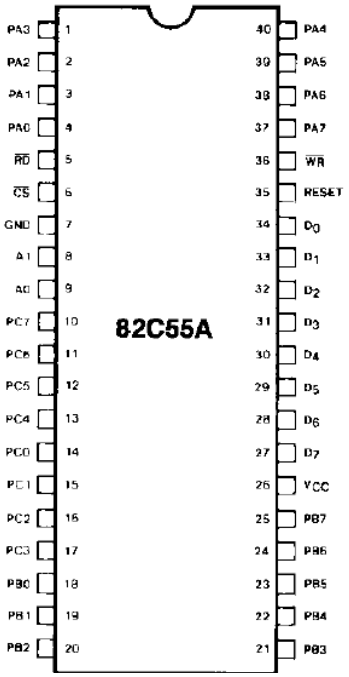
3 Thiết kế – ứng dụng

3.1 Giao tiếp với PPI8255



3.1.1 Giới thiệu PPI8255

- PPI: programmable peripheral interface – giao tiếp ngoại vi lập trình được.
- Dùng 8255 để mở rộng I/O. Từng port có thể được lập trình là input hay output một cách linh hoạt bằng phần mềm (so sánh với việc thiết kế I/O port dùng 74LS244 và 74LS373 ở chương 1 → input hay output được thiết kế "cứng", cố định).
- Các chân:

D0÷D7: bus dữ liệu 2 chiều.
PA0÷PA7: port A.
PB0÷PB7: port B.
PC0÷PC7: port C.
/RD: Read. (Nối với /RD (P3.7) của 8051.)
/WR: Write. (Nối với /WR của 8051.)
RESET: khởi động lại 8255. (Thường được nối với mạch reset của 8051 hoặc GND.
/CS: chọn chip.)
A0, A1: địa chỉ port. (Nối với bus địa chỉ.)

/CS	A1	A0	Mô tả
0	0	0	Port A
0	0	1	Port B
0	1	0	Port C
0	1	1	Từ điều khiển (control word)
1	×	×	8255 không được chọn

- Thanh ghi điều khiển:
    - Hoạt động I/O (D7 = 1)

1	D6	D5	D4	D3	D2	D1	D0
	Mode 00: mode 0 01: mode 1 1X: mode 2		PA 0: output 1: input	PC_H 0: output 1: input	Mode 0: mode 0 1: mode 1	PB 0: output 1: input	PC_L 0: output 1: input
	Nhóm A				Nhóm B		

Ghi chú: PC_H = PC7÷PC4, PC_L = PC3÷PC0.

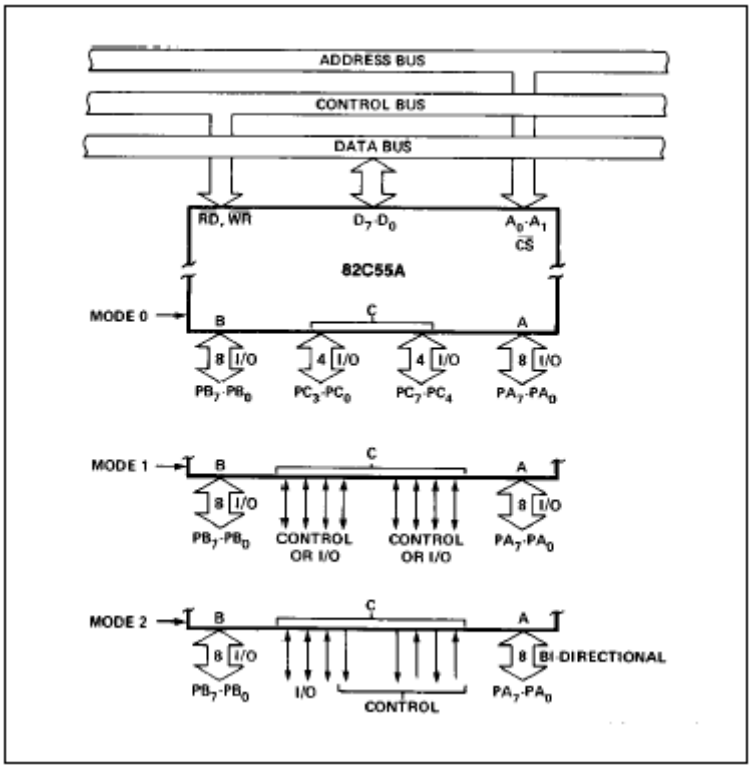
- Hoạt động BSR – Bit set/reset (D7 = 0):

0	X	X	X	D3	D2	D1	D0

Chọn bit ở port C

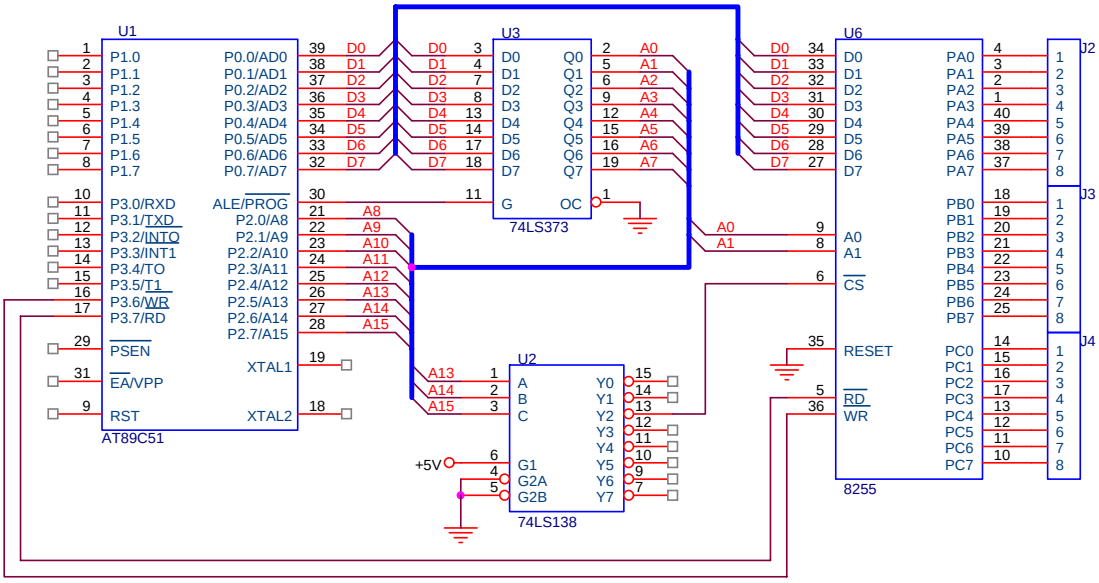
Bit set/reset
0: reset
1: set

- Hoạt động I/O cơ bản có 3 mode:
  - Mode 0: I/O đơn giản.
  - Mode 1: I/O có bắt tay.
  - Mode 2: bus 2 chiều.



