Chương 1: GIỚI THIỆU AT89C51

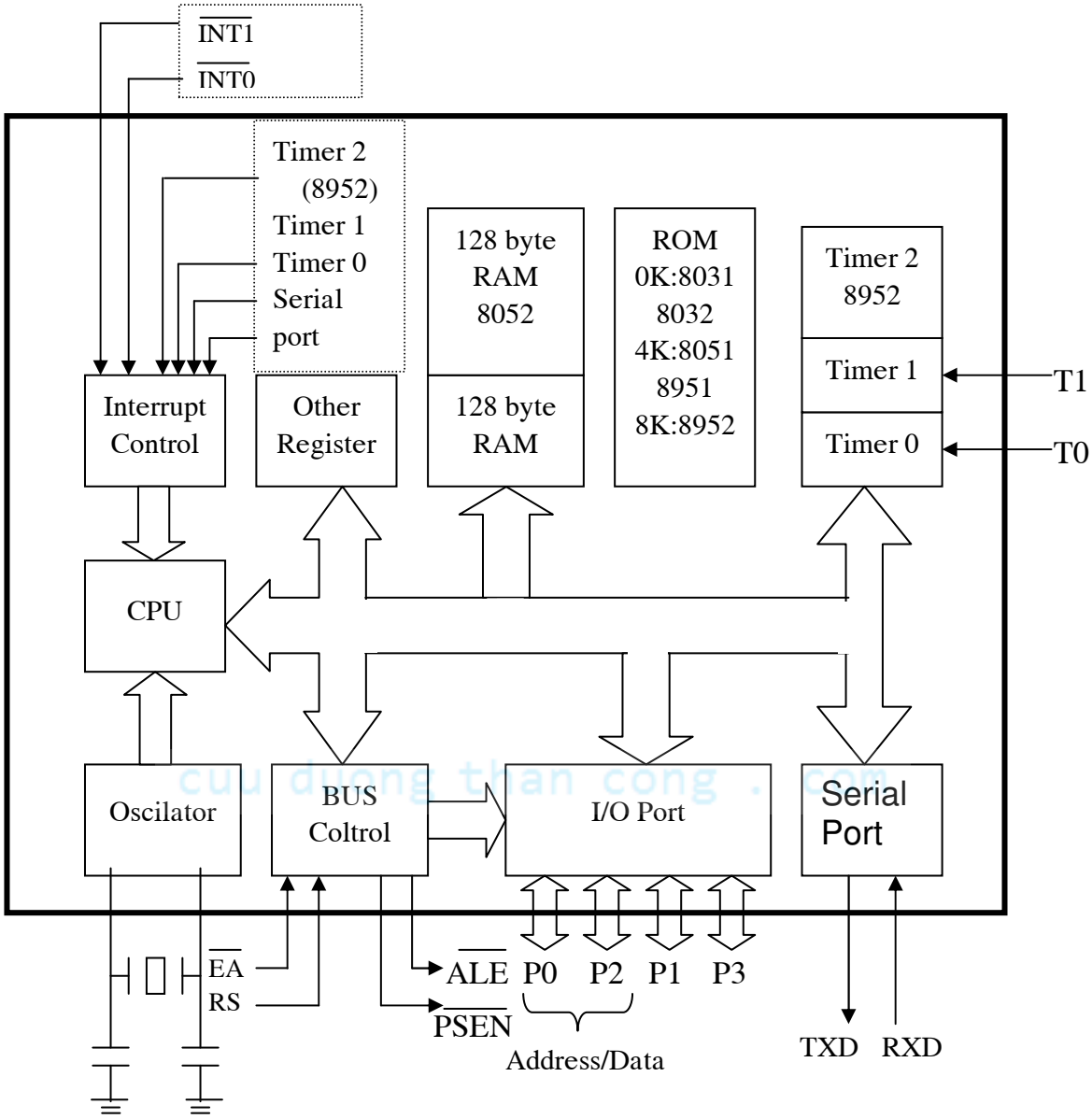
I - TÓM TẮT PHẦN CỨNG:

1) Giới thiệu họ MCS51:

MSC-51 là một họ IC vi điều khiển (microcontroller), được phát triển chế tạo và bán ở thị trường bởi hãng INTEL của Mỹ. Các nhà chế tạo IC khác như SIEMENS của Đức, ADVANCED MICRO DEVICES, FUJITSU của Nhật và PHILIPS của Hà Lan là các nhà cung cấp thiết bị trong họ MSC-51 được cấp giấy bản quyền thứ hai, hãng ALMEL cũng là một hãng được cấp bảng quyền và sản phẩm 8951, 8952 là hai IC tiêu biểu trong thiết kế của họ.

Chúng có các đặt điểm chung như sau:

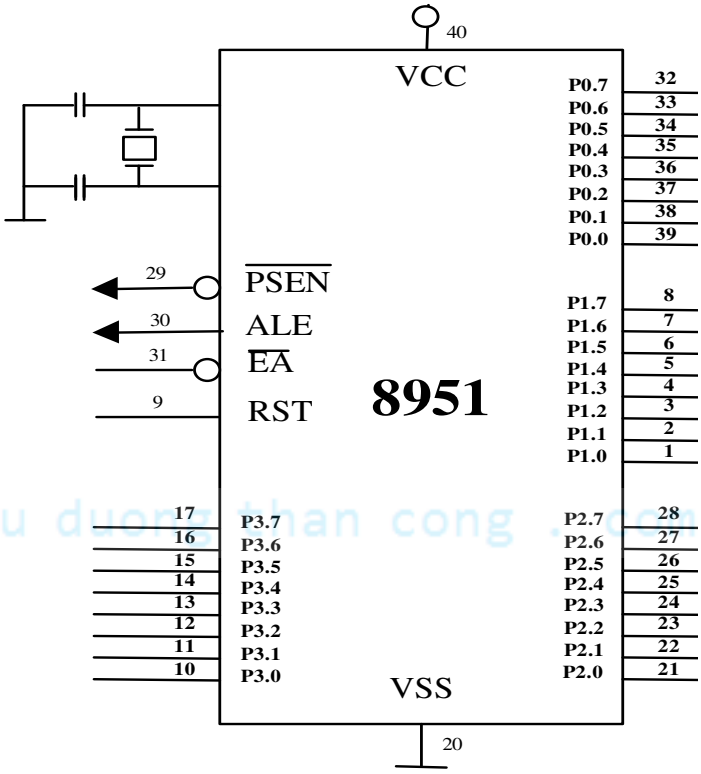
- 4Kbyte ROM (được lập trình bởi nhà sản xuất, chỉ có ở 8051).
- 4Kbyte EPROM (cho ta có thể lập trình được nhiều lần "khoảng 1000 lần", chỉ có ở 8951).
- 128 byte RAM .
- 4 port I/O 8 bit.
- 2 bộ định thời 16 bit.
- 64 K không gian bộ nhớ chương trình mở rộng.
- 64 K không gian bộ nhớ dữ liệu mở rộng.
- Một bộ xử lý luận lý (hoạt động trên bít đơn).
- 210 bit được địa chỉ hoá.
- Bộ nhân / chia 4 µs.



Sơ dồ khối họ MCS51.

2) Sơ lược về các chân của 8951 :

8951 có tất cả 40 chân có chức năng như các đường xuất nhập. Trong đó có 24 chân có công dụng kép, mỗi đường có thể hoạt động như đường xuất nhập hoặc như đường điều khiển hoặc là thành phần của bus dữ liệu và bus địa chỉ.



Sơ đồ chân 8951

a) **PORT 0 :**

Port 0 là một port hai chức năng trên các chân 32 ÷ 39. Trong thiết kế cở nhỏ (không dùng bộ nhớ mở rộng) nó có chức năng như đường I/O. Đối với thiết bị lớn với bộ nhớ mở rộng, nó được hợp kênh giữa bus địa chỉ và bus dữ liệu.

b) PORT 1 :

Port 1 là port I/O trên các chân 1 ÷ 8. Các chân được ký hiệu P1.0, P1.2, P1.3, …,P1.7 có thể dùng cho giao tiếp với các thiết bị ngoài nếu cần. Port 1 không có chức năng khác, vì vậy chúng chỉ dùng cho giao tiếp với thiết bị ngoài.

c) **PORT 2 :**

Port 2 là một port có công dụng kép trên các chân 21 ÷ 28 được dùng như các đường xuất nhập hoặc là byte cao của bus địa chỉ đối với các thiết kế dùng bộ nhớ mở rộng.

d) PORT 3:

Port 3 là một port có công dụng kép trên các chân 10 ÷ 17. Các chân của port này có nhiều cức năng. Các công dụng chuyển đổi có liên hệ với các đặc tính đặc biệt của 8951 như ở bảng sau:

Bit	Tên	Cức năng chuyển đổi
P3.0	RXD	Dữ liệu nhận cho port nối tiếp
P3.1	TXD	Dữ liệu phát cho port nối tiếp
P3.2	$\overline{\text{INT0}}$	Ngắt 0 bên ngoài
P3.3	$\overline{\text{INT1}}$	Ngắt 1 từ ngoài
P3.4	T0	Ngõ vào Timer / Counter 0
P3.5	T1	Ngõ vào Timer / Counter 1
P3.6	$\overline{\text{WR}}$	Xung ghi bộ nhớ dữ liệu ngoài
P3.7	$\overline{\text{RD}}$	Xung đọc bộ nhớ dữ liệu ngoài

Ta cần lưu ý rằng khi dùng những pin này vào những mục đích cá biệt thì cả port 3 không còn khả năng dùng làm I/O port nữa.

e) XTAL 1 và XTAL 2:

Trên chân 18 và 19 của vi mạch, được nối với bộ dao động thạch anh 12 MHz để tạo dao động trên Chip. Hai tụ 30pF được thêm vào để ổn định dao động.

f) PSEN (Program Store Enable):

Có 4 tín hiệu điều khiển.

PSEN là tín hiệu ra trên chân 29. Nó là tín hiệu điều khiển để cho phép bộ nhớ chương trình mở rộng và thường được nối đến chân OE (Output Enable) của một EPROM để cho phép đọc các byte mã lệnh.

PSEN sẽ ở mức thấp trong thời gian lấy lệnh. Các mã nhị phân của chương trình được đọc từ EPROM qua bus dữ liệu và được chốt vào thanh ghi lệnh để giải mã lệnh. Khi thi hành chươnh trình trong ROM nội PSEN ở mức thụ động (mức cao).

g) **ALE (Address Latch Enable):**

Tín hiệura ALE trên chân 30 tương hợp với các thiết bị làm việc với các vi xử lý 8085, 8088, 8086, 8051, 8951 dùng ALE một cách tương tự cho việc giải kênh các bus đia chỉ và dữ liệu.Khi Port 0 được dùng trong chế độ chuyển dổi của nó: vừa là bus dữ liệu vừa là bus thấp của bus đia chỉ, ALE là tín hiệu để chốt địa chỉ vào một thanh ghi ngoài trong suốt nửa chu kỳ nhớ đầu, trong nửa chu kỳ sau nó ở mức thấp cho phép xuất hoặc nhập dữ liệu khi dữ liệu đã di chuyển bus.

Các xung tín hiệu ALE có tốc độ bằng 1/6 lần tần số dao động trên chip và có thể được dùng làm nguồn xung nhịp cho các phần khác của hệ thống. Nếu xung nhịp trên 8951 là 12MHz thì ALE có tần số 2MHz. Chỉ ngoại trừ khi thi hành lệnh MOVX, một xung ALE bị mất. Xung này cũng được làm ngõ vào cho xung lập trình cho EPROM trong 8951.

h) EA (Eternal Acess):

Tín hiệu vào EA trên chân 31 thường được mắc lên mức cao (+5V) hoặc mức thấp (GND) . Nếu ở mức cao 8951 thi hành chương trình từ ROM nội trong khoảng địa chỉ thấp (4K). Nếu ở mức thấp chương trình chỉ được thi hành từ bộ nhớ mở rộng. Người ta còn dùng EA làm chân cấp điện áp 21V khi lập trình 8951.

i) RST (Reset) :

Ngõ vào RST trên chân 9 là ngõ Reset của 8951. Khi tín hiệu này được đưa lên mức cao (trong ít nhất 2 chu kỳ máy) , các thanh ghi bên trong 8951 được tải những giá trị thích hợp để khởi động hệ thống.

j) Các chân nguồn:

8951 vận hành với nguồn đơn +5V. V_{CC} được nối vào chân 40 và V_{SS} (GND) được nối vào chân 20.

3) Cấu trúc bộ nhớ của 8951 :

8951 có cấu trúc nhớ theo kiểu HARWARD : có vùng nhớ riêng cho chương trình và dữ liệu. Bộ nhớ ngoài có thể mở rộng đến 64KB code memory và 64KB data memory. RAM trên chip gồm 128 Byte ứng dụng cho các bộ lưu trữ mục đích chung, bộ lưu trữ có thể định vị bit, các dãy thanh ghi và thanh ghi chức năng đặc biệt**.**

Hai đặc tính cần lưu ý là:

+ Các thanh ghi và các port xuất nhập đã được xếp trong bộ nhớ và có thể được truy xuất trực tiếp giống như các địa chỉ bộ nhớ khác.

+ Ngăn xếp trong RAM nội nhỏ hơn so với RAM ngoài như trong các bộ vi xử lý khác.

Bảng tóm tắt vùng nhớ 8951 :

30	General Purpose RAM							
2F	7F	7E	7D	7C	7B	7A	79	78
2E	77	76	75	74	73	72	71	70
2D	6F	6E	6D	6C	6B	6A	69	68
2C	67	66	65	64	63	62	61	60
2B	5F	5E	5D	5C	5B	5A	59	58
2A	57	56	55	54	53	52	51	50
29	4F	4E	4D	4C	4B	4A	49	48
28	47	46	45	44	43	42	41	40
27	3F	3E	3D	3C	3B	3A	39	38
26	37	36	35	34	33	32	31	30
25	2F	2E	2D	2C	2B	2A	29	28
24	27	26	25	24	23	22	21	20
23	1F	1E	1D	1C	1B	1A	19	18
21	17	16	15	14	13	12	11	10
20	0F	0E	0D	0C	0B	0A	09	08
1F	07	06	05	04	03	02	01	00
1817	BANK 3							
100F	BANK 2							
0807	BANK 1							
00	Default Register Bank for R0 ÷ R7							

Bảng tóm tắt bản đồ vùng nhớ
RAM trên chip

FF									
F0	F7	F6	F5	F4	F3	F2	F1	F0	B
E0	E7	E6	E5	E4	E3	E2	E1	E0	ACC
D0	D7	D6	D5	D4	D3	D2	-	D0	PSW
B8	-	-	-	BC	BB	BA	B9	B8	IP
B0	B7	B6	B5	B4	B3	B2	B1	B0	P3
A8	AF	-	-	AC	AB	AA	A9	A8	IE
A0	A7	A6	A5	A4	A3	A2	A1	A0	P2
99	Not bit Addressable								SBUF
98	9F	9E	9D	9C	9B	9A	99	98	SCON
90	97	96	95	94	93	92	91	90	P1
8D	Not bit Addressable								TH1
8C	Not bit Addressable								TH0
8B	Not bit Addressable								TL1
8A	Not bit Addressable								TL0
89	Not bit Addressable								TMOD
88	8F	8E	8D	8C	8B	8A	89	88	TCON
87	Not bit Addressable								PCON
83	Not bit Addressable								DPH
82	Not bit Addressable								DPL
81	Not bit Addressable								SP
80	87	86	85	84	83	82	81	80	P0

CÁC THANH GHI CHỨC NĂNG ĐẶC BIỆT

Vùng nhớ có hai đặc điểm đáng chú ý là :

Những thanh ghi làm port I/O được bộ nhớ sắp đặt và có thể được truy xuất như bất cứ vị trí nhớ nào khác.Vùng Stack ở bên trong RAM nội bộ và giống như các con vi xử lý.

a. RAM đa dụng :

Trong bản đồ bộ nhớ trên, 80 byte từ địa chỉ 30H ÷ 7FH là RAM mục đích chung. Kể cả 32 byte phần dưới từ 00H ÷ 2FH cũng có thể sử dụng như 80 byte trên, tuy nhiên 32 byte này còn có mục đích khác sẽ đề cập sau.Bất cứ vị trí nào

trong RAM cũng đều có thể được truy xuất tùy ý giống như việc sử dụng các mode để định địa chỉ trực tiếp hay gián tiếp. Ví dụ, để đọc nội dung ở địa chỉ 5FH của RAM nội vào thanh ghi tích lũy, lệnh sau sẽ được dùng :

```
MOV  A,5FH
```

Lệnh này di chuyển 1 byte dữ liệu dùng cách đánh địa chỉ trực tiếp để xác định "địa chỉ nguồn" (5FH). Đích nhận dữ liệu được ngầm xác định trong mã lệnh là thanh ghi tích lũy A.

