

## 5

## HOẠT ĐỘNG CỦA PORT NỐI TIẾP

### 5.1 MỞ ĐẦU

Bên trong chip 8051 có một port nối tiếp hoạt động ở một vài chế độ trên một tầm tần số rộng. Chức năng cơ bản của port nối tiếp là thực hiện việc chuyển đổi dữ liệu song song thành nối tiếp khi phát và chuyển đổi dữ liệu nối tiếp thành song song khi thu.

Các mạch phần cứng bên ngoài truy xuất port nối tiếp thông qua các chân TxD ( phát dữ liệu ) và RxD ( thu dữ liệu ) đã được giới thiệu ở chương 2. Các chân này đã hợp với hai chân của port 3 : P3.1 ( TxD ) và P3.0 ( RxD ).

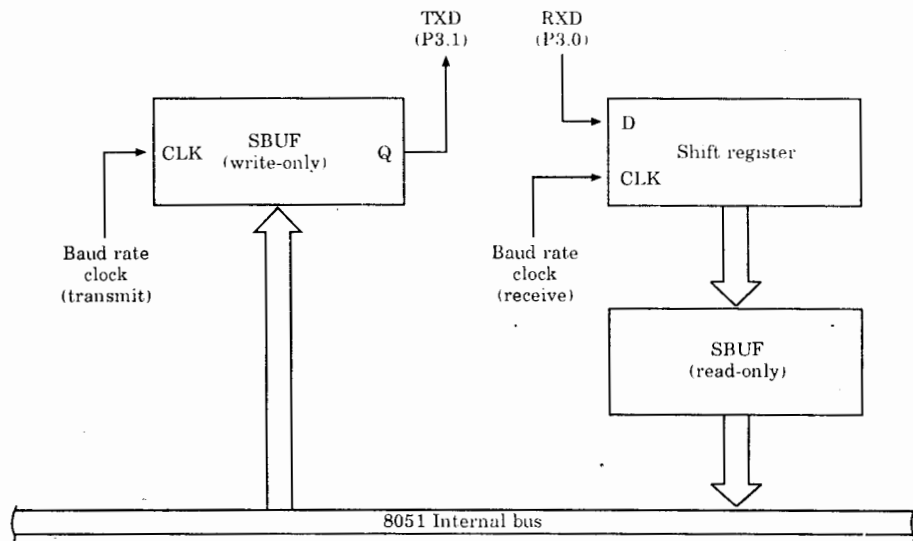
Đặc trưng của port nối tiếp là hoạt động song công ( full duplex ), nghĩa là có khả năng thu và phát đồng thời. Ngoài ra port nối tiếp còn có một đặc trưng khác, việc đệm dữ liệu khi thu của port này cho phép một ký tự được nhận và lưu giữ trong bộ đệm thu trong khi ký tự tiếp theo được nhận vào. Nếu CPU đọc ký tự thứ nhất trước khi ký tự thứ hai được nhận đầy đủ, dữ liệu sẽ không bị mất.

Phần mềm sử dụng hai thanh ghi chức năng đặc biệt SBUF và SCON để truy xuất port nối tiếp. Bộ đệm của port nối tiếp SBUF có địa chỉ byte là 99H, trên thực tế bao gồm hai bộ đệm. Việc ghi lên SBUF sẽ nạp dữ liệu để phát và việc đọc SBUF sẽ truy xuất dữ liệu đã nhận được. Điều này có nghĩa là ta có hai thanh ghi riêng rẽ và phân biệt : thanh ghi phát ( chỉ ghi ) hay bộ đệm phát và thanh ghi thu ( chỉ đọc ) hay bộ đệm thu ( xem hình 5.1 ).

Thanh ghi điều khiển port nối tiếp SCON ( có địa chỉ byte là 98H ) là thanh ghi được định địa chỉ từng bit, thanh ghi này chứa các bit trạng thái và các bit điều khiển. Các bit điều khiển sẽ thiết lập chế độ hoạt động cho port nối tiếp còn các bit trạng thái chỉ ra sự kết thúc việc thu

hoặc phát một ký tự. Các bit trạng thái được kiểm tra bởi phần mềm hoặc được lập trình để tạo ra ngắt.

Tần số hoạt động của *port* nối tiếp, hay còn gọi là tốc độ baud ( baud rate ) có thể cố định hoặc thay đổi. Khi tốc độ baud thay đổi được sử dụng, bộ định thời 1 cung cấp xung *clock* tốc độ baud và ta phải lập trình sao cho phù hợp. Trên 8032/8052, bộ định thời 2 cũng có thể được lập trình để cung cấp xung *clock* tốc độ baud.



Hình 5.1 : Sơ đồ khối của *port* nối tiếp

Write only : chỉ ghi

Read only : chỉ đọc

Shift register : thanh ghi dịch bit

Baud rate clock ( transmit ) : xung *clock* tốc độ baud ( phát )

Baud rate clock ( receive ) : xung *clock* tốc độ baud ( thu )

8051 internal bus : bus nội của 8051

## 5.2 THANH GHI ĐIỀU KHIỂN *PORT* NỐI TIẾP

Chế độ hoạt động của *port* nối tiếp được thiết lập bằng cách ghi từ điều khiển lên thanh ghi chọn chế độ SCON của *port* nối tiếp ở địa chỉ byte 99H ( xem bảng 5.1 và 5.2 ).

Trước khi sử dụng *port* nối tiếp, thanh ghi SCON phải được khởi động đúng chế độ yêu cầu. Thí dụ lệnh sau :

```
MOV SCON, #01010010B
```

khởi động *port* nối tiếp ở chế độ 1 (  $SM0/SM1 = 0/1$  ), cho phép thu (  $REN = 1$  ) và *set* cờ ngắt phát bằng 1 (  $TI = 1$  ) để chỉ ra rằng *port* nối tiếp sẵn sàng phát dữ liệu.

### 5.3 CÁC CHẾ ĐỘ HOẠT ĐỘNG

*Port* nối tiếp của 8051 có 4 chế độ hoạt động, các chế độ được chọn bằng cách ghi 1 hoặc 0 cho các bit  $SM0$  và  $SM1$  trong thanh ghi  $SCON$ . Ba trong số các chế độ hoạt động cho phép truyền không đồng bộ ( asynchronous ), trong đó mỗi một ký tự được thu hoặc được phát sẽ cùng với một bit start và một bit stop tạo thành một khung ( frame ).

