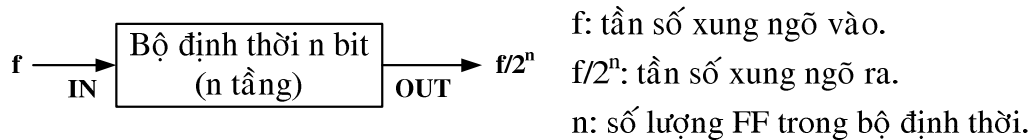


CHƯƠNG 4

HOẠT ĐỘNG CỦA BỘ ĐỊNH THỜI (TIMER)

I. MỞ ĐẦU:

Bộ định thời (TIMER) → Là chuỗi các FF (mỗi FF là 1 mạch chia 2).
 → Ngõ vào: nhận tín hiệu xung clock từ nguồn xung.
 → Ngõ ra: truyền tín hiệu xung clock cho FF báo tràn (cờ tràn).

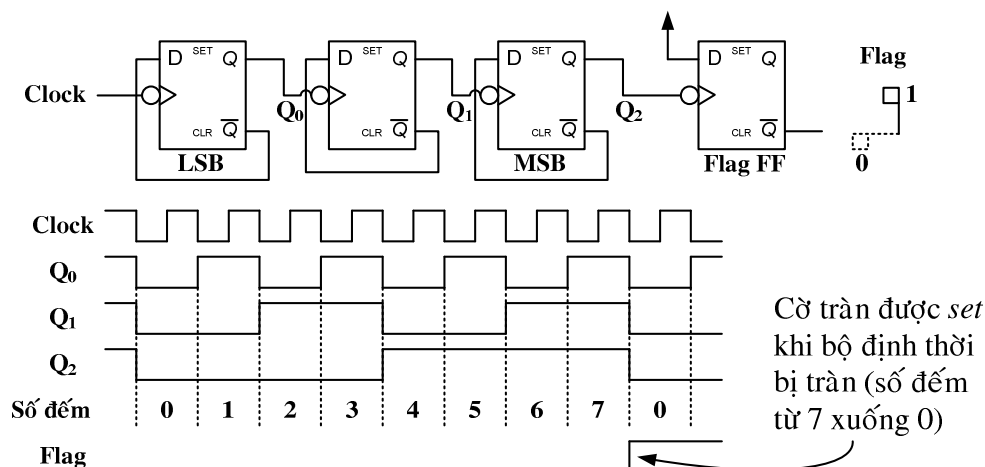


- **Tần số:** tần số xung ngõ ra bằng tần số xung ngõ vào chia cho 2^N .
- **Giá trị:** giá trị nhị phân trong các FF của bộ định thời là số đếm của các xung clock tại ngõ vào từ khi bộ định thời bắt đầu đếm.
- **Tràn:** xảy ra hiện tượng tràn (*cờ tràn = 1*) khi số đếm chuyển từ giá trị lớn nhất xuống giá trị nhỏ nhất của bộ định thời.

Ví dụ: Bộ định thời 16 bit (chứa 16 FF bên trong).

- Tần số: $f_{OUT} = \frac{f_{IN}}{2^{16}} = \frac{f_{IN}}{65536}$
- Giá trị: số đếm nằm trong khoảng 0 (0000H) → 65535 (FFFFH).
- Tràn: cờ tràn bằng 1 khi số đếm từ FFFFH chuyển xuống 0000H.

Hình minh họa đơn giản hoạt động của bộ định thời 3 bit:



Hình 4.1.1: Bộ định thời 3 bit. (a) Sơ đồ logic. (b) Biểu đồ thời gian.