RAM bên trong cũng có thể được truy xuất bằng cách đánh địa chỉ gián tiếp qua Ri. Ví dụ, hai lệnh sau thi hành cùng nhiệm vụ như lệnh đơn ở trên :

```
MOV  R0,#5FH
MOV  A,R0
```

Lệnh đầu dùng địa chỉ tức thời để di chuyển giá trị 5FH vào thanh ghi R0, và lệnh thứ hai dùng địa chỉ trực tiếp để di chuyển dữ liệu "được trỏ bởi R0" vào thanh ghi tích lũy.

b. RAM địa chỉ hóa từng bit :

8951 chứa 210 bit được địa chỉ hóa, trong đó 128 bit là các địa chỉ byte từ 20H ÷ 2FH, và phần còn lại là trong các thanh ghi chức năng đặc biệt.

Ý tưởng truy xuất từng bit riêng rẽ bằng phần mềm là một đặc tính thuận lợi của vi điều khiển nói chung. Các bit có thể được đặt, xóa, AND, OR … với một lệnh đơn. Đa số các vi xử lý đòi hỏi một chuỗi lệnh đọc-sửa-ghi để đạt được hiệu quả tương tự. Hơn nữa, các port I/O cũng được địa chỉ hóa từng bit làm đơn giản phần mềm xuất nhập từng bit.

Có 128 bit được địa chỉ hóa đa dụng ở các byte từ 20H ÷ 2FH. Các địa chỉ này được truy xuất như các byte hoặc các bit tùy thuộc vào lệnh được dùng. Ví dụ, để đặt lại bit 67H, ta dùng lệnh sau :

```
SETB 67H
```


Chú ý rằng "địa chỉ bit 67H" là bit có trọng số lớn nhất (MSB) ở "địa chỉ byte 2CH". Lệnh trên sẽ không tác động đến các bit khác ở cùng địa chỉ byte này. Các vi xử lý phải thi hành nhiệm vụ tương tự như sau :

```
MOV  A,2CH          ;đọc cả byte
ORL  A,#10000000B   ;set MSB
MOV  2CH,A          ;ghi lại byte
```

c. Các bank thanh ghi :

32 byte thấp nhất của bộ nhớ nội dành cho các bank thanh ghi. Bộ lệnh của 8951 hỗ trợ 8 thanh ghi (R0 ÷ R7) và theo mặc định (sau khi RESET hệ thống) các thanh ghi này ở địa chỉ 00H ÷ 07H. Lệnh sau đây sẽ đọc nội dung ở địa chỉ 05H vào thanh ghi tích lũy :

```
MOV     A,R5
```

Đây là lệnh 1 byte dùng địa chỉ thanh ghi. Tất nhiên, thao tác tương tự có thể được thi hành bằng lệnh 2 byte dùng địa chỉ trực tiếp nằm trong byte thứ hai :

```
MOV     A,05H
```

Các lệnh dùng thanh ghi từ R0 ÷ R7 thì sẽ ngắn hơn và nhanh hơn các lệnh tương ứng nhưng dùng địa chỉ trực tiếp. Các thanh ghi dữ liệu được dùng thường xuyên nên dùng một trong các thanh ghi này .

Bank thanh ghi tích cực có thể chuyển đổi bằng cách thay đổi các bit chọn bank thanh ghi trong từ trạng thái chương trình (Program Status Word). Giả sử rằng bank thanh ghi 3 được tích cực, lệnh sau sẽ ghi nội dung của thanh ghi A vào địa chỉ 18H :


```
MOV     R0,A
```

Ý tưởng dùng các "bank thanh ghi " cho phép "chuyển hướng" chương trình nhanh và hiệu quả (từng phần riêng rẽ của phần mềm sẽ có một bộ thanh ghi riêng không phụ thộc vào các phần khác).

4) Các thanh ghi chức năng đặc biệt :

Các thanh ghi nội của 8951 được truy xuất ngầm định bởi bộ lệnh. Ví dụ lệnh "INC A" sẽ tăng nội dung của thanh ghi tích lũy A lên 1. Tác động này được ngầm định trong mã lệnh.

Các thanh ghi trong 8951 được định dạng như một phần của RAM trên chip. Vì vậy, mỗi thanh ghi sẽ có một địa chỉ (ngoại trừ thanh ghi đếm chương trình và thanh ghi lệnh vì các thanh ghi này hiếm khi bị tác động trực tiếp). Đó là lý do tại sao trên 8951 có nhiều thanh ghi như vậy. Cũng như R0 ÷ R7, có 21 thanh ghi chức năng đặc biệt SFR (Special Function Register) ở vùng trên của địa chỉ RAM trên chip, từ địa chỉ 80H ÷ FFH. Chú ý rằng hầu hết 128 địa chỉ từ 80H đến FFH không được định nghĩa. Chỉ có 21 địa chỉ SFR là được định nghĩa.

Ngoại trừ thanh ghi tích lũy A có thể truy xuất ngầm định như đã nói, đa số các SFR được truy xuất dùng địa chỉ trực tiếp. Chú ý rằng một vài SFR có thể

được địa chỉ hóa bit hoặc byte. Người thiết kế phải cẩn trọng khi truy xuất bit và byte. Ví dụ lệnh sau:

```
SETB    0E0H
```

Sẽ set bit 0 trong thanh ghi tích lũy, các bit còn lại không đổi. Ta thấy rằng 0E0H là địa chỉ byte cũng đồng thời là địa chỉ bit có trọng số nhỏ nhất trong thanh ghi tích lũy. Vì lệnh set bit chỉ tác động lên bit nên chỉ có địa chỉ bit là có hiệu quả.

a) Từ trạng thái chương trình (Program Status Word) :

Nằm ở địa chỉ D0H chứa các bit trạng thái như bảng tóm tắt sau :

Bit	Ký hiệu	Địa chỉ	Ý nghĩa
PSW.7	CY	D7H	Cờ nhớ
PSW.6	AC	D6H	Cờ nhớ phụ
PSW.5	F0	D5H	Cờ 0
PSW.4	RS1	D4H	Bit 1 chọn Bank thanh ghi
PSW.3	RS0	D3H	Bit 1 chọn Bank thanh ghi 00 = bank 0 : địa chỉ 00H÷07H 01 = bank 1 : địa chỉ 08H÷0FH 10 = bank 2 : địa chỉ 10H÷17H 11 = bank 3 : địa chỉ 18H÷1FH
PSW.2	OV	D2H	Cờ tràn
PSW.1	-	D1H	Dự trữ
PSW.0	P	D0H	Cờ parity chẵn

*** Cờ nhớ :**

Cờ nhớ CY có công dụng kép. Thông thường nó dùng cho các lệnh toán học : nó sẽ được set nếu có một số nhớ sinh ra bởi phép cộng hoặc một số mượn bởi phép trừ. Ví dụ, nếu thanh ghi tích lũy chứa FFH, thì lệnh sau :


```
ADD  A,#1
```

Sẽ trả về thanh ghi tích lũy kết quả 00H và set cờ nhớ trong PSW.

Cờ nhớ cũng có thể xem như một thanh ghi 1 bit cho các lệnh luận lý thi hành trên bit. Ví dụ, lệnh sau sẽ AND bit 25H với cờ nhớ và đặt kết quả trở vào trong cờ nhớ

```
ALN     C,25H
```

*** Côø nhôù phuï :**

Khi cộng các số BCD, cờ nhớ phụ (AC) được set nếu kết quả của 4 bit thấp trong khoảng từ 0AH đến 0FH. Nếu các giá trị cộng được là số BCD, thì sau lệnh cần có DA A (hiệu chỉnh thập phân thanh ghi tích lũy) để mang kết quả lớn hơn 9 về tầm từ 0÷9.

*** Cờ 0 :**

Cờ 0 (F0) là một bit cờ đa dụng dành cho các ứng dụng của người dùng.

*** Các bit chọn bank thanh ghi :**

Các bit chon bank thanh ghi (RS) và RS1) được xóa sau khi reset hệ thống (chọn mặc định thanh ghi R0÷R7 của bank 0), có thể thay đổi bằng phần mềm nếu cần. Ví dụ, ba lệnh sau cho phép chọn bank thanh ghi 3 và chuyển nội dung của thanh ghi R7 vào thanh ghi tích lũy :

```
SETB RS1
SETB RS0
MOV  A,R7
```

Khi chương trình được hợp dịch, các bit địa chỉ đúng sẽ được thay thế cho các ký hiệu RS0 và RS1.

*** Cờ tràn :**

Cờ tràn (OV) được set sau một lệnh cộng hoặc trừ nếu có một phép toán bị tràn. Khi các số có dấu cộng hoặc trừ với nhau, phần mềm có thể kiểm tra bit này để xác định xem kết quả có nằm trong tầm xác định hay không. Khi các số không dấu được cộng, bit OV có thể được bỏ qua. Các kết quả lớn hơn +127 hoặc nhỏ hơn –128 sẽ set bit OV. Ví dụ, phép cộng sau bị tràn và bit OV được set:

```
MOV     A,#FFH
ADD     A,#01H
```


Kết quả trong thanh ghi A là 00H không phải là kết quả đúng, vì vậy cờ OV được set.

b) Thanh ghi B :

Thanh ghi B ở địa chỉ F0H được dùng cùng với thanh ghi tích lũy A cho các phép toán nhân và chia. Lệnh MUL AB sẽ nhân các giá trị không dấu 8 bit trong A và B rồi trả về kết quả 16 bit trong A (byte thấp) và B (byte cao). Lệnh DIV AB sẽ chia A cho B rồi trả về kết quả nguyên trong A và phần dư trong B.

Thanh ghi B cũng có thể xem như thanh ghi đệm đa dụng. Nó được địa chỉ hóa từng bit bằng các địa chỉ bit F0H đến F7H.

c) Con trỏ ngăn xếp (Stack Pointer) :

Stack Pointer là một thanh ghi 8 bit ở địa chỉ 81H. Nó chứa địa chỉ của dữ liệu đang hiện hành trên đỉnh Stack. Các hoạt động của Stack bao gồm việc đẩy dữ liệu vào stack (PUSH) và việc lấy dữ liệu ra khỏi Stack (POP).

Việc PUSH vào Stack sẽ tăng SP lên 1 trước khi dữ liệu vào.

Việc POP từ Stack sẽ lấy dữ liệu ra rồi giảm SP đi 1.

Vùng Stack của 8951 được cất trong RAM nội và được giới hạn đến những địa chỉ truy xuất bởi sự định vị gián tiếp. Ta có thể khởi gán Stack hoặc không khiở gán Stack. Nếu ta không khởi gán Stack thì mặc định vùng Stack sẽ là 07H và dữ liệu đầu tiên được đưa vào vùng nhớ 08H. Các thanh ghi Bank 1 (có thể là bank 2 hoặc 3) sẽ không dùng được nữa vì vùng RAM nội này thành vùng Stack. Ta chỉ có thể dùng các thanh ghi của Bank 0. Nếu ta khởi gán bank ghi tại vùng khác(ví dụ 60H), thì các thanh ghi của bank 0, 1, 2 và 3 vẫn còn dùng được. Để khởi gán SP với sự bắt đầu của vùng Stack tại địa chỉ 60H, ta làm như sau :

```
MOV  SP,#5F.
```


Ta đưa vào SP số 5F vì SP tăng lên 1 thành 60H trước khi dữ liệu đầu tiên vào Stack được truy xuất trực tiếp bằng lệnh PUSH, POP để cất dữ liệu tạm thời và lấy lại data, hoặc trực tiếp bằng lệnh gọi ACALL, LCALL và lệnh quay lại RET, RETI để cất giữ và lấy ra bộ đếm chương trình.

d) Con trỏ dữ liệu (Data Pointer) :

Data Pointer được dùng để truy xuất bộ nhớ mã ngoài hoặc bộ nhớ dữ liệu ngoài, nó là một thanh ghi 16 bit mà byte thấp là DPL ở địa chỉ 82H còn byte cao là DPH ở địa chỉ 83H. Để đưa nội dung 55H vào RAM ngoại có địa chỉ 1000H ta dùng 3 lệnh như sau:

```
MOV A,#55H
MOV DPTR,#1000H
MOV  @DPTR,A
```


Lệnh thứ nhất dùng sự định vị trực tiếp để đưa hằng số dữ liệu 55H vào A. Lệnh thứ hai cũng tương tự như lệnh thứ nhất đưa hằng số dữ liệu 1000H vào DPTR. Lệnh cuối cùng dùng sự định vị gián tiếp để dịch chuyển giá trị 55H trong A vào vùng nhớ RAM ngoại 1000H nằm trong DPTR.

c) Các Thanh ghi PORT xuất nhập :

Các port 0, 1, 2 và 3 có các địa chỉ tương ứng 80H, 90H, A0H, B0H. Các port 0,2,3 không còn tác dụng I/O nữa nếu bộ nhớ ngoài được dùng vào một số chức năng đặc biệt. Do đó chỉ còn port 1 có tác dụng xuất nhập I/O.

Tất cả các port đều được địa chỉ hóa bit, do đó có thể giao tiếp với bên ngoài mạnh mẽ. Giả sử như có một động cơ được nối đến bit 7 của port 1 thì ta có thể cho chạy hoặc tắt nó chỉ bằng một lệnh duy nhất : SETB P1.7 hoặc CLR P1.7.

f) Các thanh ghi Timer (Timer Register) :

8951 có 2 bộ : một bộ Timer 16 bit và một bộ Counter 16 bit, hai bộ này dùng để định giờ lúc nghỉ của chương trình hoặc đếm các sự kiện quan trọng. Timer 0 có bit thấp TL0 ở địa chỉ 8AH và có bit cao TH0 ở địa chỉ 8CH. Timer 1 có bit thấp TL1 ở địa chỉ 8BH và bit cao TH1 ở địa chỉ 8DH.

Hoạt động định thời được cho phép bởi thanh ghi TMOD (Timer Mode Register) ở địa chỉ 89H và thanh ghi điều khiển định thời TCON (Timer Control Register) ở địa chỉ 88H. Chỉ có TCON có định địa chỉ bit.

g) Thanh ghi port nối tiếp (Serial Port Register) :

8951 chứa một port nối tiếp trên chip cho việc truyền thông tin với những thiết bị nối tiếp khác như những thiết bị đầu cuối, modem hoặc để giao tiếp IC khác (như bộ biến đổi AD, những thanh ghi di chuyển, RAM …). Thanh ghi đệm dữ liệu nối tiếp SBUF ở địa chỉ 99H giữ cả việc phát dữ liệu lẫn thu dữ liệu. Việc ghi lên SBUF để LOAD dữ liệu cho việc truyền và đọc SBUF để truy xuất dữ liệu cho việc nhận, nhưng mode hoạt động khác nhau được lập trình thông qua thanh ghi điều khiển port nối tiếp SCON.

i) Các thanh ghi ngắt (Interrup Register) :

8951 có hai cấu trúc ngắt ưu tiên, 5 bộ nguồn. Những Interrup bị mất tác dụng sau khi hệ thống Reset (bị cấm) và sau đó được cho phép bởi việc ghi lên thanh ghi cho phép ngắt IE (Interrup Enable Register) ở địa chỉ A8H. Mức ưu tiên được đặt vào thanh ghi ưu tiên ngắt IP (Interrup Priolity Level) tại địa chỉ B8H. Cả hai thanh ghi trên đều có bit địa chỉ.

h) Thanh ghi điều khiển nguồn PCON (Power Control Regiester) :

Thanh ghi PCON không có bit định vị, nó ở địa chỉ 87H bao gồm các bit địa chỉ tổng hợp. Các bit PCON được tóm tắt như sau :

Bit 7 (SMOD) : Bit có tốc độ Baud ở mode 1,2,3 ở port nối tiếp khi set.
Bit 6,5,4 : không có địa chỉ.
Bit 3 (GF1) : Bit 1 của cờ đa năng.
Bit 2 (GF2) : Bit 2 của cờ đa năng.

Bit 1 (PD) : set để khởi động mode Power Down và thoát ra để reset.
Bit 0 (IDL) : set để khởi động mode Idle và thoát ra khi ngắt hoặc reset.

Các bit điều khiển Power là Power Down và Idle có tác dụng chính trong tất cả các IC họ MSC-51 nhưng chỉ được thi hành trong sự biên dịch của CMOS.

