	RX8	220

- Thực hiện đoạn chương trình sau để hiển thị số 08 ra 2 Led 7 đoạn:

```
        MOV     P2,#80h         ; Mã của số 8
        CLR     P1.0
        SETB    P1.0
        MOV     P2,#0C0H        ; Mã của số 0
        CLR     P1.1
        SETB    P1.1
END
```

- Thực hiện đoạn chương trình trên để hiển thị số tăng dần từ 00 đến 99 ra 2 Led 7 đoạn.

```
main:
        MOV     30H,#0          ; Ô nhớ 30h chứa giá trị xuất ra Led
lap:
        MOV     A,30H
        MOV     B,#10           ; A chứa số hàng chục, B, chứa số
        DIV     AB              ; hàng đơn vị

        MOV     DPTR,#Maled7
```

```
        MOVC    A,@A+DPTR       ; Chuyển sang mã Led 7 đoạn
        MOV     P2,A
        CLR     P1.1            ; Xuất số hàng chục
        SETB    P1.1

        MOV     A,B
        MOVC    A,@A+DPTR
        MOV     P2,A
        CLR     P1.0            ; Xuất số hàng đơn vị
        SETB    P1.0

        CALL    Delay
        INC     30H             ; Tăng ô nhớ 30h
        MOV     A,30H
        CJNE    A,#100,lap      ; Nếu giá trị ô nhớ đả tăng đến 100
        SJMP    main            ; thì giảm về 0
        ;-----------------
Maled7: DB      0C0h,0F9h,0A4h,0B0h,99h,92h,82h,0F8h,80h,90h
        ;-----------------
Delay:
        PUSH    07
        PUSH    06
        MOV     R6,#255
Delay1:
        MOV     R7,#255
        DJNZ    R7,$
        DJNZ    R6,Delay1
        POP     06
        POP     07
        RET
END
```

- Sửa đoạn chương trình trên để giá trị xuất ra 2 Led 7 đoạn tăng dần từ 00 - 59.
- Sửa đoạn chương trình trên để giá trị xuất ra 2 Led 7 đoạn tăng dần từ 00 - 23.

- Sửa đoạn chương trình trên để giá trị xuất ra 2 Led 7 đoạn giảm dần từ 99 - 00.
- Sửa đoạn chương trình trên để giá trị xuất ra 2 Led 7 đoạn giảm dần từ 59 - 00.
- Sửa đoạn chương trình trên để giá trị xuất ra 2 Led 7 đoạn giảm dần từ 23 - 00.

BÀI 4: CÔNG TẮC NHẤN

❖ **MỤC ĐÍCH**

Giúp sinh viên khảo sát các vấn đề sau:

- Sử dụng phần mềm Proteus để mô phỏng mạch điện.
- Tìm hiểu cách thức kiểm tra công tăc có nhấn hay không và các ứng dụng của chúng dùng trong 89C51.

❖ **THIẾT BI SỬ DUNG**

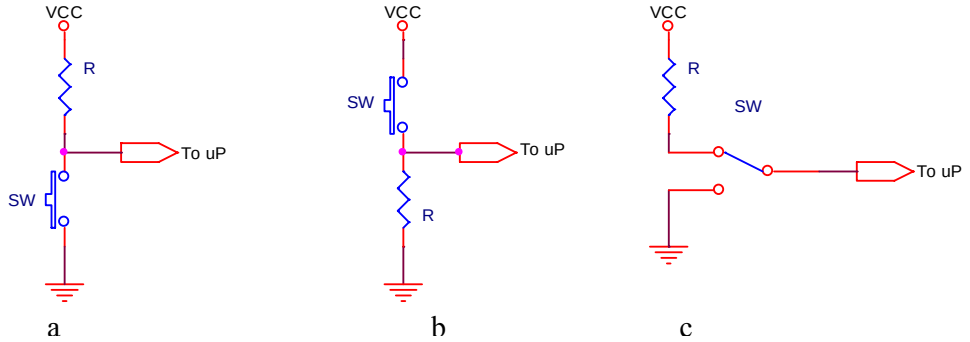
- Máy vi tính.
- Phần mềm Proteus

1. Cơ sở lý thuyết

Công tắc đơn

Các phím đơn dùng để điều khiển khi hệ thống không đòi hỏi nhiều giá trị nhập (chẳng như chỉ cần các điều khiển đóng mở thiết bị). Khi thực hiện kiểm tra phím nhấn, vấn đề cần thiết là phải thực hiện chống dội. Quá trình chống dội có thể thực hiện bằng phần mềm: Do thời gian dội của phím vào khoảng 20ms nên quá trình chống dội bằng phần mềm đơn giản là tạo một thời gian trễ đủ lớn để chương trình bỏ qua ảnh hưởng khi dội.

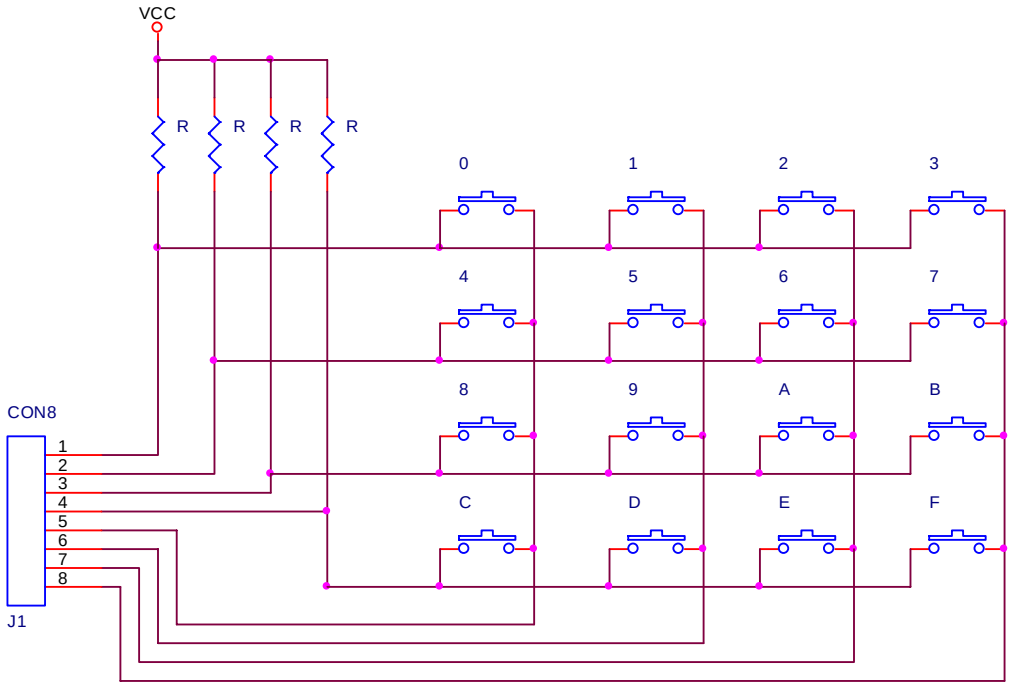
Khi thực hiện giao tiếp giữa công tắc đơn và vi điều khiển MCS-51 thì cần lưu ý phải set bit tương ứng của vi điều khiển lên mức logic 1 mới có thể đọc dữ liệu vào. Phần cứng giao tiếp có thể mô tả như hình vẽ, tuy nhiên đối với họ MCS-51, các port đã có điện trở kéo lên nguồn (trừ port 0) nên đối với sơ đồ hình a và c có thể không cần điện trở R.



Bàn phím Hex

khi không nhấn phím thì hàng của bàn phím Hex nối với Vcc thông qua điện trở R nên có mức logic 1. Để phân biệt được trạng thái của phím nhấn thì mức logic khi nhấn phím phải là mức logic 0. Mà khi nhấn một phím nào đó thì tương ứng hàng và cột của bàn phím Hex sẽ kết nối với nhau. Do đó, để thực hiện kiểm tra một phím thì ta phải cho trước cột chứa phím tương ứng ở mức logic 0, sau đó kiểm tra hàng của phím, nếu hàng = 0 thì có nhấn phím còn hàng = 1 thì không nhấn phím.

Ví dụ như muốn kiểm tra phím 4 thì ta cho cột chứa phím 4 ở mức logic 0 (chân 5 của J1, các cột khác = 1, nghĩa là dữ liệu tại J1 là 1000xxxxb), sau đó thực hiện kiểm tra chân 2 của J1 (hàng của phím 4), nếu chân này = 0 thì phím 4 được nhấn.



2. Tiến trình thực hiện

Công tắc đơn

Sử dụng mạch như hình vẽ trang bên với các linh kiện:

Keywords	Category	Sub-category	Results	Value
7seg	All	All	7SEG-COM-ANODE	
8951	All	All	AT89C51	
374	74 TTL Series	All	74LS374	
Button	Switches & Relays	All	Button	
Resistor	Resistors	Resistor Packs	RX8	220

- Thực hiện chương trình sau:

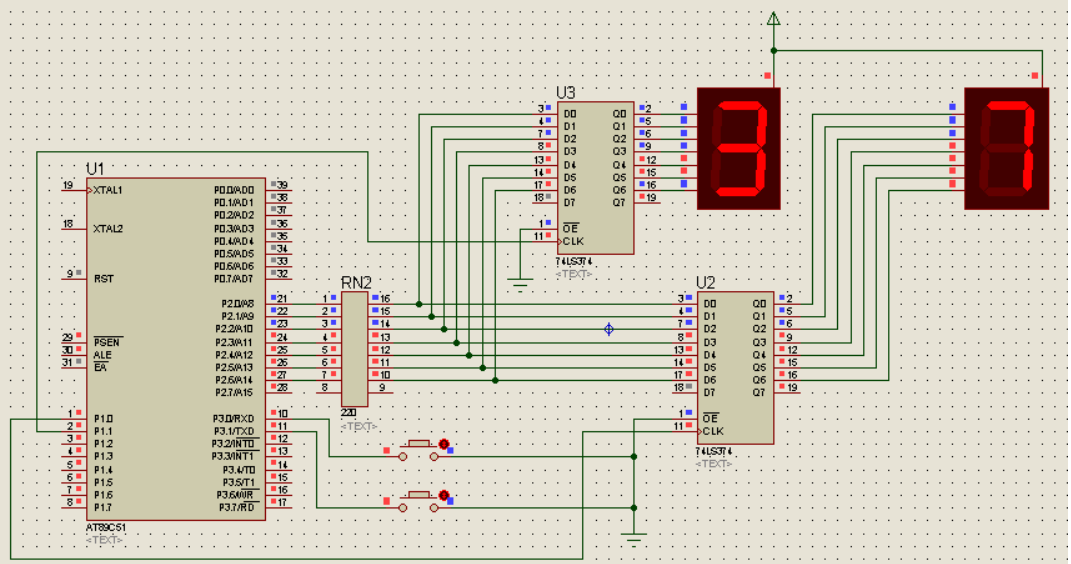
```
        MOV     30H,#0
        SJMP    Hienthi
Main:
        JNB     P3.0,Sw1     ; Nếu P3.0 = 0 thì nhấn SW1
        JNB     P3.1,Sw2     ; Nếu P3.1 = 0 thì nhấn SW2
        SJMP    Main
SW1:
        INC     30H          ; Tăng ô nhớ 30h
```

```
        MOV     A,30H
        CJNE    A,#100,Hienthi      ; Nếu giá trị ô nhớ 30h = 100
        MOV     30H,#0              ; thì gán 30h = 0
        SJMP    Hienthi             ; Hiển thị ra Led 7 đoạn
        ;-------------
Sw2:
        DEC     30H                 ; Giảm ô nhớ 30h
        MOV     A,30H
        CJNE    A,#255,Hienthi      ; Nếu giá trị giảm = -1 (255)
        MOV     30H,#99             ; thì gán 30h = 99
        SJMP    Hienthi
        ;-------------
Hienthi:
        MOV     A,30H
        MOV     B,#10
        DIV     AB
        MOV     DPTR,#Maled7
        MOVC    A,@A+DPTR
        MOV     P2,A
        CLR     P1.1
        SETB    P1.1
        MOV     A,B
        MOV     DPTR,#Maled7
        MOVC    A,@A+DPTR
        MOV     P2,A
        CLR     P1.0
        SETB    P1.0
        CALL    Delay
        SJMP    Main
Maled7: DB 0C0H,0F9H,0A4H,0B0H,99H,92H,82H,0F8H,80H,90H
        ;--------------
Delay:
        PUSH    07
        PUSH    06
        MOV     R6,#100
```

```
Delay1:
        MOV     R7,#255
        DJNZ    R7,$
        DJNZ    R6,Delay1
        POP     06
        POP     07
        RET

        END
```

Nhấn vào các công tắc và quan sát trạng thái các Led.