Nếu đã làm quen với hoạt động của *port* nối tiếp theo chuẩn RS-232C trên một máy vi tính, ta sẽ tìm thấy được sự tương tự ở các chế độ này. Với chế độ thứ tư, *port* nối tiếp hoạt động như một thanh ghi dịch bit đơn giản. Mỗi một chế độ sẽ được đề cập tóm tắt sau đây.

| Bit      | Ký hiệu | Địa chỉ | Mô tả                                                                                                                                                                        |
|----------|---------|---------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| $SCON.7$ | $SM0$   | 9FH     | bit 0 chọn chế độ của <i>port</i> nối tiếp                                                                                                                                   |
| $SCON.6$ | $SM1$   | 9EH     | bit 1 chọn chế độ của <i>port</i> nối tiếp                                                                                                                                   |
| $SCON.5$ | $SM2$   | 9DH     | bit 2 chọn chế độ của <i>port</i> nối tiếp.<br><br>Bit này cho phép truyền thông đa xử lý ở các chế độ 2 và 3; bit $RI$ sẽ không được tích cực nếu bit thứ 9 nhận được là 0. |
| $SCON.4$ | $REN$   | 9CH     | cho phép thu. Bit này phải được <i>set</i> để nhận các ký tự.                                                                                                                |
| $SCON.3$ | $TB8$   | 9BH     | bit phát 8. Bit thứ 9 được phát ở các chế độ 2 và 3; được <i>set</i> và xóa bởi phần mềm                                                                                     |
| $SCON.2$ | $RB8$   | 9AH     | bit thu 8. Bit thứ 9 nhận được.                                                                                                                                              |
| $SCON.1$ | $TI$    | 99H     | cờ ngắt phát. Cờ này được <i>set</i> ngay khi kết thúc việc phát một ký tự ; được xóa bởi phần mềm                                                                           |
| $SCON.0$ | $RI$    | 98H     | cờ ngắt thu. Cờ này được <i>set</i> ngay khi kết thúc việc thu một ký tự ; được xóa bởi phần mềm                                                                             |

Bảng 5.1 : Tóm tắt thanh ghi  $SCON$  ( điều khiển *port* nối tiếp )

### 5.3.1 Thanh ghi dịch 8-bit ( chế độ 0 )

Chế độ 0, được chọn bằng cách ghi giá trị 0 vào các bit SM0 và SM1 trong thanh ghi SCON, đặt *port* nối tiếp vào chế độ thanh ghi dịch 8-bit. Dữ liệu nối tiếp được thu và phát thông qua chân RxD, chân TxD xuất xung *clock* dịch bit. Khi phát và thu dữ liệu 8-bit, bit có ý nghĩa ( giá trị vị trí ) nhỏ nhất ( bit LSB ) được thu hoặc phát trước tiên. Tốc độ baud cố định và bằng 1/12 tần số của mạch dao động trên *chip*. Các thuật ngữ “ RxD ” và “ TxD ” bị sai lệch ý nghĩa trong chế độ này. Chân RxD được sử dụng cho cả thu và phát dữ liệu còn chân TxD được dùng làm chân xuất xung *clock* dịch bit.

| SM0 | SM1 | Chế độ | Mô tả          | Tốc độ baud                                |
|-----|-----|--------|----------------|--------------------------------------------|
| 0   | 0   | 0      | Thanh ghi dịch | Cố định ( tần số dao động / 12 )           |
| 0   | 1   | 1      | UART 8-bit     | Thay đổi ( thiết lập bởi bộ định thời )    |
| 1   | 0   | 2      | UART 9-bit     | Cố định ( tần số dao động / 12 hoặc / 64 ) |
| 1   | 1   | 3      | UART 9-bit     | Thay đổi ( thiết lập bởi bộ định thời )    |

Bảng 5.2 : Các chế độ của *port* nối tiếp

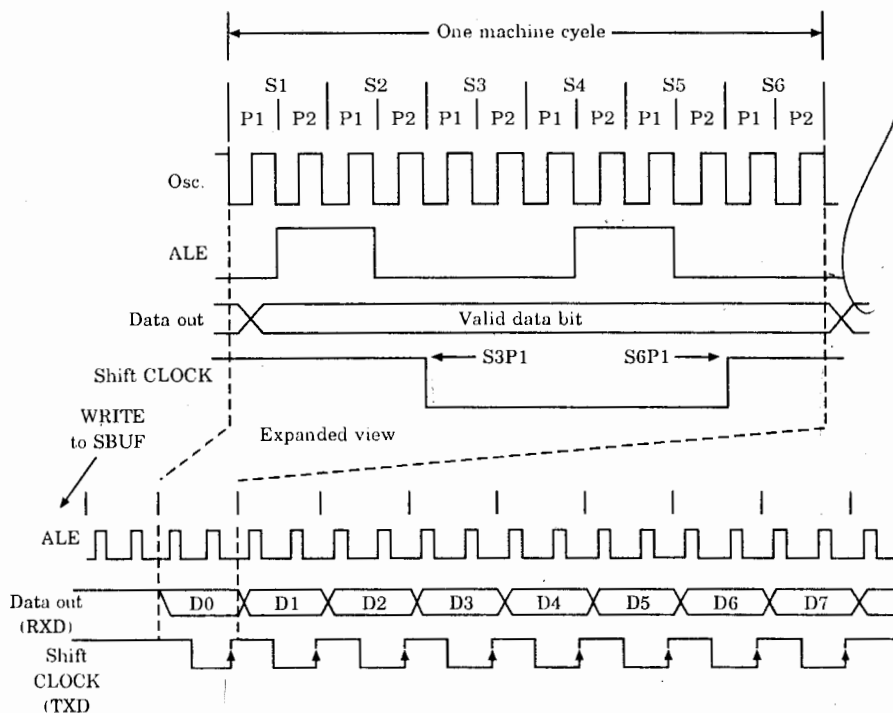
Việc phát dữ liệu được khởi động bằng một lệnh ghi dữ liệu vào SBUF. Dữ liệu được dịch ra ngoài trên chân RxD ( P3.0 ) với các xung *clock* dịch bit được gửi ra trên chân TxD ( P3.1 ). Mỗi một bit hợp lệ được truyền đi trên đường RxD trong một chu kỳ máy.

Trong mỗi một chu kỳ máy, xung *clock* dịch bit đổi thành mức thấp ở S3P1 và trở lại mức cao ở S6P1.

Giản đồ thời gian phát dữ liệu được trình bày ở hình 5.2.

Việc thu dữ liệu được khởi động khi bit cho phép thu REN ở logic 1 và cờ ngắt thu RI ở logic 0. Qui luật tổng quát là ta phải *set* bit REN bằng 1 ở thời điểm bắt đầu chương trình để khởi động *port* nối tiếp và sau đó xóa bit RI để bắt đầu công việc thu dữ liệu.

Khi bit RI được xóa, các xung *clock* dịch bit được xuất ra trên chân TxD, ta bắt đầu chu kỳ máy tiếp theo và dữ liệu được dịch vào chân RxD bởi xung *clock* dịch bit ( hiển nhiên là các mạch ghép nối để cung cấp dữ liệu trên đường RxD được đồng bộ bởi xung *clock* dịch bit trên đường TxD ( xem hình 5.3 ).