Mode Idle : khi có lệnh set bit IDL thì nó được thi hành trước khi vào mode Idle. Trong mode Idle, tín hiệu clock được đóng cổng bởi CPU, nhưng không được đóng cổng bởi bộ ngắt, timer, port nối tiếp. Các trạng thái của CPU được bảo quản, nội dung của tất cả các thanh ghi được duy trì, các chân port giữ nguyên mức logic, hai tín hiệu ALE và PSEN được giữ mức cao. Để kết thúc Idle, ta xóa bit IDL (hoặc reset hệ thống).

Mode Power Down : có một lệnh set bit PD thì lệnh này thi hành trước khi vào Mode PD. Trong mode Power Down, giao động trên chip ngừng, tất cả các lệnh ngừng hoạt động, nội dung RAM trên chip được giữ lại, các chân port giữ nguyên mức logic, hai tín hiệu ALE và PSEN được giữ mức thấp. Thoát khỏi mức này khi reset hệ thống.

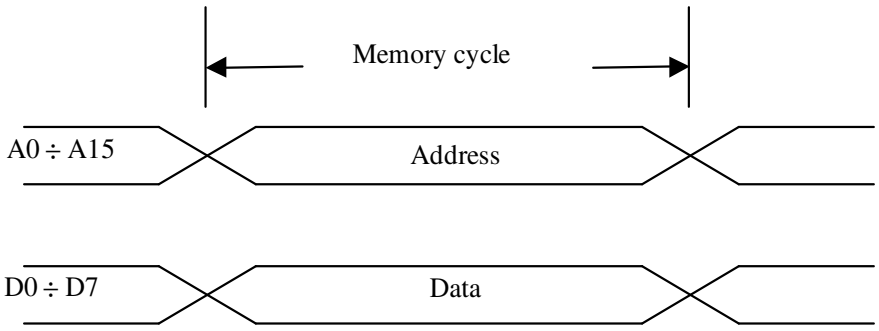
Trong suốt quá trình hoạt động của Mode PD, Vcc có thể xuống thấp 2V. Chú ý Vcc sẽ được giữ không cho hạ thấp cho đến sau khi mode PD được vào, và để khôi phục Vcc đến 5V tối thiểu 10 chu kỳ dao động trước khi chân RST xuống thấp trở lại.

5) Bộ nhớ ngoài (External Memory) :

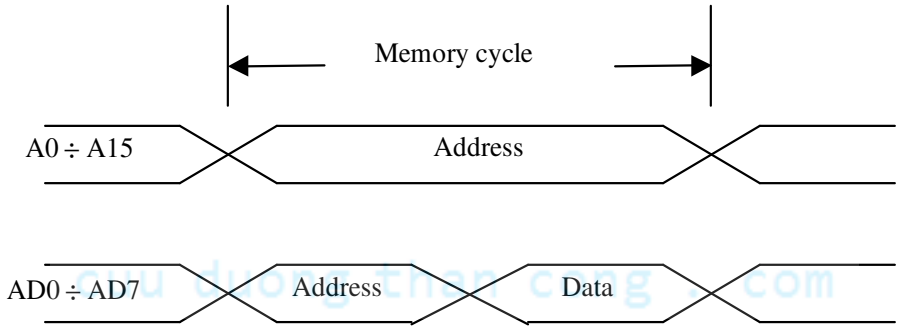
Bộ nhớ ngoài của các vi điều khiển rất quan trong để mở rộng tầm hoạt động lớn hơn Resources trên chip để tránh sự thiếu hụt của thiết bị được nhà sản xuất quy định. Họ MCS-51 cho phép mở rộng vùng nhớ mã ngoài 64KB và vùng nhớ dữ liệu ngoài 64KB. Những ngoại vi cũng được thêm vào để mở rộng thêm khả năng xuất nhập I/O và chúng trở thành vùng nhớ dữ liệu ngoài sử dụng như bản đồ nhớ I/O. Khi dùng bộ nhớ ngoài thì port trở thành địa chỉ đa kênh (A0 ÷ A7) và bus dữ liệu (D0 ÷ D7) mà tín hiệu ALE sẽ chốt byte của địa chỉ lúc bắt đầu của mỗi chu kỳ nhớ ngoài. Port 2 thường được dùng làm byte cao của bus địa chỉ. Port 0 được ghép chung để tiết kiệm chân ra.

Phương pháp chung được sắp đặt để làm việc là : trong suốt nữa đầu chu kỳ nhớ, byte thấp của địa chỉ được cung cấp bởi port 0 và được chốt bởi việc dùng tín hiệu ALE. Một con chốt 74HC373 (hoặc tương đương) sẽ giữ byte thấp của địa chỉ chỉ chốt lại trong khoảng thời gian của chu kỳ nhớ. Trong nữa chu kỳ nhớ thứ hai, port 0 được sử dụng như bus dữ liệu và dữ liệu được đọc hoặc ghi tùy thuộc hoạt động đọc hay ghi.

Việc phân kênh BUS dữ liệu và BUS địa chỉ được thể hiện trên hình vẽ sau :



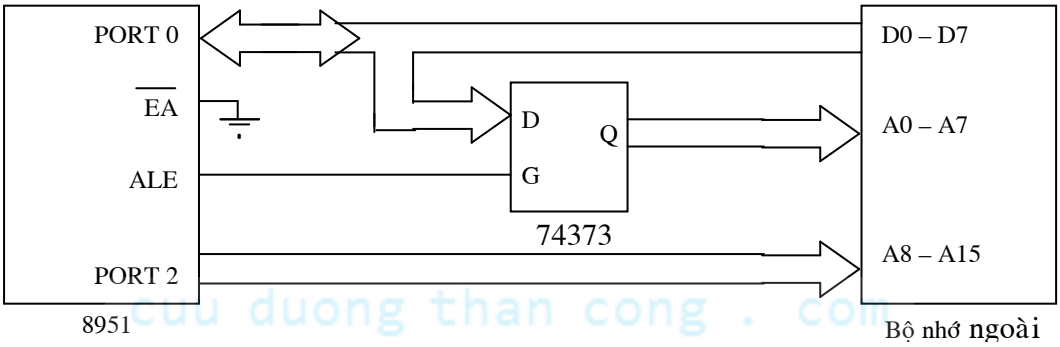
Nomultiplexed (24 pins)

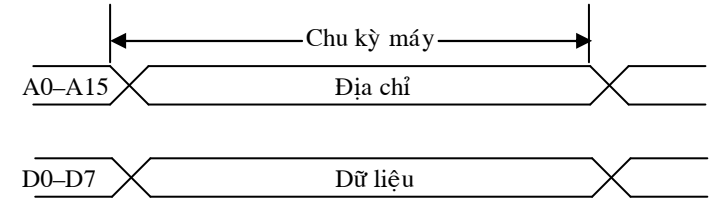


Multiplexed (16 pins)

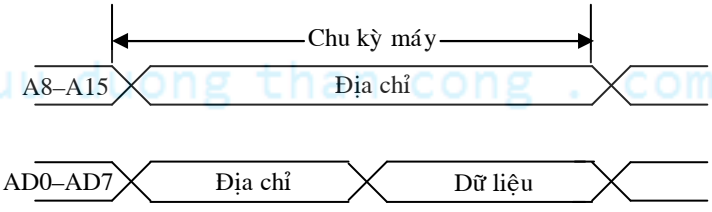
a) Bộ nhớ ngoài :

Ngoài bộ nhớ trong của 8051 ta có thể mở rộng bộ nhớ cho 8051 cụ thể là có thể thêm 64 Kb ROM và 64 Kb RAM. Khi dùng thêm bộ nhớ ngoài, nhà sản xuất khuyến khích dùng Port 0 làm Bus dữ liệu (D0 - D7) đồng thời là bus địa chỉ của byte thấp (A0 - A7) bằng tín hiệu chốt ALE để chốt byte thấp của địa chỉ khi bắt đầu mỗi chu kỳ bộ nhớ ngoài. Còn Port 2 dùng cho byte cao của bus địa chỉ. Cách tổ chức để Port 0 vừa mang dữ liệu vừa mang địa chỉ, trong nửa đầu chu kỳ bộ nhớ byte địa chỉ thấp được xuất ra từ Port 0 và được chốt lại bằng tín hiệu ALE. IC chốt 74HC373 hoặc tương đương sẽ giữ byte thấp của địa chỉ này đến hết chu kỳ bộ nhớ. Trong nửa chu kỳ thứ hai của chu kỳ bộ nhớ, Port 0 được dùng như bus dữ liệu, và dữ liệu này được đọc hay là viết tùy thuộc vào thao tác của lệnh.





a.Không Multiplexed



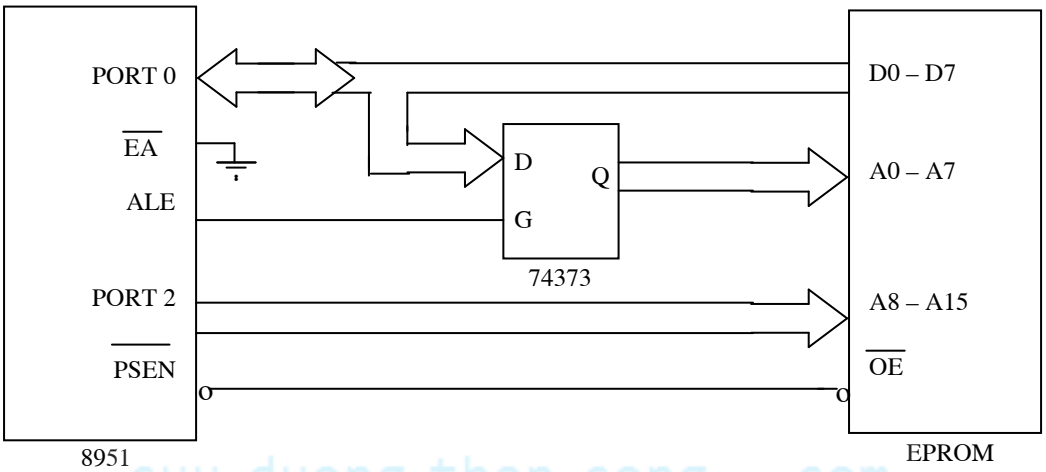
b.Có Multiplexed

b) Truy xuất ROM:

ROM là bộ nhớ chỉ đọc và được cho phép truy xuất bằng tín hiệu PSEN. Cách kết nối phần cứng giữa 8051 với ROM như hình vẽ.

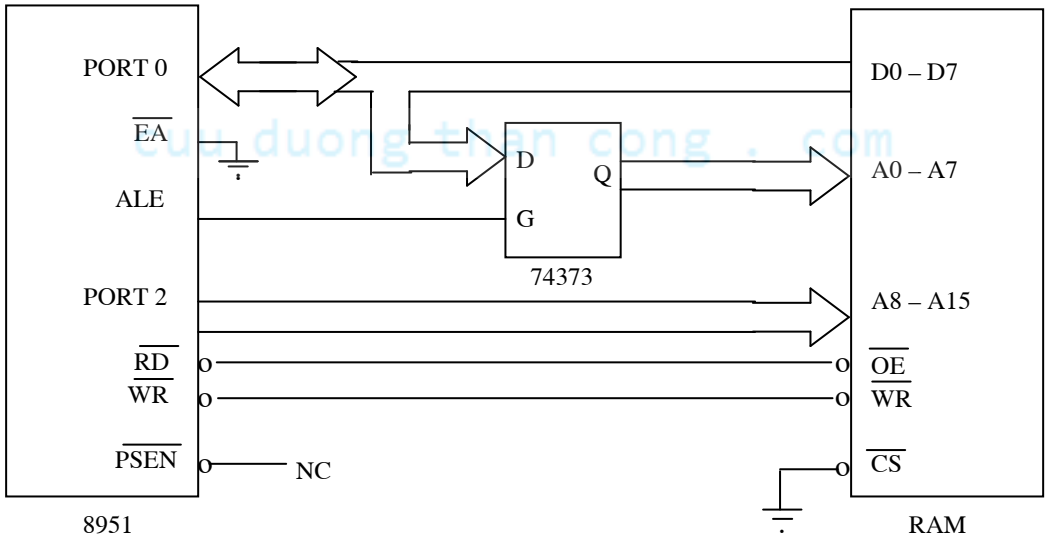
Mỗi chu kỳ máy kéo dài khoảng 12 xung clock. Nếu 8051 hoạt động với nguồn dao động là 12Mh thì thời gian của một chu kỳ máy là 1s. Giản đồ thời gian của hoạt động đọc bộ nhớ được chỉ ra ở hình.

Ta thấy trong một chu kỳ máy, xung ALE xuất hiện hai lần và hai byte được đọc từ ROM (từ lệnh hiện hành là lệnh một byte thì byte thứ hai sẽ được bỏ qua).



c) Truy xuất RAM :

RAM ngoài được truy xuất bằng các tín hiệu Read (RD) và Write (WR) ở các chân P3.7 và P3.6 tương ứng. Lệnh duy nhất truy xuất RAM là lệnh MOVX sử dụng một trong các thanh ghi data pointer (DPTR), R0, R1 như thanh ghi địa chỉ. RAM giao tiếp với 8051 giống như cách thức của EPROM chỉ khác nhau ở đường RD nối với chân Output Enable (OE) của RAM và WR nối với chân W của RAM.



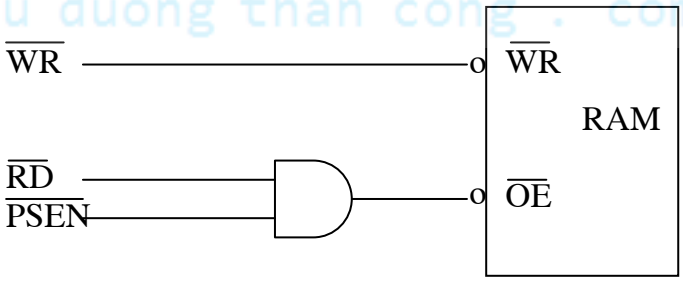
d) Giải mã địa chỉ cho 8951:

Nếu có nhiều chip ROM hoặc RAM cùng giao tiếp với 8051, vấn đề được đặt ra là phải giải mã địa chỉ cho chúng. Vấn đề giải mã cũng tương tự như các vi xử lý.

Thông thường người ta dùng IC giải mã như 74HC138, nối các chân ra của nó với các đầu chọn chip (Chip Select CS) của ROM hoặc RAM. Điều này được minh họa trong hình trên để giải mã cho hệ thống dùng nhiều IC EPROM 8Kx8bit (2764) và các IC RAM 8K (6264)

e) Xếp chồng các vùng nhớ chương trình và dữ liệu bên ngoài:

Vì bộ nhớ chương trình là ROM, nên này sinh một vấn đề bất tiện khi phát triển phần mềm cho 8951. Đó là làm cách nào phần mềm có thể sửa đổi chương trình và ghi trở lại khi nó được chứa trong bộ nhớ "chỉ đọc". Cách giải quyết là xếp chồng các vùng nhớ chương trình và dữ liệu. Một IC RAM có thể chứa cả chương trình và dữ liệu bằng cách nối đường OE của RAM vào một mạch logic AND của PSEN và RD. Mạch trên hình sau cho phép một IC RAM được dùng làm bộ nhớ chương trình và dữ liệu:



Vậy một chương trình có thể được tải vào RAM (bằng cách ghi nó như bộ nhớ dữ liệu) và thi hành (bằng cách truy xuất nó như bộ nhớ chương trình).

6. Lệnh Reset:

8951 được reset bằng cách giữ chân RST ở mức cao ít nhất trong hai chu kỳ máyvà trả nó về mức thấp.RST có thể được kích bằng tay dùng một nút bấm hoặc có thể kích khi ngắt điện dùng một mạch RC. Trạng thái của tất cả các thanh ghi của 8951 sau khi reset hệ thống được tóm tắt trong bảng sau:

Thanh ghi	Nội dung
Đếm chương trình	0000 H
Tích lũy	00 H
B	00 H
PSW	00 H
SP	07 H
DPTR	0000 H
Port 0 ÷ 3	FF H
IP	XXX00000 B
IE	0XX00000 B
Các thanh ghi định thời	00 H
SCON	00 H
SBUF	00 H
PCON (HMOS)	0XXXXXXX B
PCON (CMOS)	0XXX0000 B

Quan trọng nhất trong các thanh ghi trên là thanh ghi đếm chương trình, nó được đặt lại 0000H. Khi SRT trở lại mức thấp, việc thi hành chương trình luôn luôn bắt đầu ở địa chỉ đầu tiên trong bộ nhớ chương trình: địa chỉ 0000H. Nội dung của RAM trên chip không thay đổi bởi lệnh Reset.