Hoạt động của một bộ định thời 3 bit đơn giản được minh họa trong hình trên. Mỗi một tầng là D FF kích khởi cạnh âm hoạt động như một mạch chia 2 do ta nối ngõ ra $\bar{Q}$ với ngõ vào D. Flipflop cờ (Flag FF) là một mạch chốt D được set bằng 1 bởi tầng cuối của bộ định thời. Biểu đồ thời gian cho

thấy tầng thứ nhất (Q_0) chia 2 tần số xung clock, tầng thứ hai (Q_1) chia 4 tần số xung clock, ... Số đếm được ghi ở dạng thập phân và được kiểm tra dễ dàng bằng cách khảo sát trạng thái của 3 flipflop. Ví dụ, số đếm là 4 xuất hiện khi $Q_2 = 1, Q_1 = 0, Q_0 = 0$. Các flipflop ở trên là các flipflop tác động cạnh âm (nghĩa là trạng thái của các flipflop sẽ thay đổi theo cạnh âm của xung clock). Khi số đếm tràn từ 111 xuống 000, ngõ ra Q_2 có cạnh âm làm cho trạng thái của flipflop cờ đổi từ 0 lên 1 (ngõ vào D của flipflop này luôn luôn ở logic 1).

Chip 8051 có 2 bộ định thời

- Dùng để định một khoảng thời gian.
- Dùng để đếm sự kiện (đếm số xung).
- Dùng để tạo tốc độ baud cho port nối tiếp.

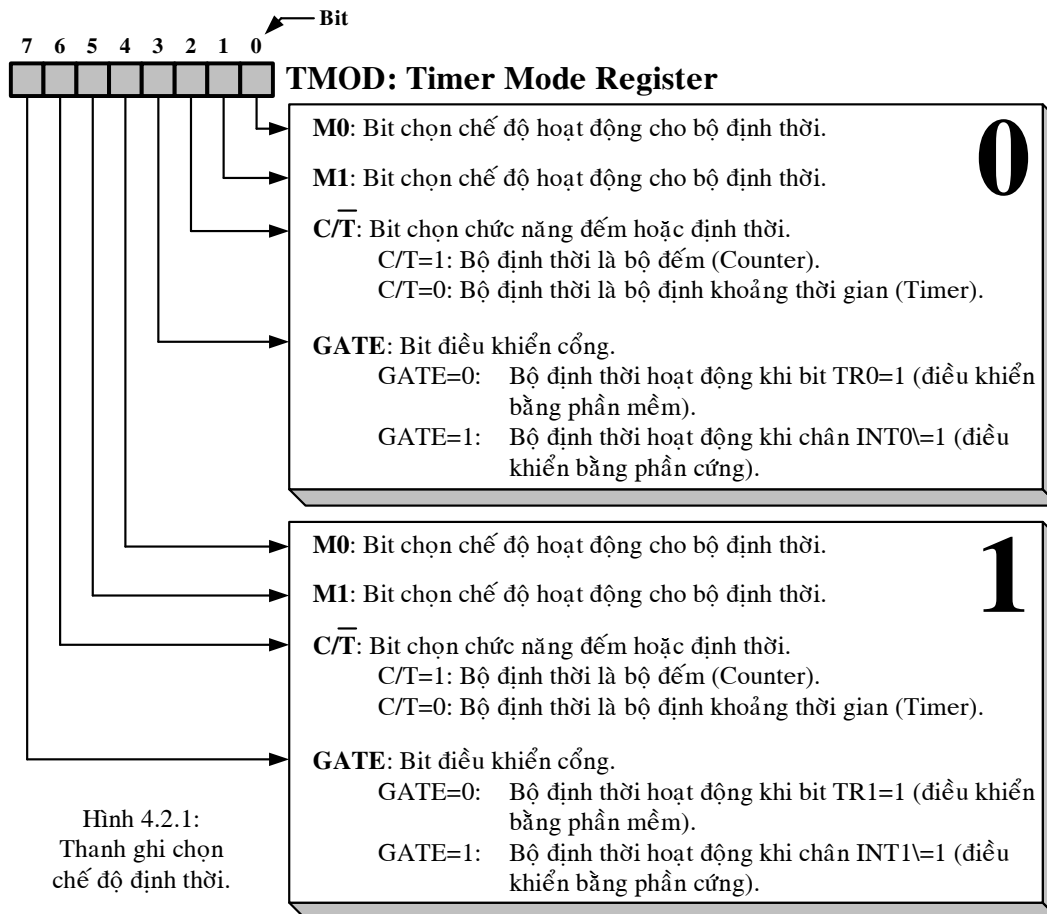
- **Ứng dụng định thời gian (TIMER):** bộ định thời được lập trình sao cho sẽ tràn sau một khoảng thời gian đã qui định và khi đó cờ tràn của bộ định thời sẽ bằng 1.
- **Ứng dụng đếm sự kiện (COUNTER):** để xác định số lần xuất hiện của một kích thích từ bên ngoài tới một chân của chip 8051 (kích thích là sự chuyển trạng thái từ 1 xuống 0).
- **Ứng dụng tạo tốc độ baud cho port nối tiếp:** xem thêm trong chương “Chương 5: Hoạt động port nối tiếp.”.

