ĐẠI HỌC BÁCH KHOA HÀ NỘI
VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

# Hệ nhúng (Embedded Systems)

**Đỗ Công Thuần**

Bộ môn Kỹ thuật Máy tính

Email: thuandc@soict.hust.edu.vn

# Chương 4:
# Ghép nối ngoại vi với 8051

# Nội dung

- Ghép nối cổng vào ra song song
- Ghép nối ngắt ngoài
- Ghép nối nút bấm
- Ghép nối bộ định thời
- Lập trình C với 8051

# Ghép nối cổng vào ra song song

- 8051 có 4 cổng vào ra GPIO (mỗi cổng 8 bit): P0, P1, P2, P3
- Sau khi reset, các cổng ở chế độ mặc định là cổng ra (output)
- Để các cổng/chân làm việc ở chế độ cổng/chân vào (input) phải tiến hành ghi các bit 1 ra các cổng/chân tương ứng
  - Ví dụ: MOV P1,#0FF; Cổng 1 thành cổng vào

```
SETB  P1.0      ; Chân P1.0 làm chân vào
MOV   P1,#03    ; Chân P1.0 và P1.1 làm chân vào
                ; các chân còn lại làm chân ra
```

# Xuất dữ liệu ra cổng/chân

- Xuất dữ liệu ra cổng ra

  **MOV tên_cổng, giá trị**

  - Ví dụ: MOV P1, #55h
- Xuất dữ liệu ra từng chân
  - Đưa chân cổng lên mức cao:

    **SETB bit**

    Ví dụ: SETB P1.0
  - Đưa chân cổng xuống mức thấp:

    **CLR bit**

    Ví dụ: CLR P1.0

# Đọc dữ liệu từ cổng vào

- Bước 1: Thiết lập cổng làm việc ở chế độ input
- Bước 2: Đọc dữ liệu từ chân cổng

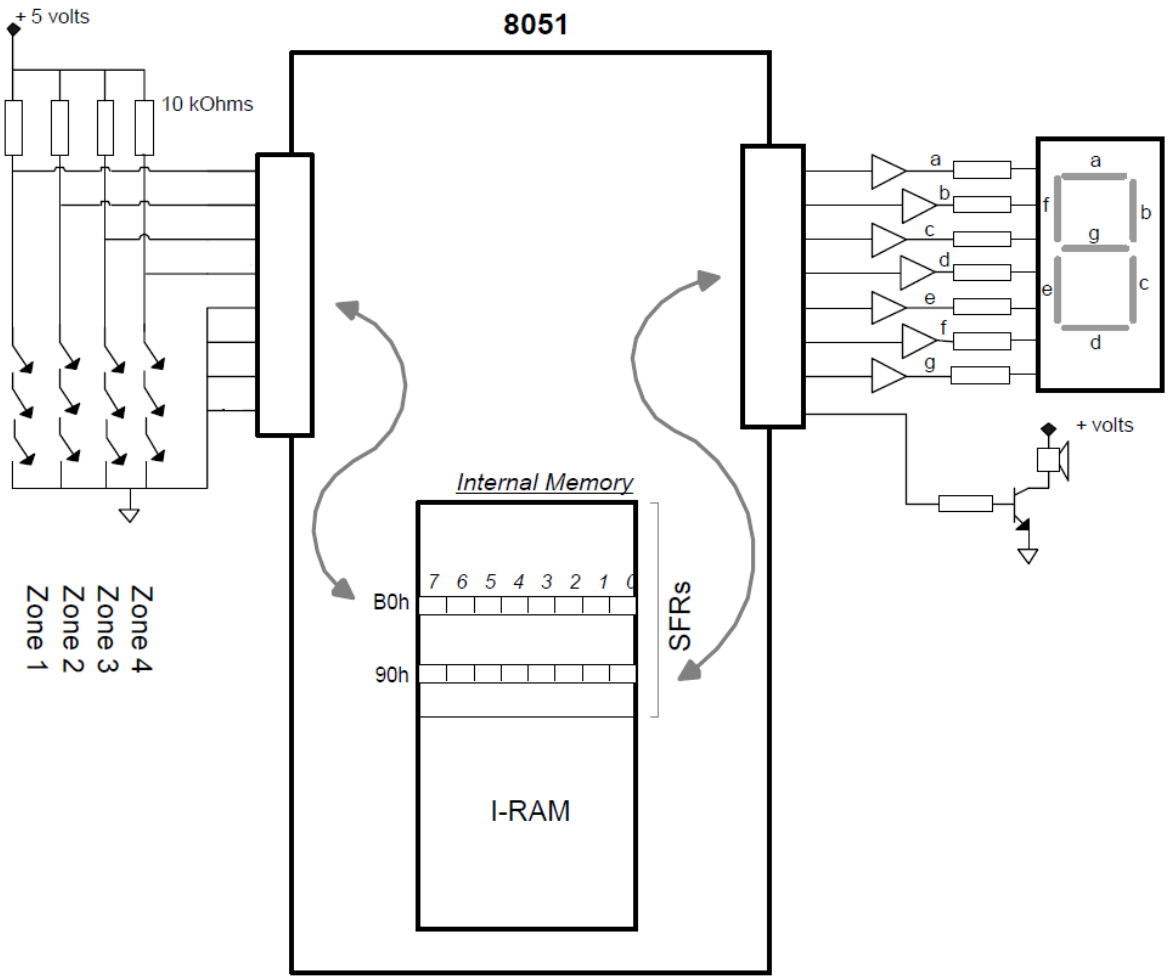
Ví dụ:

```
MOV P1, #0FFh
MOV A, P1        ; Đọc giá trị tại
                 ; cổng P1, lưu vào A
MOV P2, A        ; Xuất ra cổng P2
```

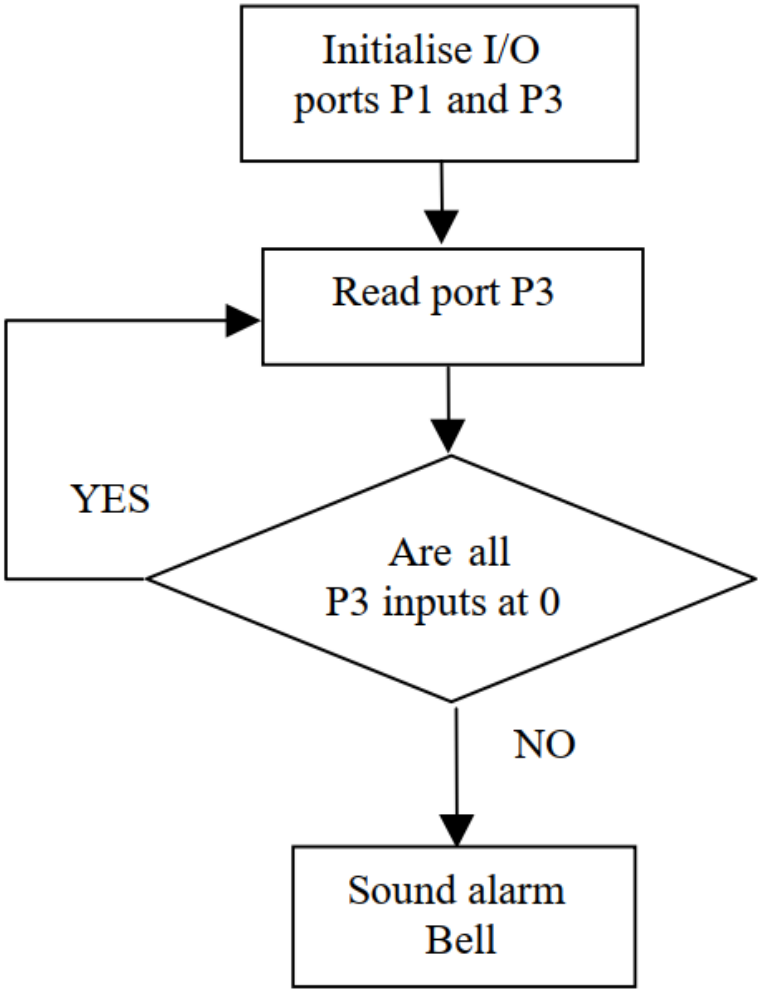
# Ví dụ 1

- Thiết kế mạch phát hiện đột nhập dùng công tắc đóng/mở
  - Nhà có 4 khu vực, mỗi khu vực có 3 công tắc lắp nối tiếp.
  - Tủ trung tâm lắp 1 chuông báo động, và một đèn LED 7 thanh báo vị trí khu vực phát hiện đột nhập