3.1.2 Thiết kế - Giao tiếp

Thiết kế 1



Địa chỉ 8255 (base addr.): 4000h (16 bit)

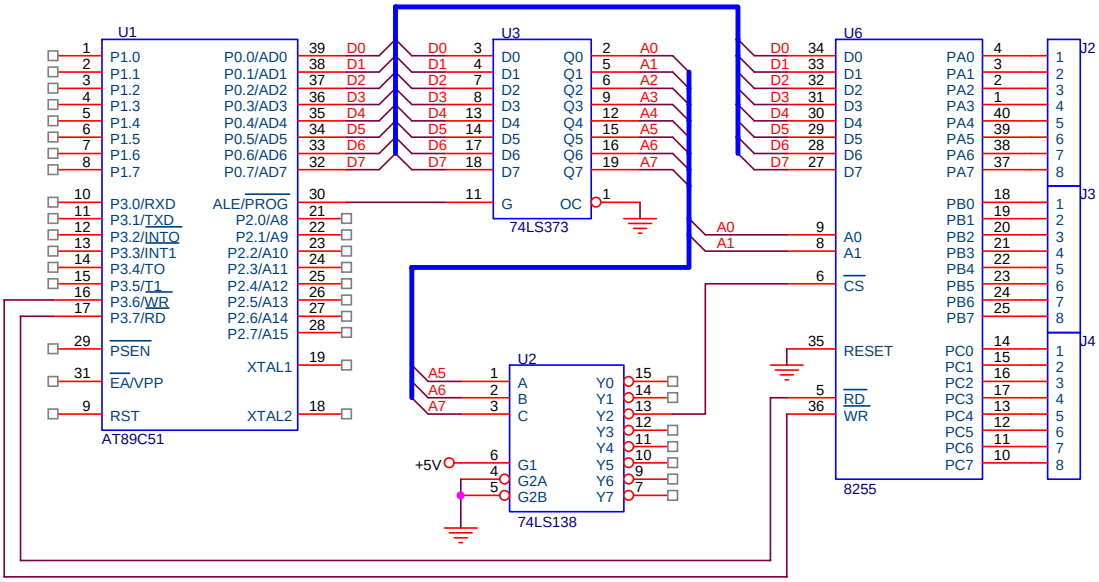
PA (base + 00h): 4000h

PB (base + 01h): 4001h

PC (base + 02h): 4002h

Control word (base + 03h): 4003h

Thiết kế 2



Địa chỉ 8255 (base addr.): 40h (8 bit)

PA (base + 00h): 40h

PB (base + 01h): 41h

PC (base + 02h): 42h

Control word (base + 03h): 43h

VD1: Khởi động 8255:
PA xuất, PB xuất, PC xuất → Từ điều khiển: 80h
PA xuất, PB nhập, PC xuất → Từ điều khiển: 82h
PA xuất, PB nhập, PC nhập → Từ điều khiển: 8Bh

VD2: Viết chương trình.
- Khởi động 8255: PA xuất, PB nhập, PC nhập.
- Liên tục đọc dữ liệu từ Port 1 của 8951, xuất dữ liệu đó ra 8255.

Chương trình cho sơ đồ 'Thiết kế 1':

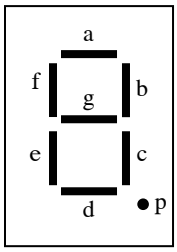
```
        ORG     0
        MOV     DPTR,#4003h     ; tu+` ddie^`u khie^?n
        MOV     A, #8Bh         ; PA: output, PB: input, PC: input
        MOVX    @DPTR, A
        MOV     P1, #0FFh       ; P1 (8951): input
AGAIN:  MOV     DPTR,#4000h     ; Port A
        MOV     A, P1
        MOVX    @DPTR,A
        SJMP    AGAIN
```

Chương trình cho sơ đồ 'Thiết kế 2':

```
        ORG     0
        MOV     R0,#43h         ; tu+` ddie^`u khie^?n
        MOV     A, #8Bh         ; PA: output, PB: input, PC: input
        MOVX    @R0, A
        MOV     P1, #0FFh       ; P1 (8951): input
AGAIN:  MOV     R0,#40h         ; Port A
        MOV     A, P1
        MOVX    @R0,A
        SJMP    AGAIN
```

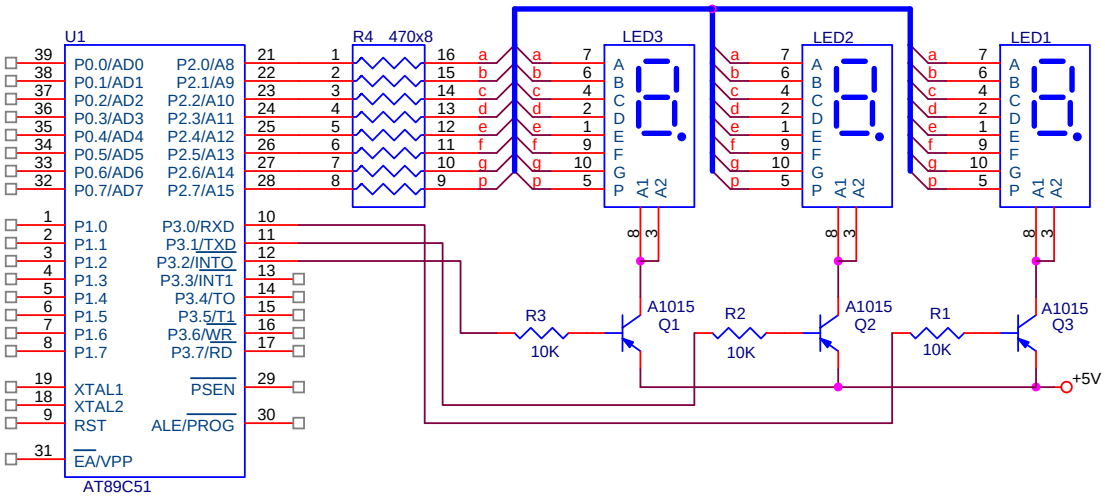
3.2 Giao tiếp với LED 7 đoạn

Đoạn:	p	g	f	e	d	c	b	a
	↑	↑	↑	↑	↑	↑	↑	↑
Bit:	D7	D6	D5	D4	D3	D2	D1	D0



Hiển thị	Anod chung	Cathode chung
0	C0h	3Fh
1	F9h	06h
2	A4h	5Bh
3	B0h	4Fh
4	99h	66h
5	92h	6Dh
6	82h	7Dh
7	F8h	07h
8	80h	7Fh
9	98h	67h
A	88h	77h
B	C6h	39h
C	86h	79h
D	8Eh	71h
E	82h	70h
F	89h	76h
.	7Fh	80h
[trắng]	FFh	00h