Thanh ghi chức năng đặc biệt của bộ định thời

- **TCON:** Điều khiển bộ định thời.
- **TMOD:** Chọn chế độ cho bộ định thời.
- **TL0:** Byte thấp của bộ định thời 0.
- **TL1:** Byte thấp của bộ định thời 1.
- **TH0:** Byte cao của bộ định thời 0.
- **TH1:** Byte cao của bộ định thời 1.

II. THANH GHI CHẾ ĐỘ ĐỊNH THỜI (TMOD):

- Thanh ghi TMOD (Timer Mode Register) chứa các bit dùng để thiết lập chế độ hoạt động cho bộ định thời 0 và bộ định thời 1.
- Thanh ghi TMOD được nạp giá trị một lần tại thời điểm bắt đầu của chương trình để qui định chế độ hoạt động của các bộ định thời.
- Cấu trúc thanh ghi TMOD:



Hình 4.2.1:
Thanh ghi chọn
chế độ định thời.

- Các chế độ hoạt động của bộ định thời:

M1	M0	Chế độ	Mô tả
0	0	→ 0	→ Chế độ định thời 13 bit.
0	1	→ 1	→ Chế độ định thời 16 bit.
1	0	→ 2	→ Chế độ định thời 8 bit tự động nạp lại.
1	1	→ 3	→ Chế độ định thời chia xẻ.

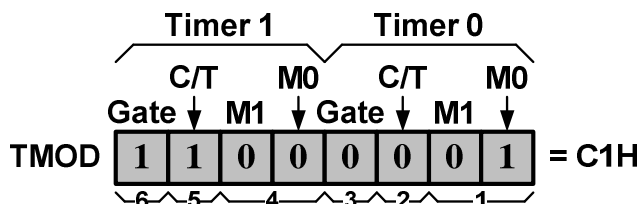
Hình 4.2.2: Các chế độ định thời.

- Ví dụ 1: Cho biết giá trị cần nạp cho thanh ghi TMOD để
 - Timer 0: là bộ định thời gian 16 bit, được điều khiển bằng phần mềm (bit $TR0$).
 - Timer 1: là bộ đếm xung 13 bit, được điều khiển bằng phần cứng (chân $\overline{INT1}$).

Giải

Phân tích:

- | | | |
|---------------------------------|---------------|-----------------------|
| (1): Chế độ 16 bit. | $\Rightarrow$ | $M1 = 0, M0 = 1.$ |
| (2): Bộ định thời gian. | $\Rightarrow$ | $C/\overline{T} = 0.$ |
| (3): Điều khiển bằng phần mềm. | $\Rightarrow$ | $GATE = 0.$ |
| (4): Chế độ 13 bit. | $\Rightarrow$ | $M1 = 0, M0 = 0.$ |
| (5): Bộ đếm xung. | $\Rightarrow$ | $C/\overline{T} = 1.$ |
| (6): Điều khiển bằng phần cứng. | $\Rightarrow$ | $GATE = 1.$ |



Từ đó ta có: (TMOD) = 11000001B = C1H.

- Ví dụ 2: Cho biết giá trị cần nạp cho thanh ghi TMOD để
 - Timer 0: không sử dụng.
 - Timer 1: là bộ định thời gian 8 bit tự nạp lại, được điều khiển bằng phần mềm (bit $TR1$).