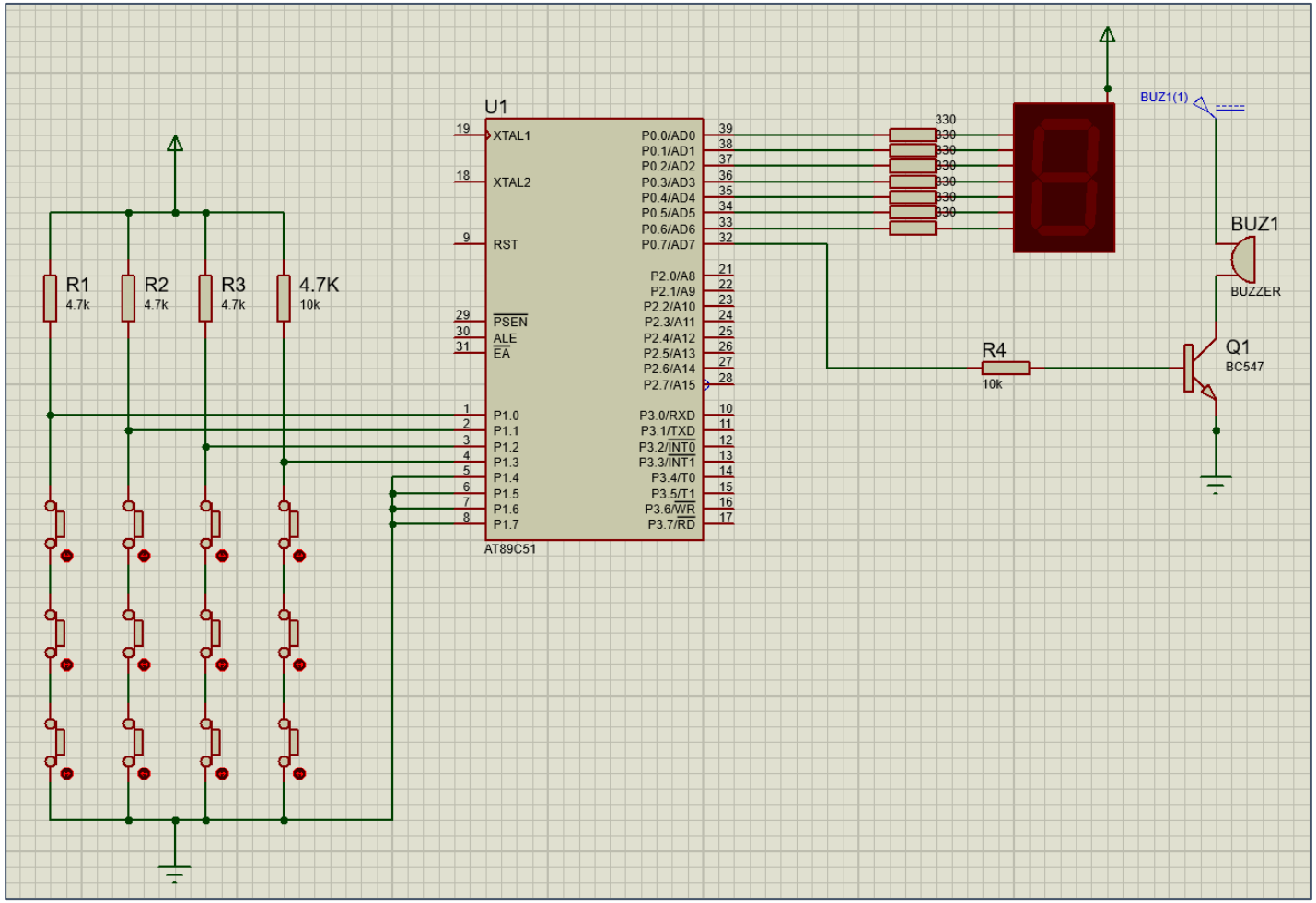
# Sơ đồ khối


+ 5 volts
8051
10 kOhms
a
b
c
d
e
f
g
+ volts
Internal Memory
7 6 5 4 3 2 1 0
B0h
90h
SFRs
I-RAM
Zone 1
Zone 2
Zone 3
Zone 4