Quét LED



VD: Hiển thị '123' lên LED 7 đoạn.

```
; a,b,c,d,e,f,g -> Port 2
; P3.0 -> LED1
; P3.1 -> LED2
; P3.1 -> LED3
        ORG     0H
        MOV     P3,#0FFh        ; ta('t ta^'t ca? ca'c LED

BEGIN:  MOV     P2,#0B0h        ; xua^'t ra P2 ma~ cu?a '3'
        CLR     P3.0            ; ba^.t LED1
        ACALL   DELAY           ; delay
        SETB    P3.0            ; ta('t LED1

        MOV     P2,#0A4h        ; xua^'t ra P2 ma~ cu?a '2'
        CLR     P3.1            ; ba^.t LED2
        ACALL   DELAY           ; delay
        SETB    P3.1            ; ta('t LED2

        MOV     P2,#0F9h        ; xua^'t ra P2 ma~ cu?a '1'
        CLR     P3.2            ; ba^.t LED3
        ACALL   DELAY           ; delay
        SETB    P3.2            ; ta('t LED3
        SJMP    BEGIN

DELAY:  MOV     R1,#10
        MOV     R0,#0FFh
LOOP:   DJNZ    R0,LOOP
        DJNZ    R1,LOOP
        RET
        END
```

VD: Đếm xung ở ngõ vào T0 (P3.4) → hiển thị trị đếm lên LED 7 đoạn

```
; Que't LED
; a,b,c,d,e,f,g -> Port 2
; P3.0 -> LED1
; P3.1 -> LED2
; P3.2 -> LED3
; P3.4(T0) -> Button
; 40h: ha`ng do+n vi.
; 41h: ha`ng chu.c
; 42h: ha`ng tra(m
        ORG     0H
        MOV     DPTR,#LED7SEG   ; DPTR tro? dde^'n ba?ng ma~ LED
```

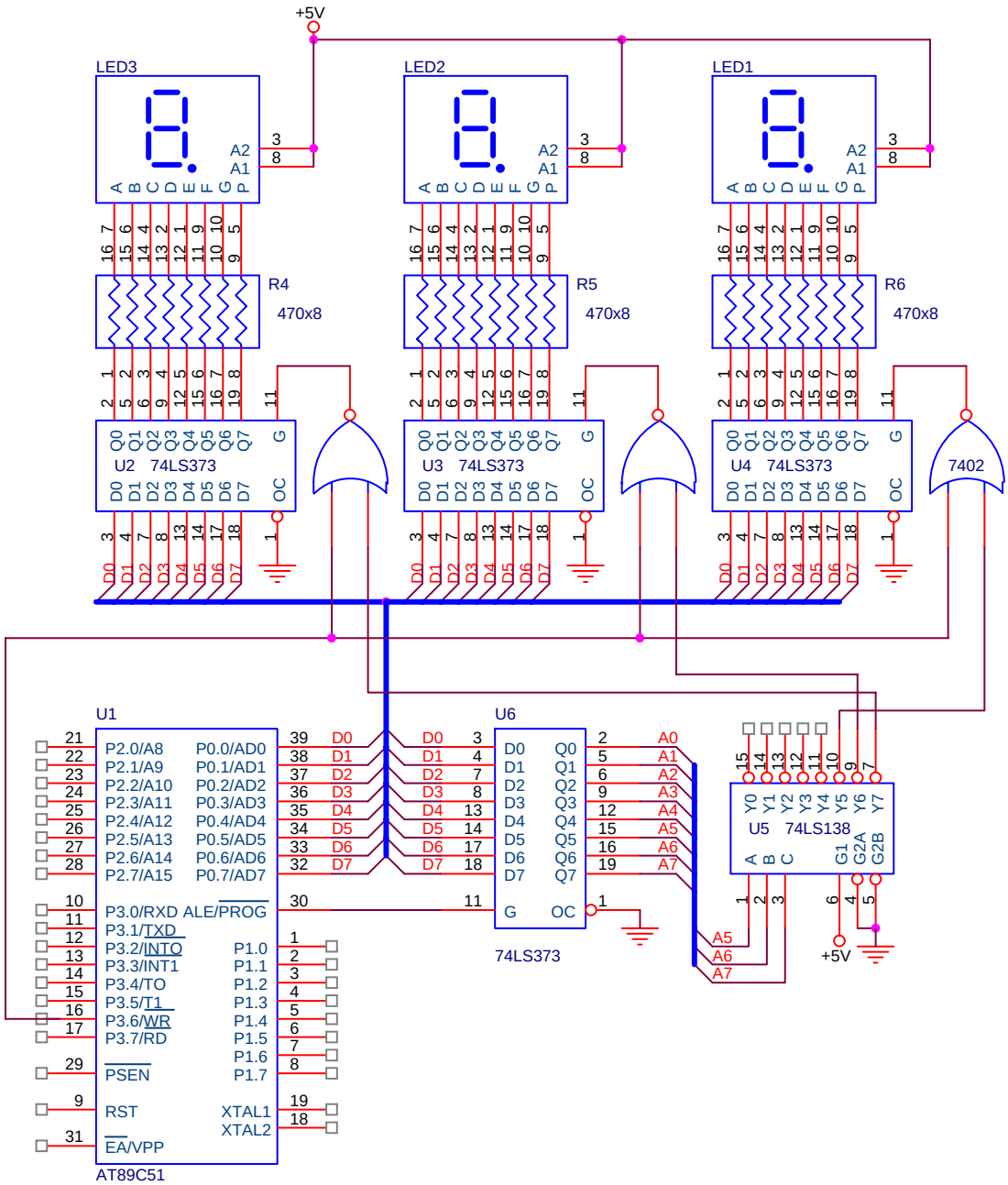
```
        MOV     TMOD,#06h       ; counter 0, mode 2
        MOV     TH0,#0
        SETB    P3.0            ; ta('t ta^'t ca? ca'c LED
        SETB    P3.1
        SETB    P3.2
        SETB    P3.4            ; P3.4: input
        SETB    TR0             ; cho phe'p counter 0 cha.y
BEGIN:  MOV     A,TL0
        LCALL   BIN2BCD
        ; tra ba?ng, ddo^?i BCD -> LED 7 ddoa.n
        MOV     A,40h
        MOVC    A,@A+DPTR
        MOV     40h,A
        MOV     A,41h
        MOVC    A,@A+DPTR
        MOV     41h,A
        MOV     A,42h
        MOVC    A,@A+DPTR
        MOV     42h,A
        LCALL   DISPLAY
        SJMP    BEGIN
DISPLAY:
        MOV     P2,40H          ; LED1
        CLR     P3.0            ; ba^.t LED1 sa'ng
        ACALL   DELAY           ; delay
        SETB    P3.0            ; ta('t LED1

        MOV     P2,41H          ; LED2
        CLR     P3.1            ; ba^.t LED2 sa'ng
        ACALL   DELAY           ; delay
        SETB    P3.1            ; ta('t LED2

        MOV     P2,42H          ; LED 3
        CLR     P3.2            ; ba^.t LED3 sa'ng
        ACALL   DELAY           ; delay
         SETB   P3.2            ; ta('t LED3
        RET

BIN2BCD:
        MOV     B,#10           ; B=10
        DIV     AB              ; chia cho 10
        MOV     40h,B           ; lu+u digit tha^'p
        MOV     B,#10           ;
        DIV     AB              ; chia cho 10
        MOV     41h,B           ; lu+u digit tie^'p theo va`o 41h
        MOV     42h,A           ; lu+u digit cuo^'i va`o 42h
        RET
; su+?a cho SV
DELAY:  PUSH    7
        PUSH    6
        MOV     R7,#10
LP2:    MOV     R6,#0FFh
LP1:    DJNZ    R6,LP1
        DJNZ    R7,LP2
        POP     6
        POP     7
        RET
LED7SEG:
        DB      0C0H,0F9H,0A4H,0B0H,99H,92H,82H,0F8H,80H,98H
        DB      88H,0C6H,86H,8EH,82H,89H
        END
```