Hình 5.2 : Giảm đồ thời gian phát dữ liệu ở chế độ 0

One machine cycle : một chu kỳ máy

Osc : xung clock của mạch dao động

ALE : xung ALE

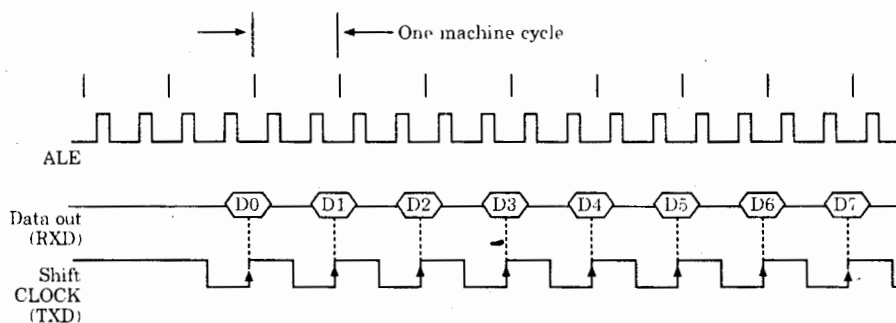
Data out : dữ liệu xuất

Valid data bit : bit dữ liệu hợp lệ

Shift CLOCK : xung clock dịch bit

WRITE to SBUF : ghi vào SBUF

Expanded view : quan sát được phóng đại



Hình 5.3 : Giảm đồ thời gian thu dữ liệu ở chế độ 0

One machine cycle : một chu kỳ máy

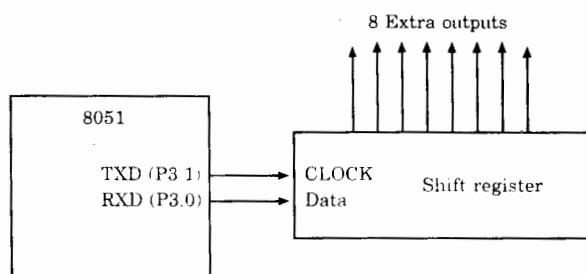
ALE : xung ALE

Data out : dữ liệu xuất

Shift CLOCK : xung *clock* dịch bit

Việc dịch dữ liệu vào port nối tiếp xảy ra ở cạnh dương ( cạnh lên ) của Tx.D.

Một ứng dụng khả thi của chế độ 0 ( chế độ thanh ghi dịch bit ) là mở rộng thêm các ngõ ra cho 8051. Một vi mạch thanh ghi dịch nối tiếp-song song có thể được nối với các chân Tx.D và Rx.D của 8051 để cung cấp thêm 8 đường xuất ( xem hình 5.4 ). Các thanh ghi dịch bit khác có thể ghép *cascade* với thanh ghi dịch bit đầu tiên để mở rộng thêm nữa.



Hình 5.4 : Chế độ thanh ghi dịch bit của port nối tiếp

Shift register : thanh ghi dịch bit

8 extra outputs : 8 ngõ ra mở rộng

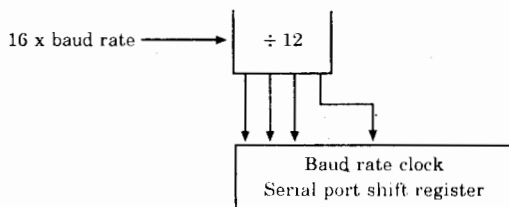
### 5.3.2 UART 8-bit có tốc độ baud thay đổi ( chế độ 1 )

Trong chế độ 1, port nối tiếp của 8051 hoạt động như một bộ thu phát không đồng bộ ( universal asynchronous receiver transmitter ) UART 8-bit có tốc độ baud thay đổi. UART là một bộ thu và phát dữ liệu nối tiếp với mỗi một ký tự dữ liệu được đứng trước bởi một bit start ( logic 0 ) và được đứng sau bởi một bit stop ( logic 1 ). Thỉnh thoảng một bit chẵn lẻ được chèn giữa bit dữ liệu sau cùng và bit stop. Hoạt động chủ yếu của một UART là biến đổi dữ liệu phát từ song song thành nối tiếp và biến đổi dữ liệu thu từ nối tiếp thành song song.

Như vậy ở chế độ 1 ta có 10 bit được thu trên chân Rx.D và 10 bit được phát trên chân Tx.D cho mỗi một ký tự dữ liệu, chúng bao gồm 1 bit start ( luôn luôn là 0 ), 8 bit dữ liệu ( bit LSB trước tiên ) và 1 bit stop ( luôn luôn là 1 ). Khi hoạt động thu, bit stop đưa đến bit RB8 của SCON. Với 8051, tốc độ baud được thiết lập bởi tốc độ tràn ( overflow rate ) của bộ định thời 1 còn ở 8052, tốc độ baud được thiết lập bởi tốc

độ tràn của bộ định thời 1 hoặc bộ định thời 2 hoặc tổ hợp của cả hai ( một cho phát và một cho thu ).

Việc cấp xung *clock* dịch bit và đồng bộ các thanh ghi dịch bit của port nối tiếp ở các chế độ 1, 2 và 3 được thiết lập bởi một bộ đếm 16, ngõ ra của bộ đếm là xung *clock* tốc độ baud ( xem hình 5.5 ). Ngõ vào của bộ đếm vừa nêu được chọn bằng phần mềm và sẽ được trình bày sau.



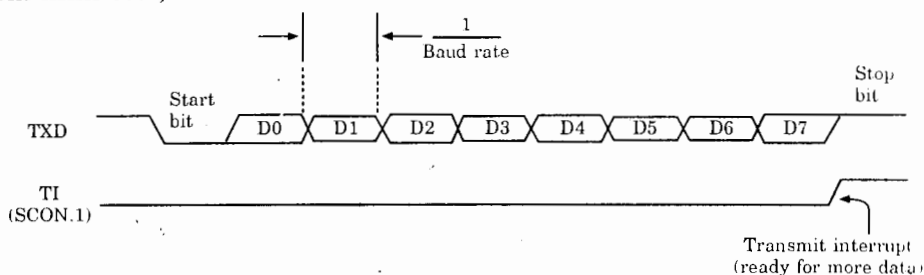
Hình 5.5 : Cấp xung *clock* cho port nối tiếp

16 x baud rate : 16 x tốc độ baud

Baud rate clock : xung *clock* tốc độ baud

Serial port shift register : thanh ghi dịch bit của port nối tiếp

Việc phát được khởi động bằng cách ghi vào SBUF nhưng việc phát không thực sự bắt đầu cho đến lần tràn kế của bộ đếm 16, bộ đếm cung cấp tốc độ baud cho port nối tiếp. Dữ liệu được dịch bit để được xuất ra trên đường TxD sẽ bắt đầu bằng bit start, tiếp theo là 8 bit dữ liệu rồi đến bit stop. Thời gian của mỗi một bit là giá trị nghịch đảo của tốc độ baud, tốc độ baud có được bằng cách lập trình cho bộ định thời. Cờ ngắt phát TI được set bằng 1 ngay khi bit stop xuất hiện trên đường TxD ( xem hình 5.6 ).