# Thủ tục phát hiện đột nhập



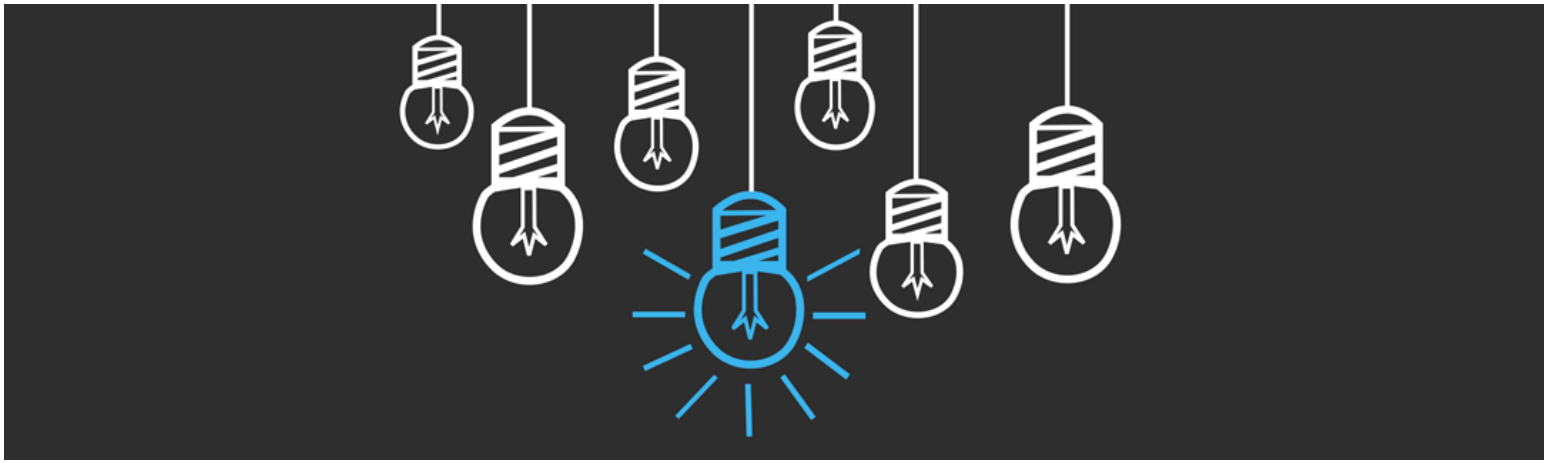
# Mạch Proteus



# Sample Code: Phát hiện đột nhập

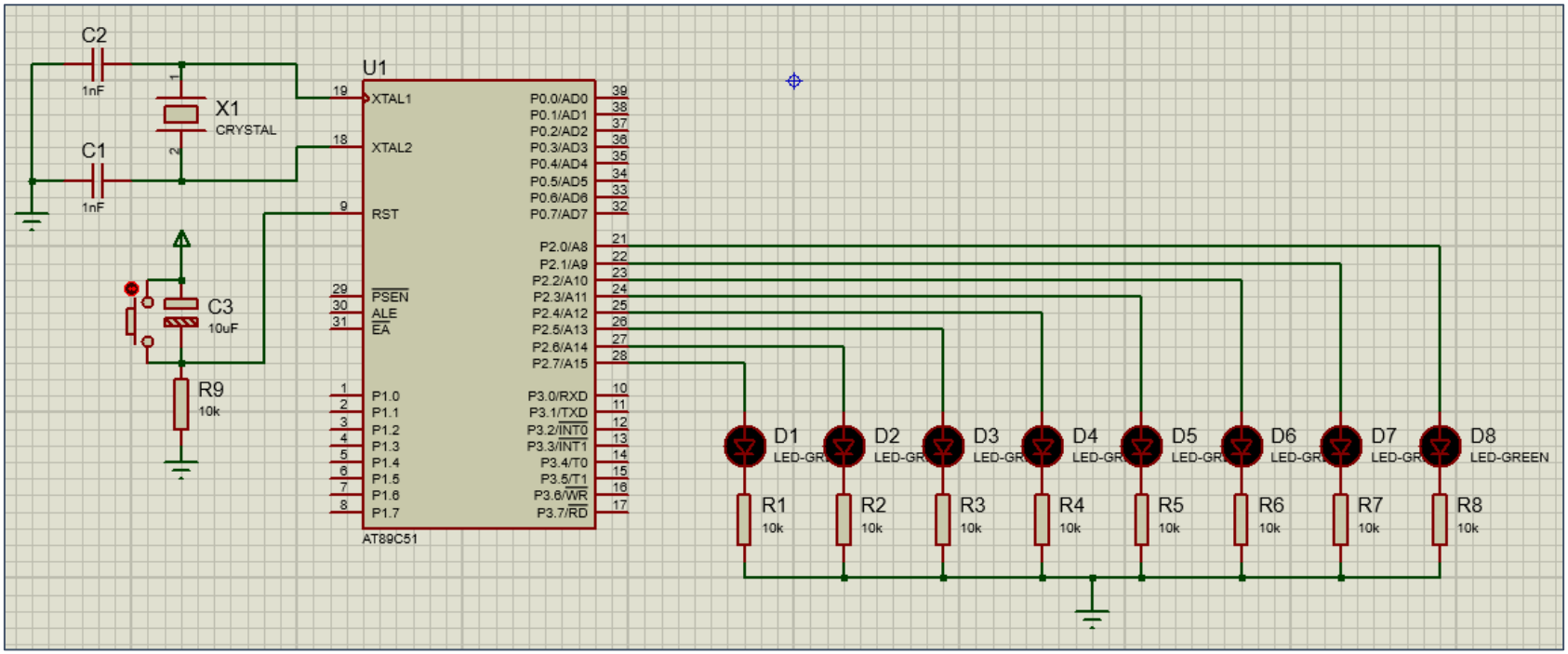
```
ORG 0000h
        MOV P1, #0ffh           ;set P1 as input
        MOV P0, #00             ;initialize P0
POLL:
        MOV A, P1               ;read sensors
        CJNE A, #00h, ALARM     ;fire alarm if detected
        LJMP POLL               ;otherwise polling sensor
ALARM:
        SETB P0.7               ;turn on buzzer
END_LOOP:
        LJMP END_LOOP           ;final endless loop
END
```

# Bài tập: Hiển thị vị trí đột nhập?



# Ví dụ 2

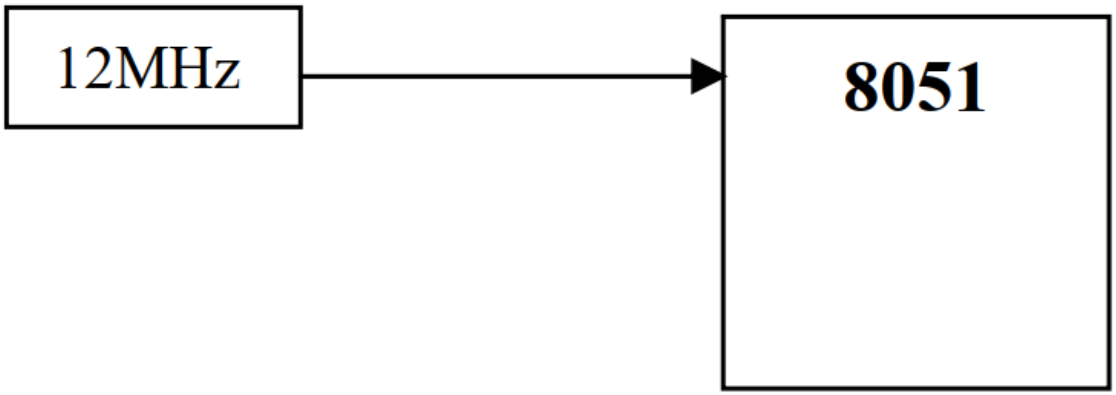
- Lập trình cho dãy đèn LED sáng lần từ trái sang phải



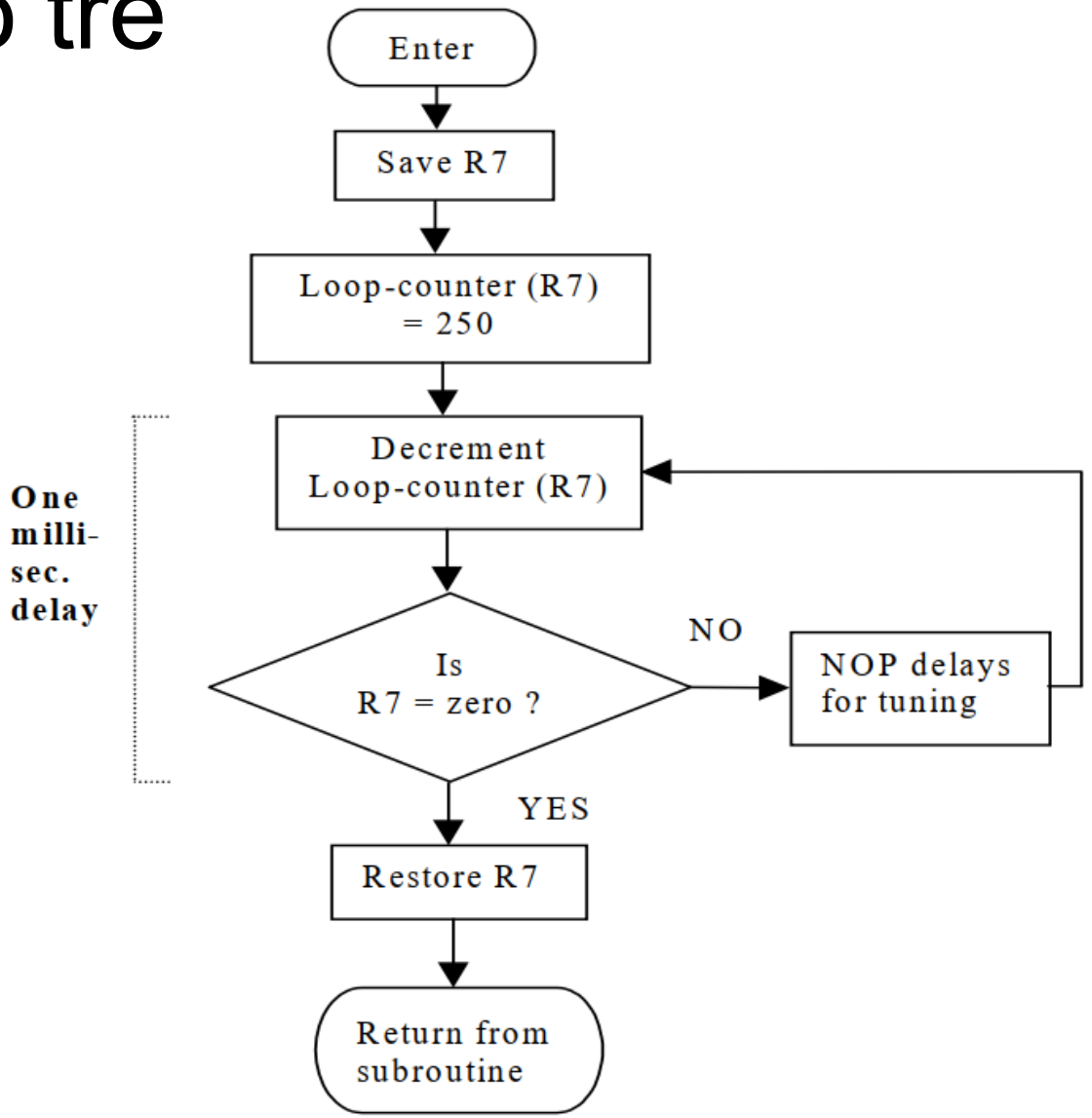
# Tính thời gian trễ chính xác?