Thiết kế 3 port xuất ra LED

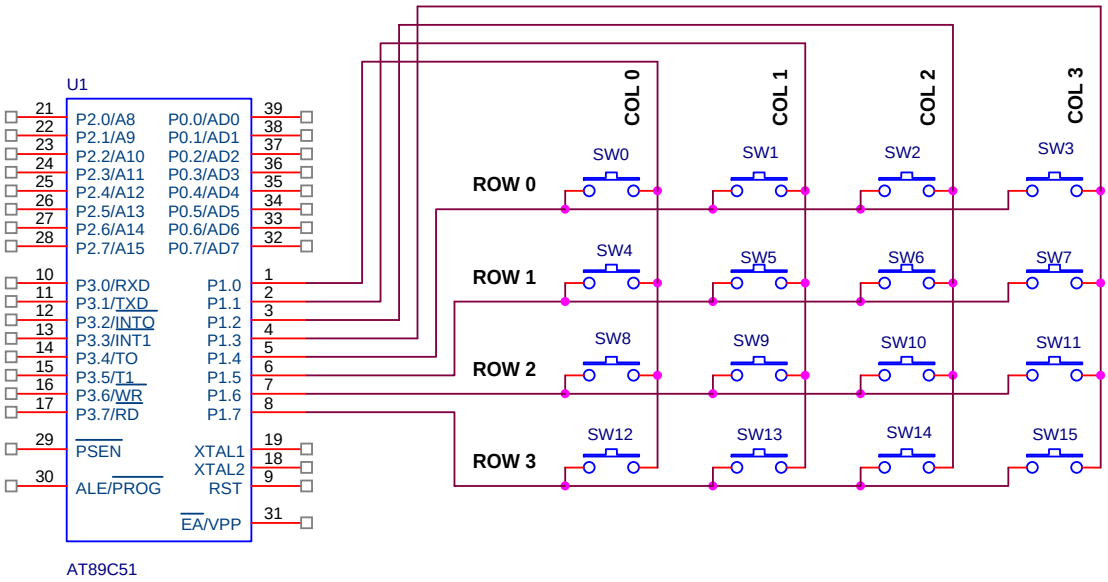
Địa chỉ (8-bit): LED1: A0h
LED2: C0h
LED3: E0h

VD: Xuất ra LED 7 đoạn '123'

```
ORG     0
MOV     R0,#0A0h        ; LED1
MOVX    @R0,#0B0h
MOV     R0,#0C0h        ; LED2
MOVX    @R0,#0A4h
MOV     R0,#0E0h        ; LED3
MOVX    @R0,#0F9h
SJMP    $
```