II – HOẠT ĐỘNG CỦA BỘ ĐỊNH THỜI (Timer) :

1) Giới thiệu:

Một định nghiã đơn giản của Timer là một chuỗi các Flip – Flop chia đôi tần số nối tiếp với nhau, chúng nhận tín hiệu vào làm xung nhịp. Ngõ ra cuả tầng cuối làm xung nhịp cho Flip – Flop báo tràn của Timer (Flip – Flop cờ). Giá trị nhị phân trong các Flip – Flop của Timer có thể xem như một số đếm số xung nhịp (hoạt các sự kiện) từ khi khởi động Timer. Ví dụ Timer 16 Bit sẽ đếm lên từ 0000H đến FFFFH. Cờ báo tràn sẽ lên 1khi số đếm tràn từ FFFFH đến 0000H.

8951 có 2 Timer 16 bit, mỗi Timer có bốn cách làm việc. Người ta sử dụng các Timer để:

a) định khoảng thời gian,
b) đếm sự kiện,
c) tạo tốc độ baud cho port nối tiếp.

Trong các ứng dụng định khoảng thời gian, người ta lập trình Timer trong khoảng thời gian đều đặn và đặt cờ tràn timer. Cờ được dùng để đồng bộ hoá chương trình để thực hiện một tác động như kiển tra trạng thái cuả các ngõ vào hoặc gởi sự kiện ra các ngõ ra. Các ứng dụng khác có thể sử dụng việc tạo xung nhịp đều đặn cuả Timer để đo thời gian trôi qua giữa hai sự kiện (ví dụ : đo độ rộng xung).

Đếm sự kiện dùng để xác định số lần xảy ra của một sự kiện. Một sự kiện là bất cứ tác động ngoài nào có thể cung cấp một chuyển trạng thái trên một chân của 8951. Các Timer cũng có thể cung cấp xung nhịp tốc độ Baud cho port nối tiếp.

Truy xuất Timer của 8951 dùng sáu thanh ghi chức năng đặc biệt cho trong bảng sau:

Tên thanh ghi đặc biệt	Mục đích	Địa chỉ	Bit được địa chỉ hóa
TCON	Điều khiển	88H	Có
TMOD	Chế độ	89H	Không
TL0	Byte thấp bộ Timer 0	8AH	Không
TL1	Byte thấp bộ Timer 1	8BH	Không
TH0	Byte cao bộ Timer 0	8CH	Không
TH1	Byte cao bộ Timer 1	8DH	Không

2) Thanh ghi chế độ định thời (TMOD) :

Bit	Tên	Bộ Timer	Mô tả chức năng
7	GAT	1	Khi set bộ Timer 1 chỉ chạy khi INT1 ở mức cao
6	E	1	Bit lựa chọn bộ đếm/đồng hồ.
5	C/T	1	Bit 1 chọn chế độ.
4	M1	1	Bit 0 chọn chế độ.
3	M0	0	Bit (mở) cổng.
2	GAT	0	Bit lựa chọn bộ đếm/đồng hồ.
1	E	0	Bit 0 chọn chế độ.
0	C/T	0	Bit 1 chọn chế độ.
	M1		
	M0		

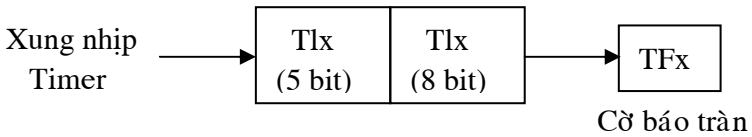
Tóm tắt thanh ghi TMOD

3) Các chế độ hoạt động cho Timer :

M1	M0	Chế độ	Mô tả
0	0	0	Chế độ Tiner 13 bit
0	1	1	Chế độ Timer 16 bit
1	0	2	Chế độ tự động nạp lại giá trị đếm Timer 0 : TL0 là Timer 8 bit được điều khiển bằng các bit chế độ của Timer 0 TH0 tương tự nhưng được điều khiển bằng các bit của chế độ Timer 1
1	1	3	Chế độ tách đôi Timer Timer 0 : TL0 là bộ định thời Timer 8 bit được điều khiển bằng các bit của chế độ Timer 0 TH0 tương tự nhưng được điều khiển bằng các bit của chế độ Timer 1 Timer 1 : Ngưng hoạt động

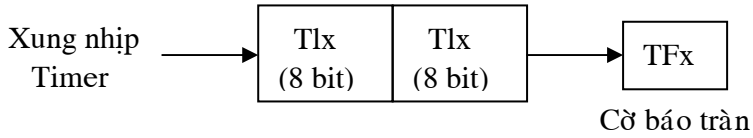
a. Chế độ 0 - chế độ Timer 13 bit :

- Để tương thích với 8084 (có trước 8051).
- 3 bit cao của Tlx (TL0 và / hoặc TL1) không dùng.



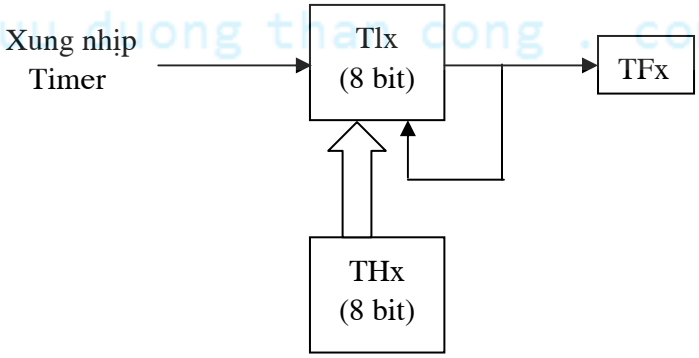
b. Chế độ 1- chế độ Timer 16 bit :

- Hoạt động như Timer 16 bit đầy đủ.
- Cờ báo tràn là bit Tlx trong TCON có thể đọc hoặc ghi bằng phần mềm.
- MSB của giá trị trong các thanh ghi Timer là bit 7 của Thx là bit 0 của Tlx. Các thanh ghi Timer (Tlx / Thx) có thể được đọc hoặc ghi bất cứ lúc nào bằng phần mềm.



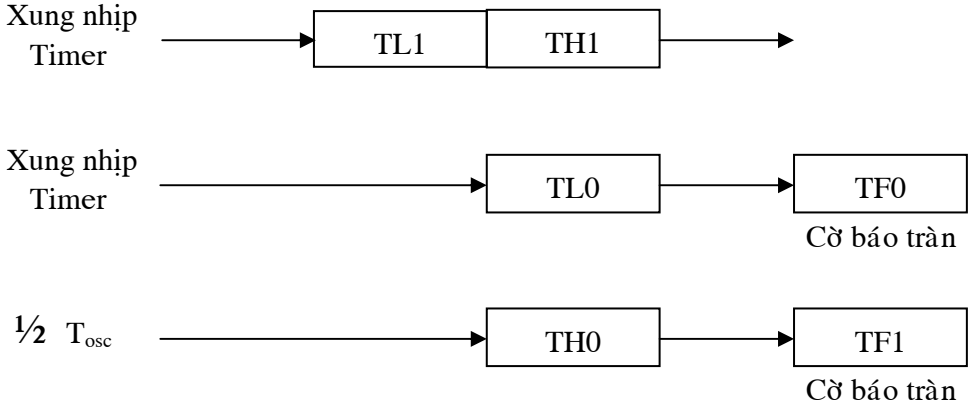
c. Chế độ 2- chế độ tự động nạp lại :

Tlx hoạt động như là một Timer 8 bit, trong khi đó THx vẫn giữ nguyên giá trị được nạp. Khi số đếm tràn từ FFH đến 00H, không những cờ timer được set mà các giá trị trong Thx đồng thời được nạp vào TLx, việc đếm tiếp tục từ giá trị này lên đến FFH xuống 00H và nạp lại …Chế độ này rất thông dụng vì sự tràn timer xảy ra trong những khoảng thời gian nhất định và tuần hoàn một khi đã khởi động TMOD và THx.



d. Chế độ 3 - chế độ tách Timer :

Timer 0 tách thành hai Timer 8 bit (TL0 và TH0), TL0 có cờ báo tràn là TF0 và TH0 có cờ là TF1.



4) Thanh ghi điều khiển Timer (TCON) :

Thanh ghi này chứa các bit trạng thái và các bit điều khiển cho Timer 0 và Timer 1.

Xem bảng bảng tóm tắt thanh ghi TCON.

Bit	Ký hiệu	Bit địa chỉ	Mô tả
TCON.7	TF1	8FH	Cờ tràn bộ Timer 1. Được set bằng phần cứng lúc tràn, được xóa bằng phần mềm hay phần cứng khi bộ xử lý chỉ đến chương trình phục ngắt.
TCON.6	TR1	8EH	Bit điều khiển Timer 1 chạy. Đặt hay xóa bằng phần mềm.
TCON.5	TF0	8DH	Cờ báo tràn Timer 0.
TCON.4	TR0	8EH	Bit điều khiển Timer 0 chạy.
TCON.3	IE1	8BH	Cờ cạnh ngắt 1 bên ngoài. Đặt bởi phần cứng khi phát hiện cạnh xuống ở chân INT1, xóa bằng phần mềm hoặc phần cứng khi CPU chỉ đến chương trình phục vụ ngắt.
TCON.2	IT1	8AH	Cờ kiểu ngắt 1 bên ngoài. Đặt hay xóa bằng phần mềm để ngắt ngoài tích cực cạnh xuống hay mức thấp.
TCON.1	IE0	89H	Cờ cạnh ngắt 0 bên ngoài.
TCON.0	IT0	88H	Cờ kiểu ngắt 0 bên ngoài.

5) Nguồn tạo xung nhịp:

Có hai nguồn tạo xung nhịp có thể có, được chọn bằng cách ghi vào các bit C/T (cuunter/timer) trong TMOD khi khởi động timer. Một nguồn tạo xung nhịp dùng cho định khoảng thời gian, các khác cho đếm sự kiện.

- Định khoảng thời gian (interval timing):

Nếu C/T = 0, hoạt động timer liên tục được chọn và timer được dùng cho việc định khoảng thời gian. Lúc đó, timer lấy xung nhịp từ bộ dao động trên chip. Bộ chia 12 được thêm vào để giảm tần số xung nhịp đến giá trị thích hợp cho phần lớn các ứng dụng. Như vậy, thạch anh 12MHz sẽ cho tốc độ xung nhịp timer 1MHz. Báo tràn timer xảy ra sau một số (cố định) xung nhịp, phụ thuộc vào giá trị ban đầu được nạp vào các thanh ghi timer TLx/THx.

- Đếm sự kiện (Event counting):

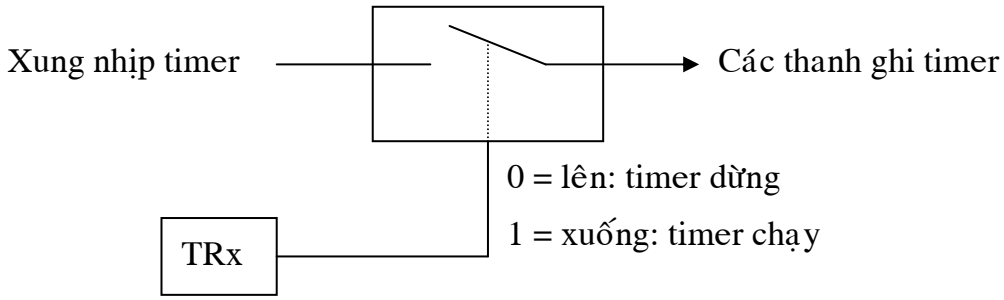
Nếu C/T = 1, timer lấy xung nhịp từ bên ngoài.Trong hầu hết các ứng dụng, nguồn bên ngoài này cung cấp cho time một xung khi xảy ra một "sự kiện"- timer dùng đếm sự kiện. Số sự kiện được xác định bằng phần mềm bằng cách đọc các thanh ghi TLx/THx vì giá trị 16 bit trong các thanh ghi này tăng lên 1 cho mỗi sự kiện.

Nguồn xung nhịp ngoài có từ thay đổi chức năng của các chân Port 3. Bit 4 của Port 3 (P3.4) dùng làm ngõ vào tạo xung nhịp bên ngoài cho timer 0 và được gọi là "T0". Và P3.% hau "T1" là ngõ vào tạo xung nhịp cho timer 1.

Trong các ứng dụng bộ đếm, các thanh ghi timer được tăng thêm 1 tương ứng với chuyển từ 1 xuống 0 ở ngõ vào bên ngoài : Tx. Ngõ vào bên ngoài được lấy mẫu trong S5P2 của mọi chu kỳ máy. Như vậy, khi ngõ vào cao trong một chu kỳ và thấp trong chu kỳ kế thì số đếm được tăng thêm 1. Giá trị mới được xuất hiện trong các thanh ghi trong S3P1 của chu kỳ theo sau chu kỳ trong đó phát hiện sự chuyển tiếp. Do đó mất hai chu kỳ máy (2μs) để ghi nhận một sự chuyển 1 sang 0, tần số ngoài tối đa là 500KHz (giả sử hoạt động ở 12MHz).

6. Bắt đầu, dừng và điều khiển các timer:

Phương pháp đơn giản nhất để bắt đầu (cho chạy) và dừng các timer là dùng các bit điều khiển chạy: TRx trong TCON. TRx bị xoá sau khi reset hệ thống. Như vậy, các timer theo mặc nhiên là bị cấm (bị dừng). TRx được đặt bằng phần mềm cho các timer chạy.



Cho chạy và dừng các timer

Vì TRx ở trong thanh ghi TCON có địa chỉ bit, nên dễ dàng cho việc điều khiển các timer trong chương trình. Ví dụ: cho timer 0 chạy bằng lệnh:

```
SETB    TR0
```

và dừng lệnh:

```
SETB    TR0
```

Trình biên dịch sẽ thực hiện việc biến đổi ký hiệu cần thiết từ "TR0" sang địa chỉ bit đúng. SETB TR0 chính xác giối như SETB 8CH.

Một phương pháp khác để điều khiển các timer là dùng bit GATE trong TMOD và ngõ vào bên ngoài INTx. Việc này hữu dụng cho việc đo độ rộng xung như sau: Giả sử INT0 ở mức thấp nhưng các xung ở mức cao trong khoảng thời gian đo. Khởi động timer 0 ở chế độ 2 (chế độ timer 16 bit), với TL0/TH0 = 0000H, GATE = 1 và TR0 =1. Khi INT0 ở mức cao, timer được "mở cổng" và được cấp xung nhịp 1MHz (nếu 8951 hoạt động ở tần số 12MHz). Khi INT0 xuống thấp, timer bị "đóng cổng" và thời khoảng của xung tính bằng µs là số đếm được trong TL0/TH0. (Có thể lập trình INT0 để tạo ra một ngắt khi nó xuống thấp).

7. Khởi động và truy xuất các thanh ghi timer:

Thông thường các thanh ghi được khởi động một lần ở đầu chương trình để đặt chế độ làm việc đúng. Sau đó, trong thân chương trình, các timer được cho chạy, dừng, các bit cờ được kiểm tra và xoá, các thanh ghi timer được đọc và cập nhật, .v.v. theo đòi hỏi của các ứng dụng.

TMOD là thanh ghi thứ nhất được khởi động vì nó đặt chế độ hoạt động. Ví dụ, các lệnh sau khởi động timer 1 như timer 16 bit (chế độ) có xung nhịp từ bộ dao động trên chip cho việc định khoảng thời gian:

```
MOV    TMOD, #00010000B
```

Lệnh này sẽ đặt M1 = 0 và M0 = 1 cho chế độ 1, C/T = 0 và GATE = 0 cho xung nhịp nội và các bit chế độ timer 0. Dĩ nhiên, timer không bắt đầu định thời cho đến khi bit điều khiển chạy TR1 được đặt lên 1.