- Bỏ lệnh **CALL Delay** trong chương trình trên và quan sát trạng thái các Led, có nhận xét gì so với khi có thời gian Delay.

Bàn phím Hex

Sử dụng mạch như hình vẽ trang bên với các linh kiện:

Keywords	Category	Sub-category	Results	Value
7seg	All	All	7SEG-COM-ANODE	
8951	All	All	AT89C51	
374	74 TTL Series	All	74LS374	
Resistor	Resistors	Resistor Packs	RX8	220
Keypad	Switches & Relays	All	KEYPAD-SMALLCALC	

- Thực hiện chương trình sau:

```
Main:
        MOV     P3,#0FEH        ; Chọn cột chứa các phím 7,8,9,÷
        JNB     P3.4,Sw7
```

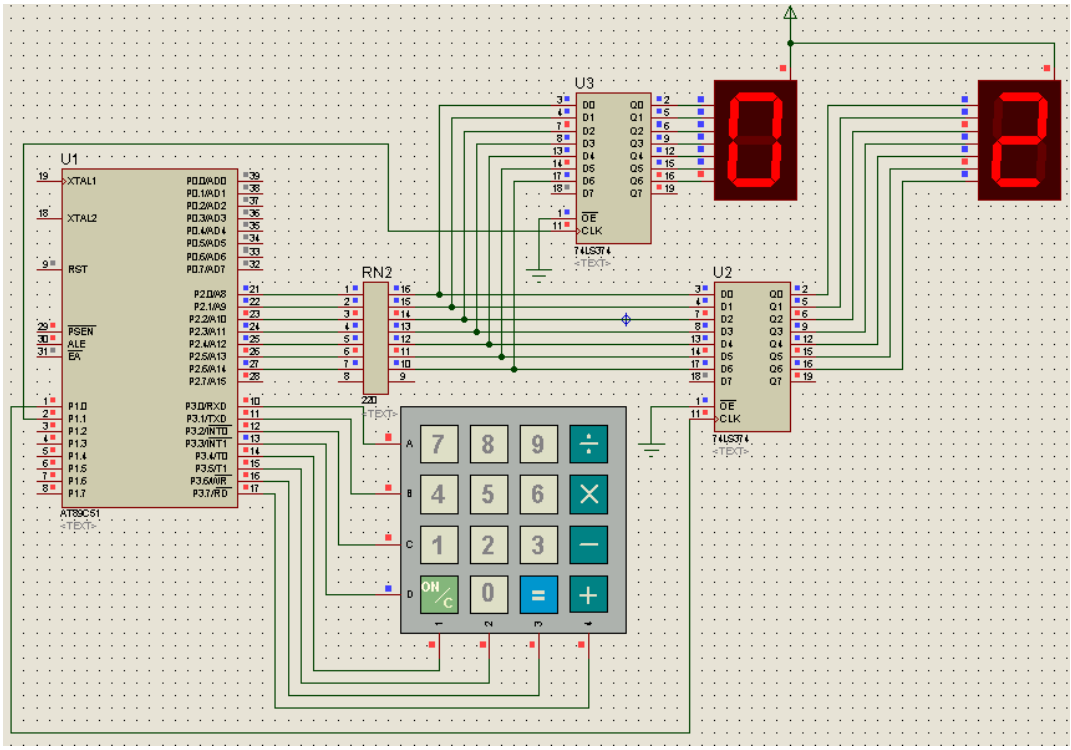
```
        JNB     P3.5,Sw8
        JNB     P3.6,Sw9
        JNB     P3.7,Swchia
        MOV     P3,#0FDH    ; Chọn cột chứa các phím 4,5,6,X
        JNB     P3.4,Sw4
        JNB     P3.5,Sw5
        JNB     P3.6,Sw6
        JNB     P3.7,Swnhan
        MOV     P3,#0FBH    ; Chọn cột chứa các phím 1,2,3,-
        JNB     P3.4,Sw1
        JNB     P3.5,Sw2
        JNB     P3.6,Sw3
        JNB     P3.7,Swtru
        MOV     P3,#0F7H    ; Chọn cột chứa các phím ON/C,0,=,+
        JNB     P3.4,Swon
        JNB     P3.5,Sw0
        JNB     P3.6,Swbang
        JNB     P3.7,Swcong
        SJMP Main
        ;-------------
SW0:
        MOV 30H,#0
        SJMP Hienthi
        ;-------------
SW1:
        MOV 30H,#1
        SJMP Hienthi
        ;-------------
SW2:
        MOV 30H,#2
        SJMP Hienthi
        ;-------------
SW3:
        MOV 30H,#3
        SJMP Hienthi
        ;-------------
SW4:
```

```
        MOV 30H,#4
        SJMP Hienthi
        ;-------------
SW5:
        MOV 30H,#5
        SJMP Hienthi
        ;-------------
SW6:
        MOV 30H,#6
        SJMP Hienthi
        ;-------------
SW7:
        MOV 30H,#7
        SJMP Hienthi
        ;-------------
SW8:
        MOV 30H,#8
        SJMP Hienthi
        ;-------------
SW9:
        MOV 30H,#9
        SJMP Hienthi
        ;-------------
Swon:
        MOV 30H,#10
        SJMP Hienthi
        ;-------------
Swbang:
        MOV 30H,#11
        SJMP Hienthi
        ;-------------
Swcong:
        MOV 30H,#12
        SJMP Hienthi
        ;-------------
Swtru:
        MOV 30H,#13
```

```
        SJMP Hienthi
        ;-------------
Swnhan:
        MOV 30H,#14
        SJMP Hienthi
        ;-------------
Swchia:
        MOV 30H,#15
        SJMP Hienthi
        ;-------------
Hienthi:
        MOV     A,30H
        MOV     B,#10
        DIV     AB
        MOV     DPTR,#MALED7
        MOVC    A,@A+DPTR
        MOV     P2,A
        CLR     P1.1
        SETB    P1.1
        MOV     A,B
        MOV     DPTR,#MALED7
        MOVC    A,@A+DPTR
        MOV     P2,A
        CLR     P1.0
        SETB    P1.0
        CALL    Delay
        LJMP    Main
Maled7: DB 0C0H,0F9H,0A4H,0B0H,99H,92H,82H,0F8H,80H,90H
        ;--------------
Delay:
        PUSH    07
        PUSH    06
        MOV     R6,#100
Delay1:
        MOV     R7,#255
```

```
DJNZ    R7,$
DJNZ    R6,Delay1
POP     06
POP     07
RET

END
```



- Sửa chương trình để:
    - Nhấn phím +: thực hiện cộng nội dung ô nhớ 30h với 20 và xuất ra 2 Led 7 đoạn.
    - Nhấn phím -: thực hiện trừ nội dung ô nhớ 30h với 1 và xuất ra 2 Led 7 đoạn.
    - Nhấn phím x: thực hiện nhân nội dung ô nhớ 30h với 3 và xuất ra 2 Led 7 đoạn.
    - Nhấn phím ÷: thực hiện chia nội dung ô nhớ 30h với 2 và xuất kết quả ra 2 Led 7 đoạn.
    - Nhấn phím ON/C: thực hiện xoá nội dung ô nhớ 30h (gán bằng 0) và xuất ra 2 Led 7 đoạn.

BÀI 5: TIMER

❖ **MỤC ĐÍCH**

Giúp sinh viên khảo sát các vấn đề sau:

- Sử dụng phần mềm Proteus để mô phỏng mạch điện.
- Tìm hiểu cách sử dụng Timer trong 89C51.

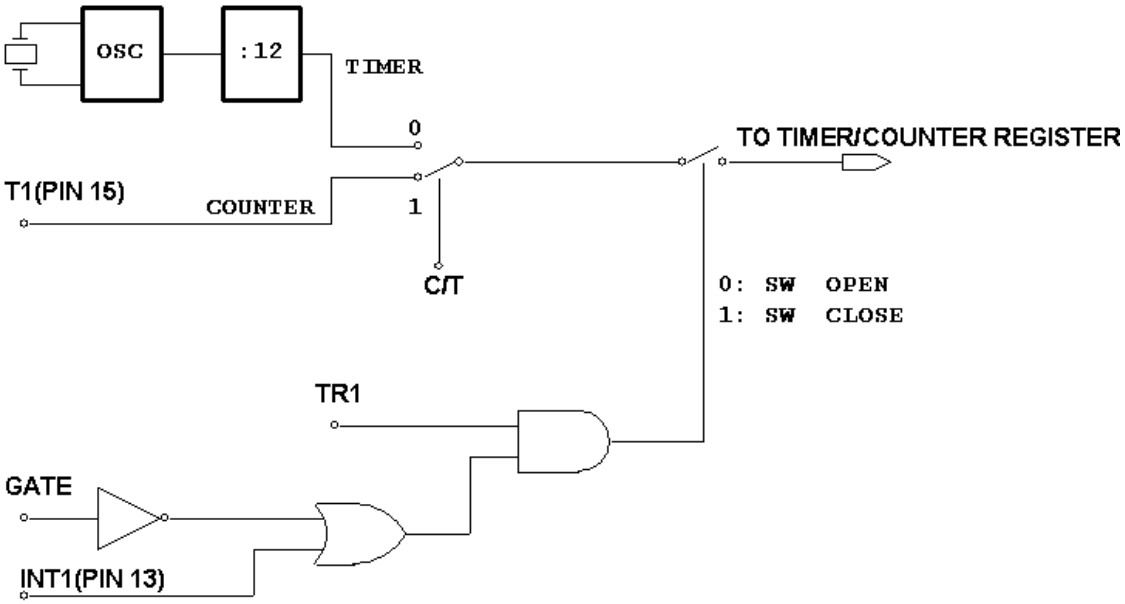
❖ **THIẾT BỊ SỬ DỤNG**

- Máy vi tính.
- Phần mềm Proteus

1. Cơ sở lý thuyết

Vi điều khiển 89C51 có hai thanh ghi timer/counter 16 bit. Các thanh ghi này có thể hoạt động ở một trong hai trạng thái timer hoặc counter. Mỗi thanh ghi gồm 2 thanh ghi 8 bit ghép lại:

Cấu trúc của bộ Timer/ Counter trong 89C51 như hình sau.