- CPU 8051 chạy ở tần số 12 MHz

  → Chu kỳ lệnh là 1us



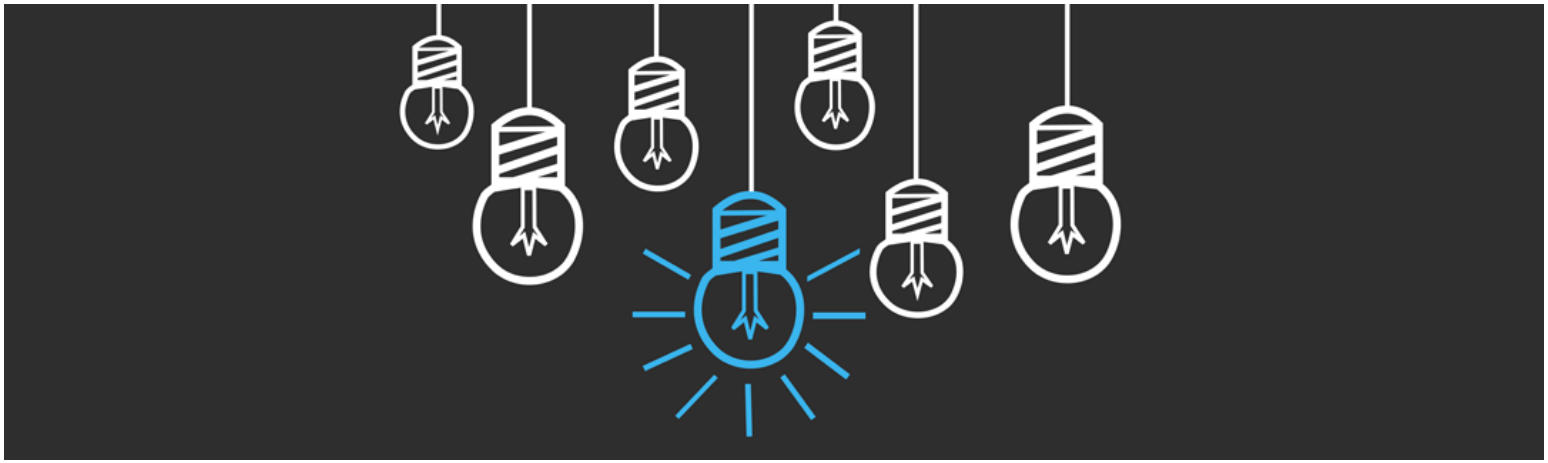
# Thủ tục tạo trễ 1ms



# Sample Code: Thủ tục tạo trễ 1ms

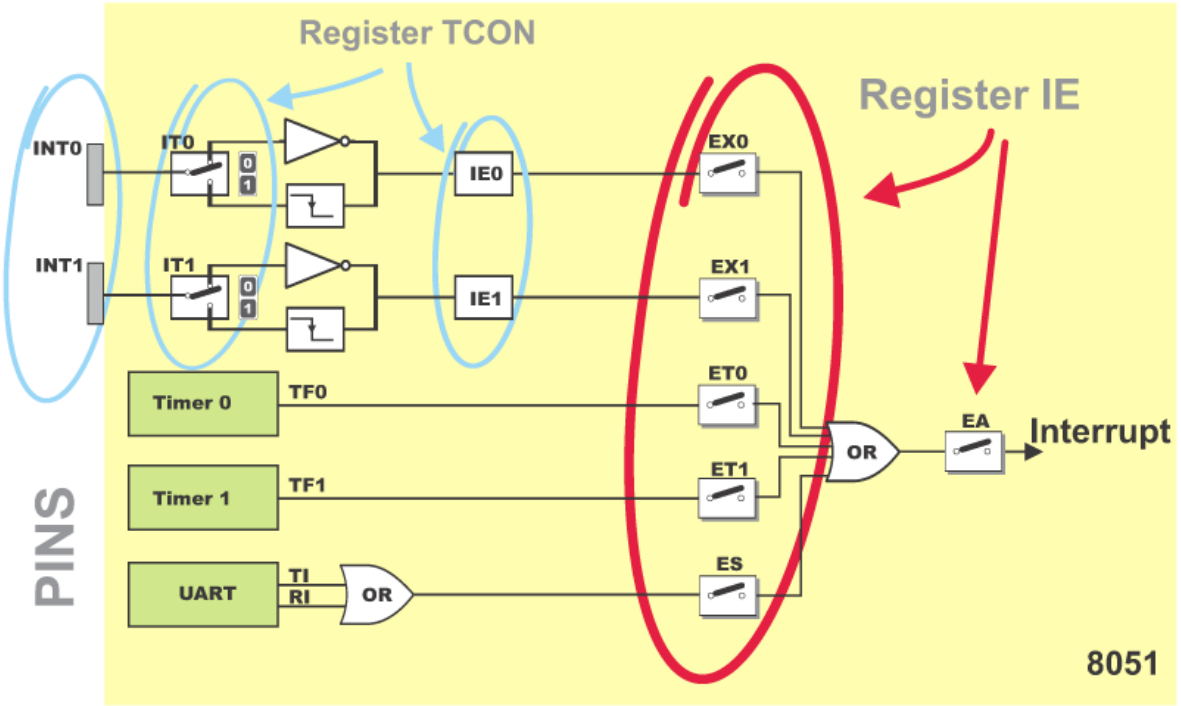
```
ONE_MILLI_SUB:
        PUSH 07h                    ; save R7 to stack
        MOV R7, #250d               ;
LOOP_1_MILLI:                       ; loops 250 times
        NOP                         ; two NOPs
        NOP                         ;
        DJNZ R7, LOOP_1_MILLI       ; loop until zero
        POP 07h                     ; restore R7
        RET                         ; return to caller
```

# Bài tập: Điều khiển dãy LED?



# Lập trình ngắt với 8051

- Có 5 nguồn ngắt lập trình được gồm:
- (INT0, INT1, TF0, TF1, TI & RI)