3.3 Giao tiếp với bàn phím hex

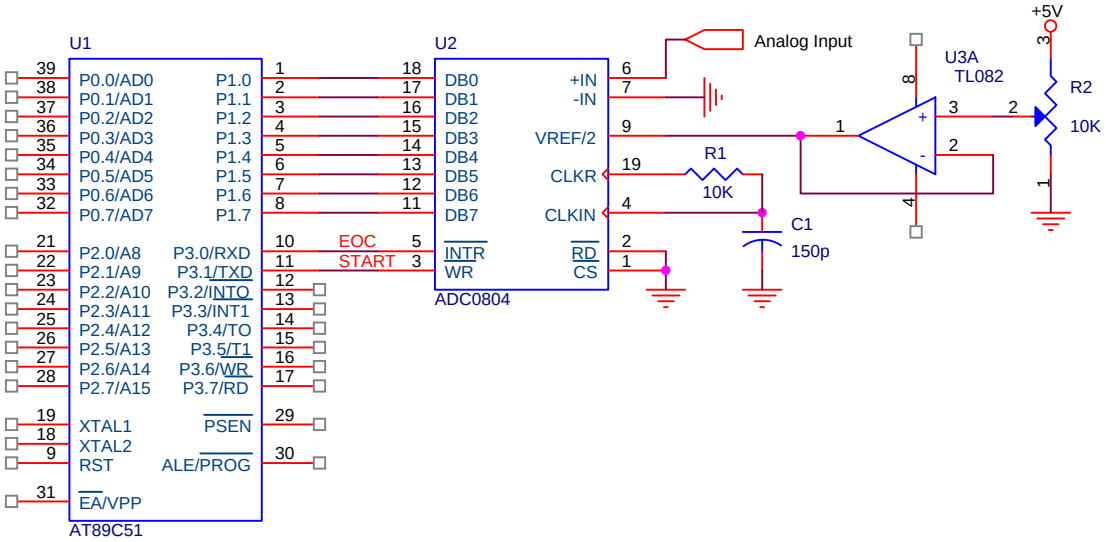


0	1	2	3
4	5	6	7
8	9	A	B
C	D	E	F

```
; Ba`n phi'm hex no^'i va`o P1
; Chuo+ng tri`nh hie^?n thi. phi'm nha^'n ra LED 7 ddoa.n
; P1.0-P1.3: columns
; P1.4-P1.7: rows
; DDi.a chi? LED: A000h

LOOP:    LCALL    READKB          ; tri. tra? ve^`: A = 0-15
         MOV      DPTR,#T7SEG
         MOVC     A,@A+DPTR
         MOV      DPTR,#0A000H    ; A000h: ddi.a chi? LED 1
         MOVX     @DPTR,A
         SJMP     LOOP

READKB:  PUSH     7
SCAN:    MOV      A,#11111110B    ; col_0 -> GND
         MOV      R7,#0           ; R7 = i
CONT:    MOV      P1,A            ; no^'i col i -> GND
         MOV      A,P1            ; ddo.c row
         JNB      ACC.4,ROW_0     ; xe't xem row na`o?
         JNB      ACC.5,ROW_1
         JNB      ACC.6,ROW_2
         JNB      ACC.7,ROW_3
         RL       A               ; chua^?n bi. no^'i GND
         INC      R7              ; co^.t tie^'p theo
         CJNE     R7,#4,CONT      ; la^`n luo+.t no^'i GND 4 co^.t
         SJMP     SCAN            ; quay la.i que't tu+` co^.t 0
ROW_0:   MOV      A,R7            ; Row=0, Col=R7
```

```
        ADD     A,#0            ; A = 0 + R7
        SJMP    EXIT
ROW_1:  MOV     A,R7            ; Row=1, Col=R7
        ADD     A,#4            ; A = 4 + R7
        SJMP    EXIT
ROW_2:  MOV     A,R7            ; Row=2, Col=R7
        ADD     A,#8            ; A = 8 + R7
        SJMP    EXIT
ROW_3:  MOV     A,R7            ; Row=3, Col=R7
        ADD     A,#12           ; A = 12 + R7
EXIT:   POP     7
        RET

T7SEG:  DB      40H,79H,24H,30H,19H,12H,02H,78H,00H,10H,
        DB      08H,03H,46H,21H,04H,0EH
        END
```

3.4 Giao tiếp với ADC0804



ADC0804 là bộ chuyển đổi tương tự sang số 8 bit.

Xét sơ đồ như hình:

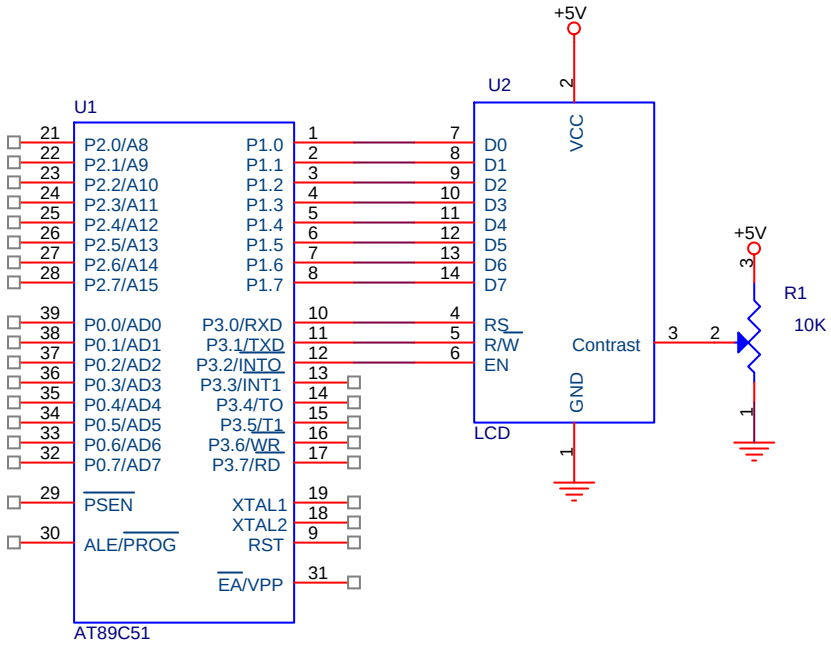
- Điện trở 10K và tụ 150pF nối với đầu vào CLKR và CLKIN như hình → bộ phát xung nhịp bên trong tạo tần số hoạt động là 640KHz.
- Một lần biến đổi được bắt đầu bằng một xung START (tích cực mức thấp) ngắn hạn ở ngõ vào /WR. Sau thời gian biến đổi khoảng 100µs, ngõ ra /INTR chuyển sang LOW báo hiệu là kết thúc quá trình biến đổi (EOC – End of Conversion)

VD: Đọc AD từ port 1, lưu vào ô nhớ 40h và xuất ra Port 2

```
;P1 <- D0-D7
;P3.0 <- /INTR
;P3.1 -> /WR
;P1 <- D0-D7
;P3.0 <- /INTR
;P3.1 -> /WR
        ORG     0
        MOV     P1,#0FFH        ;P1: input
        SETB    P3.0            ;P3.0: input
LOOP:   CLR     P3.1            ;pha't xung START
        SETB    P3.1
```

```
JB      P3.0,$          ;cho+` bie^'n ddo^?i xong
MOV     A,P1            ;ddo.c data va`o A
MOV     40h,A           ;lu+u va`o o^ nho+' 40h
MOV     P2,A            ;xua^'t ra P2
SJMP    LOOP
```

3.5 Giao tiếp với màn hình LCD



RS	R/W	D7	D6	D5	D4	D3	D2	D1	D0	Lệnh	Diễn giải
0	0	0	0	0	0	0	0	0	1	Clear display	
0	0	0	0	0	0	0	0	1	-	Return Cursor and LCD to Home Position	
0	0	0	0	0	0	0	1	ID	S	Set Cursor Move Direction	ID: increment the cursor after each byte written to display is set. S: shift the display when each byte is written to display
0	0	0	0	0	0	1	D	C	B	Enable Display/Cursor	D: display on(1)/ off(0) C: cursor on(1) / off(0) B: cursor blink on(1)/ off(0)
0	0	0	0	0	1	SC	RL	-	-	Move Cursor / Shift Display	SC: display shift on(1)/ off(0) RL: direction shift righ(1)/ left(0)
0	0	0	0	1	DL	N	F	-	-	Set Interface Length	DL: set data length 8(1)/ 4(0) N: number of line 1(0)/ 2(1) F: character font 5x10(1)/ 5x7(0)
0	0	0	1	A	A	A	A	A	A	Move Cursor into CG RAM	A: address
0	0	1	A	A	A	A	A	A	A	Move Cursor to Display	A: address
0	1	BF	-	-	-	-	-	-	-	Poll Busy Flag	BF: this bit is set while the LCD is processing.
1	0	D	D	D	D	D	D	D	D	Write a Character on the Display at the Current Cursor Position	D: data
1	1	D	D	D	D	D	D	D	D	Read the Character on the Display at the Current Cursor Position	D: data