Hoạt động của bộ Timer/Counter được điều khiển bởi hai thanh ghi TCON và TMOD

Thanh ghi TCON (timer control): Là thanh ghi 8 bit, có thể truy xuất byte hoặc bit dùng để điều khiển hoạt động của Timer.

7	6	5	4	3	2	1	0
TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0

TF1: báo trạng thái tràn cho bộ Timer/Counter1

TR1: điều khiển cấp xung cho bộ Timer/Counter1

TF0: báo trạng thái tràn cho bộ Timer/Counter0

TR0: điều khiển cấp xung cho bộ Timer/Counter0

IE1, IT1, IE0, IT0: sử dụng cho ngắt ngoài 1 và ngắt ngoài 0 (không dùng cho Timer).

Thanh ghi TMOD (timer mode): Là thanh ghi 8 bit, chỉ có thể truy xuất byte dùng để xác định chế độ hoạt động của Timer.

7	6	5	4	3	2	1	0
GATE	C/$\overline{T}$	M1	M0	GATE	C/$\overline{T}$	M1	M0
ĐIỀU KHIỂN TIMER 1				**ĐIỀU KHIỂN TIMER 0**			

GATE, C/$\overline{T}$: điều khiển trạng thái hoạt động cho Timer/

M1, M0: chọn chế độ hoạt động cho Timer/Counter

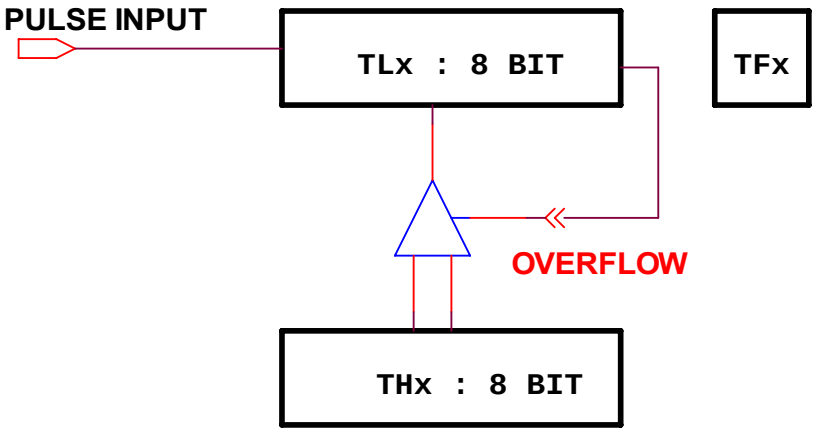
M1	M0	CHẾ ĐỘ	MÔ TẢ
0	0	0	Timer/Counter 13 bit
0	1	1	Timer/Counter 16 bit
1	0	2	Timer/Counter 8 bit, auto reload
1	1	3	Timer/Counter 8 bit

a/ Chế độ 0:

Thanh ghi THx và TLx kết hợp tạo thành bộ Timer/Counter 13 bit, khi tràn 13 bit thì cờ TFx sẽ đặt lên logic 1.

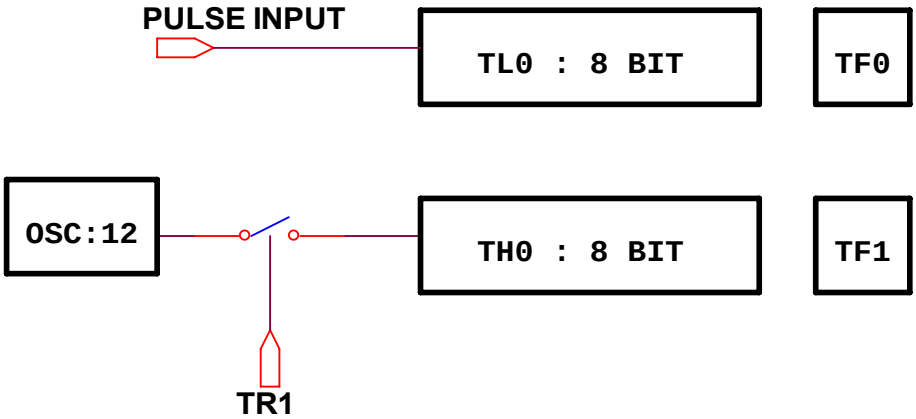
b/ Chế độ1 (16 bit):

c/ Chế độ 2 (8 bit Auto-Reload):



TLx được nạp giá trị ban đầu từ THx và bắt đầu đếm từ giá trị này khi có xung ở ngõ vào, khi tràn thì TFx sẽ đặt lên logic 1 đồng thời kích hoạt bộ khóa để nạp giá trị trong THx vào TLx.

d/ Chế độ 4:

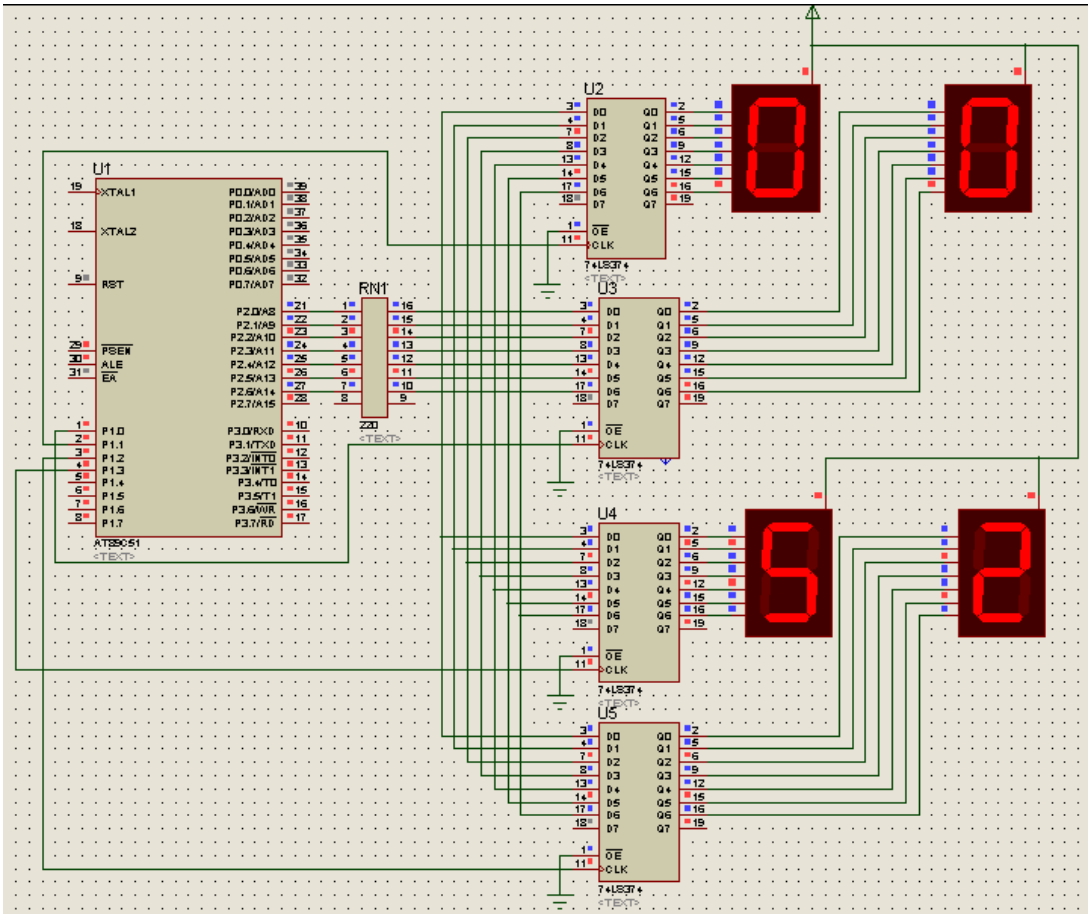


Trong chế độ này, TH1 và TL1 không được sử dụng thay vào đó là TH0 và TL0 hoạt động như 2 bộ Timer 8 bit (TL0) và Timer/Counter 8 bit (TL0). Tuy nhiên, tín hiệu mở xung cho TH0 không phải là TR0 mà là TR1.

2. Tiến trình thực hiện

Sử dụng mạch như hình vẽ trang bên với các linh kiện:

Keywords	Category	Sub-category	Results	Value
7seg	All	All	7SEG-COM-ANODE	
8951	All	All	AT89C51	
374	74 TTL Series	All	74LS374	
Resistor	Resistors	Resistor Packs	RX8	220



- Thực hiện chương trình sau (tạo một mạch đồng hồ đếm phút, giây):

```
        MOV     TMOD,#01H       ; Sử dụng Timer0, chế độ 16 bit
        MOV     30H,#0
        MOV     31H,#0
Lap:
        MOV     A,30H
        MOV     B,#10
        DIV     AB
        MOV     DPTR,#Maled7
        MOVC    A,@A+DPTR
        MOV     P2,A
        CLR     P1.1
        SETB    P1.1
        MOV     A,B
        MOVC    A,@A+DPTR
        MOV     P2,A
```

```
        CLR     P1.0
        SETB    P1.0

        MOV     A,31H
        MOV     B,#10
        DIV     AB
        MOV     DPTR,#Maled7
        MOVC    A,@A+DPTR
        MOV     P2,A
        CLR     P1.3
        SETB    P1.3
        MOV     A,B
        MOVC    A,@A+DPTR
        MOV     P2,A
        CLR     P1.2
        SETB    P1.2

        CALL    Delay
        INC     31H
        MOV     A,31H
        CJNE    A,#60,Lap     ; Đủ 60s (1 phút) thì
        MOV     31H,#0        ; giây = 0
        INC     30H           ; và tăng phút lên 1
        MOV     A,30H
        CJNE    A,#60,Lap     ; Đủ 60 phút thì
        MOV     30H,#0        ; phút = 0
        SJMP    Lap
Maled7: DB 0C0H,0F9H,0A4H,0B0H,99H,92H,82H,0F8H,80H,90H
Delay:
        PUSH    07
        MOV     R7,#20
        SETB    TR0
Delay1: ; Tạo trễ 50000 chu kỳ = 50000µs = 50ms
        MOV     TH0,#HIGH(-50000)
        MOV     TL0,#LOW(-50000)
```

```
        JNB     TF0,$ ; Chờ đến khi Timer tràn
        CLR     TF0
        DJNZ    R7,Delay1 ; R7 = 20 → lặp lại 20 lần → tạo trễ 20*50ms
        CLR     TR0
        POP     07
        RET

        END
```

- Sửa chương trình để thực hiện đếm giây và %giây.

```
        …
        CALL    Delay
        INC     31H
        MOV     A,31H
        CJNE    A,#100,Lap   ; Đủ 100% s (1 s) thì
        MOV     31H,#0       ; %giây = 0
        INC     30H          ; và tăng giây lên 1
        MOV     A,30H
        CJNE    A,#60,Lap    ; Đủ 60 s thì
        MOV     30H,#0       ; giây = 0
        SJMP    Lap
Delay: ; Tạo trễ 10ms
        PUSH    07
        SETB    TR0
        MOV     TH0,#HIGH(-10000)
        MOV     TL0,#LOW(-10000)
        JNB     TF0,$
        CLR     TF0
        CLR     TR0
        POP     07
        RET

        END
```

BÀI 6: INTERRUPT (NGẮT)

❖ **MỤC ĐÍCH**

Giúp sinh viên khảo sát các vấn đề sau:

- Sử dụng phần mềm Proteus để mô phỏng mạch điện.
- Tìm hiểu cách sử dụng ngắt trong 89C51.