Nếu cần số đếm ban đầu, các thanh ghi timer TL1/TH1 cũng phải được khởi động. Nhớ lại là các timer đếm lên và đặt cờ báo tràn khi có sự chuyển tiếp FFFFH sang 0000H. một khoảng 100 µs có thể định thời bằng cách khởi động cho TL1/TH1 là FF9CH :

```
MOV    TL1, #9CH
MOV    TH1, #0FFH
```

Rồi timer cho chạy bằng cách đặt bit điều khiển chạy như sau:

```
SET    TR1
```

Cờ báo tràn được tự động đặt lên 1 sau 100 µs. Phần mềm có thể đợi trong 100 µs bằng cách dùng lệnh rẽ nhánh có điều kiện nhảy đến chính nó trong khi cờ báo tràn chưa được đặt lên 1:

```
WAIT:   JNB    TF1, WAIT
```

Khi timer tràn, cần dừng timer và xoá cờ báo tràn trong phần mềm

```
CLR    TR1
CLR    TF1
```

- Đọc timer đang chạy:

Trong một số ứng dụng cần đọc giá trị trong thanh ghi timer đang chạy. Vì phải đọc hai thanh ghi timer, "sai pha" có thể xảy ra nế byte thấp tràn vào byte cao giữa hai lần đọc. Giá trị có thể đọc được không đúng. Giải pháp là đọc byte cao trước, kế đó đọc byte thấp rồi đọc byte cao lại một lần nữa. Nếu byte cao đã thay đổi thì lặp lại các hoạt động đọc. Các lệnh dưới đây đọc nội dung của các thanh ghi timer TL1/TH1 vào các thanh ghi R6/R7:

```
AGAIN :    MOV    A, TH1
           MOV    R6, TL1
           CJNZ   A, TH1, AGAIN
           MOV    R7, A
```


8. Các khoảng ngắn và khoảng dài:

Dãy các thời gian có thể định thời thời là bao nhiêu ? Vấn đền này đã được khảo sát với 8951 hoạt động với tần số 12MHz. Như vậy xung nhịp của các timer có tần số là 1MHz.

Khoảng thời gian ngắn nhất có bị giới hạn không chỉ bởi tần số xung nhịp của timer mà còn bởi phần mềm. Do ảnh hưởng của thời khoảng thực hiện một lệnh, lệnh ngắn nhất của 8951 là 1 chu kỳ máy hay 1 µs. Sau đây là bảng tóm tắt các kỹ thuật để tạo những khoảng thời gian có chiều dài khác nhau (với giả sử xung nhịp cho 8951 có tần số 12MHz):

Khoảng thời gian tối đa (µs)	Kỹ thuật
≈ 10	- bằng phần mềm
256	- timer 8 bit với tự động nạp lại
65535	- timer 16 bit
không giới hạn	- timer 16 bit cộng với các vòng lặp phần mềm

Các kỹ thuật để lập trình các khoảng thời gian (F_{OCS} = 12mhZ).

III – HOẠT ĐỘNG PORT NỐI TIẾP:

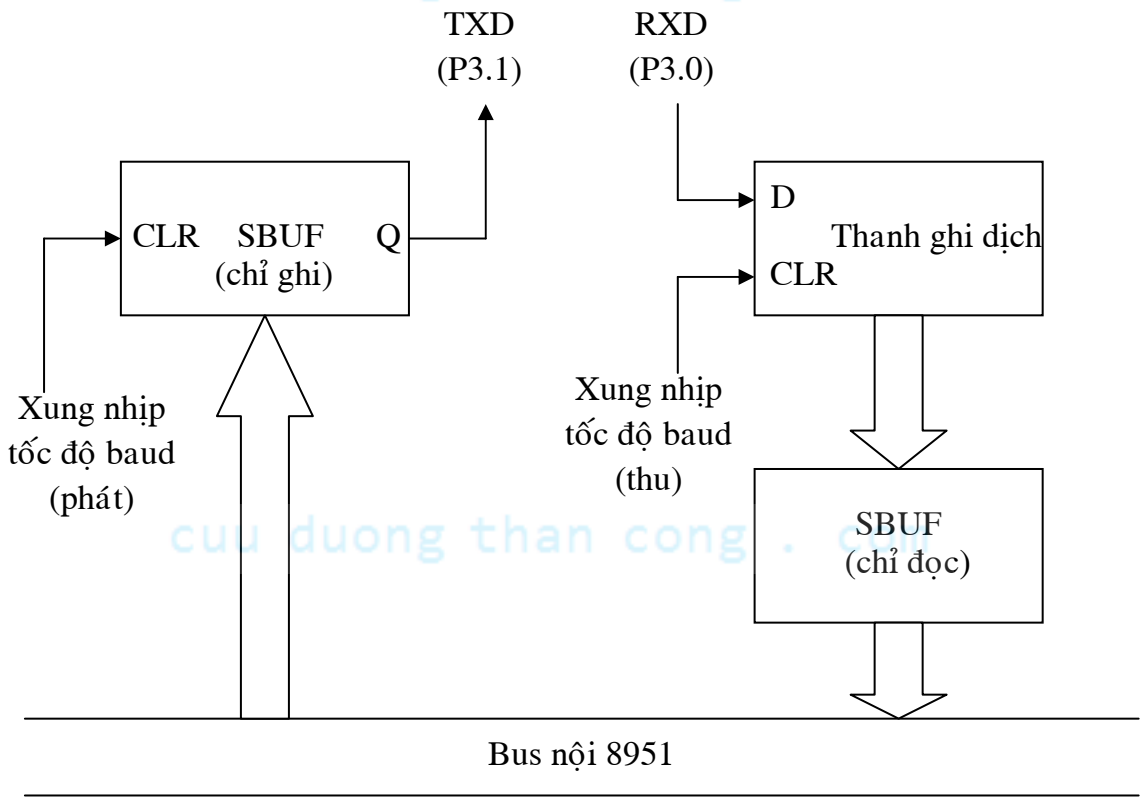
1) Giới thiệu:

8951có một port nối tiếp trong chip có thể hoạt động ở nhiều chế độ trên một dãi tần số rộng. Chức năng chủ yếu của port nối tiếp là thực hiện chuyển đổi song song sang nối tiếp đối với dữ liệu xuất, và chuyển đổi dữ liệu nối tiếp sang song song đối với dữ liệu nhập.

Truy xuất phần cứng đến port nối tiếp qua các chân TXD và RXD. Các chân này có các chức năng khác với hai bit của port 3, P3.1 ở chân 11 (TXD) và P3.0 ở chân 10 (RXD).

Port nối tiếp cho hoạt động song công (full duplex: thu và phát đồng thời), và đệm lúc thu (receiver buffering) cho phép một ký tự sẽ được thu và được giữ trong khi ký tự thứ hai được nhận. Nếu CPU đọc ký tự thứ nhất trước ký tự thứ hai được thu đầy đủ thì dữ liệu sẽ không bị mất.

Hai thanh ghi chức năng đặc biệt cho phép phần mềm truy xuất đến port nối tiếp là: SBUF và SCON. Bộ đệm port nối tiếp (SBUF) ở địa chỉ 99H thật sự là hai bộ đệm. Viết vào SBUF để nạp dữ liệu sẽ được phát, và đọc SBUF để truy xuất dữ liệu thu được. Đây là hai thanh ghi riêng biệt: thanh ghi chỉ ghi để phát và thanh ghi chỉ đọc để thu.



Thanh ghi điều khiển port nối tiếp (SCON) ở địa chỉ 98H là thanh ghi có địa chỉ bit chứa các bit trạng thái và các bit điều khiển. Các bit điều khiển đặt chế độ hoạt động cho port nối tiếp, và các bit trạng thái báo kết thúc việc phát hoạt thu ký tự. Các bit trạng thái có thể kiểm tra bằng phần mềm hoặc có thể lập trình để tạo ngắt.

Tần số làm việc của port nối tiếp, còn gọi là tốc độ baud có thể cố định (lấy từ bộ dao động trên chip). Nếu sử dụng tốc độ baud thay đổi, Timer 1 sẽ cung cấp xung nhịp tốc độ baud và phải được lập trình.

2) **Thanh ghi điều khiển port nối tiếp:**

Chế độ hoạt động cuả port nối tiếp được đặt bằng cách ghi vào thanh ghi chế độ port nối tiếp (SCON) ở địa chỉ 98H. Sau đây là các bảng tóm tắt thanh ghi SCON và các chế độ cuả port nối tiếp:

Bit	Ký hiệu	Địa chỉ	Mô tả
SCON.7	SMO	9FH	Bit 0 của chế độ port nối tiếp.
SCON.6	SM1	9EH	Bit 1 của chế độ port nối tiếp.
SCON.5	SM2	9DH	Bit 2 của chế độ port nối tiếp. Cho phép tryền thông đa sử lý trong các chế độ 2 và 3, R1 sẽ không bị tác động nếu bit thứ 9 thu được là 0.
SCON.4	REN	9CH	Cho phép bộ thu được dặt lên 1 để thu (nhận) các ký tự.
SCON.3	TB8	9BH	Bit 8 phát, bit thứ 9 được phát trong các chế độ 2 và 3, được đặt và xoá bằng phần mềm.
SCON.2	RB8	9AH	Bit 8 thu, bit 9 thu được.
SCON.1	T1	99H	Cờ ngắt phát. Đặt lên 1 khi kết thúc phát ký tự, được xoá bằng phần mềm.
SCON.0	R1	98H	Cờ ngắt thu. Đặt lên 1 khi kết thúc thu ký tự, được xoá bằng phần mềm.

Tóm tắt thanh chế độ port nối tiếp SCON.

SM0	SM1	Chế độ	Mô tả	Tốc độ baud
0	0	0	Thanh ghi dịch	Cố định(Fosc/12)
0	1	1	AURT 8 bit	Thay đổi (đặt bằng timer)
1	0	2	AURT 9 bit	Cố định (Fosc chia cho 12 hoặc 64)
1	1	3	AURT 9 bit	Thay đổi (đặt bằng timer)

Các chế độ port nối tiếp.

Trước khi sữ dụng port nối tiếp, phải khởi động SCON cho đúng chế độ. Ví dụ lệnh sau:

MOV SCON. #01010010B

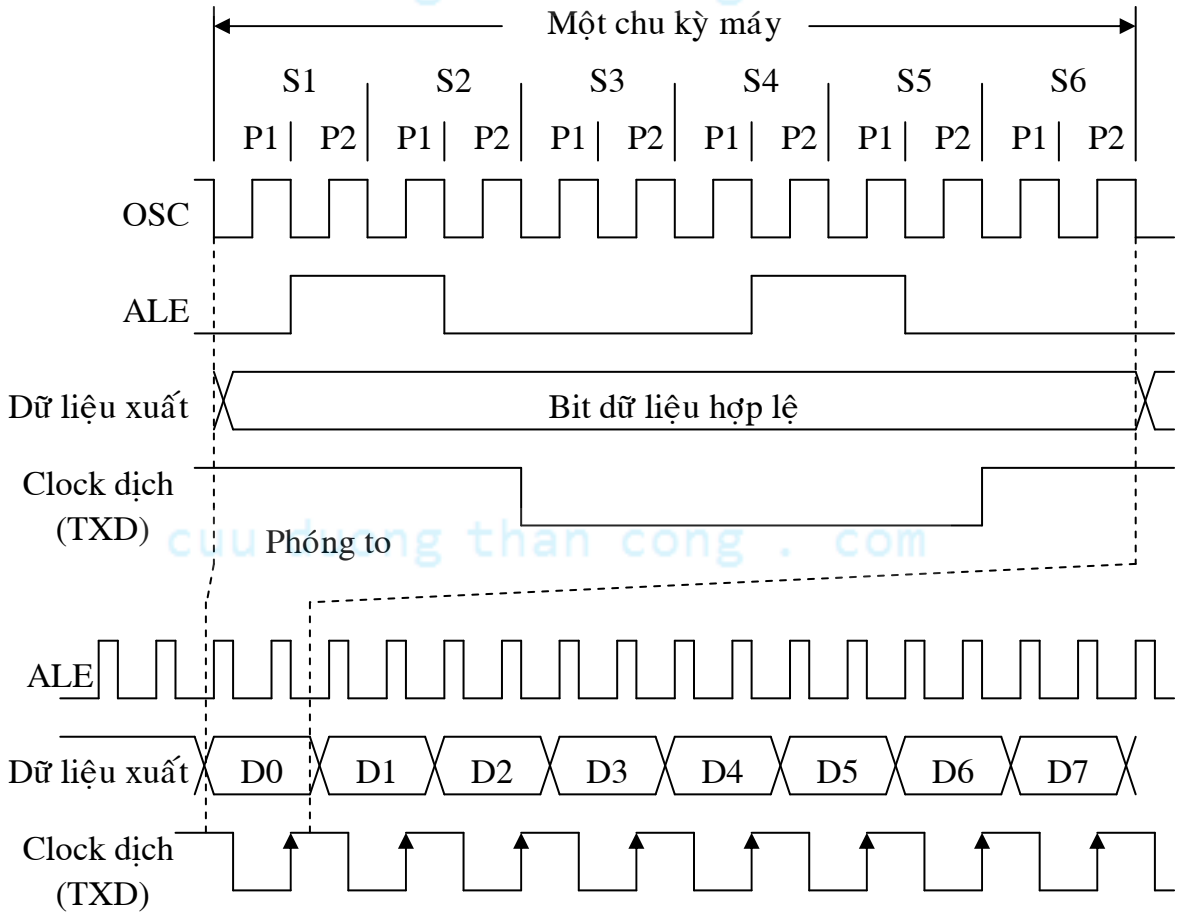
Khởi động port nối tiếp cho chế độ 1(SM0/SM1 = 0/1), cho phép bộ thu (REN = 1) và đặt cờ ngắt phát (T1 = 1) để chỉ bộ phát sẵn sàng hoạt động.

3) **Các chế độ hoạt động:**

Port nối tiếp có 4 chế độ hoạt động, có thể chọn được bằng cách viết các số 1 hay 0 và các bit SM0 và SM1 trong SCON. Có 3 chế độ cho phép truyền thông bất đồng bộ, với mỗi ký tự được thu (nhận) hoặc phát đều được đóng khung bằng một bit start và một bit stop. Ở chế độ thứ tư, port nối tiếp hoạt động như một thanh ghi dịch đơn giản.

a) Thanh ghi dịch 8 bit (chế độ 0):

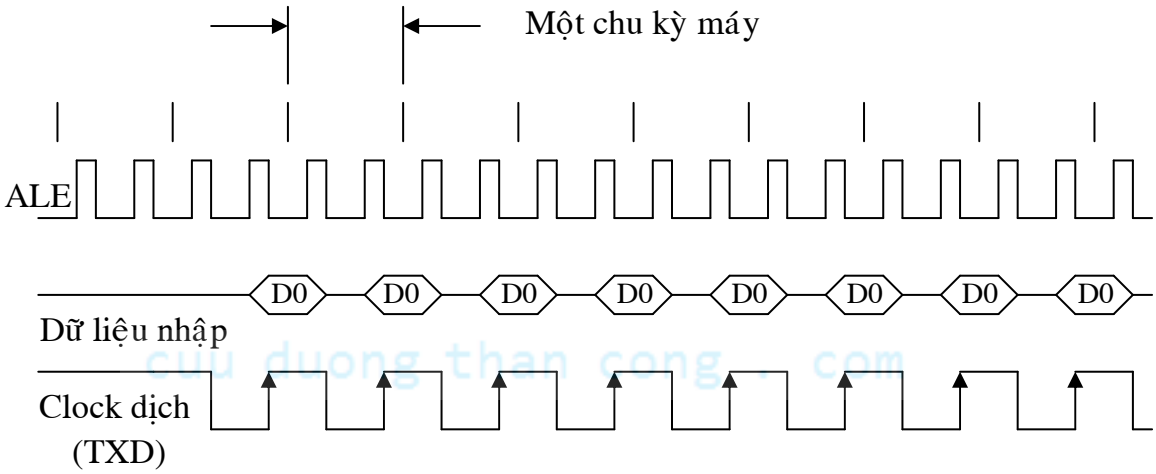
Chế độ 0 được chọn bằng cách ghi các bit 0 vào SM1và SM0 cuả SCON, đưa port nối tiếp vào chế độ thanh ghi dịch 8 bit. Dữ liệu nối tiếp vào và ra qua RXD và TXD xuất xung nhịp dịch 8 bit được phát hoặc thu với bit đầu tiên là LSB. Tốc độ baud cố định ở 1/12 tần số dao động trên chip.



Giản đồ thời gian port nối tiếp phát ở chế độ 0.

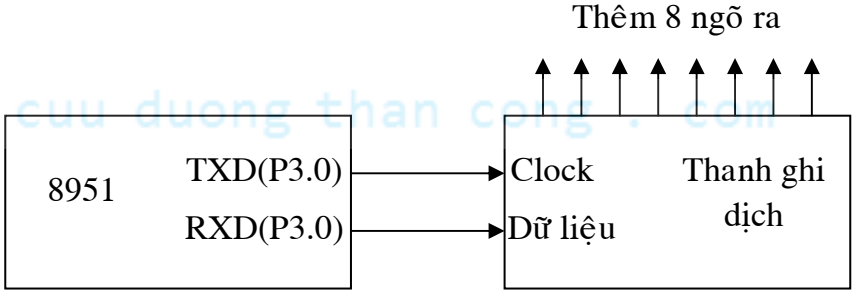
Việc phát đi được khởi động bằng bất cứ lệnh nào ghi dữ liệu vào SBUF. Dữ liệu được dịch ra ngoài trên đường RXD (P3.0) với các xung nhịp được gởi ra đường TXD (P3.1). Mỗi bit phát đi hợp lệ (trên RXD) trong một chu kỳ máy. Trong mỗi chu kỳ máy, tín hiệu xung nhịp xuống thấp ở S3P1và trở về mức cao ở S6P1.