## Cấu hình ngắt sử dụng thanh ghi IE (Interrupt Enable)

| | 0 | X | 0 | 0 | 0 | 0 | 0 | 0 | Value after Reset |
|---|---|---|---|---|---|---|---|---|---|
| IE | EA | | ET2 | ES | ET1 | EX1 | ET0 | EX0 | Bit name |
| | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | |

# Các ngắt của 8051

- Các nguồn ngắt lập trình được
  - 2 ngắt ngoài (INT0, INT1)
  - 2 ngắt cho bộ đếm và định thời (TF0, TF1)
  - 1 ngắt cho bộ truyền thông nối tiếp UART
- Bảng vector ngắt của 8051

| Ngắt | Địa chỉ ROM (Hexa) | Chân |
|---|---|---|
| RESET | 0000 | 9 |
| Ngắt phần cứng ngoài (INT0) | 0003 | 12 (P3.2) |
| Ngắt bộ Timer 0 (TF0) | 000B | |
| Ngắt phần cứng ngoài 1 (INT1) | 0013 | 13 (P3.3) |
| Ngắt bộ Timer 1 (TF1) | 001B | |
| Ngắt COM nối tiếp (RI và TI) | 0023 | |

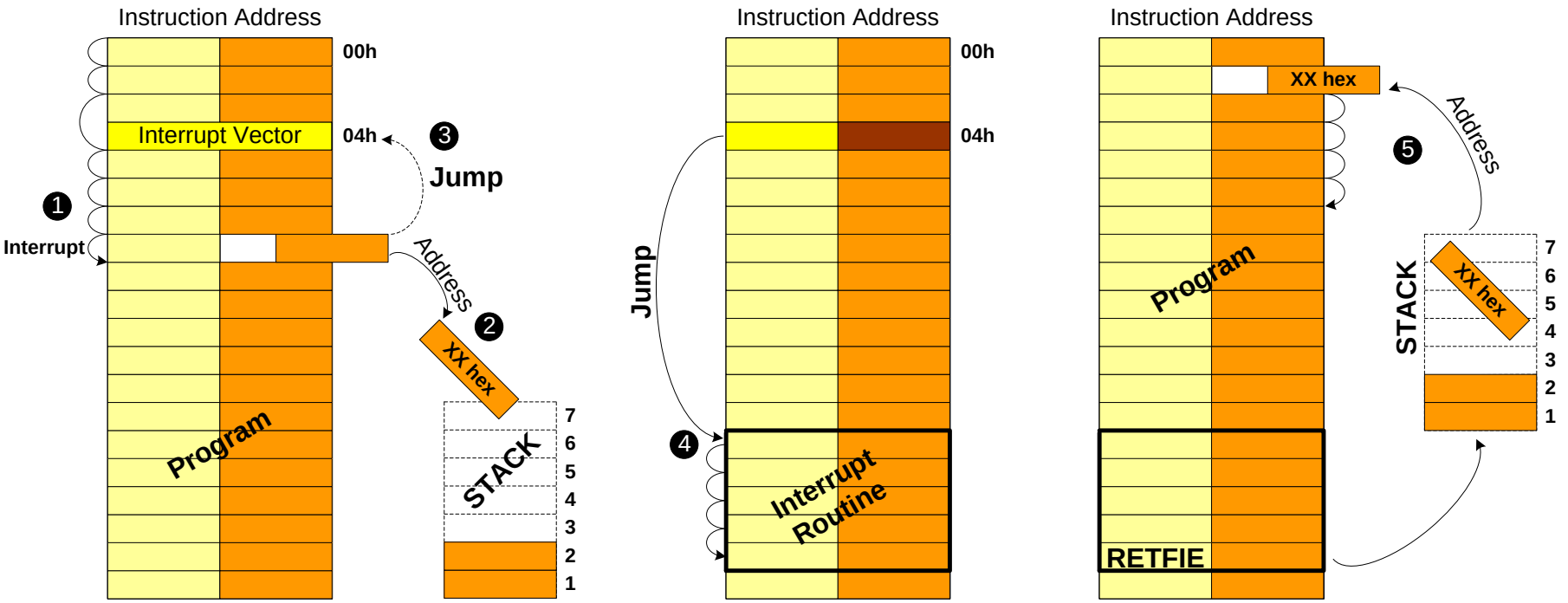
# Lập trình xử lý ngắt

- Trình phục vụ ngắt:
  - Mỗi ngắt luôn có một trình phục vụ ngắt (ISR)
  - Khi một ngắt được kích hoạt thì vi điều khiển thực thi trình phục vụ ngắt
  - Trình phục vụ ngắt của mỗi ngắt có một vị trí cố định trong bộ nhớ
  - Tập hợp các ô nhớ lưu giữ địa chỉ của tất cả các trình phục vụ ngắt gọi là bảng vector ngắt (**Interrupt Table**)

# Trình tự phục vụ ngắt của 8051

1. Kết thúc lệnh hiện tại, lưu địa chỉ của lệnh kế tiếp (PC) vào ngăn xếp
2. Nhảy đến vị trí cố định trong bảng vector ngắt
3. Nhận địa chỉ của trình phục vụ ngắt, nhảy tới địa chỉ đó và bắt đầu thực thi chương trình con phục vụ ngắt cho đến lệnh cuối cùng là lệnh RETI
4. Kết thúc chương trình con phục vụ ngắt, bộ vi điều khiển trở về thực thi tiếp lệnh nơi nó đã bị ngắt

# Trình tự phục vụ ngắt



# Nguồn ngắt ngoài

- Ngắt phần cứng ngoài (0 và 1)
  - Tín hiệu yêu cầu ngắt được gửi đến chân INT0 (P3.2) và chân INT1 (P3.3)
  - 8051 có thể nhận ngắt theo 2 kiểu:
    - Ngắt theo mức thấp: ngắt được kích hoạt khi có tín hiệu mức thấp đưa đến chân ngắt **(chế độ mặc định)**
    - Ngắt theo sườn âm: ngắt được kích hoạt khi có sườn âm (chuyển từ mức cao xuống mức thấp) đưa đến chân ngắt
  - Thiết lập chế độ kích hoạt ngắt qua thanh ghi **TCON (bit IT0 cho INT0 và IT1 cho INT1)**

# Thanh ghi TCON

| D7 | | | | | | | D0 |
|---|---|---|---|---|---|---|---|
| TF1 | TR1 | TF0 | TR0 | IE1 | IT1 | IE0 | IT0 |

| | | |
|---|---|---|
| TF1 | TCON.7 | Cờ tràn của Timer 1, được thiết lập bởi phần cứng khi bộ đếm/bộ định thời 1 tràn, và được xoá bởi phần cứng khi bộ xử lý nhảy đến trình phục vụ ngắt. |
| TR1 | TCON.6 | Bit điều khiển hoạt động của Timer 1, được thiết lập và xoá bởi phần mềm để bật/tắt bộ đếm/bộ định thời 1. |
| TF0 | TCON.5 | Tương tự như TF1 nhưng là cho Timer 0. |
| TR0 | TCON.4 | Tương tự như TR1 nhưng là cho Timer 0. |
| IE1 | TCON.3 | Cờ ngắt ngoài 1 kích phát sườn, được CPU thiết lập khi phát hiện có sườn xuống ngắt ngoài và được CPU xóa khi ngắt được xử lý.<br>Lưu ý: Cờ này không chốt ngắt kích phát mức thấp. |
| IT1 | TCON.2 | Bit điều khiển kiểu ngắt 1 (Interrrupt 1 Type Control Bit) được thiết lập và xoá bởi phần mềm để xác định kiểu ngắt ngoài kích phát sườn xuống hay mức thấp. |
| IE0 | TCON.1 | Tương tự như IE1 nhưng là cho ngắt ngoài 0. |
| IT0 | TCON.0 | Tương tự như bit IT1 nhưng là cho ngắt ngoài 0. |