❖ **THIẾT BỊ SỬ DỤNG**

- Máy vi tính.
- Phần mềm Proteus

1. Cơ sở lý thuyết

Thanh ghi IE (Interrupt Enable Register)

Thanh ghi IE dùng để cho phép hay cấm các ngắt hoạt động. Mặc định khi khởi động chương trình thì tất cả các ngắt đều bị cấm. Chức năng các bit trong thanh ghi IE cho trong bảng sau:

D7	D6	D5	D4	D3	D2	D1	D0
EA	-	-	ES	ET1	EX1	ET0	EX0

EA = 0: cấm tất cả các ngắt
= 1: cho phép ngắt tuỳ theo trạng thái các bit điều khiển tương ứng

ES = 0: cấm ngắt tại port nối tiếp
= 1: cho phép ngắt tại port nối tiếp

ET1 = 0: cấm ngắt tại Timer 1
= 1: cho phép ngắt tại Timer 1

EX1 = 0: cấm ngắt tại ngắt ngoài 1 ($\overline{INT1}$: chân P3.3)
= 1: cho phép ngắt tại ngắt ngoài 1

ET0 = 0: cấm ngắt tại Timer 0
= 1: cho phép ngắt tại Timer 0

EX0 = 0: cấm ngắt tại ngắt ngoài 1 ($\overline{INT0}$: chân P3.2)
= 1: cho phép ngắt tại ngắt ngoài 1

Để cho phép ngắt tại Timer 0, ta phải có: EA = 1 và ET0 = 1. Nội dung của thanh ghi IE khi đó là:

D7	D6	D5	D4	D3	D2	D1	D0	
1	0	0	0	0	0	1	0	= 82h

Chương trình có thể thực hiện như sau:

```
SETB EA
SETB ET0
```

Hay có thể viết:

```
MOV  IE,#82h
```

Danh sách các ngắt trong 89C51

- Ngắt ngoài 0:

Địa chỉ vector ngắt	0003H
Khai báo sử dụng ngắt	- SETB EA - SETB EX0 - SETB IT0 (ngắt cạnh)
Sự kiện ngắt:	Xuất hiện cạnh xuống (hoặc mức thấp) tại chân INT0

- Ngắt timer0:

Địa chỉ vector ngắt	000BH
Khai báo sử dụng ngắt	- SETB EA - SETB ET0
Sự kiện ngắt	Tràn timer 0

- Ngắt ngoài 1:

Địa chỉ vector ngắt	00013H
Khai báo sử dụng ngắt	- SETB EA - SETB EX1 - SETB IT1 (ngắt cạnh)
Sự kiện ngắt:	Xuất hiện cạnh xuống (hoặc mức thấp) tại chân INT1

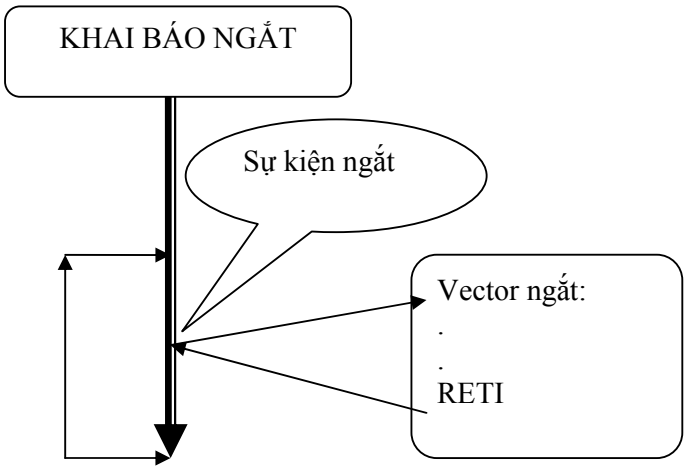
- Ngắt timer1:

Địa chỉ vector ngắt	0001BH
Khai báo sử dụng ngắt	- SETB EA - SETB ET1
Sự kiện ngắt	Tràn timer 1

- Ngắt truyền thông (truyền/nhận UART):

Địa chỉ vector ngắt	00023H
Khai báo sử dụng ngắt	- SETB EA - SETB ES
Sự kiện ngắt	Nhận được một byte hoặc truyền xong một byte trong SBUF
Tốc độ truyền nhận MODE autoreload (timer1)	- TH1 = -3 ; 9600bps - TH1 = -6 ; 4800bps - TH1 = -12 ; 2400bps - TH1 = -24 ; 1200bps

SƠ ĐỒ CHƯƠNG TRÌNH KHI CÓ SỬ DỤNG NGẮT:



Cấu trúc chương trình sử dụng ngắt

Chương trình sử dụng ngắt bao gồm 2 phần: phần chương trình chính và phần xử lý ngắt. Một đoạn chương trình ví dụ sử dụng ngắt ngoài 0 như sau:

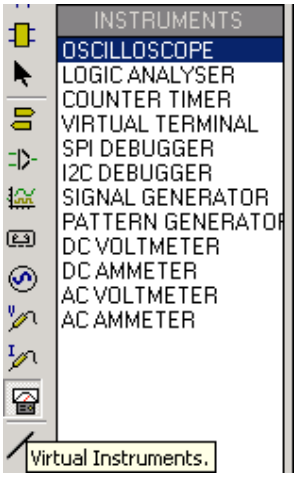
```
        ORG  0          ; Địa chỉ bắt đầu chương trình
        LJMP main
        ORG  0003h      ; Địa chỉ chương trình phục vụ ngắt cho INT0
        LJMP int0_isr
Main:
        ; khai báo sử dụng ngắt
        SETB EA
        SETB EX0        ; Có thể thay 2 lệnh này bằng lệnh MOV IE,#81h
        ; chương trình chính
Int0_isr:
        ; chương trình phục vụ ngắt
        RETI

        END
```

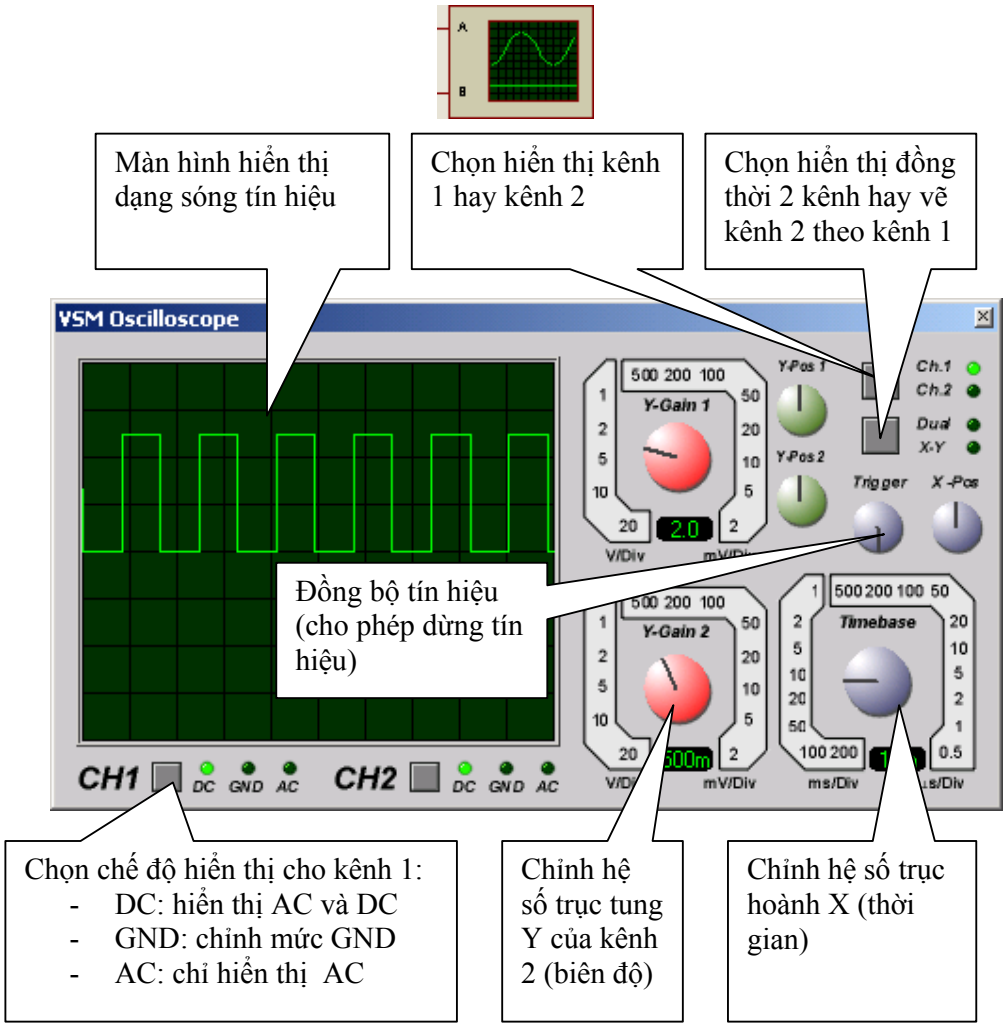
VSM Oscilloscope

VSM Oscilloscope là thiết bị cho phép hiển thị dạng tín hiệu tương tự, bao gồm 2 kênh A và B. VSM có khả năng đo 2 tín hiệu đồng thời.