Hình 5.6 : Set cờ TI của port nối tiếp

Baud rate : tốc độ baud

Start bit : bit start

Stop bit : bit stop

Transmit interrupt : ngắt phát

Việc nhận được khởi động bởi một chuyển trạng thái từ 1 xuống 0 trên đường RxD ( bắt đầu bit start ). Bộ đếm 16 ngay lập tức được xóa để gán các số đếm cho dòng bit đến chân RxD ( bit kế tiếp đến khi bộ đếm tràn lần nữa và v.v... ). Dòng bit đến được lấy mẫu ở giữa 16 số đếm.

Bộ thu bao gồm việc phát hiện bit start sai bằng cách yêu cầu 8 số đếm ở trạng thái 0 sau khi có sự chuyển trạng thái từ 1 xuống 0 lần đầu tiên. Nếu điều này không xảy ra, bộ thu được giả sử rằng đã được nhận được nhiều thay vì là nhận một bit hợp lệ. Bộ thu sẽ được thiết lập lại, quay về trạng thái nghỉ và chờ sự chuyển trạng thái từ 1 xuống 0 kế. Giả sử một bit start hợp lệ được phát hiện, việc nhận ký tự sẽ tiếp tục. Bit start được bỏ qua và 8 bit dữ liệu được nhận tuần tự vào thanh ghi dịch bit của port nối tiếp. Khi cả 8 bit đã được nhận, các điều sau sẽ xảy ra :

- ✓ 1. Bit thứ 9 ( bit stop ) được đưa đến bit RB8 trong thanh ghi SCON.
- ✓ 2. 8 bit dữ liệu được nạp vào SBUF.
- ✓ 3. Cờ ngắt thu RI được *set*.

Tuy nhiên các điều trên chỉ xảy ra nếu các điều kiện sau tồn tại :

1. RI = 0
2. SM2 = 1 và bit stop nhận được là bit 1, hoặc SM2 = 0.

Yêu cầu RI = 0 đảm bảo rằng phần mềm đã đọc ký tự trước ( và xóa RI ). Điều kiện thứ hai có vẻ phức tạp nhưng chỉ áp dụng trong chế độ truyền thông đa xử lý. Yêu cầu có nghĩa là không *set* RI bằng 1 trong chế độ truyền thông đa xử lý khi bit dữ liệu thứ 9 là 0.

### 5.3.3 UART 9-bit có tốc độ baud cố định ( chế độ 2 )

Khi SM1 = 1 và SM0 = 0, *port* nối tiếp hoạt động ở chế độ 2, chế độ UART 9-bit có tốc độ baud cố định. 11 bit được thu hoặc phát cho việc thu phát một ký tự dữ liệu : bit start, 8 bit dữ liệu, bit dữ liệu thứ 9 lập trình được và bit stop. Khi phát, bit thứ 9 là bit bất kỳ được đặt vào bit TB8 trong thanh ghi SCON ( có thể là bit chẵn lẻ ). Khi thu, bit thứ 9 nhận được sẽ đặt vào bit RB8. Tốc độ baud ở chế độ 2 bằng 1/32 hoặc bằng 1/64 tần số của mạch dao động trên *chip* ( xem mục 5.6 ).

### 5.3.4 UART 9-bit có tốc độ baud thay đổi ( chế độ 3 )

Chế độ 3, UART 9-bit có tốc độ thay đổi, tương tự như chế độ 2 ngoại trừ tốc độ baud được lập trình và được cung cấp bởi bộ định thời. Thật

ra các chế độ 1, 2 và 3 đều tương tự nhau. Chúng khác nhau ở tốc độ baud ( cố định ở chế độ 2 và thay đổi ở các chế độ 1 và 3 ) và ở số bit dữ liệu ( 8 bit ở chế độ 1 và 9 bit ở các chế độ 2 và 3 ).

## 5.4 KHỞI ĐỘNG VÀ TRUY XUẤT CÁC THANH GHI

### 5.4.1 Cho phép thu

Bit cho phép thu REN trong thanh ghi SCON phải được *set* bằng 1 bởi phần mềm để cho phép nhận các ký tự. Điều này thường được thực hiện ở đầu chương trình khi port nối tiếp, các bộ định thời v.v... được khởi động và có thể được thực hiện theo 2 cách. Lệnh :

```
SETB REN
```

set bit REN bằng 1 hoặc lệnh :

```
MOV SCON, #xxx1xxxxB
```

set bit REN bằng 1 và xóa hoặc set các bit khác trong SCON nếu cần.

### 5.4.2 Bit dữ liệu thứ 9

Bit dữ liệu thứ 9 được phát ở các chế độ 2 và 3 phải được nạp cho bit TB8 bằng phần mềm. Bit dữ liệu thứ 9 thu được phải đặt vào bit RB8 của SCON. Phần mềm có thể yêu cầu hoặc không yêu cầu bit dữ liệu thứ 9 tùy vào các đặc tính của thiết bị nối tiếp mà với thiết bị này việc truyền dữ liệu được thiết lập ( bit thứ 9 còn đóng vai trò quan trọng trong truyền thông đa xử lý ).

### 5.4.3 Thêm vào bit chắn lẻ

Bit thứ 9 thường được dùng làm bit chắn lẻ cho một ký tự. Như đã đề cập ở chương 2, bit P trong từ trạng thái chương trình PSW được set hoặc xóa ở mỗi một chu kỳ máy để thiết lập việc kiểm tra chắn cho 8 bit chứa trong thanh chứa A.

Thí dụ nếu việc truyền thông yêu cầu 8 bit dữ liệu cộng với một bit kiểm tra chắn, các lệnh sau được dùng để phát đi 8 bit trong thanh chứa với bit kiểm tra chắn được đưa vào bit thứ 9 :

```
MOV C, P           ; đưa bit kiểm tra chắn vào TB8
MOV TB8, C         ; bit này trở thành bit thứ 9
MOV SBUF, A        ; di chuyển 8 bit dữ liệu từ ACC đến SBUF
```

Nếu kiểm tra lẻ được yêu cầu, các lệnh trên phải được sửa đổi như sau :

```
MOV C, P           ; đưa bit kiểm tra chắn vào TB8 ✓
CPL C              ; biến đổi thành kiểm tra lẻ
```

MOV TB8, C ; bit này trở thành bit thứ 9  
 MOV SBUF, A ; di chuyển 8 bit dữ liệu từ ACC đến SBUF

Dĩ nhiên việc sử dụng bit chuẩn lễ không bị giới hạn ở các chế độ 2 và chế độ 3. Trong chế độ 1, 8 bit dữ liệu được phát đi bao gồm 7 bit dữ liệu cộng với bit chuẩn lễ. Để phát đi một mã ASCII 7-bit cùng với bit kiểm tra chuẩn ( bit thứ 8 ), ta có thể sử dụng các dòng lệnh sau :

CLR ACC.7 ; đảm bảo bit MSB được xóa  
 ; kiểm tra chuẩn  
 MOV C, P ; sao chép bit P vào C  
 MOV ACC.7, C ; đặt bit kiểm tra chuẩn vào bit MSB  
 MOV SBUF, A ; phát ký tự  
 ; 7 bit dữ liệu cộng với bit kiểm tra chuẩn

#### 5.4.4 Các cờ ngắt

Các cờ ngắt thu RI và ngắt phát TI trong thanh ghi SCON đóng một vai trò quan trọng trong việc truyền dữ liệu nối tiếp của 8051. Cả 2 bit đều được *set* bằng 1 bằng phần cứng nhưng phải được xóa bằng phần mềm.