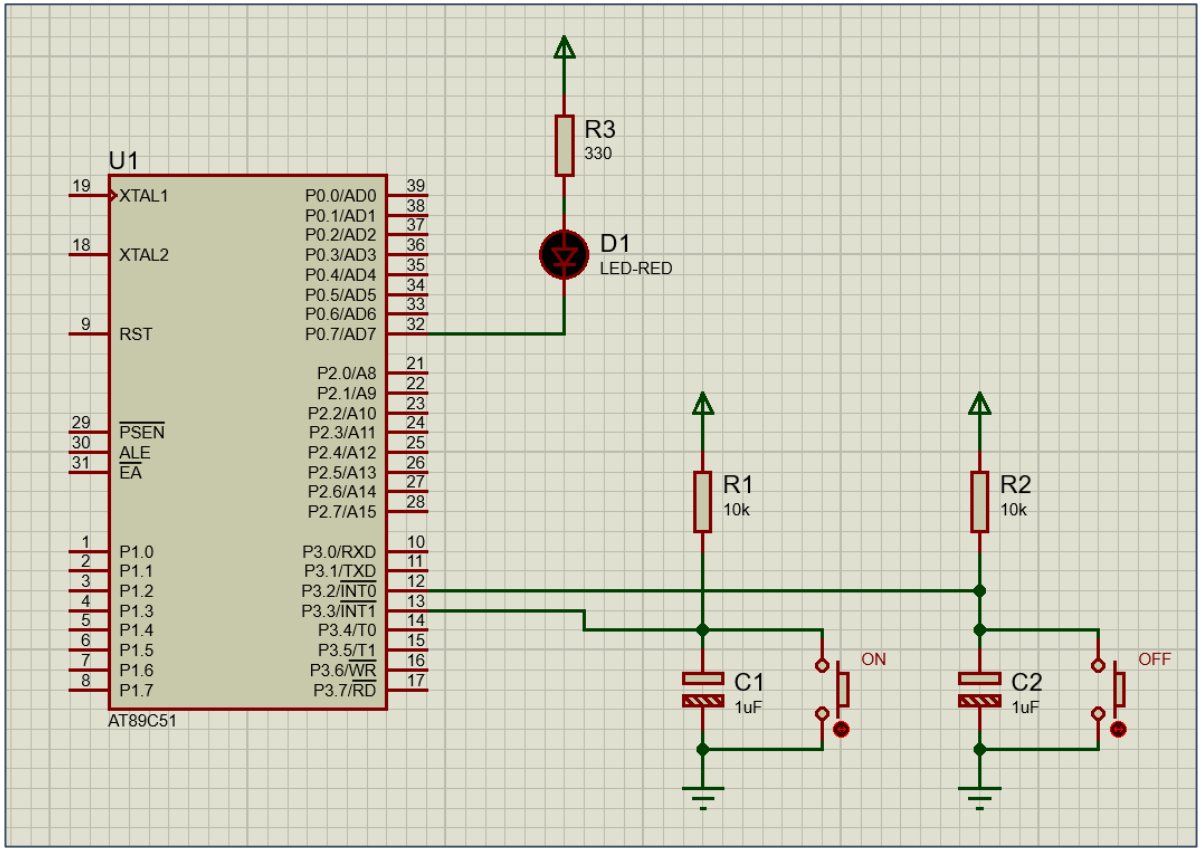
# Lập trình với ngắt ngoài

```
;Interrupt table
ORG 0000h                ; entry address for 8051 RESET
        LJMP MAIN        ; move MAIN away from interrupt vector table
ORG 0003h                ; vector address for interrupt 0
        LJMP ISR0        ; jump to ISR0
ORG 0013h                ; vector address for interrupt 1
        LJMP ISR1        ; jump to ISR1
ORG 0100h                ; MAIN starts here
MAIN:
        MOV IE, #10000101B      ; enable external interrupts IE0, IE1
        SETB IT0                ; negative edge trigger for interrupt 0
        SETB IT1                ; negative edge trigger for interrupt 1
        SETB P1.0               ; LED ON
LOOP:
LJMP LOOP                       ; end loop
```

# Lập trình với ngắt ngoài (tiếp)

```
;=========================================================
; ISR0
;=========================================================
ISR0:
        SETB P1.0                       ; LED ON
        RETI                            ; return from interrupt
;=========================================================
; ISR1
;=========================================================
ISR1:
        CLR P1.0                        ; LED OFF
        RETI                            ; return from interrupt
END                                     ; end of program
```

# Ghép nối nút bấm

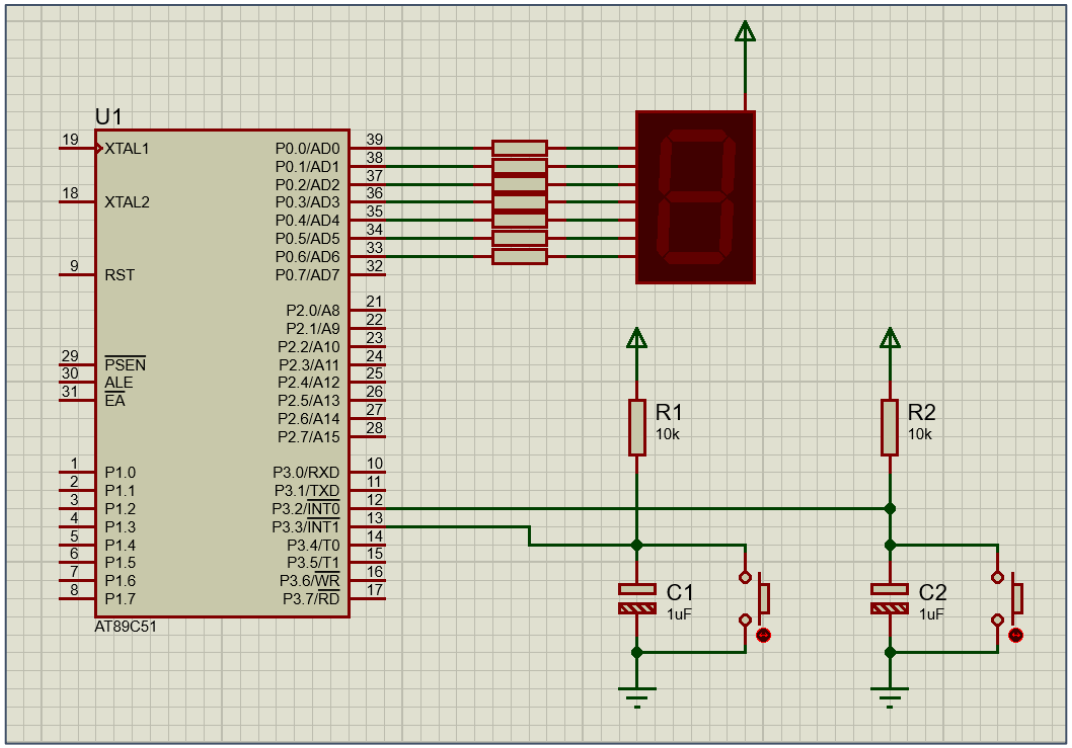


# Bài tập 1

- Lập trình chương trình cho phép điều khiển bật/tắt LED bằng 2 nút bấm ON và OFF
- Tham khảo code mẫu
- **Chú ý**:
  - Chống nảy phím bằng phần cứng
  - Nắm được nguyên tắc hoạt động của ngắt

# Bài tập 2

- Thay LED bằng LED 7 thanh. Lập trình để hai nút bấm thực hiện chức năng tăng/giảm giá trị hiện trên LED.