Vị trí của Oscilloscope trong cửa sổ thiết kế như sau:



Hình ảnh của Oscilloscope khi thiết kế và thực thi chương trình như sau:

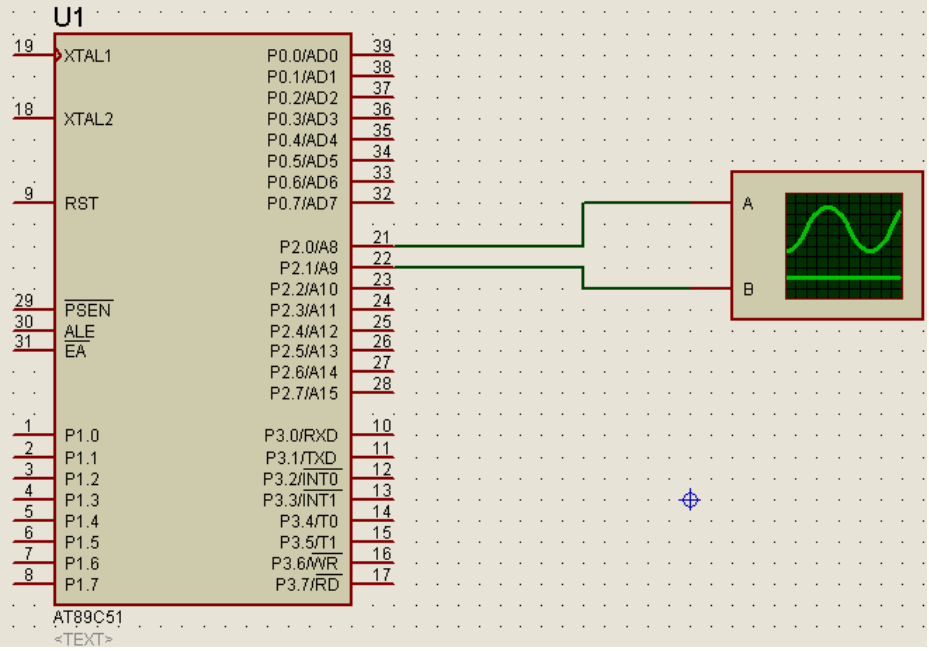


2. Tiến trình thực hiện

Ngắt Timer

Tạo xung vuông

Sử dụng mạch như hình vẽ:



- Thực hiện chương trình dùng ngắt Timer 0 tạo xung vuông tần số f = 5 KHz tại P2.0 như sau:

```
        ORG 0000h
        LJMP main
        ORG 000Bh
        LJMP timer0_isr
main:
        MOV IE,#82h
        MOV  TMOD,#02h
        MOV TH0,#(-100) ; f = 5KHz → T = 1/f = 0.2ms = 200 µs → thời gian
        MOV TL0,#(-100) ; trễ là 100 µs → giá trị đếm của Timer là 100
        SETB TR0
Here:
        SJMP here
        ;-----------
Timer0_isr:
        CPL P2.0
```

```
        RETI
        END
```

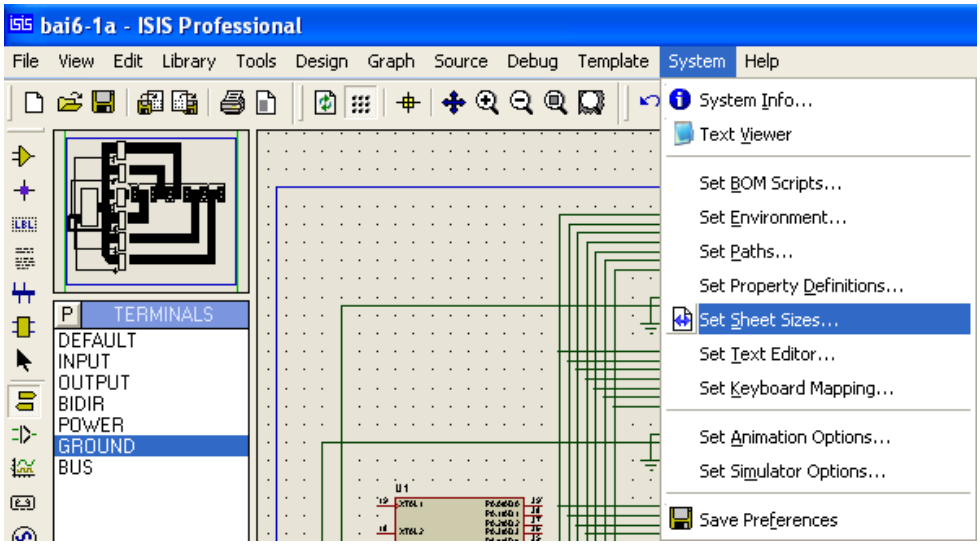
- Thực hiện chương trình dùng ngắt Timer 0 và Timer 1 đồng thời tạo xung vuông tần số f = 5 KHz tại P2.0 và f = 500Hz tại P2.1 như sau:

```
        ORG 0000h
        LJMP main
        ORG 000Bh
        LJMP Timer0_isr
        ORG 001Bh
        LJMP Timer1_isr
main:
        SETB EA
        SETB ET0
        SETB ET1
        MOV TMOD,#12h
        MOV TH0,#(-100)
        MOV TL0,#(-100)
        SETB TR0
        MOV TH1,#HIGH(-1000)
        MOV TL1,#LOW(-1000)
        SETB TR1
here:
        SJMP here
        ;-----------
Timer0_isr:
        CPL P2.0
        RETI
        ;-----------
Timer1_isr:
        MOV TH1,#HIGH(-1000)
        MOV TL1,#LOW(-1000)
        CPL P2.1
        RETI
        ;-----------
        END
```

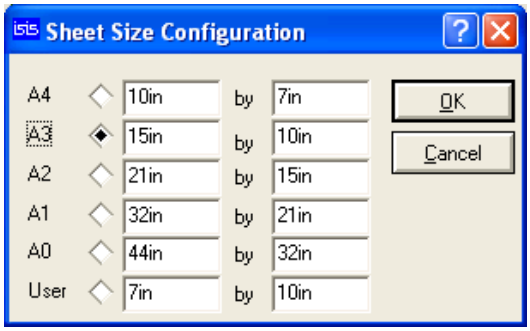
- Viết chương trình dùng ngắt Timer 0 và Timer 1 đồng thời tạo xung vuông tần số f = 20 KHz tại P2.3 và f = 100Hz tại P2.4.

Mạch đồng hồ

- Chọn menu **System > Set Sheet Sizes**

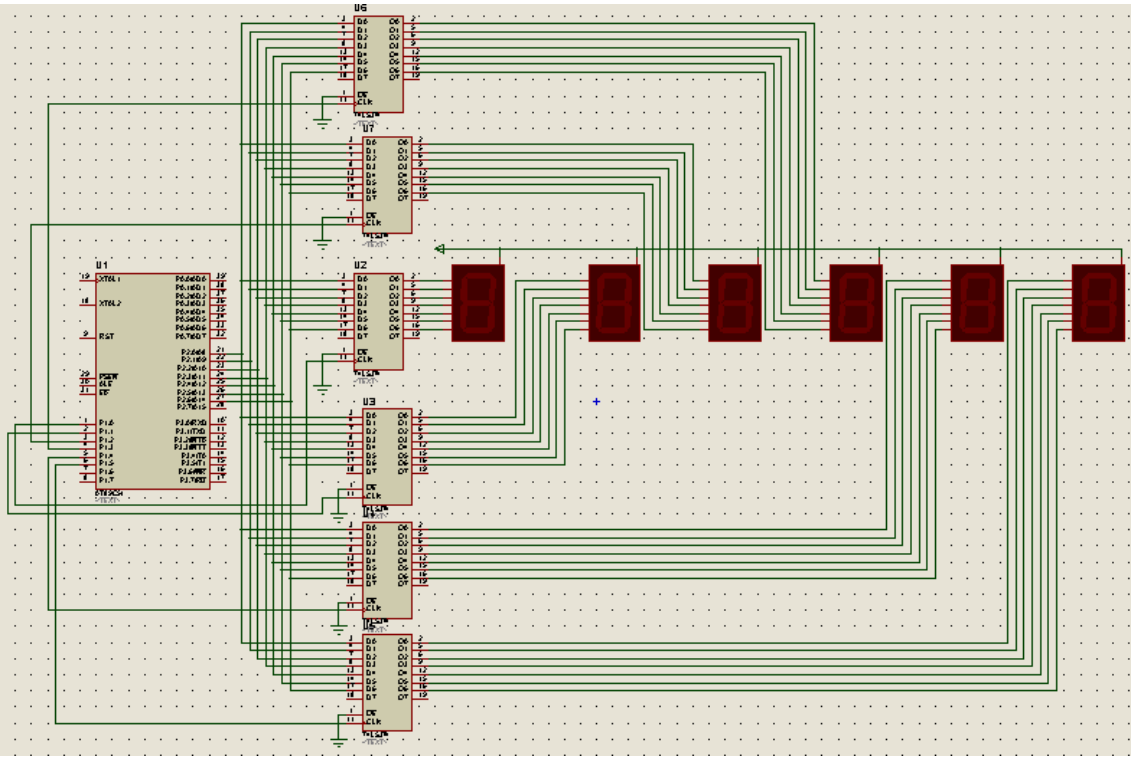


Chọn kích thước Sheet là A3:



Sử dụng mạch như hình vẽ (các IC chốt sử dụng là 74LS374). Các linh kiện cho như sau:

Keywords	Category	Sub-category	Results
7seg	All	All	7SEG-COM-ANODE
8951	All	All	AT89C51
374	74 TTL Series	All	74LS374
Switch	Switches & Relays	Switches	SW-SPDT



Chương trình đồng hồ:

```
        ORG 0
        LJMP main
        ORG 0Bh
        LJMP Timer0_ISR
main:
        SETB EA
        SETB ET0
        MOV TMOD,#01h
        MOV TH0,#HIGH(-50000)       ; Đếm 50000 chu kỳ = 50 ms
        MOV TL0,#LOW(-50000)
        MOV R7,#20                  ; 20 x 50ms = 1000ms = 1s
        SETB TR0
        MOV 30h,#0                  ; Giờ
        MOV 31h,#0                  ; Phút
        MOV 32h,#0                  ; Giây
        CALL display
here:
        SJMP here
```

```
;----------
Timer0_ISR:
        MOV TH0,#HIGH(-50000)
        MOV TL0,#LOW(-50000)
        DJNZ R7,exittimer0
        MOV R7,#20
        CALL IncTime            ; Cứ mỗi 1s thì tăng thời gian
        CALL display            ; và hiển thị ra Led
exittimer0:
        RETI
;---------
IncTime:
        INC 32h                 ; Tăng giây
        MOV A,32h
        CJNE A,#60,ExitIncTime  ; Nếu Giây = 60
        MOV 32h,#0              ; thì Giây = 0
        INC 31h                 ; và tăng phút
        MOV A,31h
        CJNE A,#60,ExitIncTime  ; Nếu Phút = 60
        MOV 31h,#0              ; thì Phút = 0
        INC 30h                 ; và tăng giờ
        MOV A,30h
        CJNE A,#24,ExitIncTime  ; Nếu giờ = 24
        MOV 30h,#0              ; thì Giờ = 0
ExitIncTime:
        RET
;---------
display:
        MOV A,30h               ; Hiển thị 2 Led chỉ giờ
        MOV B,#10
        DIV AB
        MOV DPTR,#MaLed7
        MOVC A,@A+DPTR
        MOV P2,A
        CLR P1.0
```