Diễn hình là RI được *set* bằng 1 khi kết thúc việc nhận một ký tự và chỉ ra rằng bộ đệm thu đầy. Điều kiện này được kiểm tra bằng phần mềm hoặc được lập trình để tạo ra một ngắt ( các ngắt được đề cập trong chương 6 ).

Nếu phần mềm muốn nhập một ký tự từ một thiết bị ghép với *port* nối tiếp, phần mềm phải chờ cho đến khi RI được *set* bằng 1, kể đến phần mềm xóa RI và đọc ký tự từ SBUF. Điều này được thể hiện như sau :

WAIT: JNB RI, WAIT ; kiểm tra RI cho đến khi bằng 1  
 CLR RI ; xóa RI  
 MOV A, SBUF ; đọc ký tự

(CONTINUE)

Cờ TI được *set* bằng 1 khi kết thúc việc phát một ký tự và chỉ ra rằng bộ đệm phát rỗng. Nếu phần mềm muốn phát một ký tự đến một thiết bị ghép với *port* nối tiếp, phần mềm trước tiên phải kiểm tra để biết *port* nối tiếp đã sẵn sàng. Nói cách khác, nếu một ký tự trước đó đã được phát, phần mềm phải chờ việc phát kết thúc trước khi gửi tiếp ký tự kế.

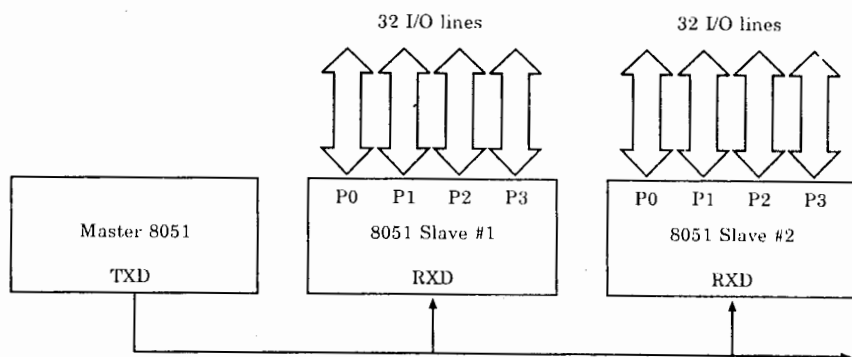
Các lệnh sau đây phát một ký tự chứa trong thanh chứa :

WAIT: JNB TI, WAIT ; kiểm tra TI cho đến khi bằng 1  
 CLR TI ; xóa TI  
 MOV SBUF, A ; phát ký tự

Các chuỗi lệnh thu và phát ở trên là một phần của các chương trình con xuất nhập ký tự. Chúng sẽ được mô tả chi tiết trong thí dụ 5.2 và thí dụ 5.3.

## 5.5 TRUYỀN THÔNG ĐA XỬ LÝ

Các chế độ 2 và 3 là các chế độ dự phòng cho việc truyền thông đa xử lý. Trong các chế độ này, 9 bit dữ liệu được thu và bit thứ 9 được đưa đến **RB8**. Port có thể được lập trình sao cho khi bit stop được nhận, ngắt do port nối tiếp được tích cực chỉ nếu **RB8 = 1**. Đặc trưng này có được bằng cách set bit SM2 trong thanh ghi SCON bằng 1. Một ứng dụng cho điều này là một môi trường mạng sử dụng nhiều 8051 được sắp xếp theo mô hình chủ / tớ ( master / slave ) như trình bày trong hình 5.7.



Hình 5.7 : Truyền thông đa xử lý

Master 8051 : 8051 chủ

8051 slave # 1, # 2 : 8051 tớ 1, 2

32 I/O lines : 32 đường xuất/nhập

Khi bộ xử lý chủ muốn truyền một khối dữ liệu đến một trong nhiều bộ xử lý tớ, trước tiên bộ xử lý chủ phát đi một byte địa chỉ nhận dạng bộ xử lý tớ đích. Một byte địa chỉ khác với một byte dữ liệu ở chỗ bit thứ 9 là 1 trong byte địa chỉ và là 0 trong byte dữ liệu. Một byte địa chỉ ngắt tất cả các bộ xử lý tớ để cho mỗi một bộ xử lý tớ có thể khảo sát byte nhận được để kiểm tra xem có phải là bộ xử lý tớ đang được định địa chỉ không. Bộ xử lý tớ được định địa chỉ sẽ xóa bit SM2 của mình và chuẩn

bị nhận các byte dữ liệu theo sau. Các bộ xử lý tổ không được định địa chỉ có các bit SM2 của chúng được set bằng 1 và thực thi các công việc của riêng chúng, bỏ qua không nhận các byte dữ liệu. Các bộ xử lý này sẽ được ngắt lần nữa khi bộ xử lý chủ phát tiếp byte địa chỉ kế. Các sơ đồ cụ thể có thể được nêu ra sao cho một khi liên kết chủ tớ đã được thiết lập, bộ xử lý tớ cũng có thể phát đến bộ xử lý chủ. Mưu mẹo ở đây là không sử dụng bit dữ liệu thứ 9 sau khi liên kết vừa được thiết lập ( ngược lại các bộ xử lý tớ khác có thể được chọn một cách không cố ý ).

SM2 không ảnh hưởng đến chế độ 0, và trong chế độ 1 bit này có thể được dùng để kiểm tra sự hợp lệ của bit stop. Ở chế độ 1 thu, nếu SM2 = 1, ngắt thu sẽ không được tích cực trừ phi bit stop thu được là hợp lệ.

## 5.6 TỐC ĐỘ BAUD CỦA PORT NỐI TIẾP

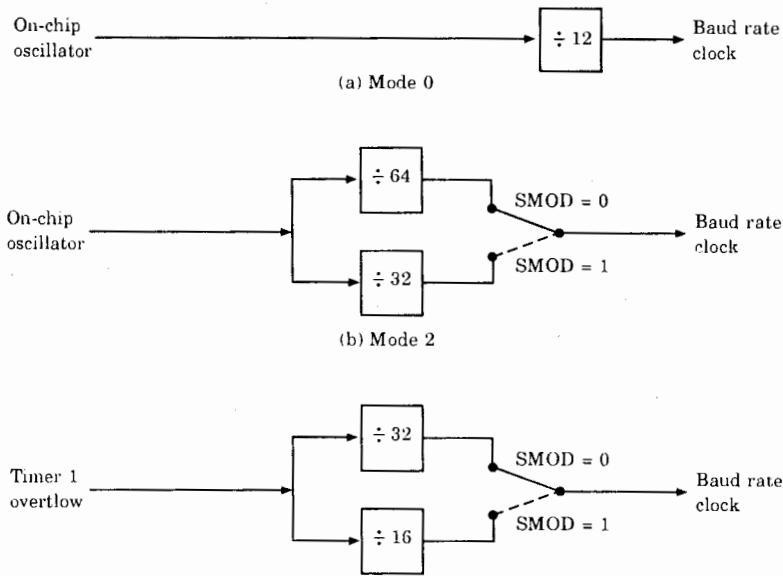
Như đã thấy trong bảng 5.2, tốc độ baud sẽ cố định trong các chế độ 0 và 2. Trong chế độ 0, tốc độ baud luôn luôn bằng tần số của mạch dao động trong chip chia cho 12. Thông thường người ta sử dụng một thạch anh bên ngoài *chip* cho mạch dao động này. Giả sử tần số của mạch dao động là 12 MHz, tốc độ baud của chế độ 0 là 1 MHz ( xem hình 5.8a ).