Việc thu được khởi động khi bit cho phép bộ thu (REN) là 1 và bit ngắt thu (R1) là 0. Qui tắc tổng quát là đặt REN khi bắt đầu chương trình để khởi động port nối tiếp, rồi xóa R1 để bắt đầu hoạt động nhập dữ liệu. Khi R1 bị xoá, các xung nhịp được đưa ra đường TXD, bắt đầu chu kỳ máy kế tiếp, và dữ liệu theo xung nhịp ở đường RXD. Lấy xung nhịp cho dữ liệu vào port nối tiếp xảy ra ở cạnh dương của TXD.



Giản đồ thời gian port nối tiếp thu ở chế độ 0.

Một ứng dụng của chế độ thanh ghi dịch là mở rộng khả năng xuất của 8951. IC thanh ghi dịch nối tiếp ra song song có thể được nối vào các đường TXD và RXD của 8951 để có thêm 8 đường ra. Có thể nối xâu chuỗi thêm các thanh ghi dịch để mở rộng thêm.

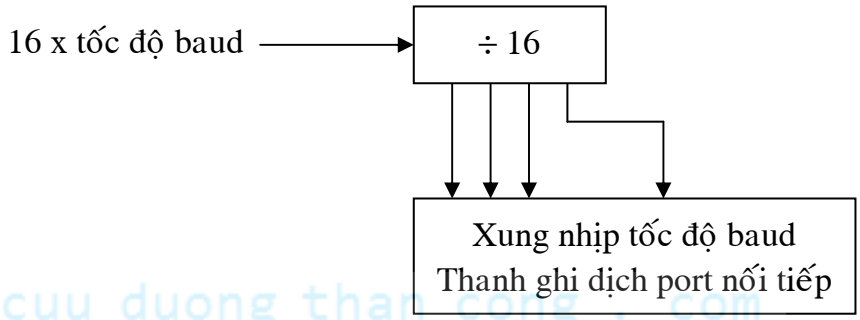


Chế độ thanh ghi dịch của port nối tiếp

b) UART 8 bit với tốc độ baud thay đổi được (chế độ 1):

ở chế độ 1, port nối tiếp cuả 8951 làm việc như một UART 8 bit với tốc độ baud thay đổi được. Một UART (Universal Asynchronous Receiver/Transmitter: Bộ thu/ phát bất đồng bộ vạn năng) là một dụng cụ thu và phát dữ liệu nối tiếp với mỗi ký tự dữ liệu đi trước là bit start ở mức thấp và theo sau là bit Stop ở mức cao. Đôi khi xen thêm bit kiểm tra chẵn lẻ giữa bit dữ liệu cuối cùng và bit Stop. Hoạt động chủ yếu cuả UART là chuyển đổi song song sang nối tiếp với dữ liệu xuất và chuyển đổi nối tiếp sang song song với dữ liệu nhập.

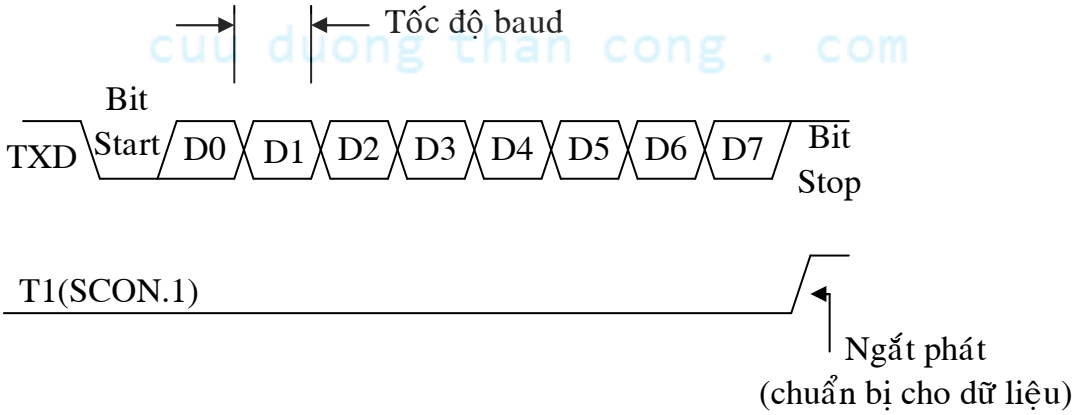
Ở chế độ 1, 10 bit được phát trên TXD hoặc thu trên RXD. Những bit đó là: 1 bit start (luôn luôn là 0), 8 bit dữ liệu (LSB đầu tiên) và 1 bit stop (luôn luôn là 1). Với hoạt động thu, bit stop được đưa vào RB8 trong SCON. Trong 8951 chế độ baud được đặt bằng tốc độ báo tràn của timer 1.



Tạo xung nhịp port nối tiếp

Tạo xung nhịp và đồng bộ hoá các thanh ghi dịch cuả port nối tiếp trong các chế độ 1, 2 và 3 được thiết lập bằng bộ đếm 4 bit chia cho 16, ngõ ra là xung nhịp tốc độ baud. Ngõ vào cuả bộ đếm này được chọn qua phần mềm.

Truyền dữ liệu (phát) được khởi động bằng cách ghi vào SBUF, nhưng vẫn chưa thật sự bắt đầu chạy cho đến khi sự thay thế kế tiếp cuả bộ đếm chia cho 16 cung cấp tốc độ baud cổng nối tiếp. Dữ liệu được dịch ra ngoài trên đường TXD bắt đầu bằng bit Start, theo sau là 8 bit dữ liệu và sau cùng là bit Stop. Độ rộng (theo thời gian của mỗi bit) là ngịch đảo cuả tốc độ baud được lập trình trong timer. Cờ ngắt phát (T1) được đặt lên 1khi xuất hiện bit Stop trên TXD.



Đặt cờ T1 port nối tiếp

Việc thu dữ liệu được khởi động bằng một chuyển trạng thái từ 1 xuống 0 trên RXD. Bộ đếm 16 bit tức thời được xoá để đồng bộ số đếm với luồng bit đến. Luồng bit đến được lấy mẫu giữa 16 lần đếm.

Bộ thu sẽ phát hiện được bit Start sai bằng cách yêu cầu trạng thái 0 ở (bit Start ở lần đếm thứ 8 sau khi có chuyển trạng thái từ 1 xuống 0 đầu tiên. Nếu điều này không xảy ra, người ta giả sử bộ thu được kích bởi nhiễu chứ không phải do một ký tự hợp lệ. Bộ tìm được Reset và quay về trạng thái nghỉ (idle), tìm kiếm (đợi) chuyển trạng thái từ 1 xuống 0 kế.

Giả sử đã phát hiện được bit Start hợp lệ, thì tiếp tục thu ký tự. Bit Start được bỏ qua và 8 bit dữ liệu được đưa vào thanh ghi dịch cổng nối tiếp theo xung nhịp. Khi có được tất cả 8 bit thì điều sau đây xảy ra:

1. Bit thứ 9 (bit Stop) được chốt vào RB8 trong SCON.

2. SBUF được nạp với 8 bit dữ liệu.

3. Cờ ngắt bộ thu (R1) được đặt lên 1.

Tuy nhiên điều này xảy ra nếu đã có những điều kiện sau:

1. R1 = 0.

2. SM2 = 1 và bit Stop thu được là 1, hoặc SM2 = 0.

Đòi hỏi R1 = 0 để bảo đảm là phần mềm đã đọc ký tự trước (và R1 được xoá). Điều kiện thứ hai hơi phức tạp nhưng chỉ áp dụng trong chế độ truyền thông đa xử lý. Điều đó hàm ý là không đặt R1 lên 1 trong chế độ truyền thông đa xử lý khi bit dữ liệu thứ 9 là 0.

b) <u>UART 9 bit với tốc độ baud cố định (chế độ 2):</u>

Khi SM1 = 1 và SM2 = 0, cổng nối tiếp làm việc ở chế độ 2, như một UART 9 bit có chế độ baud cố định. 11 bit sẽ được phát hoặc thu: 1bit Start, 8 bit dữ liệu, bit dữ liệu thứ 9 có thể lập trình được và 1 bit Stop. Khi phát bit thứ 9 là bất cứ gì đã được đưa vào TB8 trong SCON (có thể là bit parity). Khi thu bit thứ 9 thu được sẽ ở trong RB8. Tốc độ baud ở chế độ 2 là 1/32 hoặc 1/16 tần số dao động trên chip.

c) <u>UART 9 bit với tốc độ baud thay đổi được:</u>

Chế độ này giống chế độ 2 ngoại trừ tốc độ baud có thể lập trình được và được cung cấp bởi timer. Thật ra các chế độ 1, 2 và 3 rất giống nhau. Các khác biệt ở tốc độ baud (cố định trong chế độ 2 và thay đổi trong chế độ 1 và 3) và ở số bit dữ liệu (8 trong chế độ 1, 9 trong chế đồ và 3).

4.Khởi động và truy xuất các thanh ghi cổng nối tiếp:

a) Cho phép thu:

Bit cho phép bộ thu (REN = Receiver Enable) trong SCON phải được đặt lên 1 bằng phần mềm để cho phép thu các ký tự. Thông thường thực hiện phần này ở đầu chương trình khi khởi động cổng nối tiếp, timer, … Có thể thực hiện việc này theo hai cách. Lệnh:

```
SETB REN
```

Sẽ đặt lệnh REN 1, hoặc lệnh:

```
MOV SCON, #xxx1xxxxB
```

Sẽ đặt REN lên 1 và đặt hoặc xoá các bit khác trong SCON khi cần (các X phải là 0 hoặc 2 để đặt chế độ làm việc.

b) Bit dữ liệu thứ 9:

Bit dữ liệu thứ 9 cần phát trong chế độ 2 và 3 phải được nạp trong TB8 bằng phần mềm. Bit dữ liệu thứ 9 thu được đặt ở RB8. Phần mềm có thể cần hoặc không cần bit thứ 9, phụ thuộc vào đặt tính kỹ thuật của thiết bị nối tiếp sử dụng. (Bit dữ liệu thứ 9 cũng đóng một vai trò quan trọng trong truyền thông đa xử lý).

c) Thêm 1 bit parity:

Thường sử dụng bit dữ liệu thứ 9 để thêm parity vào ký tự. Bit P trong từ trạng thái chương trình (PSW) được đặt lên 1 hoặc bị xoá mỗi chu kỳ máy để thiết lập kiểm tra chẵn với 8 bit trong thanh ghi tích luỹ. Ví dụ: nếu truyền thông cần 8 bit dữ liệu cộng thêm kiểm tra chẵn, có thể sử dụng các lệnh sau để phát 8 bit trong thanh ghi tích luỹ với kiểm tra chẵn thêm vào bit thứ 9:

```
MOV   C, P          : Đặt bit parity chẵn vào TB8.
MOV   TB8, C        : nó trở thành bit dữ liệu thứ 9.
MOV   SBUF, A       : Chuyển 8 bit từ ACC vào SBUF.
```

Nếu cần parity lẻ thì sử lại các lệnh sau:

```
MOV   C, P          : Đặt bit parity chẵn vào cờ C.
CPL   C             : Đổi sang parity lẻ.
MOV   TB8, C
MOV   SBUF, A
```

Dĩ nhiên việc sử dụng parity không giới hạn ở các chế độ 2 và 3. Ở chế độ 1, 8 bit dữ liệu được truyền đi có thể bao gồm 7 bit dữ liệu cộng thêm 1 bit parity. Để truyền mã ASCII 7 bit với parity chẳn ở bit 8, có thể sử dụng các lệnh sau:

```
CLR   ACC, 7          : Bảo đảm MSB được xoá.
MOV   C,P             : parity chẳn ở trong P.
MOV   ACC, 7, C       : Đặt bit parity chẳn vào MSB.
MOV   SBUF, A         : Gởi ký tự đi.
                      : 7 bit dữ liệu cộng parity chẳn.
```

d) <u>Các cờ ngắt:</u>

Hai cờ ngắt thu và phát (R1 và T1) trong SCON đóng một vai trò quan trọng trong truyền thông nối tiếp dùng 8951. Cả hai bit được đặt lên 1bằng phần cứng, nhưng phải được xoá bằng phần mềm.

Ví dụ, thường R1 được đặt lê 1 khi kết thúc việc thu ký tự và báo "bộ đệm thu tràn". Điều kiện này có thể kiểm tra trong phần mềm hoặc có thể lập trình để xảy ra một ngắt. Nếu phần mềm muốn nhập một ký tự từ thiết bị được nối vào cổng nối tiếp (có thể là thiết bị đầu cuối hiển thị video), nó phải đợi đến khi R1 được đặt lên 1, rồi xoá R1 và đọc ký tự từ SBUF.

Chương trình như sau:


```
WAIT :   JNB   R1, WAIT    :Kiểm tra R1 đến khi nó bằng 1.
         CLR   R1          : Xoá R1.
         MOV   A, SBUF     : Đọc ký tự
```

T1 được đặt lên 1 ở cuối lúc phát ký tự và báo "bộ đệm thu tràn". Nếu phần mềm muốn gởi đi một ký tự đến một thiết bị được nối vào cổng nối tiếp, trước hết nó kiểm tra cổng nối tiếp đã sẳn sàng chưa. Nói cách khác, nếu ký tự trước đã gởi đi, đợi đến khi việc truyền dữ liệu hoàn tất trước khi gởi ký tự kế. Các lệnh sau sẽ truyền ký tự trong thanh ghi tích luỹ:

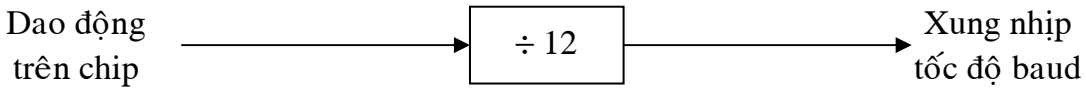

```
WAIT :   JNB   T1, WAIT    :Kiểm tra T1 đến khi nó bằng 1.
         CLR   T1          : Xoá T1.
         MOV   A, SBUF     : Gởi ký tự đi.
```

Các đoạn chương trình trên là một phần cuả các hàm nhập và xuất ký tự chuẩn

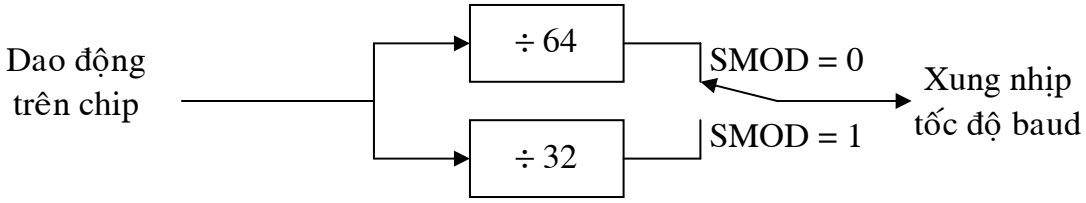
<u>5.Tốc độ baud port nối tiếp:</u>

Như đã nói tốc độ baud cố định ở chế độ 0 và 2. Trong chế độ 0 nó luôn luôn là tần số dao động trên chip được chia cho 12. Thông thường thạch anh ấn định

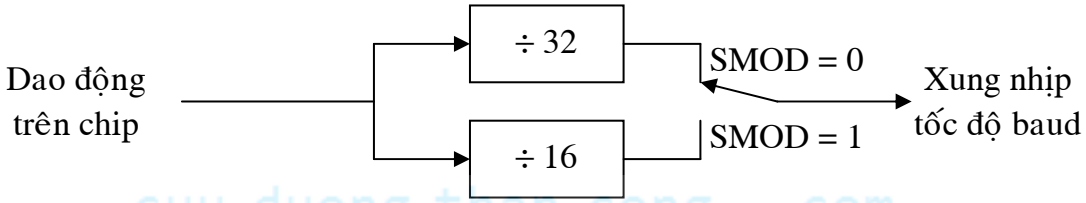
tần số dao động trên chip của 8951, nhưng cũng có thể sử dụng nguồn xung nhịp khác. Giả sử với tần số danh định là 12MHz, thì tốc độ baud chế độ 0 là 1MHz.



a) chế độ 0



b) chế độ 2



c) chế độ 1 và 3

Các nguồn tạo xung nhịp cho port nối tiếp

Mặc nhiên sau khi reset hệ thống, tốc độ baud chế độ 2 là tần số dao động chia co 64. Tốc độ baud cũng ảnh hưởng bởi một bit trong thanh ghi điều khiển nguồn cung cấp (PCON). Bit 7 cuả PCON là bit SMOD. Đặt bit SMOD lên 1 làm gấp đôi tốc độ baud trong các chế độ 1, 2 và 3. Trong chế độ 2 tốc độ baud có thể bị gấp đôi từ giá trị mặc nhiên của 1/64 tần số dao động (SMOD = 0) đến 1/32 tần số dao động (SMOD = 1).