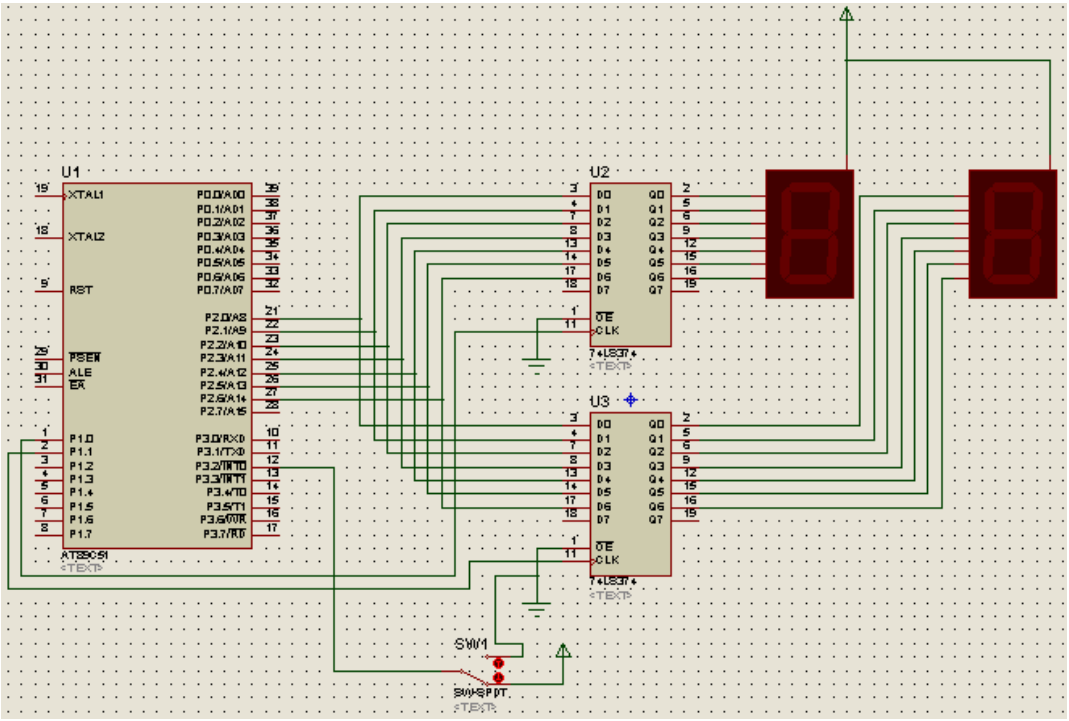
```
        SETB P1.0
        MOV A,B
        MOVC A,@A+DPTR
        MOV P2,A
        CLR P1.1
        SETB P1.1
        MOV A,31h               ; Hiển thị 2 Led chỉ phút
        MOV B,#10
        DIV AB
        MOV DPTR,#MaLed7
        MOVC A,@A+DPTR
        MOV P2,A
        CLR P1.2
        SETB P1.2
        MOV A,B
        MOVC A,@A+DPTR
        MOV P2,A
        CLR P1.3
        SETB P1.3
        MOV A,32h               ; Hiển thị 2 Led chỉ giây
        MOV B,#10
        DIV AB
        MOV DPTR,#MaLed7
        MOVC A,@A+DPTR
        MOV P2,A
        CLR P1.4
        SETB P1.4
        MOV A,B
        MOVC A,@A+DPTR
        MOV P2,A
        CLR P1.5
        SETB P1.5
        RET
;---------
MaLed7: DB 0C0h,0F9h,0A4h,0B0h,99h,92h,82h,0F8h,80h,90h
```

END

- Sửa chương trình trên để cho phép đếm giờ thể thao: 2 Led chỉ phút, 2 Led chỉ giây và 2 Led chỉ %giây.

Ngắt ngoài

Sử dụng mạch như hình vẽ:



Các linh kiện cho như sau:

Keywords	Category	Sub-category	Results
7seg	All	All	7SEG-COM-ANODE
8951	All	All	AT89C51
374	74 TTL Series	All	74LS374
Switch	Switches & Relays	Switches	SW-SPDT

- Viết chương trình đếm sử dụng ngắt ngoài 0 như sau (mỗi lần có ngắt xảy ra thì tăng nội dung ô nhớ 30h lên 1 và xuất ra Led 7 đoạn):

```
ORG 0
LJMP main
ORG 03h
LJMP Int0_ISR
main:
SETB EA
SETB EX0
MOV 30h,#0
CALL display
```

```
here:
SJMP here
;----------
Int0_ISR:
INC 30h
MOV A,30h
CJNE A,#100,next
MOV 30h,#0
next:
CALL display
RETI
;---------
display:
MOV A,30h
MOV B,#10
DIV AB
MOV DPTR,#MaLed7
MOVC A,@A+DPTR
MOV P2,A
CLR P1.0
SETB P1.0
MOV A,B
MOVC A,@A+DPTR
MOV P2,A
CLR P1.1
SETB P1.1
RET
;---------
MaLed7: DB 0C0h,0F9h,0A4h,0B0h,99h,92h,82h,0F8h,80h,90h

END
```

- Thêm vào lệnh cho phép ngắt bằng cạnh như sau (tại đoạn in đậm):

```
SETB EA
SETB EX0
SETB IT0
```

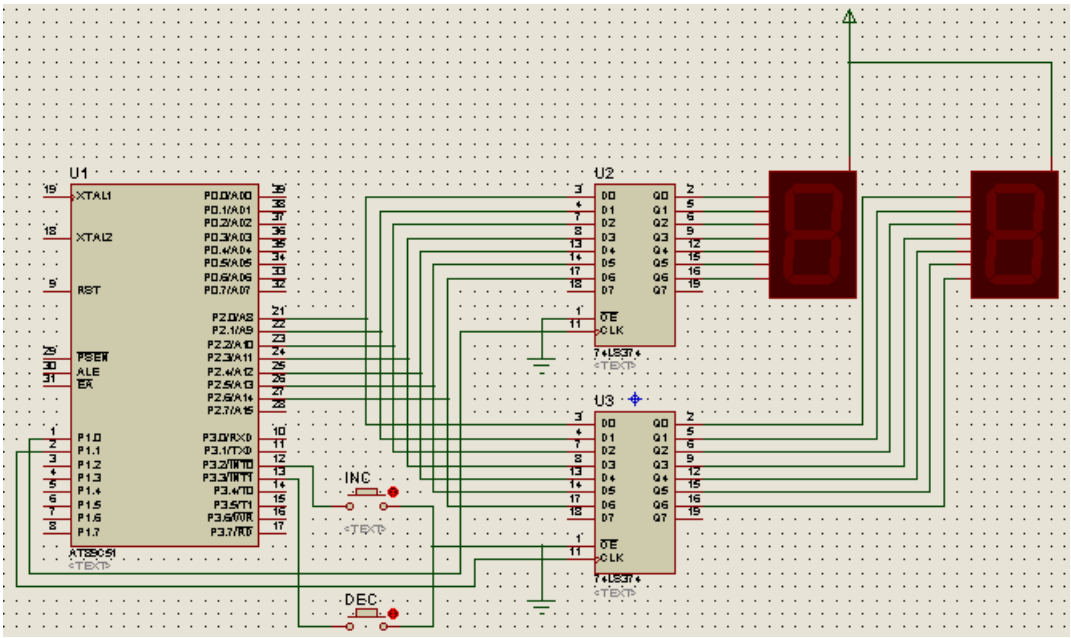
- So sánh trường hợp sử dụng ngắt bằng cạnh và bằng mức logic.
- Sửa chương trình trên để cho phép đếm từ 10 – 50.

Kết hợp ngắt ngoài và ngắt Timer.

Các linh kiện cho như sau:

Keywords	Category	Sub-category	Results	Componet Reference
7seg	All	All	7SEG-COM-ANODE	
8951	All	All	AT89C51	
374	74 TTL Series	All	74LS374	
Switch	Switches & Relays	Switches	BUTTON	INC
Switch	Switches & Relays	Switches	BUTTON	DEC

Sử dụng mạch như hình vẽ:



Chương trình đếm giây và điều chỉnh giá trị hiển thị bằng 2 công tắc (nhấn INC thì tăng giá trị hiển thị và nhấn DEC thì giảm giá trị hiển thị) như sau:

```
ORG 0
LJMP main
ORG 03h
LJMP Int0_ISR
ORG 0Bh
LJMP Timer0_ISR
ORG 13h
LJMP Int1_ISR
```

```
main:
        SETB EA
        SETB EX0
        SETB EX1
        SETB ET0
        SETB IT0
        SETB IT1
        MOV TMOD,#01h
        MOV TH0,#HIGH(-50000)
        MOV TL0,#LOW(-50000)
        MOV R7,#20
        SETB TR0
        MOV 30h,#0
        CALL display
        here:
        SJMP here
        ;----------
        Int0_ISR:
        INC 30h
        MOV A,30h
        CJNE A,#100,next
        MOV 30h,#0
next:
        CALL display
        RETI
;----------
Int1_ISR:
        DEC 30h
        MOV A,30h
        CJNE A,#255,next1
        MOV 30h,#99
next1:
        CALL display
        RETI
;---------
```

```
Timer0_ISR:
        MOV TH0,#HIGH(-50000)
        MOV TL0,#LOW(-50000)
        DJNZ R7,exittimer0
        MOV R7,#20
        INC 30h
        MOV A,30h
        CJNE A,#100,next2
        MOV 30h,#0
next2:
        CALL display
exittimer0:
        RETI
;---------
display:
        MOV A,30h
        MOV B,#10
        DIV AB
        MOV DPTR,#MaLed7
        MOVC A,@A+DPTR
        MOV P2,A
        CLR P1.0
        SETB P1.0
        MOV A,B
        MOVC A,@A+DPTR
        MOV P2,A
        CLR P1.1
        SETB P1.1
        RET
;---------
MaLed7: DB 0C0h,0F9h,0A4h,0B0h,99h,92h,82h,0F8h,80h,90h
        END
```

- Sửa chương trình để khi nhấn INC thì tăng 2 đơn vị và nhấn DEC thì giảm 10 đơn vị.

BÀI 7: GIAO TIẾP CÁC THIẾT BỊ CƠ BẢN

❖ **MỤC ĐÍCH**

Giúp sinh viên khảo sát các vấn đề sau:

- Sử dụng phần mềm Proteus để mô phỏng mạch điện.
- Khảo sát phương pháp hiển thị trên ma trận Led, điều khiển động cơ, đóng ngắt Relay.

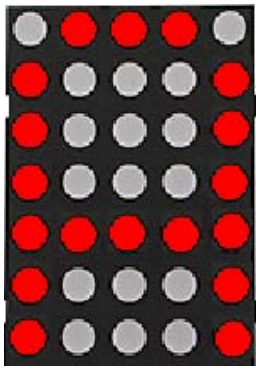
❖ **THIẾT BỊ SỬ DỤNG**

- Máy vi tính.
- Phần mềm Proteus

1. Cơ sở lý thuyết

Ma trận Led

Ma trận LED bao gồm nhiều LED cùng nằm trong một vỏ chia thành nhiều cột và hàng, mỗi giao điểm giữa hàng và cột có thể có 1 LED (ma trận LED một màu) hay nhiều LED (2 LED tại một vị trí tạo thành ma trận LED 3 màu). Để LED tại một vị trí nào đó sáng thì phải cấp hiệu điện thế dương giữa Anode và Cathode. Trên cơ sở cấu trúc như vậy, ta có thể mở rộng hàng và cột của ma trận LED để tạo thành các bảng quang báo.



Kết nối của ma trận Led có 2 cách: anode nối với hàng, cathode nối với cột hay ngược lại. Sơ đồ kết nối mô tả như hình vẽ trang bên.