Sau khi hệ thống được *reset*, tốc độ baud của chế độ 2 bằng tần số của mạch dao động chia cho 64. Tốc độ baud cũng bị ảnh hưởng bởi một bit trong thanh ghi điều khiển nguồn PCON. Bit 7 của PCON là bit SMOD và việc set bit này bằng 1 sẽ làm tăng tốc độ baud của các chế độ 1, 2 và 3 lên gấp đôi. Ở chế độ 2, tốc độ baud có thể được nhân 2 từ giá trị mặc định là 1/64 tần số của mạch dao động ( SMOD = 0 ) trở thành 1/32 tần số của mạch dao động ( SMOD = 1 ) ( xem hình 5.8b ).

Vì thanh ghi PCON không được định địa chỉ từng bit, việc set bit SMOD lên 1 mà không làm thay đổi các bit khác của thanh ghi này được thực hiện bằng những dòng lệnh sau :

```
MOV    A, PCON           ; lấy giá trị hiện hành của PCON
SETB   ACC.7             ; set bit 7 bằng 1 ( SMOD )
MOV    PCON, A           ; ghi giá trị mới vào PCON
```

Các tốc độ baud của 8051 ở chế độ 1 và chế độ 3 được xác định bởi tốc độ tràn của bộ định thời 1. Vì bộ định thời hoạt động ở tần số tương đối cao, ta cần chia tốc độ tràn cho 32 ( hoặc 16 nếu SMOD = 1 ) trước khi trở thành xung *clock* tốc độ baud cung cấp cho *port* nối tiếp. Tốc độ baud của 8052 ở các chế độ 1 và 3 được xác định bởi tốc độ tràn của bộ định thời 1 hoặc bộ định thời 2 hoặc cả hai.



Hình 5.8 : Các nguồn xung *clock* cho port nối tiếp (a) chế độ 0 (b) chế độ 2 (c) chế độ 1 và 3

On chip oscillator : bộ dao động trong *chip*

Baud rate clock : xung *clock* tốc độ baud

Timer 1 overflow : tràn bộ định thời 1

### 5.6.1 Sử dụng bộ định thời 1 làm xung *clock* tốc độ baud

Kỹ thuật thường dùng để tạo xung *clock* tốc độ baud là khởi động thanh ghi TMOD ở chế độ tự nạp lại 8-bit ( chế độ định thời 2 ) và đặt giá trị nạp lại thích hợp vào thanh ghi TH1 để có tốc độ tràn đúng, từ đó tạo ra tốc độ baud.

Thanh ghi TMOD được khởi động như sau :

```
MOV TMOD, #0010xxxxB
```

xxxx dành cho bộ định thời 0.

Đây không phải là khả năng duy nhất. Các tốc độ baud rất chậm có thể nhận được bằng cách sử dụng chế độ 16-bit, chế độ định thời 2 với (TMOD) = 0001xxxxB. Tuy nhiên có một lỗi phần mềm ở đây do các thanh ghi TH1/TL1 phải được khởi động lại sau mỗi lần tràn. Điều này cần được thực hiện trong một trình phục vụ ngắt. Một lựa chọn khác là cung cấp xung *clock* bên ngoài cho bộ định thời 1 bằng cách sử dụng ngõ vào T1 ( P3.5 ). Dù là lựa chọn nào, tốc độ baud cũng bằng tốc độ tràn của bộ định thời 1 chia cho 32 ( hoặc chia cho 16 nếu SMOD = 1 ).

Do vậy, công thức dùng để xác định tốc độ baud ở các chế độ 1 và 3 là :

$$\text{BAUD RATE} = \text{TIMER 1 OVERFLOW RATE} \div 32$$

$$\text{Tốc độ baud} = \text{tốc độ tràn bộ định thời 1 chia cho 32}$$

Thí dụ nếu cần tốc độ baud là 1200, tốc độ tràn của bộ định thời 1 phải là 38.4 KHz ( = 1200 x 32 ).

Nếu tần số của mạch dao động bên trong *chip* là 12 MHz, bộ định thời 1 được cấp xung *clock* là 1 MHz hay 1000 KHz. Do bộ định thời phải tràn ở tốc độ là 38.4 KHz, việc tràn cần xảy ra sau mỗi 26.04 xung *clock* ( 1000 ÷ 38.4 ) và được làm tròn là 26. Vì bộ định thời đếm lên và tràn khi có số đếm từ FFH chuyển thành 00H, 26 số đếm nhỏ hơn 0 là giá trị nạp lại cần có để nạp cho thanh ghi TH1. Giá trị này là -26. Cách dễ dàng nhất để đặt giá trị nạp lại vào TH1 là :

```
MOV TH1, #-26
```

Trình dịch hợp ngữ sẽ thực hiện việc biến đổi cần thiết. Trong trường hợp này -26 được biến đổi thành 0E6H, vậy thì lệnh trên trở thành :

```
MOV TH1, #0E6H
```

Do ta làm tròn số xung đếm nên sẽ có một sai số nhỏ trong kết quả tính tốc độ baud. Trong trường hợp tổng quát, một sai số 5% là sai số chấp nhận được trong truyền dữ liệu không đồng bộ. Các tốc độ baud chính xác có thể nhận được bằng cách sử dụng một thạch anh 11.059 MHz cho mạch dao động trong *chip*. Bảng 5.3 tóm tắt các giá trị nạp lại cần đưa vào thanh ghi TH1 cho các tốc độ baud thường dùng nhất ( sử dụng thạch anh 12 MHz hoặc 11.059 MHz ).

| Tốc độ baud | Tần số thạch anh | SMOD | Giá trị nạp cho TH1 | Tốc độ baud thực tế | Sai số |
|-------------|------------------|------|---------------------|---------------------|--------|
| 9600        | 12.000 MHz       | 1    | -7 (F9H)            | 8923                | 7%     |
| 2400        | 12.000 MHz       | 0    | -13 (F3H)           | 2404                | 0.16%  |
| 1200        | 12.000 MHz       | 0    | -26 (E6H)           | 1202                | 0.16%  |
| 19200       | 11.059 MHz       | 0    | -3 (FDH)            | 19200               | 0      |
| 9600        | 11.059 MHz       | 0    | -3 (FDH)            | 9600                | 0      |
| 2400        | 11.059 MHz       | 0    | -12 (F4H)           | 2400                | 0      |
| 1200        | 11.059 MHz       | 0    | -24 (E8H)           | 1200                | 0      |