Vì PCON không được định địa chỉ theo bit, nên để đặt bit SMOD lên 1 cần phải theo các lệnh sau:


```
MOV   A, PCON        : Lấy giá trị hiện thời của PCON.
SETB  ACC. 7         : Đặt bit 7 (SMOD) lên 1.
MOV   PCON, A        : Ghi giá trị ngược về PCON.
```

Các tốc độ baud trong các chế độ 1 và 3 được xác định bằng tốc độ tràn của timer 1. Vì timer hoạt động ở tần số tương đối cao, tràn timer được chia thêm cho 32 (16 nếu SMOD = 1) trước khi cấp xung nhịp tốc độ baud cho port nối tiếp.

- **Sử dụng timer 1 làm xung hnng nhip tốc độ baud :**

Cách thông dụng để tạo tốc độ baud là khởi động TMOD cho chế độ 8 bit tự động nạp lại (chế độ 2) và đặt giá trị nạp lại đúng vào TH1 để cho tốc độ tràn đúng với tốc độ baud. TMOD được khởi động như sau:

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

Các x là bit 1 hoặc 0 cần cho timer.

Cũng có thể đạt được tốc độ baud thấp bằng cách sử dụng timer chế độ 1 với TMOD = 0001xxxxB, Tuy nhiên, tốn thêm phần mềm vì các thanh ghi TH1/TL1 phải được khởi động lại sau mỗi lần tràn. Việc này sẽ được thực hiện trong chương trình ngắt. Một chọn lựa khác là cấp xung nhịp cho timer 1 từ ngoài dùng T1 (P3.5). Và luôn luôn tốc độ baud là tốc độ tràn cuả timer 1 được chia cho 32 (hoặc 16, nếu SMOD = 1).

Công thức tổng quát để xác định tốc độ baud trong chế độ 1 và 3 là:

Tốc độ baud = Tốc độ tràn cuả timer 1 ÷ 32.

Ví dụ, muốn làm việc với tốc độ baud là 1200 baud, thì tốc độ tràn cuả timer phải là:

1200 x 32 = 38,4 KHz.

Nếu dùng thạch anh 12MHz. Timer được cấp xung nhịp 1MHz, hay 1000KHz, thì cần tràn sau 1000 ÷ 38,4 = 26,04 xung nhịp (làm tròn là 26). Vì timer đếm lên và tràn xảy ra khi có sự thay đổi từ FFH xuống 00H ở số đếm. Như vậy giá tri đúng cần nạp cho TH1 là 26. Cách dễ nhất để đặt giá trị nạp lại vào TH1 là:

```
MOV    TH1, # - 16
```

Trình hợp dịch sẽ thực hiện chuyển đổi cần thiết. Trong trường hợp này –26 được chuyển thành 0E6H. Như vậy, lệnh trên hoàn toàn giống với lệnh:

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


Do việc làm tròn nên có sai số nhỏ trong tốc độ baud. Tổng quát thì cho phép dung sai 5% trong truyền thông bất đồng bộ (Start/Stop). Có thể có được tốc độ baud chính xác nếu dùng thạch anh 11,059MHz. Bảng sau đây tóm tắt các giá trị nạp lại cho các tốc độ baud thông dụng nhất, dùng thạch anh 12MHz hoặc 11,059MHz:

Tốc độ baud	Tần số thạch anh	SMOD	Giá tri nạp lải vào TH1	Tốc độ baud thật	Sai số
9600	12,000MHz	1	-7(F9H)	8923	7%
2400	12,000MHz	0	-13(F3H)	2404	0,16%
1200	12,000MHz	0	-26(E6H)	1202	0,16%
19200	11,059MHz	1	-3(FDH)	19200	0
9600	11,059MHz	0	-3(FDH)	9600	0
2400	11,059MHz	0	-12(F4H)	2400	0
1200	11,059MHz	0	-24(E8H)	1200	0

Bảng tóm tắt tốc độ baud.

IV – NGẮT (Interrupt) :

1. Giới thiệu:

Một ngắt là sự xảy ra một điều kiện – một sự kiện – mà nó gây ra treo tạm thời chương trình trong khi điều kiện đó được phục vụ bởi mốt chương trình khác.

Các ngắt đóng một vai trò quan trọng trong thiết kế và cài đặt các ứng dụng vi điều khiển. Chúng cho phép hệ thống đáp ứng bất đồng bộ với một sự kiện và giải quyết sự kiện đó trong khi một chương trình khác đang thực thi.

Một hệ thống điều khiển bằng ngắt cho ảo giác là làm nhiều việc đồng thời. Dĩ nhiên CPU mỗi lần không thể thực thi hơn một lệnh. Nhưng nó có thể tạm treo việc thực thi một chương trình để thực thi một chương trình khác, rồi quay về chương trình thứ nhất. Theo cách này nó giống như một chương trình con, nhưng có sự khác biệt trong hệ thống được điều khiển bằng ngắt là sự ngắt quảng không xảy ra như kết quả của một lệnh (lệnh CALL subroutine), mà là đáp ứng với một sự kiện xảy ra bất đồng bộ với chương trình chính. Người ta không biết khi nào và ở đâu chương trình chính sẽ bị ngắt quảng.

Chương trình giải quyết ngắt được gọi là chương trình phục vụ ngắt (ISR: Interrupt Service Routine) hoặc bộ sử lý ngắt. ISR thực thi khi đáp ứng ngắt và thông thường thực hiện tác vụ nhập hay xuất với một thiết bị. Khi ngắt xảy ra chương trình chính tạm thời bị treo và rẽ nhánh đến ISR: ISR thực thi và kết thúc bằng lệnh trở về từ ngắt, chương trình chính tiếp tục thực thi chổ mà nó tạm dừng. Thường người ta xem chương trình chính như thực thi ở mức nền (cơ sở) (base –level) và các ISR thực thi ở mức ngắt (interrupt – level). Người ta cũng dùng các thuật ngữ foreground chỉ mức nền và background chỉ mức ngắt.

2. Tổ chức ngắt của 8951:

8951 có 5 nguồn ngắt : hai ngắt ngoài, hai ngắt Timer và một ngắt cổng nối tiếp. Tất cả các ngắt theo mặc nhiên đều bị cấm sau khi reset hệ thống và được cho phép từng cái một bằng phần mềm.

Khi có hai hoặc nhiều ngắt đồng thời, hoặc một ngắt xảy ra khi một ngắt khác đang phục vụ, có cả hai sự tuần tự hỏi vòng và sơ đồ ưu tiên hai mức dùng để xác định việc thực hiện các ngắt. Việc hỏi vòng tuần tự thì cố định như ưu tiên ngắt thì có thể lập trình được.

a) Cho phép và cấm các ngắt :

Mỗi nguồn ngắt được cho phép hoặc cấm từng ngắt một qua thanh ghi chức năng đặc biệt có định địa chỉ bit IE (Interrupt Enable: cho phép ngắt) ở địa chỉ A8H. Cũng như các bit cho phép riêng biệt cho mỗi nguồn ngắt, có một bit cho phép ngắt/cấm toàn bộ được xoá để cấm tất cả các ngắt hoặc được đặt lên 1để cho phép tất cả các ngắt.

Bit	Ký hiệu	Địa chỉ	Mô tả
IE.7	EA	AFH	Cho phép / cấm toàn cục
IE.6	_	AEH	Không dùng
IE.5	ET2	ADH	Cho phép ngắt từ Tiner 2
IE.4	ES	ACH	Cho phép ngắt cổng nối tiếp
IE.3	ET1	ABH	Cho phép ngắt timer 1
IE.2	EX1	AAH	Cho phép ngắt ngoài 1
IE.1	ET0	A9H	Cho phép ngắt timer 0
IE.0	EX0	A8H	Cho phép ngắt ngoài 0

Tóm tắt thanh ghi IE

Hai bit được đặt lên 1 để cho phép bất kỳ ngắt nào: bit cho phép riêng và bit cho phép toàn bộ. Ví dụ, các ngắt từ timer 1 dược cho phép như sau:

```
SETB    ET1          : Cho phép ngắt từ Timer 1
SETB    EA           : Đặt bit cho phép toàn bộ
```

Cũng có thể sử dụng lệnh sau:

```
MOV    IE, #10001000B
```


Mặc dù hai cách nàycó cùng hiệu quả sau khi reset hệ thống, nhưng hiệu quả sẽ khác đến IE được ghi ở chương trình. Cách thứ nhất không ảnh hưởng đến 5 bit khác trong thanh ghi IE, trái lại cách thứ hai sẽ xoá các bit khác. Nên khởi trị IE theo cách thứ hai ở đầu chương trình (nghiã là sau khi mở máy hoặc reset hệ thống), nhưng cho phép cấm các ngắt ngay trong chương trình nên dùng cách thứ nhất để tránh ảnh hưởng đến các bit khác trong thanh ghi IE.

b) Quyền ưu tiên của ngắt :

Có hai mức độ ưu tiên ngắt, mỗi ngắt được lập trình riêng cho một trong hai mức ưu tiên qua các bit được địa chỉ trong thanh ghi IP (Interrupt Priority) ở địa chỉ 0B8H. Thanh ghi IP đặt tất cả các ngắt về mức ưu tiên thấp nhất khi reset hệ thống. Nếu có ngắt ưu tiên cao hơn ngắt đang được phục vụ thì chương trình con phục vụ ngắt sẽ phục vụ cho mức ngắt ưu tiên này.

Bảng tóm tắt thanh ghi IP :

Bit	Ký hiệu	Địa chỉ bit	Mô tả
IP.7	-	-	Không dùng
IP.6	-	-	Không dùng
IP.5	PT2	BDH	Không dùng
IP.4	PS	BCH	Ưu tiên cho ngắt cổng nối tiếp
IP.3	PT1	BBH	Ưu tiên cho ngắt timer1
IP.2	PX1	BAH	Ưu tiên cho ngắt ngoài 1
IP.1	PT0	B9H	Ưu tiên cho ngắt timer 0
IP.0	PX0	B8H	Ưu tiên cho ngắt ngoài 0

Khi một ngắt được gọi chương trình phục vụ ngắt sẽ thi hành. Nếu chương trình phục vụ ngắt ưu tiên thấp đang thi hành mà có một ngắt ưu tiên cao được gọi thì CPU sẽ thi hành chương trình hoạt động cho ngắt ưu tiên cao .

c) hỏi vòng tuần tự:

Nếu có hai ngắt cùng mức ưu tiên, thì tuần tự hỏi vòng sẽ xác định cho ngắt nào ưu tiên. Tuần tự là external 0, timer 0, external 1, timer 1, serial port, timer 2. Trạng thái của tất cả các mức ngắt được cho phép bởi từng bit riêng trong SFR. Dĩ nhiên, nếu ngắt bất kỳ không cho phép ngắt xảy ra, nhưng phần mền có thể kiểm tra lại cờ ngắt. Bộ timer và serial port sử dụng cờ ngắt thông dụng mà không cần dùng ngắt thật sự.

Các bit cờ ngắt :

Ngắt	Cờ	Thanh ghi chức năng và vị trí bit
Ngoài 0	IE0	TCON.1
Ngoài 1	IE1	TCON.3
Timer 1	TF1	TCON.7
Timer 0	TFO	TCON.5
Port nối tiếp	T1	SCON.1
Port nối tiếp	R1	SCON.0

3. Xử lý ngắt:

khi ngắt xảy ra và được CPU chấp nhận, chương trình chính bị ngắt quảng. Những hoạt động sau xảy ra:

- Lệnh hiện hành hoàn tất việc thực thi.
- Cất PC vào ngăn xếp.

- Trạng thái ngắt hiện hành được cất bên trong.
- Các ngắt bị chặn ở mức ngắt.
- Nạp vào PC địa chỉ vector cuả ISR.
- ISR thực thi.

ISR thực thi và đáp ứng ngắt. ISR hoàn tất bằng lệnh RET1 (quay về từ ngắt). Điều này làm lấy lại giá trị cũ cuả PC từ ngăn xếp và lấy lại trạng thái ngắt cũ. Thực thi chương trình chính tiếp tục ở chổ mà nó bị dừng.

*** Các vetor ngắt :**

Vetor ngắt là địa chỉ được vào trong PC để thi hành chương trình con phục vụ ngắt khi ngắt được gọi

Bảng vector ngắt :

Loại ngắt	Flag	Địa chỉ voter ngắt
Khởi động hệ thống	RST	0000H
Ngoài 0	IE0	0003H
Timer 0	TF0	000BH
Ngoài 1	IE1	0013H
Timer 1	TF1	001BH
Cổng nối tiếp	R1 hoặc T1	0023H

Khi xảy ra ngắt, cờ ngắt được xóa tự động bằng phần cứng ngoại trừ R1 và T1 của cổng nối tiếp, TF2 và EXF2 của Timer 2. Vì có hai ngắt này nên CPU không thể thực hiện thao tác xóa cờ ngắt. Các bit này phải được kiểm tra trong ISK để xác định ngắt nào, sau đó cờ ngắt được xóa bằng phần mềm. Thông thường rẽ nhánh chương trình phụ thuộc vào việc rẽ nhánh thích hợp.

4. Các ngắt của 8951:

a) Các ngắt timer:

Các ngắt timer có địa chỉ vecter ngắt là 000BH (Timer 0) và 001BH (Timer1). Ngắt timer xảy ra khi các thanh gji timer (TLx/THx) tràn và set cờ báo tràn (TFx) lên 1.

Chú ý rằng các cờ timer (TFx) không bị xoá bằng phần mềm. Khi cho phép các ngắt, TFx tự động bị xoá bằng phần cứng khi CPU chuyển đến ngắt.

b) Các ngắt cổng nối tiếp:

Ngắt cổng nối tiếp xảy ra khi hoặc cờ ngắt phát (T1) hoặc cờ ngắt thu (R1) được đặt lên 1. Ngắt phải xảy ra khi truyền một ký tự vừa được ghi vào SBUF. Ngắt thu xảy ra khi một ký tự đã được nhận xong và đang đợi trong SBUF để được đọc.

Ngắt cổng nối tiếp hơi khác với ngắt timer. Cờ gây ra ngắt cổng nối tiếp không bị xoá bằng phần cứng khi CPU chuyển tới ngắt . Nguyên do là có hai nguồn ngắt cổng nối tiếp: T1 và R1. Nguồn ngắt phải được xác định trong ISR và cờ tạo V sẽ được xoá bằng phần mềm.

c) Các ngắt ngoài:

Các ngắt ngoaì xảy ra khi có một mức thấp hoặc cạnh xuống trên chân INT0 hoặc INT1 của 8951. Đây là các chức năng chuyển đổi của các bit port3 : P3.2(chân 12) và P3.3(chân 13).

Các cờ tạo ngắt này là các bit IE0 và IE1 trong TCON. KKhi quyền điều khiển đã chuyển đến ISR, cờ tạo ra ngắt chỉ được xoá nếu ngắt được tích cực bằng cạnh xuống. Nếu ngắt được tích cực theo mức, thì nguồn yêu cầu ngắt bên ngoài sẽ điều khiển mức của cờ thay cho phần cứng.

Sự lựa chọn ngắt tích cực mức thấp hay tích cực cạnh xuống được lập trình qua các bit IT0 và IT1 trong TCON. Ví dụ, nếu IT1 = 0, ngắt ngoài 1 được kích khởi bởi mức thấp ở chân IT1. Nếu IT1 = 1, ngắt ngoài sẽ được kích khởi bằng cạnh. Trong chế độ này nếu mẫu liên tiếp trên chân IT1 chỉ mức cao trong một chu kỳ và thấp trong chu kỳ kế, cờ yêu cầu ngắt IE1 trong TCON được đặt lên 1. Rối bit cờ IE1 yêu cầu ngắt.