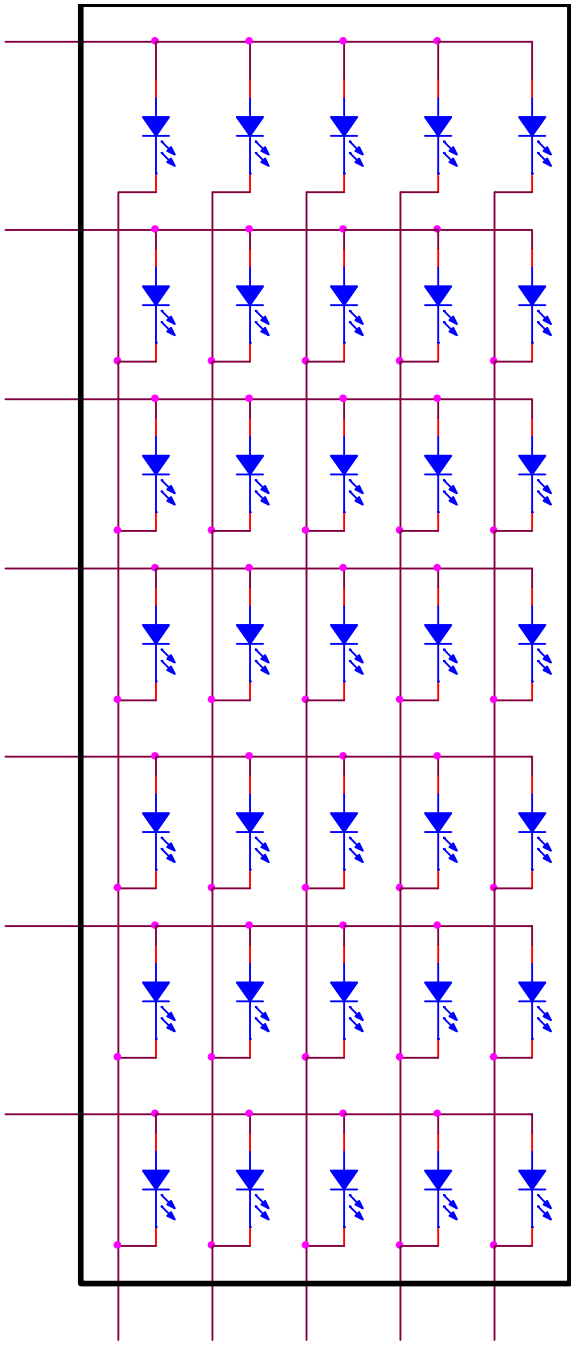
Theo cấu trúc kết nối như hình vẽ, 2 Led trên 2 cột không thể sáng đồng thời. Xét sơ đồ kết nối như mạch hình b, một Led sáng khi tương ứng hàng của Led = 0 và cột = 1.

Giả sử ta cần sáng Led đồng thời tại hàng 1, cột 1 và hàng 2, cột 2. Như vậy, ta phải có hàng 1 = 0, cột 1 = 1 (sáng Led tại hàng 1, cột 1) và hàng 2 = 0, cột 2 = 1 (sáng Led tại hàng 2, cột 2).

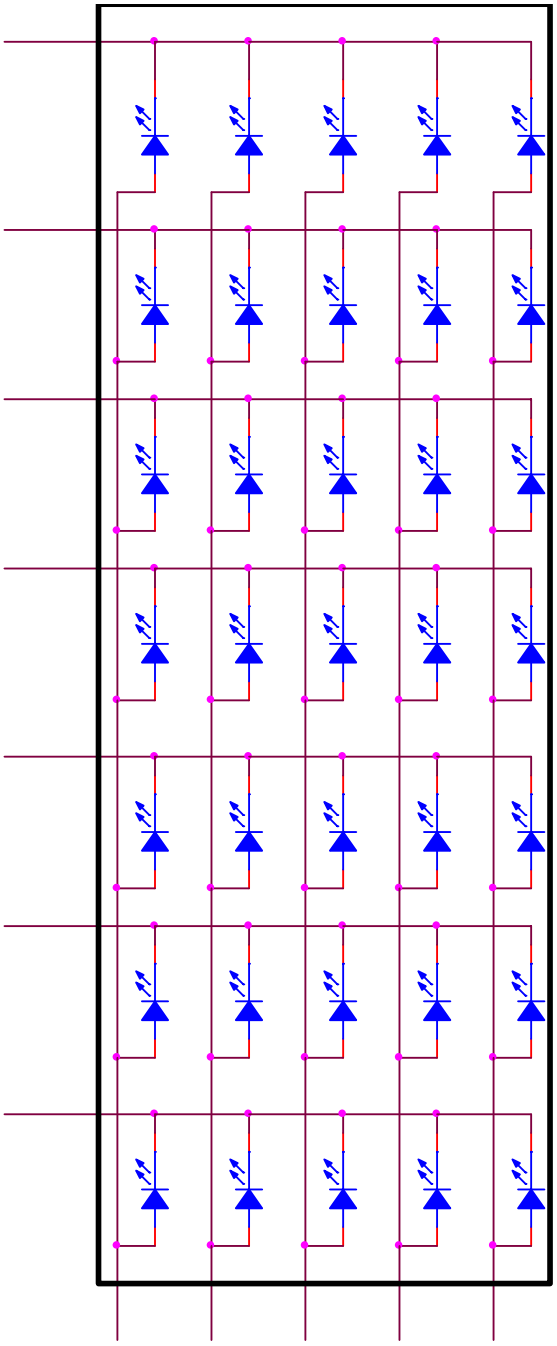
Từ đó, do hàng 1 = 0, cột 2 = 1 và hàng 2 = 0, cột 2 = 1 nên ta cũng có các Led tại hàng 1, cột 2 và hàng 2, cột 1 cũng sáng.

Nghĩa là, khi ta cho 2 Led tại hàng 1, cột 1 và hàng 2, cột 2 sáng đồng thời thì sẽ dẫn đến các Led tại hàng 1, cột 2 và hàng 2, cột 1 cũng sáng.

Do đó, để thực hiện sáng một ký tự trên ma trận Led, ta phải dùng cơ chế quét, tại mỗi thời điểm chỉ sáng 1 cột, các cột còn lại tắt đi nhưng nếu cho thời gian quét đủ nhanh thì ta vẫn thấy giống như các cột sáng đồng thời.



Hình a



Hình b

Dữ liệu cho số 0:

	X	X	X	
X				X
X				X
X				X
X				X
X				X
	X	X	X	

Để sáng số 0 trên ma trận Led, ta thực hiện quá trình quét như sau:

Lần 1: Hàng = 0100 0001b, cột = 0001 0000b

Lần 2: Hàng = 0011 1110b, cột = 0000 1000b

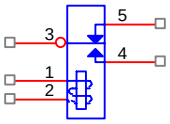
Lần 3: Hàng = 0011 1110b, cột = 0000 0100b

Lần 4: Hàng = 0011 1110b, cột = 0000 0010b

Lần 5: Hàng = 0100 0001b, cột = 0000 0001b

Điều khiển Relay

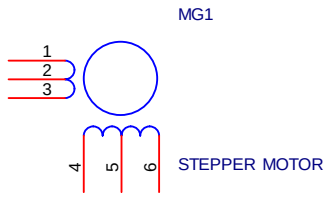
Relay hoạt động như một công tắc điện tử bao gồm một chân thường đóng (NC – Normally Closed: chân 5) và một chân thường mở (NO - Normal Open: chân 4). Khi có dòng điện đi qua cuộn dây (1-2) của Relay, vị trí của công tắc sẽ thay đổi từ NC sang NO. Như vậy, khi không có dòng điện qua cuộn dây thì chân 3 nối với chân 5 và khi có dòng điện thì chân 3 nối với chân 4. Dạng của Relay và sơ đồ mạch cho như hình vẽ.



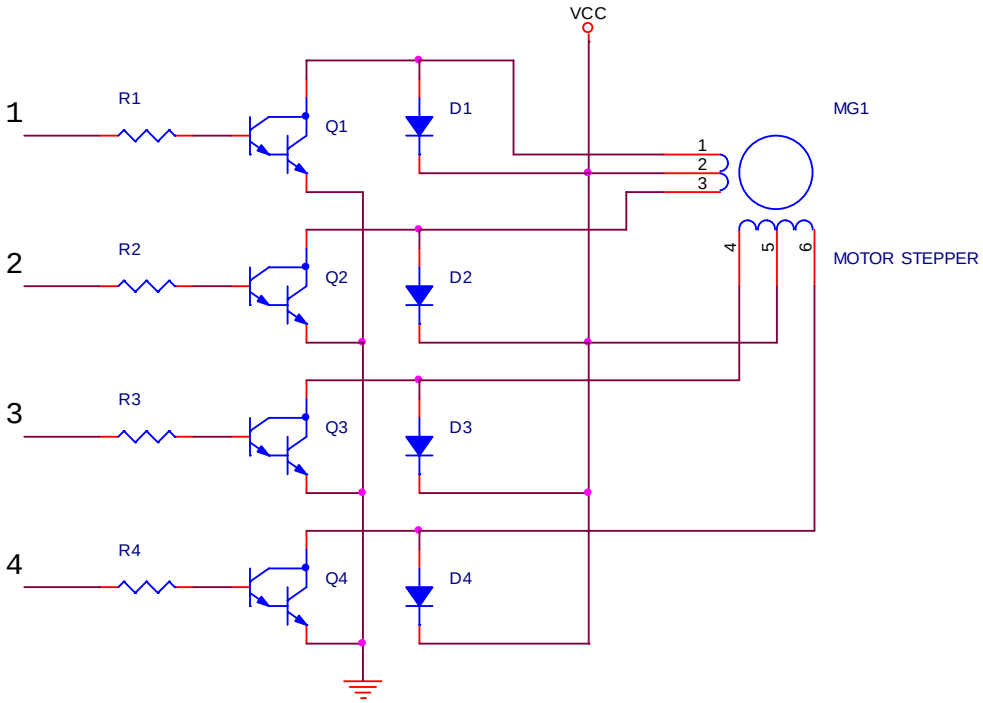
Điều khiển động cơ bước

Động cơ bước là động cơ cho phép dịch chuyển mỗi lần một bước hay nửa bước tuỳ theo xung điều khiển. Góc quay của mỗi bước tuỳ theo loại động cơ, thường là 1.8^0/bước.