Bảng 5.3 : Tóm tắt tốc độ baud

**Thí dụ 5.1 : Khởi động port nối tiếp**

*Viết một chuỗi lệnh để khởi động port nối tiếp sao cho port này hoạt động như một UART 8-bit với tốc độ 2400 baud. Sử dụng bộ định thời 1 để cung cấp xung clock tốc độ baud.*

Với thí dụ này ta phải khởi động 4 thanh ghi : SMOD, TMOD, TCON và TH1. Các giá trị yêu cầu được tóm tắt dưới đây :

|       | SM0 | SM1 | SM2 | REN | TB8 | RB8 | TI  | RI  |
|-------|-----|-----|-----|-----|-----|-----|-----|-----|
| SCON: | 0   | 1   | 0   | 1   | 0   | 0   | 1   | 0   |
|       | GTE | C/T | M1  | M0  | GTE | C/T | M1  | M0  |
| TMOD: | 0   | 0   | 1   | 0   | 0   | 0   | 0   | 0   |
|       | TF1 | TR1 | TF0 | TR0 | IE1 | IT1 | IE0 | IT0 |
| TCON: | 0   | 1   | 0   | 0   | 0   | 0   | 0   | 0   |
| TH1:  | 1   | 1   | 1   | 1   | 0   | 0   | 1   | 1   |

Việc thiết lập SM0/SM1 = 0/1 nhằm đặt port nối tiếp ở chế độ UART 8-bit. REN = 1 cho phép port nối tiếp thu các ký tự. Việc set TI bằng 1 cho phép phát ký tự đầu tiên bằng cách chỉ ra rằng bộ đệm phát rỗng. Với thanh ghi TMOD, việc thiết lập M1/M0 = 1/0 đặt bộ định thời 1 vào chế độ tự nạp lại 8-bit. Việc set TR1 trong TCON bằng 1 sẽ khởi động bộ định thời 1 hoạt động.

Các bit khác được cho bằng 0 do chúng điều khiển các đặc trưng hoặc các chế độ không được sử dụng trong thí dụ này.

Giá trị cần nạp cho TH1 là giá trị cung cấp tốc độ tràn :  $2400 \times 32 = 76.8 \text{ KHz}$ . Giả sử tần số của mạch dao động trong chip là 12 MHz, bộ định thời 1 được cung cấp xung clock có tần số 1 MHz hay 1000 KHz và số xung clock cho mỗi lần tràn là  $1000 \div 76.8 = 13.02$  (làm tròn là 13). Giá trị nạp lại là -13 hoặc 0F3H.

Chuỗi lệnh khởi động port nối tiếp như sau :

```

ORG    8100H

INIT:  MOV    SCON, #52H    ; port nối tiếp, chế độ 1
        MOV    TMOD, #20H   ; bộ định thời 1, chế độ 2
        MOV    TH1, #-13    ; giá trị nạp lại để có 2400 baud
        SETB   TR1          ; bộ định thời 1 hoạt động
        END

```

**Thí dụ 5.2 :** Chương trình con xuất ký tự

Viết một chương trình con gọi là *OUTCHR* để phát mã ASCII 7-bit chứa trong thanh chứa A ra port nối tiếp của 8051 với bit kiểm tra lẻ là bit thứ 8. Việc trở về từ chương trình con không làm thay đổi nội dung thanh chứa (nghĩa là thanh chứa có nội dung giống như nội dung trước khi chương trình con được gọi).

Thí dụ này và thí dụ kế minh họa hai trong nhiều chương trình con thông dụng nhất trên các hệ máy vi tính có thiết bị đầu cuối ghép qua chuẩn RS-232 : xuất ký tự ( *OUTCHR* ) và nhập ký tự ( *INCHAR* ).

```

                                ORG    8100H
OUTCHR:                        MOV    C, P          ; đặt bit chẵn lẻ vào cờ C
                                CPL    C           ; đổi thành lẻ
                                MOV    ACC.7, C     ; đưa vào bit 7 của ACC
AGAIN:                         JNB    TI, AGAIN     ; bộ đếm phát rỗng ?
                                ; sai, kiểm tra lại
                                CLR    TI          ; đúng, xóa cờ TI và
                                MOV    SBUF, A      ; phát 1 ký tự
                                CLR    ACC.7
                                RET
                                END

```

Ba lệnh đầu tiên đặt bit kiểm tra lẻ vào bit 7 của thanh chứa. Do bit P trong từ trạng thái chương trình PSW thiết lập kiểm tra chẵn cho thanh chứa, bit này phải được lấy bù trước khi đặt vào ACC.7. Lệnh JNB tạo ra một vòng lặp chờ để kiểm tra cờ ngắt phát TI cho đến khi cờ này được set bằng 1. Khi TI = 1 ( do việc phát ký tự trước đó vừa kết thúc ), bit này được xóa và sau đó ký tự trong thanh chứa được ghi vào bộ đếm của port nối tiếp SBUF và việc phát ký tự bắt đầu ở lần tràn kế của bộ đếm 16 tạo xung clock cho port nối tiếp. Sau cùng bit ACC.7 được xóa để giá trị trả về giống như khi mã 7-bit được chuyển đến chương trình con.

Chương trình con *OUTCHR* thường được gọi bởi một chương trình gọi để phát một ký tự hoặc một chuỗi ký tự. Thí dụ các lệnh sau phát mã ASCII của ký tự Z đến thiết bị nối tiếp ghép với port nối tiếp của 8051 :

```

MOV    A, # 'Z'
CALL   OUTCHR

```

Ta cũng có thể sử dụng chương trình con OUTCHR như là một khối được thiết kế trong một chương trình con có tên là OUTSTR chẳng hạn, chương trình con này phát một chuỗi ký tự ASCII kết thúc bởi byte NULL ( 00H ) đến một thiết bị nối tiếp ghép với port nối tiếp của 8051.

**Thí dụ 5.3 :** Chương trình con thu một ký tự

*Viết một chương trình con có tên là INCHAR để thu một ký tự từ port nối tiếp của 8051 và trả về mã ASCII 7-bit trong thanh chứa. Sử dụng kiểm tra lẻ trong bit thứ 8 thu được và set cờ nhớ bằng 1 nếu có lỗi chẵn lẻ.*

```

                ORG      8100H
INCHAR:  JNB      RI, $      ; chờ ký tự
          CLR      RI        ; xóa cờ
          MOV      A, SBUF    ; đọc ký tự vào thanh chứa
          MOV      C, P       ; với kiểm tra lẻ trong A
                               ; P cần được set bằng 1
          CPL      C          ; việc lấy bù chỉ ra có lỗi hay
                               ; không ?
          CLR      ACC.7      ; xóa
          RET
          END

```

Chương trình con này bắt đầu bằng việc chờ cờ ngắt thu RI được set bằng 1 để chỉ ra rằng ký tự đã sẵn sàng trong bộ đệm thu SBUF ( để được đọc ). Khi RI = 1, lệnh JNB chuyển điều khiển đến lệnh tiếp theo lệnh này. Cờ RI được xóa và mã trong SBUF được đọc vào thanh chứa. Bit P trong PSW thiết lập kiểm tra chẵn cho thanh chứa, do vậy bit này cần được set bằng 1 nếu bản thân thanh chứa chứa bit kiểm tra lẻ ở bit thứ 7 của thanh ghi này. Việc di chuyển bit P vào cờ nhớ làm cho CY = 0 nếu không có lỗi. Mặt khác, nếu thanh chứa chứa một lỗi chẵn lẻ, cờ CY sẽ bằng 1. Cuối cùng bit ACC.7 được xóa để đảm bảo rằng chỉ có mã 7-bit được trả về cho chương trình gọi.