VD: Xuất ra LCD chuỗi “Hello”

```
;P1=data pin
;P3.0 -> RS pin
;P3.1 -> R/W pin
;P3.2 -> E pin

RS      EQU     P3.0
RW      EQU     P3.1
E       EQU     P3.2
        ORG     0
        MOV     A,#38H          ;init. LCD 2 do`ng, ma tra^.n 5x7
        ACALL   CSTROBE
        MOV     A,#0CH          ;LCD on, cursor on
        ACALL   CSTROBE
        MOV     A,#01H          ;clear LCD
        ACALL   CSTROBE
        MOV     A,#06H          ;cursor di.ch pha?i
        ACALL   CSTROBE

        MOV     A,#86H          ;chuye^?n cursor dde^'n line 1, pos. 6
        ACALL   CSTROBE
        MOV     A,#'H'
        ACALL   DSTROBE
        MOV     A,#'e'
        ACALL   DSTROBE
        MOV     A,#'l'
        ACALL   DSTROBE
        MOV     A,#'l'
        ACALL   DSTROBE
        MOV     A,#'o'
        ACALL   DSTROBE
HERE:   SJMP    HERE

CSTROBE:                        ;command strobe
        ACALL   READY           ;is LCD ready?
        MOV     P1,A            ;xua^'t ma~ le^.nh
        CLR     RS              ;RS=0: le^.nh
        CLR     RW              ;R/W=0 -> ghi ra LCD
        SETB    E               ;E=1 -> ta.o ca.nh xuo^'ng
        CLR     E               ;E=0 ,cho^'t
        RET

DSTROBE:                        ;data strobe
        ACALL   READY           ;is LCD ready?
        MOV     P1,A            ;xua^'t du+~ lie^.u
        SETB    RS              ;RS=1 for data
        CLR     RW              ;R/W=0 to write to LCD
        SETB    E               ;E=1 -> ta.o ca.nh xuo^'ng
        CLR     E               ;E=0, cho^'t
        RET

; kie^?m tra co+` BF
READY:  SETB    P1.7            ;P1.7: input
        CLR     RS              ;RS=0: thanh ghi le^.nh
        SETB    RW              ;R/W=1: ddo.c
BACK:   CLR     E               ;E=0 -> ta.o ca.nh le^n
        SETB    E               ;E=1
        JB      P1.7,BACK       ;cho+` busy flag=0
        RET
        END
```

VD2: Đọc bàn phím Hex → xuất ra LCD

```
;P1 = data/command pin
;P3.0 -> RS pin
;P3.1 -> R/W pin
;P3.2 -> E pin
;P2 -> Keypad
        ORG     0
RS      EQU     P3.0
RW      EQU     P3.1
EN      EQU     P3.2

        MOV     A,#38H          ;init. LCD 2 lines,5x7 matrix
        ACALL   CSTROBE
        MOV     A,#0EH          ;LCD on, cursor on
        ACALL   CSTROBE
        MOV     A,#01H          ;clear LCD
        ACALL   CSTROBE
        MOV     A,#06H          ;cursor di.ch pha?i
        ACALL   CSTROBE
        MOV     A,#80H          ;cursor: line 1, pos. 0
        ACALL   CSTROBE

AGAIN:  LCALL   READKP
        ORL     A,#30h
        ACALL   DELAY
        ACALL   DSTROBE
        SJMP    AGAIN

;command strobe
CSTROBE:
        ACALL   READY           ;is LCD ready?
        MOV     P1,A            ;xua^'t ma~ le^.nh
        CLR     RS              ;RS=0: le^.nh
        CLR     RW              ;R/W=0: ghi ra LCD
        SETB    EN              ;EN=1 -> ta.o ca.nh xuo^'ng
        CLR     EN              ;EN=0 ,cho^'t
        RET

;data strobe
DSTROBE:
        ACALL   READY           ;is LCD ready?
        MOV     P1,A            ;xua^'t du+~ lie^.u ra P1
        SETB    RS              ;RS=1: du+~ lie^.u
        CLR     RW              ;R/W=0 ghi ra LCD
        SETB    EN              ;EN=1 -> ta.o ca.nh xuo^'ng
        CLR     EN              ;EN=0, cho^'t
        RET

READY:  SETB    P1.7            ;P1.7: input
        CLR     RS              ;RS=0: le^.nh
        SETB    RW              ;R/W=1: ddo.c
BACK:   CLR     EN              ;EN=0 -> ta.o ca.nh le^n
        SETB    EN              ;EN=1
        JB      P1.7,BACK       ;cho+` busy flag=0
        RET

; DDo.c ba`n phi'm
READKP: PUSH    7
SCAN:   MOV     A,#11111110B    ; col_0 -> GND
        MOV     R7,#0           ; R7 = i
```

```
CONT:   MOV     P2,A            ; no^'i col i -> GND
        MOV     A,P2            ; ddo.c row
        JNB     ACC.4,ROW_0     ; xe't xem row na`o?
        JNB     ACC.5,ROW_1
        JNB     ACC.6,ROW_2
        JNB     ACC.7,ROW_3
        RL      A               ; chua^?n bi. no^'i GND
        INC     R7              ; co^.t tie^'p theo
        CJNE    R7,#4,CONT      ; la^`n luo+.t no^'i GND 4 co^.t
        SJMP    SCAN            ; quay la.i que't tu+` co^.t 0
ROW_0:  MOV     A,R7            ; Row=0, Col=R7
        ADD     A,#0            ; A = 0 + R7
        SJMP    EXIT
ROW_1:  MOV     A,R7            ; Row=1, Col=R7
        ADD     A,#4            ; A = 4 + R7
        SJMP    EXIT
ROW_2:  MOV     A,R7            ; Row=2, Col=R7
        ADD     A,#8            ; A = 8 + R7
        SJMP    EXIT
ROW_3:  MOV     A,R7            ; Row=3, Col=R7
        ADD     A,#12           ; A = 12 + R7
EXIT:   POP     7
        RET

DELAY:  PUSH    6
        PUSH    7
        MOV     R7,#0FFh
LP1:    MOV     R6,#0FFh
LP0:    DJNZ    R6,LP0
        DJNZ    R7,LP1
        POP     7
        POP     6
        RET
        END
```

4 Lập trình hợp ngữ

4.1 Một số cấu trúc lập trình

Nhảy có điều kiện:

<condition>	Jump_if_not <conditon>	Jump_if_<conditon>
C = 1 bit = 1	**JNC** rel **JNB** bit, rel	**JC** rel **JB** bit, rel / **JBC** bit, rel
A = 0 Rn = 0 direct = 0	**JNZ** rel **DJNZ** Rn, rel **DJNZ** direct, rel	**JZ** rel
A ≠ direct A ≠ #data Rn ≠ #data @Ri ≠ #data	**CJNE** A, direct, rel **CJNE** A, #data, rel **CJNE** Rn, #data, rel **CJNE** @Ri, #data, rel	

Nhảy không điều kiện: **AJMP**, **LJMP**, **SJMP**.