Động cơ bước gồm 4 cuộn dây: 1-2, 2-3, 4-5 và 5-6 như sơ đồ sau:



Mạch điều khiển động cơ như sau:



Xung điều khiển động cơ như sau:

- Điều khiển một bước:

Ngược				Thuận			
1	**2**	**3**	**4**	**1**	**2**	**3**	**4**
1	0	0	0	1	0	0	0
0	1	0	0	0	0	0	1
0	0	1	0	0	0	1	0
0	0	0	1	0	1	0	0
1	0	0	0	1	0	0	0

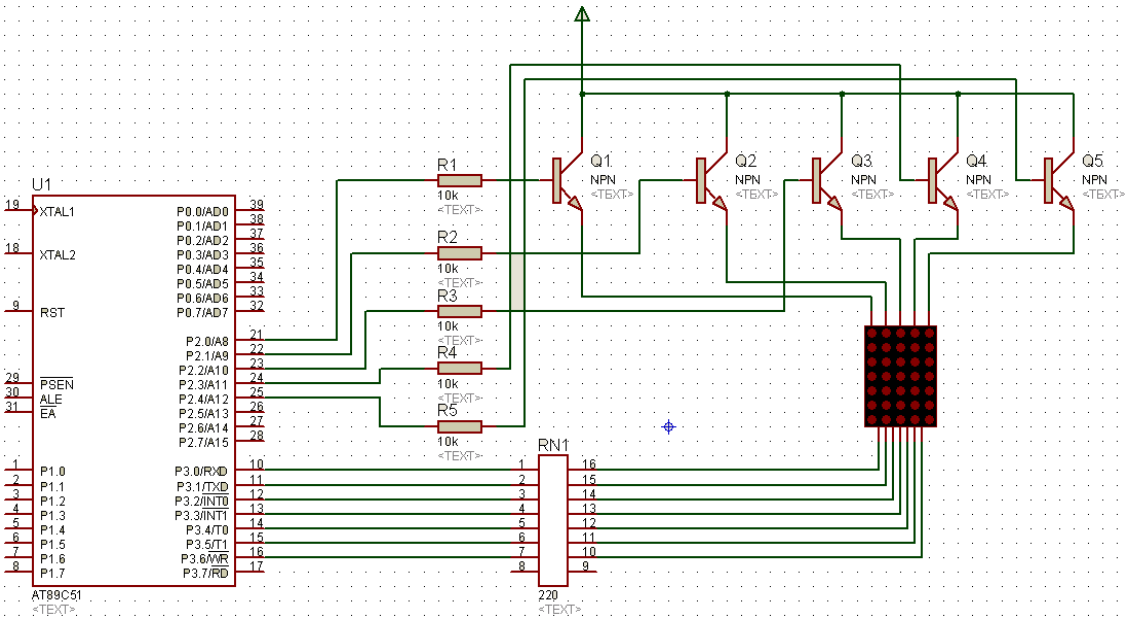
- Điều khiển nửa bước:

Ngược				Thuận			
1	**2**	**3**	**4**	**1**	**2**	**3**	**4**
1	0	0	1	1	0	0	1
1	0	0	0	0	0	0	1
1	1	0	0	0	0	1	1
0	1	0	0	0	0	1	0
0	1	1	0	0	1	1	0
0	0	1	0	0	1	0	0
0	0	1	1	1	1	0	0
0	0	0	1	1	0	0	0
1	0	0	1	1	0	0	1

2. Tiến trình thực hiện

Ma trận Led

Thực hiện mạch như hình vẽ sau:



Các linh kiện cho như sau:

Keywords	Category	Sub-category	Results	Value
Matrix	Optoelectronics	All	MATRIX-5X7-RED	
8951	All	All	AT89C51	
Npn	Transistors	Generic	Npn	
Resistor	Resistors	Generic	RES	
Resistor	Resistors	Resistor Packs	Rx8	220

Chương trình hiển thị số 0 trên ma trận Led:

```
main:
        MOV R0,#0
lap:
        MOV A,R0
        MOV DPTR,#cot
        MOVC A,@A+DPTR
        MOV P2,A
        MOV A,R0
        MOV DPTR,#hang
        MOVC A,@A+DPTR
        MOV P3,A
        CALL delay
        INC R0
        CJNE R0,#5,lap
        SJMP main
        ;-----------
delay:
        MOV TMOD,#01h
        MOV TL0,#LOW(-500)
        MOV TH0,#HIGH(-500)
        SETB TR0
        JNB TF0,$
        CLR TF0
        CLR TR0
        RET
cot: DB 01h,02h,04h,08h,10h
hang: DB 41h,3Eh,3Eh,3Eh,41h
END
```

- Viết chương trình hiển thị dấu ? trên ma trận Led.
- Viết chương trình hiển thị số 1 trên ma trận Led.

Chương trình chạy chuỗi ‘CDKTCN’ trên ma trận Led:

```
main2:
        MOV R2,#0
main1:
```

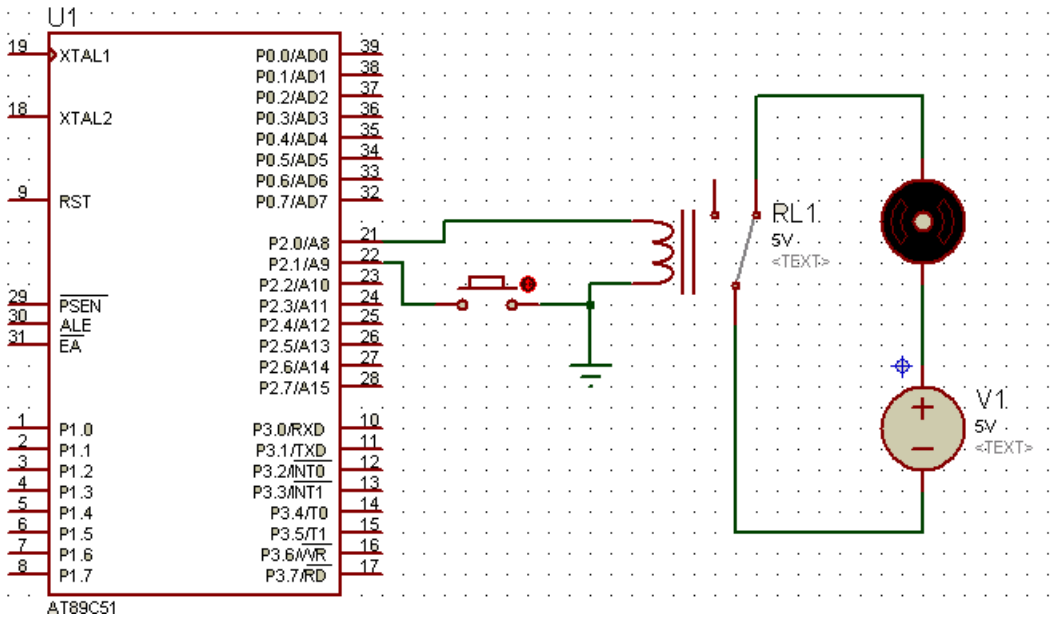
```
        MOV R1,#20
main:
        MOV R0,#0
lap:
        MOV A,R0
        MOV DPTR,#cot
        MOVC A,@A+DPTR
        MOV P2,A
        MOV A,R0
        ADD A,R2
        MOV DPTR,#hang
        MOVC A,@A+DPTR
        MOV P3,A
        CALL delay
        INC R0
        CJNE R0,#5,lap
        DJNZ R1,main
        INC R2
        CJNE R2,#37,main1
        SJMP main2
        ;-----------
delay:
        MOV TMOD,#01h
        MOV TL0,#LOW(-500)
        MOV TH0,#HIGH(-500)
        SETB TR0
        JNB TF0,$
        CLR TF0
        CLR TR0
        RET
cot: DB 01h,02h,04h,08h,10h
hang: DB 41h,3Eh,3Eh,3Eh,5Dh,7Fh ;C
        DB 00h,3Eh,3Eh,3Eh,41h,7Fh ;D
        DB 00h,77h,6Bh,5Dh,3Eh,7fh ;K
        DB 7Eh,7Eh,00h,7Eh,7Eh,7Fh ;T
```

```
        DB 41h,3Eh,3Eh,3Eh,5Dh,7Fh ;C
        DB 00h,7Dh,7Bh,77h,00h,7Fh ;N
        DB 7Fh,7Fh,7Fh,7Fh,7Fh
END
```

- Viết chương trình để chuỗi ‘TRUONG CAO DANG KINH TE CONG NGHE’ chạy trên ma trận Led.

Điều khiển Relay

Thực hiện mạch như hình vẽ sau:



Các linh kiện cho như sau:

Keywords	**Category**	**Sub-category**	**Results**	**Value**
Relay	Switches & Relays	Relays (Generic)	Relay	5V
8951	All	All	AT89C51	
Button	Switches & Relays	All	Button	
motor	Electromechanical	All	Motor	
Source	Simulator Primitives	All	Vsource	Voltage=5V

Chương trình điều khiển đóng / ngắt Relay mỗi lần nhấn công tắc như sau:

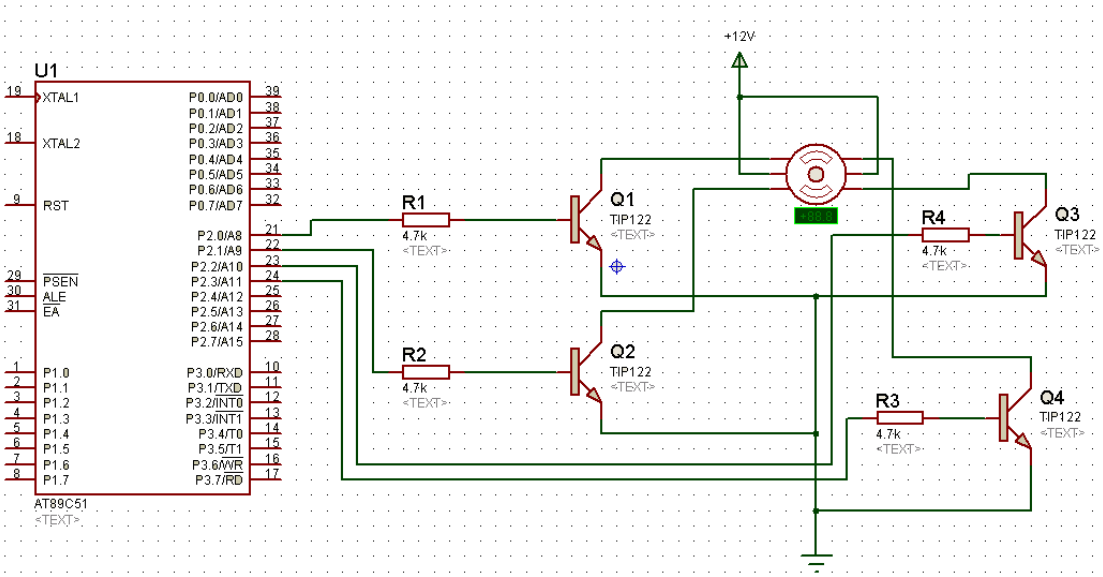
```
main:
        JB P2.1,$
        CALL DELAY
        CPL P2.0
        SJMP main
```

```
delay:
        MOV R7,#255
delay1:
        MOV R6,#255
        DJNZ R6,$
        DJNZ R7,delay1
        RET
END
```

- Thêm vào một nút nhấn tại P2.2, viết chương trình điều khiển để khi nhấn SW1 thì đóng Relay, nhấn SW2 thì ngắt Relay.

Điều khiển động cơ

Thực hiện mạch như hình vẽ sau:



Các linh kiện cho như sau:

Keywords	Category	Sub-category	Results	Value
Resistor	Resistors	Generic	RES	4.7k
8951	All	All	AT89C51	
Tip122	Transistors	All	Tip122	
Step	Electromechanical	All	Motor-stepper	

Chương trình điều khiển động cơ quay thuận liên tục, mỗi lần một bước như sau:

```
main:
        MOV  R2,#4
        MOV R0,#0
```

```
        MOV DPTR,#thuan1buoc
begin:
        MOV A,R0
        MOVC A,@A+DPTR
        MOV  P2,A
        CALL Delay
        INC R0
        DJNZ R2, begin
        SJMP  main
delay:
        MOV TMOD,#01h
        MOV TH0,#HIGH(-50000)
        MOV TL0,#LOW(-50000)
        SETB TR0
        JNB TF0,$
        CLR TF0
        CLR TR0
        RET

thuan1buoc: DB 08h,04h,02h,01h
END
```

- Viết chương trình điều khiển quay ngược động cơ liên tục, mỗi lần một bước.
- Viết chương trình điều khiển quay thuận động cơ liên tục, mỗi lần nửa bước.
- Viết chương trình điều khiển quay ngược động cơ liên tục, mỗi lần nửa bước.