Các nội dung trong các chương 4 và 5 trình bày các chi tiết chính cần đến khi ta lập trình định thời và thu phát nối tiếp. Các chương trình thí dụ trong 2 chương này hoàn toàn không sử dụng ngắt mà sử dụng các vòng lặp. Việc sử dụng các vòng lặp trong lập trình xuất nhập

gặp phải khuyết điểm là bộ vi xử lý không làm gì cả để chờ các điều kiện xuất nhập sẵn sàng.

Trên thực tế, nhiều ứng dụng liên quan đến các bộ định thời và *port* nối tiếp của 8051 yêu cầu phải được đồng bộ cũng như để bộ vi xử lý không phải chờ, người ta sử dụng ngắt. Đây là chủ đề của chương tiếp theo, hoạt động ngắt.

## 6

## HOẠT ĐỘNG NGẮT

### 6.1 MỞ ĐẦU

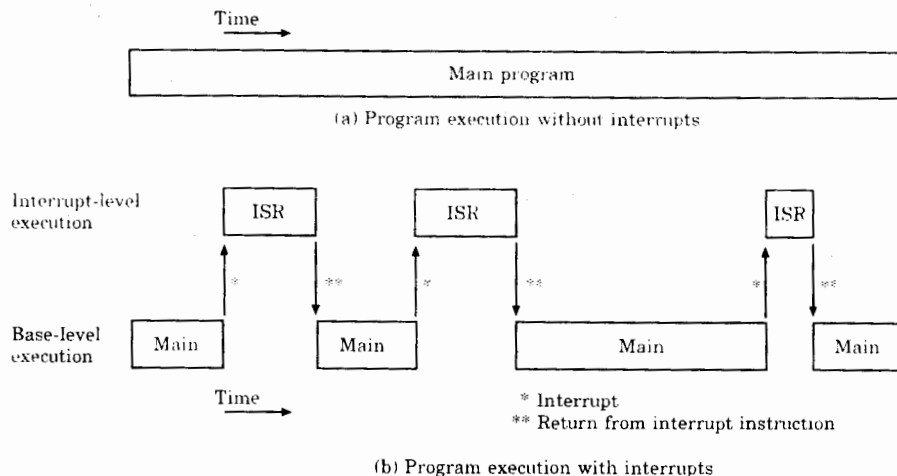
Ngắt ( interrupt ) là sự xảy ra của một điều kiện – một sự kiện – làm cho chương trình hiện hành bị tạm ngưng trong khi điều kiện được phục vụ bởi một chương trình khác. Các ngắt đóng vai trò quan trọng trong việc thiết kế và hiện thực các ứng dụng của bộ vi điều khiển. Các ngắt cho phép hệ thống đáp ứng một sự kiện theo cách không đồng bộ và xử lý sự kiện trong khi một chương trình khác đang thực thi. Một hệ thống được điều khiển bởi ngắt cho ta ảo tưởng đang làm nhiều công việc đồng thời.

CPU dĩ nhiên không thể thực thi nhiều hơn một lệnh ở một thời điểm nhưng CPU có thể ngưng tạm thời việc thực thi một chương trình để thực thi một chương trình khác rồi sau đó quay trở về thực thi tiếp chương trình đang bị tạm ngưng, điều này giống như CPU rời khỏi chương trình gọi để thực thi chương trình con bị gọi để rồi sau đó quay trở về chương trình gọi. Sự khác nhau của hai vấn đề vừa nêu là trong một hệ thống được điều khiển bởi ngắt, việc ngắt nhằm đáp ứng một sự kiện mà sự kiện này xuất hiện không đồng bộ với chương trình chính đang được thực thi và chương trình chính ( hay nói cách khác là CPU ) không biết trước là sẽ bị ngắt khi nào.

Chương trình xử lý một ngắt được gọi là trình phục vụ ngắt ISR ( interrupt service routine ) hay quản lý ngắt ( interrupt handler ). ISR được thực thi nhằm đáp ứng một ngắt và trong trường hợp tổng quát thực hiện việc xuất nhập đối với một thiết bị. Khi một ngắt xuất hiện, việc thực thi chương trình chính tạm thời bị dừng và CPU thực hiện việc rẽ nhánh đến trình phục vụ ngắt ISR. CPU thực thi ISR để thực hiện một công việc và kết thúc việc thực thi này khi gặp lệnh “ quay về từ một trình phục vụ ngắt ”; chương trình chính được tiếp tục tại nơi bị tạm dừng. Ta có thể nói chương trình chính được thực thi ở mức nền ( base level ) còn ISR được thực thi ở mức ngắt ( interrupt level ).

Cách nhìn ngắn gọn này được mô tả ở hình 6.1, hình này trình bày :

- a- Việc thực thi một chương trình không có ngắt.
- b- Việc thực thi ở mức nền có ngắt và các ISR được thực thi ở mức ngắt.



Hình 6.1 : Thực thi chương trình có và không có ngắt (a) không có ngắt (b) có ngắt

Time : thời gian

Main program : chương trình chính

Program execution without interrupts : thực thi chương trình không có ngắt

Program execution with interrupts : thực thi chương trình có ngắt

Interrupt level execution : thực thi ở mức ngắt

Base level execution : thực thi ở mức nền

Interrupt : ngắt

Return from interrupt instruction : trở về từ lệnh ngắt

Một thí dụ về ngắt điển hình là nhập bằng tay sử dụng bàn phím. Ta hãy khảo sát một ứng dụng của lò vi ba. Chương trình chính có thể điều khiển thành phần công suất của lò để thực hiện việc nấu nướng ; tuy nhiên trong khi đang nấu, hệ thống phải đáp ứng việc nhập bằng tay trên cửa lò, chẳng hạn như một yêu cầu rút ngắn bớt hay kéo dài thêm thời gian nấu. Khi người sử dụng bấm phím nhấn, một ngắt được tạo ra ( chẳng hạn một tín hiệu từ mức cao chuyển xuống mức thấp ) và chương trình chính bị ngắt. ISR được thực thi để đọc mã phím và thay đổi các điều kiện nấu tương ứng, sau đó kết thúc bằng cách chuyển điều khiển trở về chương trình chính.