Cấu trúc "repeat... until"

```
repeat <action> until <condition>

REPEAT:  <action>
         JUMP_if_not_<condition>,REPEAT
```

Cấu trúc "while... do"

```
while <condition> do <action>

START:    JUMP_if_not_<condition>,STOP
          <action>
          SJMP    START
STOP:     ...
```

Cấu trúc "if... then... else"

```
if <condition> then <action 1> else <action 2>

     JUMP_if_not_<condition>,ELSE
     <action 1>
     SJMP DONE
ELSE:
     <action 2>
DONE:
```

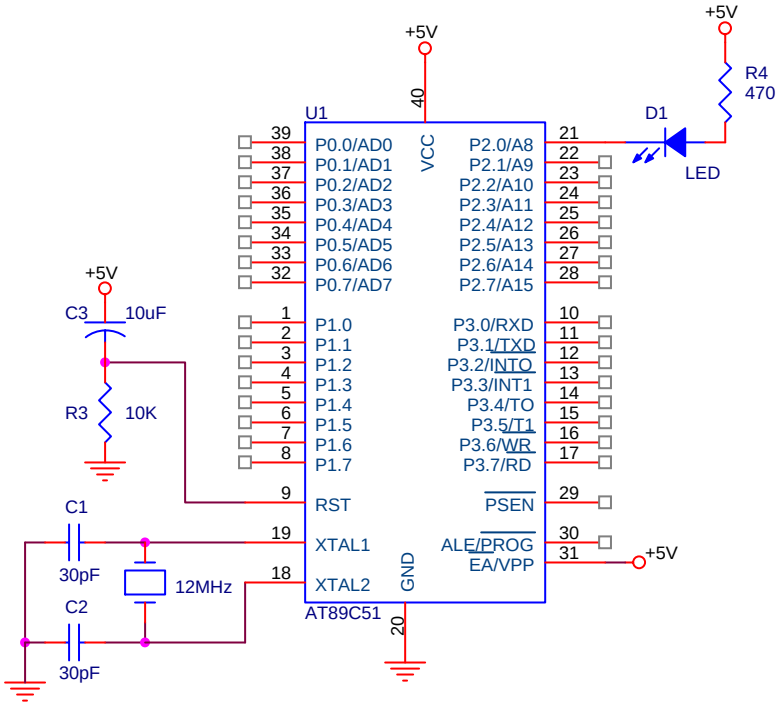
Cấu trúc "case... of..."

```
case <var> of
        val1: <action 1>
        val2: <action 2>
        val3: <action 3>
        else: <action else>
end
```

```
        CJNE   <var>,<val1>,SKIP1
        <action 1>
        SJMP   EXIT
SKIP1:  CJNE   <var>,<val2>,SKIP2
        <action 2>
        SJMP   EXIT
SKIP2:  CJNE   <var>,<val2>,SKIP3
        <action 3>
        SJMP   EXIT
SKIP3:  CJNE   <var>,<val2>,EXIT
        <action else>
EXIT:
```

4.2 Một số ví dụ

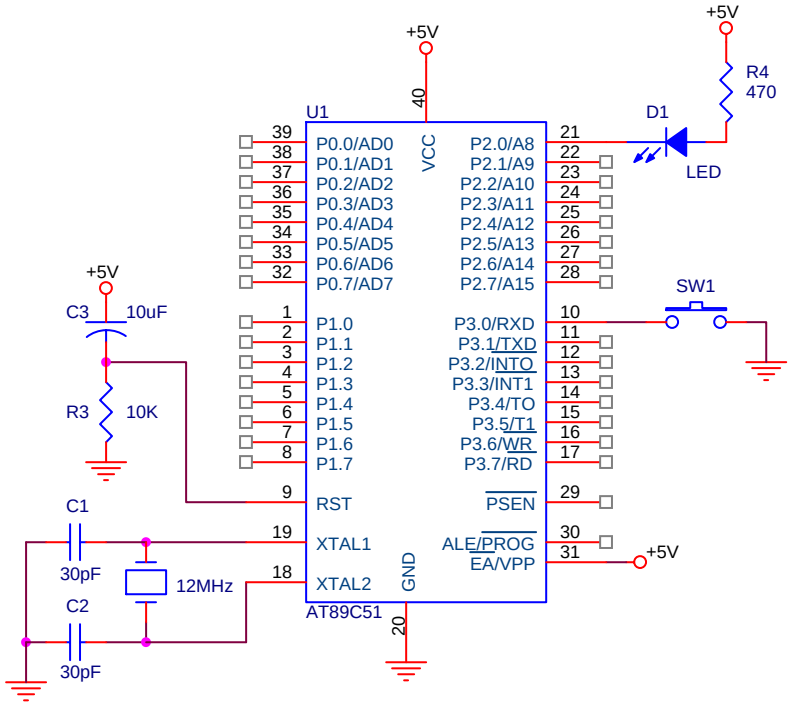
VD1: LED nhấp nháy.




```
        ORG     0
LOOP:   SETB    P2.0
        ACALL   DELAY
        CLR     P2.0
```

```
            ACALL   DELAY
            SJMP    LOOP

DELAY:      MOV     R6, #0FFh
LP2:        MOV     R7, #0FFh
LP1:        DJNZ    R7, LP1
            DJNZ    R6, LP2
            RET
```

VD2: Nhấn SW1 (tạo cạnh xuống) → LED sáng một lúc rồi tắt.



Pseudo code:

```
            Repeat until P3.0 = 1
            Repeat until P3.0 = 0
            P2.0 = 0
            Delay
            P2.1 = 1
```

Assembly code:

```
            ORG     0
            SETB    P3.0            ;P3.0: input
LOOP:       JNB     P3.0, LOOP
LOOP1:      JB      P3.0, LOOP1
            CLR     P2.0
            ACALL   DELAY
            SETB    P2.0
```

```
        SJMP    LOOP
DELAY:  MOV     R6, #0FFh
LP2:    MOV     R7, #0FFh
LP1:    DJNZ    R7, LP1
        DJNZ    R6, LP2
        RET
```

Ôn tập

Chương 1: Khái niệm cơ bản.

- Sơ đồ khối một hệ vi xử lý tổng quát.
- Bộ nhớ: ROM (các loại?), RAM.
    - Các chân địa chỉ: số chân ↔ dung lượng.
    - Các chân dữ liệu.
    - Các chân điều khiển: RAM có /OE và /WE, ROM chỉ có /OE. Trong hệ vi xử lý: /OE ↔ /RD, /WE ↔ /WR.
    - Địa chỉ chip nhớ = địa chỉ làm cho chân /CS (/CE) tích cực → mạch giải mã địa chỉ.
- Giải mã địa chỉ: toàn phần, một phần.
    - Bus địa chỉ có 16-bit, chip nhớ có n chân địa chỉ:
        - (16-n) đường tín hiệu đưa vào mạch GMĐC → GM toàn phần,
        - ít hơn (16-n) đường tín hiệu đưa vào mạch GMĐC → GM một phần.
    - Mạch GMĐC: thường dùng 74LS138, 74LS139, các cổng Logic.
- Thiết kế port nhập (dùng 74LS244), port xuất (dùng 74LS373).

Yêu cầu chương 1:

- Nhìn sơ đồ → xác định địa chỉ.
- Bản đồ địa chỉ → vẽ sơ đồ (thiết kế).