# Bài tập 2

- Tham khảo code mẫu
- **Chú ý**:
  - Tổ chức dữ liệu và gán chức năng cụ thể cho từng tiến trình
    - Dữ liệu: lưu trữ trạng thái hệ thống
    - Ngắt ngoài: tiến trình ngắn, chỉ cập nhật dữ liệu hoặc trạng thái hệ thống.
    - Main loop: thực hiện chức năng chính của hệ thống.

# Counter/Timer

- 8051 có hai bộ đếm/định thời
  - Timer/Counter 0 (T0) và Timer/Counter 1 (T1)
- Ứng dụng
  - Bộ định thời (Timer): Tạo ra các sự kiện định thời sau khoảng thời gian thực chính xác, có thể dùng để tạo trễ.
  - Bộ đếm (Counter): Đếm xung

# Các thanh ghi Counter/Timer

- Thanh ghi TCON
  - TR1/TR0: bit khởi động/tắt
  - TF1/TF0: cờ báo tràn bộ đếm/định thời
  - IE1, IT1, IE0, IT0: liên quan tới ngắt phần cứng ngoài

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | Value after Reset |
| TCON | TF1 | TR1 | TF0 | TR0 | IE1 | IT1 | IE0 | IT0 | Bit name |
| | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | |

# Các thanh ghi Counter/Timer

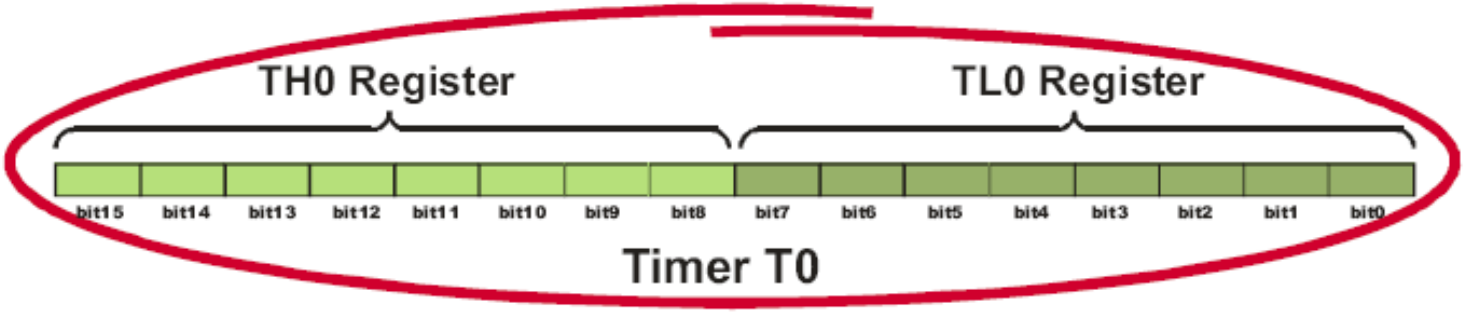
- Thanh ghi TMOD
  - Gate: sử dụng cho bộ đếm
  - C/T: chọn chế độ (bộ đếm hay bộ định thời)
    - C/T=0: bộ định thời
    - C/T=1: bộ đếm
  - M1,M0: chọn chế độ làm việc. Hai chế độ thông dụng
    - M1=0, M0=1: chế độ 1, bộ Timer/Counter 16 bit
    - M1=1, M0=0: chế độ 2, bộ Timer/Counter 8 bit tự động nạp lại

**TMOD**

| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | Value after reset |
|---|---|---|---|---|---|---|---|---|
| GATE1 | C/T1 | T1M1 | T1M0 | GATE0 | C/T0 | T0M1 | T0M0 | Bit name |
| bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 | |

# Các thanh ghi Counter/Timer

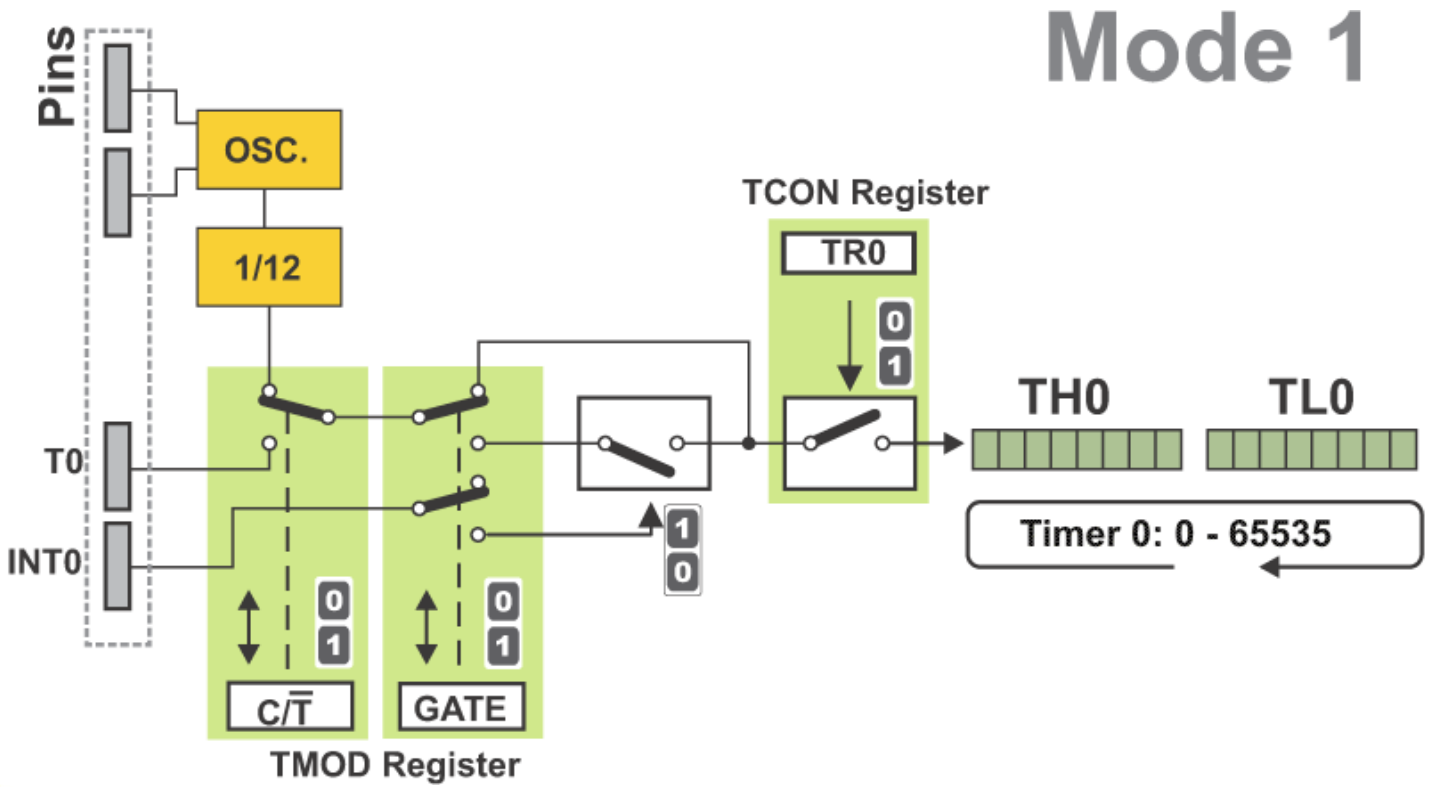
- Thanh ghi giá trị Timer (Timer T0) 16 bit
- Gồm 2 thanh ghi 8 bit: TH0, TL0