Vì các chân ngắt ngoài được lấy mẫu một lần ở mỗi chu kỳ máy, ngõ vào nên được giữ trong tối thiểu 12 chu kỳ dao động để bảo đảm lấy mẫu đúng. Nếu ngắt ngoài được tác động theo cạnh xuống, nguồn bên ngoài phải giữ chân yêu cầu cao tối thiểu 1 chu kỳ và giữ nó ở mức thấp thêm một chu kỳ nữa để đảm bảo phát hiện được cạnh xuống. IE0 và IE1 tự động được xoá khi CPU chuyển tới ngắt.

Nếu ngắt ngoài được tác động theo mức, nguồn bên ngoài phải giữ yêu cầu tác động cho đến khi ngắt được yêu cầu thật sự được tạo ra. Rồi nó không tác động yêu cầu trước khi ISR được hoàn tất, nếu không một ngắt khác sẽ được lập lại. Thông thường khi vào ISR người ta làm nguồn yêu cầu đưa tín hiệu tạo ngắt về trạng thái không tác động.

V - TẬP LỆNH CỦA 8051 (8951) :

Các kiểu lệnh (Instruction types) :

8051 chia ra 5 nhóm lệnh chính :

Các lệnh số học.
Lệnh logic.
Dịch chuyển dữ liệu.
Lý luận.
Rẽ nhánh chương trình.

Từng kiểu lệnh được mô tả như sau :

1. Các lệnh số học (Arithmetic Instruction) :

```
ADD     A, Rn           : (A) ← (A) + (Rn)
ADD     A, direct       : (A) ← (A) + (direct)
ADD     A, @ Ri         : (A) ← (A) + ((Ri))
ADD     A, # data       : (A) ← (A) + # data

ADDC    A, Rn       : (A) ← (A) + (C) + (Rn)
ADDC    A, direct   : (A) ← (A) + (C) + (direct)
ADDC    A, @ Ri     : (A) ← (A) + (C) +((Ri))
ADDC    A, # data   : (A) ← (A) + (C) + # data

SUBB    A, < src, byte >
SUBB    A, Rn           : (A) ← (A) - (C) - (Rn)
SUBB    A, direct       : (A) ← (A) - (C) - (direct)
SUBB    A, @ Ri         : (A) ← (A) - (C) - ((Ri))
SUBB    A, # data   : (A) ← (A) - (C) - # data

INC < byte >
INC     A               : (A) ← (A) + 1
INC     direct      : (direct) ← (direct) + 1
INC     @ Ri        : ((Ri)) ← ((Ri)) +1
INC     Rn          : ((Rn)) ← ((Rn)) +1
INC     DPTR        : (DPTR) ← (DPTR) +1

DEC < byte >
DEC     A           : (A) ← (A) -1
```

DEC direct : (direct) ← (direct) -1
DEC Ri : (Ri) ← (Ri) -1
DEC Rn : (Rn) ← (Rn) –1

MUL AB : (A) ← LOW {(A) x (B)}; có ảnh hưởng cờ OV
(B) ← HIGH {(A) x(B)}; cờ Carry được xóa

DIV AB : (A) ← Integer Result of {(A)/(B)}; cờ Carry xóa

DA A : Điều chỉnh thanh ghi A thành số BCD đúng trong phép cộng BCD (thường DA A đi kèm với : ADD, ADDC).

Nếu {(A3 – A0) > 9} và {(AC) =1}→ (A3 ÷ A0) ← (A1 ÷ A0) + 6
Nếu {(A7 – A4) > 9} và {(AC) =1}→ (A3 ÷ A0) ← (A7 ÷ A4) + 6

2. Các hoạt động logic (Logic Operation) :

Tất cả các lệnh logic sử dụng thanh ghi A như là một trong những toán hạng thực thi một chu kỳ máy, ngoài A ra mất hai chu kỳ mày. Những hoạt động logic có thể được thực hiện trên bất kỳ byte nào trong vị trí nhớ dữ liệu nội mà không qua thanh ghi A.

Các hoạt động logic được tóm tắt như sau :

ALN <**dest** - byte> <**srs**- byte>
ALN A, Rn : (A) ← (A) AND (Rn)
ALN A, direct : (A) ← (A) AND (direct)
ALN A, @Ri : (A) ← (A) AND (Ri)
ALN A, # data : (A) ← (A) AND (# data)
ALN direct, A : (direct) ← (direct) AND (A)
ALN direct, # data : (direct) ← (direct) AND (# data)

ORL <**dest** - byte> <**srs** - byte>
ORL A, Rn : (A) ← (A) OR (Rn)
ORL A, direct : (A) ← (A) OR (direct)
ORL A, @Ri : (A) ← (A) OR (Ri)
ORL A, # data : (A) ← (A) OR (# data)
ORL direct, A : (direct) ← (direct) OR (A)
ORL direct, # data : (direct) ← (direct) OR (# data)

XLR <**dest** - byte> <**srs** - byte>

XLR A, Rn : (A) ← (A) ⊕ (Rn)
XLR A, direct : (A) ← (A) ⊕ (direct)
XLR A, @Ri : (A) ← (A) ⊕ (Ri)
XLR A, # data : (A) ← (A) ⊕ (# data)
XLR direct, A : (direct) ← (direct) ⊕ (A)
XLR direct, # data : (direct) ← (direct) ⊕ (# data)

CLR A : (A) ← 0
CLR C : (C) ← 0
CLR Bit : (Bit) ← 0

RL A : quay vòng thanh ghi A qua trái 1 bit
(An + 1) ← (An) ; n = 0 ÷ 6
(A0) ← (A7)

RLC A : quay vòng thanh ghi A qua trái 1 bit có cờ carry
(An + 1) ← (An) ; n = 0 ÷ 6
(C) ← (A7)
(A0) ← (C)

RRA : quay vòng thanh ghi A qua phải 1 bit
(An) ← (An + 1) ; n = 0 ÷ 6
(A0) ← (A7)

RLC A : quay vòng thanh ghi A qua trái 1 bit có cờ carry
(An) ← (An + 1) ; n = 0 ÷ 6
(C) ← (A7)
(A7) ← (C)

SWAP A : đổi chỗ 4 bit thấp và 4 bit cao của A
cho nhau (A0 ÷ A3) ↔ (A7 ÷ A4)

3. Các lệnh rẽ nhánh chương trình (lệnh nhảy) :

Có nhiều lệnh để điều khiển chương trình bao gồm việc gọi hoặc trả lại từ chương trình con hoặc chia nhánh có điều kiện hay không có điều kiện.

Tất cả các lệnh rẽ nhánh đều không ảnh hưởng đến cờ. Ta có thể định nhãn cần nhảy tới mà không cần định rõ địa chỉ , trình biên dịch sẽ đặt lại địa chỉ nơi cần nhảy tới vào đúng lệnh đã đưa ra.

Sau đây là tóm tắt hoạt động của lệnh nhảy :

JC rel : nhảy đến "rel" nếu carry set.

JNC rel : nhảy đến "rel" nếu carry = 0.
JB bit,rel : nhảy đến "rel" nếu bit set.
JNB bit,rel : nhảy đến "rel" nếu bit = 0.
JBC bit,rel : nhảy đến "rel" nếu bit set và xóa bit.

ACALL addr11 : lệnh gọi tuyệt đối trong page 2K
(PC) ← (PC) + 2
(SP) ← (SP) + 1
((SP)) ← (PC7 ÷ PC0)
(SP) ← (SP) + 1
((SP)) ← (PC15 ÷ PC8)
(PC10 ÷ PC0) ← page Address.

LCALL addr16 : lệnh gọi dài chương trình con trong 64K
(PC) ← (PC) + 3
(SP) ← (SP) + 1
((SP)) ← (PC7 ÷ PC0)
(SP) ← (SP) + 1
((SP)) ← (PC15 ÷ PC8)
(PC) ← Addr ÷ Addr0.

RET : kết thúc chương trình con và quay về chương trình chính
(PC15 ÷ PC8) ← ((SP))
(SP) ← (SP) – 1
(PC7 ÷ PC0) ← ((SP))
(SP) ← (SP) – 1
RETI :kết thúc thủ tục phục vụ ngắt quay về chương trình chính, hoạt động tương tự RET.

AJMP Addr11:Nhảy tuyệt đối không điều kiện trong 2K.
(PC) ← (PC) + 2
(PC10 ÷ PC0) ← page Address
AJMP Addr16: Nhảy dài không điều kiện trong 64K.
(PC) ← (PC) + 2
(PC15 ÷ PC0) ← page Address
SJMP rel :nhảy ngắn không điều kiện trong -128 ÷ +127 byte
(PC) ← (PC) + 2
(PC) ← (PC) + byte 2
JMP @A + DPTR :nhảy không điều kiện đến add (A) + (DPTR)
(PC) ← (A) + (DPTR)
JZ rel :nhảy đến A=0. Thực hành lệnh kế nếu A # 0

(PC) ← (PC) + 2
(A) = 0 → (PC) ← (PC) + 2

JNZ rel :nhảy đến A # 0. Thực hành lệnh kế nếu A = 0
(PC) ← (PC) + 2
(A) # 0 → (PC) ← (PC) + 2byte

CJNE A,direct,rel :so sánh và nhảy đến rel nếu A # direct
(PC) ← (PC) + 3
(A) < > (direct) → (PC) ← (PC) + Relative Add
(A) < (direct) → C = 1
(A) > (direct) → C = 0
(A)= direct. Thực hành lệnh kế tiếp

CJNE A,#data,rel :tương tự lệnh CJNE A,direct,rel.
CJNE Rn,#data,rel :tương tự lệnh CJNE A,direct,rel.
CJNE @Ri,#data,rel :tương tự lệnh CJNE A,direct,rel.

DJNZ Rn,rel :giảm Rn và nhảy nếu Rn # 0
(PC) ← (PC) + 2
(Rn) ← (Rn) – 1
(Rn) < > 0 → (PC) ← (PC) + byte 2
DJNZ direct,rel :tương tự lệnh DJNZ Rn,rel.

4. Các lệnh dịch chuyển dữ liệu :

Các lệnh dịch chuyển dữ liệu trong những vùng nhớ nội thực thi 1 hoặc 2 chu kỳ máy. Mẫu lệnh MOV <des> <source> cho phép di chuyển dữ liệu bất kỳ 2 vùng nhớ nào của RAM nội hoặc các vùng nhớ của các thanh ghi chức năng đặc biệt mà không thông qua thanh ghi A.

Vùng stack của 8951 chỉ chứa 128 byte RAM nôi, nếu con trỏ Stack SP được tăng quá địa chỉ 7FH thì các byte được PUSH vào sẽ mất đi và các byte POP ra sẽ không biết rõ.

Các lệnh dịch chuyển bộ nhớ nội và bộ nhớ ngoại dùng sự định vị gián tiếp. Địa chỉ gián tiếp có thể dùng địa chỉ 1 byte (@Ri) hoặc địa chỉ 2 byte (@DPTR). Tất cả các lệnh dịch chuyển hoạt động trên bộ nhớ ngoài thực thi trong hai chu kỳ máy và dùng thanh ghi A làm toán hạng SOURCE hoặc toán hạng DESTINATION.

Việc đọc và ghi RAM ngoài (RD và WR) chỉ tích cực trong suốt quá trình thực thi của lệnh MOVX, còn bình thường thì RD và WR không tích cực (mức 1).

Tất cả các lệnh dịch chuyển đều không ảnh hưởng đến cờ. Hoạt động của từng lệnh được tóm tắt như sau :

```
MOV     A,Rn            : (A) ← (Rn)
MOV     A,direct        : (A) ← (direct)
MOV     A,@Ri           : (A) ← ((Ri))
MOV     A,#data         : (A) ← #data
MOV     Rn,A            : (Rn) ← (A)
MOV     Rn,direct       : (Rn) ← (direct)
MOV     Rn,#data        : (Rn) ← #data
MOV     direct,A        : (direct) ← (A)
MOV     direct,Rn       : (direct) ← (Rn)
MOV     direct,direct   : (direct) ← (direct)
MOV     direct,@Ri      : (direct) ← ((Ri))
MOV     direct,#data    : (direct) ← #data
MOV     @Ri,A           : ((Ri)) ← (A)
MOV     @Ri,direct      : ((Ri)) ← (direct)
MOV     @Ri,#data       : ((Ri)) ← #data
MOV     DPTR,#data16        : (DPTR) ← #data16
MOV     A,@A + DPTR         : (A) ← (A) + (DPTR)
MOV     A,@A + PC           : (PC) ← (PC) + 1
                              (A)← (A) + (PC)
MOVX    A,@Ri           : (A) ← ((Ri))
MOVX    A,@DPTR         : (A) ← ((DPTR))
MOVX    @Ri,A           : ((Ri)) ← (A)
MOVX    @DPTR,A         : ((DPTR)) ← (A)

PUSH            direct          : cất dữ liệu vào stack
                                  (SP) ← (SP) + 1
                                  (SP) ← (direct)

POP             direct          : lấy dữ liệu ra khỏi stack
                                  (direct) ← (SP)
                                  (SP) ← (SP) - 1

XCH             A,Rn            :đổi chỗ nội dung thanh ghi A với Rn
                                  A ⇔ (Rn)
XCH             A,direct        : (A) ⇔ (direct)
XCH             A,@Ri           : (A) ⇔ ((Ri))
XCHD    A,@Ri           : đổi chỗ 4 bit thấp của A với ((Ri))
                          (A3 ÷ A0) ⇔ ((Ri3 ÷ Ri0))
```

5. Các lệnh luận lý (Boolean Instruction) :

8951 chứa một bộ xử lý luận lý đầy đủ cho các hoạt động đơn bit, đây là một điểm mạnh của họ vi điều khiển MSC-51 mà các họ khác không có.

RAM nội chứa 128 bit định vị và vùng nhớ các thanh ghi chức năng đặc biệt lên đến 128 bit định vị khác. Tất cả các đường port là bit định vị, mỗi đường có thể xử lý như một port đơn bit riêng biệt. Các truy xuất các bit này không chỉ là các lệnh rẽ nhánh không, mà là một danh mục đầy đủ các lệnh MOVE, SET, CLEAR, COMPLEMENT, OR, AND.

Toàn bộ sự truy xuất của bit dùng sự định vị trực tiếp với những địa chỉ từ 00H ÷ 7FH trong 128 vùng nhớ thấp và 80H ÷ FFH ở vùng các thanh ghi chức năng đặc biệt.

Bit Carry C trong thanh ghi PSW của từ trạng thái chương trình được dùng như một sự tích lũy đơn của bộ xử lý luận lý. Bit carry cũng là một bit định vị và có địa chỉ trực tiếp vì nó nằm trong PSW. Hai lệnh CLR C và CLR CY đều có cùng tác dụng là xóa cờ Carry nhưng lệnh đầu mất 1 byte còn lệnh sau mất 2 byte.

Hoạt động của các lệnh luận lý được tóm tắt như sau :

CLR C : xóa cờ carry xuống 0, có ảnh hưởng cờ Carry
CLR BIT : xóa bit xuống 0, không ảnh hưởng cờ Carry

SET C : set cờ carry lên 1, có ảnh hưởng cờ Carry
SET BIT : set bit lên 1, không ảnh hưởng cờ Carry

CPL C : đảo bit cờ carry, có ảnh hưởng cờ carry
CPL BIT : đảo bit, không ảnh hưởng cờ carry

ALN C,BIT : (C) ← (C) AND (BIT) : có ảnh hưởng cờ Carry
ALN C,/BIT: (C) ← (C) AND NOT (BIT) : có ảnh hưởng cờ Carry
ORL C,BIT : (C) ← (C) OR (BIT) : có ảnh hưởng cờ Carry
ALN C,/BIT: (C) ← (C) OR NOT (BIT) : có ảnh hưởng cờ Carry

MOV C,BIT : (C) ← (BIT) : có ảnh hưởng cờ Carry
MOV BIT,C : (BIT) ← (C) : không ảnh hưởng cờ Carry

6. Các lệnh xen vào (Miscellameous Instruction) :

NOP : không hoạt động gì cả, chỉ tốn 1 byte và một chu kỳ thực thi máy, ta dùng để DELAY những khoảng thời gian nhỏ.

Chú ý :

(A) : nội dung của thanh ghi A.

((Ri)) : nội dung của địa chỉ trong RAM nội mà Ri trỏ đến.

((DPTR)) : nội dung của địa chỉ trong RAM ngoại mà DPTR đang trỏ đến.

Rn : với n = 0 ÷ 7 (tức R0, R1, R2 … : 8 thanh ghi).