Chương 2: Họ VĐK 8051

- Đặc tính kỹ thuật:
    - Không gian bộ nhớ dữ liệu: 64KB, không gian bộ nhớ chương trình: 64KB. (Bộ nhớ on-chip 89C51: 128 byte RAM, 4K EEPROM.)
    - 4 port I/O 2 chiều.
    - 2 timer.
    - 1 port nối tiếp.
    - 5 nguồn ngắt
- Truy xuất ô nhớ → phải biết các kiểu định địa chỉ (cách chỉ định ô nhớ).
- Truy xuất RAM nội? Truy xuất bộ nhớ dữ liệu mở rộng (RAM ngoài)? Truy xuất bộ nhớ chương trình?
- Một số lệnh thường dùng (các lệnh trong các ví dụ).
- Kết hợp các lệnh nhảy để thực hiện các cấu trúc: repeat … until, while … do, if … then … else, …
- Timer:
    - Thanh ghi TMOD? Các bit: TFi, TRi (thanh ghi TCON)?
    - Dùng timer để định thời như thế nào?
- Sử dụng port: muốn 1 port là input thì làm như thế nào?
- Port nối tiếp:

    - Thanh ghi SCON?
    - Dùng Timer 1 để tạo baud rate → xác định trị nạp cho TH1?
    - Xuất một ký tự ra port nối tiếp?
    - Nhận một ký tự từ port nối tiếp?
- Ngắt:
    - Thanh ghi IE, IP? Các bit: ITi?
    - Bảng vector ngắt?
    - Cấu trúc một chương trình có sử dụng ngắt?

Yêu cầu chương 2 :
Viết chương trình cho 8051:
- Tra bảng.
- Truy xuất RAM ngoài, RAM trong.
- Copy khối dữ liệu.
- Đổi binary → BCD.
- Delay (ngắn/dài) không dùng Timer.
- Delay (ngắn/dài) dùng Timer.
- Phát/thu 1 ký tự qua port nối tiếp.
- Trình phục vụ ngắt thu/phát dữ liệu qua port nối tiếp.
- Tạo xung vuông dùng ngắt.
- Xử lý ngắt ngoài tác động mức/cạnh.

Chương 3: Ứng dụng
- Cách quét LED 7 đoạn.
- Các cách đọc A/D.
- Cách quét bàn phím HEX.
- Khởi động 8255. Đọc/xuất dữ liệu qua các port A, B, C (mode 0).

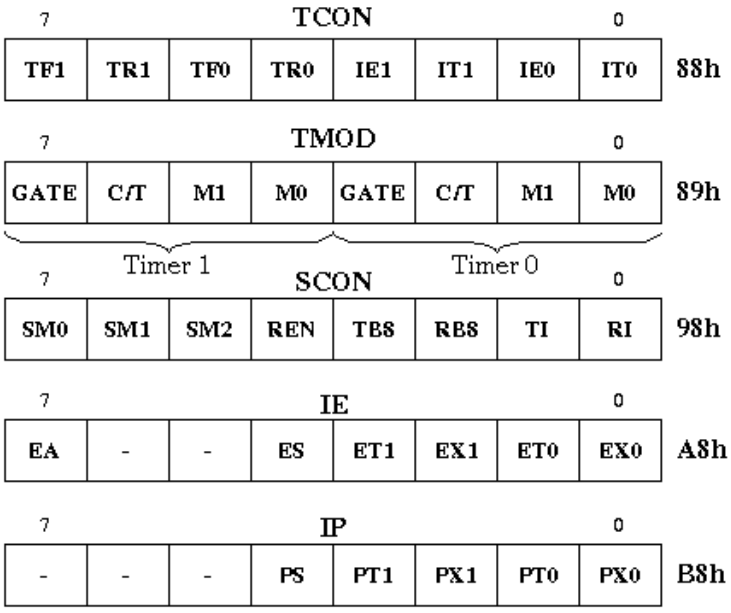
Tóm tắt

Sinh viên nên tự lập bảng tóm tắt:

- Bảng tổng kết các lệnh nhảy.

<condition>	Jump_if_not <conditon>	Jump_if_<conditon>
C = 1 bit = 1	**JNC** rel **JNB** bit, rel	**JC** rel **JB** bit, rel / **JBC** bit, rel
A = 0 Rn = 0 direct = 0	**JNZ** rel **DJNZ** Rn, rel **DJNZ** direct, rel	**JZ** rel
A ≠ direct A ≠ #data Rn ≠ #data @Ri ≠ #data	**CJNE** A, direct, rel **CJNE** A, #data, rel **CJNE** Rn, #data, rel **CJNE** @Ri, #data, rel	

- Các thanh ghi SFR



- Công thức tính giá trị nạp cho TH1 để tạo baud rate cho port nối tiếp.
  - SMOD = 0:

$$TH1 = 256 - \frac{f_{osc}}{384 \times Baud}$$

  - SMOD = 1:

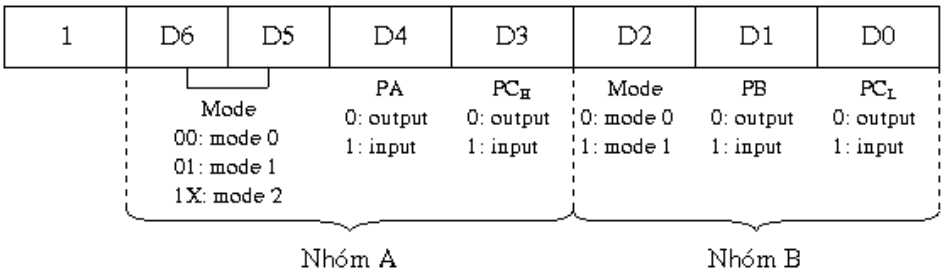
$$TH1 = 256 - \frac{f_{osc}}{192 \times Baud}$$

- Bảng vector ngắt:

Bảng vector ngắt

Ngắt	Cờ	Vector ngắt
Ngắt ngoài 0	IE0	0003h
Timer 0	TF0	000Bh
Ngắt ngoài 1	IE1	0013h
Timer 1	TF1	001Bh
Port nối tiếp	RI/TI	0023h

- Thanh ghi điều khiển 8255 ở mode 0:



- Bảng mã LED 7 đoạn.

Hiển thị	Anod chung	Cathode chung
0	C0h	3Fh
1	F9h	06h
2	A4h	5Bh
3	B0h	4Fh
4	99h	66h
5	92h	6Dh
6	82h	7Dh
7	F8h	07h
8	80h	7Fh
9	98h	67h
A	88h	77h
B	C6h	39h
C	86h	79h
D	8Eh	71h
E	82h	70h
F	89h	76h
.	7Fh	80h
[trắng]	FFh	00h