- Công thức: $T0 = (TH0 << 8) + TL0$
  - Ví dụ: $(3 \times 256) + 232 = 1000$

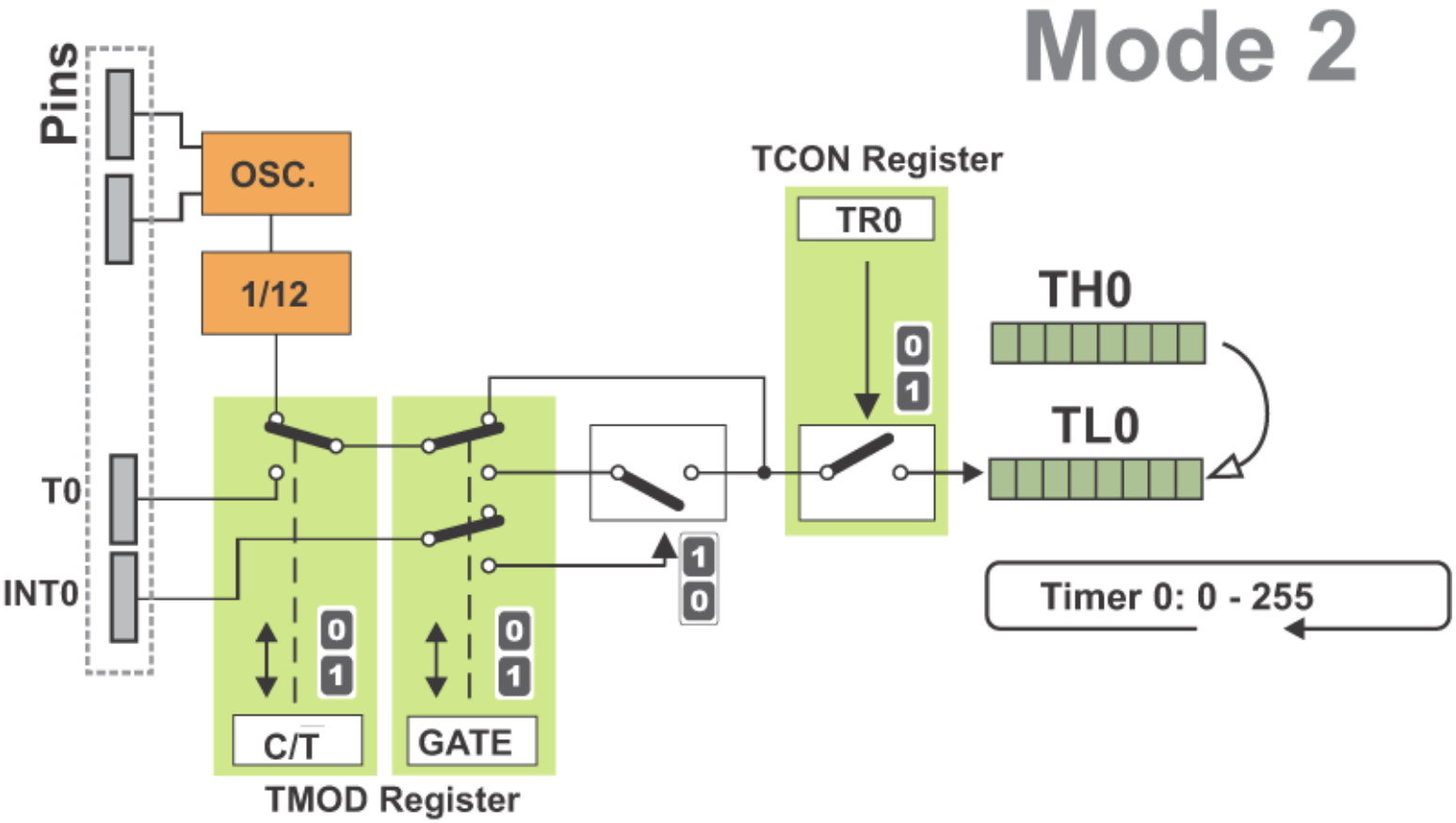
# Counter/Timer

- Timer 0 Mode 1 (16 bit):
  - Start from 0 to 65535 -> Timer Interrupt
  - Clear to 0



# Counter/Timer

- Timer 0 Mode 2 (8 bit auto-reload)



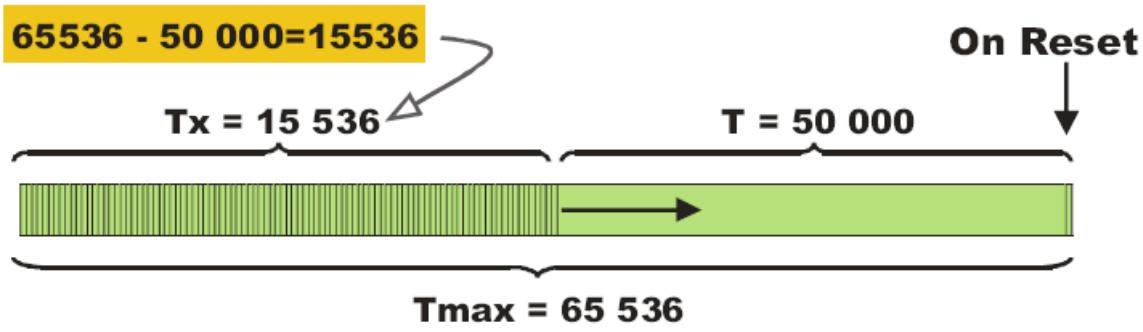
# Lập trình bộ định thời

**Các bước lập trình:**

- Tính toán giá trị ban đầu cho thanh ghi bộ định thời.
- Nạp giá trị này vào thanh ghi bộ định thời.
- Khởi động bộ định thời.
- Thanh ghi bộ định thời sẽ tự động tăng cho đến giá trị tối đa và thiết lập cờ TF báo tràn.

# Ví dụ tạo trễ 50ms (với tần số 12MHz)

- **Bước 1**: Tính toán giá trị ban đầu cho thanh ghi bộ định thời:
  - Chu kỳ timer: 1 us → cần tạo trễ 50,000 chu kỳ
  - 65536 – 50000 =15536 (0x3CB0)
- **Bước 2**: Nạp giá trị vào thanh ghi bộ định thời
  - Với Timer 0: TH0=0x3C, TL0=0xB0
  - Với Timer 1: TH1=0x3C, TL1=0xB0
- **Bước 3**: Khởi động bộ định thời
  - Với Timer 0: TR0=1
  - Với Timer 1: TR1=1
- **Bước 4**: Chờ cờ tràn được thiết lập (TF0=1 hoặc TF1=1)



# Bài tập

- Tạo trễ 50 ms trước khi bắt đầu khởi tạo chương trình.
- Mục đích để có cơ hội vào boot loader ở các mạch dùng boot loader.
- **Chú ý**: Proteus 8.7 không mô phỏng chính xác được thời gian trễ của timer.

# Bài tập

- Vẽ và lập trình mạch kitchen timer
  - 4 phím (+), (-), OK, Cancel
  - 1 màn hình LED 7 thanh
  - 1 còi báo
- Hoạt động:
  - Bấm +, - để set thời gian.
  - Cancel để xóa về 0 và reset trạng thái mạch
  - Bấm OK để bắt đầu đếm lùi theo phút. Khi về 0 thì báo còi. Còi kêu tới khi nút OK hoặc Cancel được bấm.
- **Sử dụng ngôn ngữ Assembly**