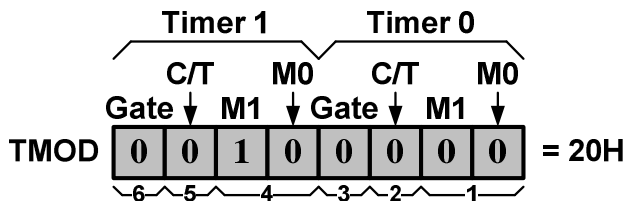
Giải

Phân tích:

- | | | |
|---------------------|---------------|-----------------------|
| (1): Không sử dụng. | $\Rightarrow$ | $M1 = 0, M0 = 0.$ |
| (2): Không sử dụng. | $\Rightarrow$ | $C/\overline{T} = 0.$ |
| (3): Không sử dụng. | $\Rightarrow$ | $GATE = 0.$ |

Do Timer 0 không sử dụng, nên ta có thiết lập nó ở bất cứ chế độ nào. Thông thường để dễ dàng ta nên cho: $GATE=0, C/\overline{T} = 0, M1 = 0$ và $M0 = 0.$

- | | | |
|------------------------------------|---------------|-----------------------|
| (4): Chế độ 8 bit tự động nạp lại. | $\Rightarrow$ | $M1 = 1, M0 = 0.$ |
| (5): Bộ định thời gian. | $\Rightarrow$ | $C/\overline{T} = 0.$ |
| (6): Điều khiển bằng phần mềm. | $\Rightarrow$ | $GATE = 0.$ |

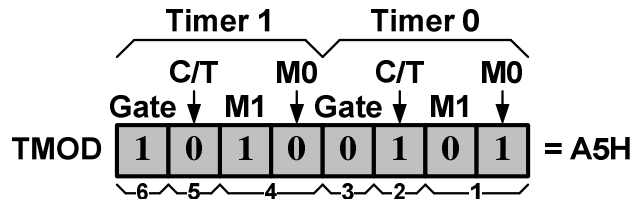


Từ đó ta có: (TMOD) = 00100000B = 20H.

- Ví dụ 3: Cho biết (TMOD) = A5H. Hãy cho biết chế độ hoạt động của các Timer 0 và Timer 1.

Giải

Ta có: (TMOD) = A5H = 10100101B.



Giải thích:

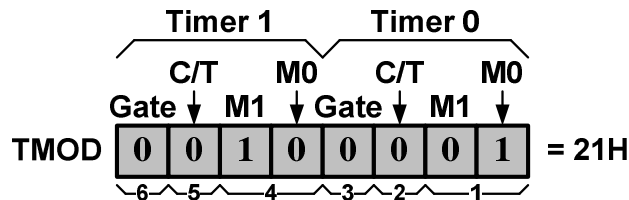
- M1 = 1, M0 = 0. $\Rightarrow$ (4): Chế độ 8 bit tự động nạp lại.
- $C/\overline{T} = 0$. $\Rightarrow$ (5): Bộ định thời gian.
- GATE = 1. $\Rightarrow$ (6): Điều khiển bằng phần cứng.
- M1 = 0, M0 = 1. $\Rightarrow$ (1): Chế độ 16 bit.
- $C/\overline{T} = 1$. $\Rightarrow$ (2): Bộ đếm xung.
- GATE = 0. $\Rightarrow$ (3): Điều khiển bằng phần mềm.

Từ đó ta có:

- Timer 0: là bộ đếm xung 16 bit, được điều khiển bằng phần mềm (*bit TR0*).
 - Timer 1: là bộ định thời gian 8 bit tự nạp lại, được điều khiển bằng phần cứng (*chân INT1*).
- Ví dụ 4: Cho biết (TMOD) = 21H. Hãy cho biết chế độ hoạt động của các Timer 0 và Timer 1.

Giải

Ta có: (TMOD) = 21H = 00100001B.



Giải thích:

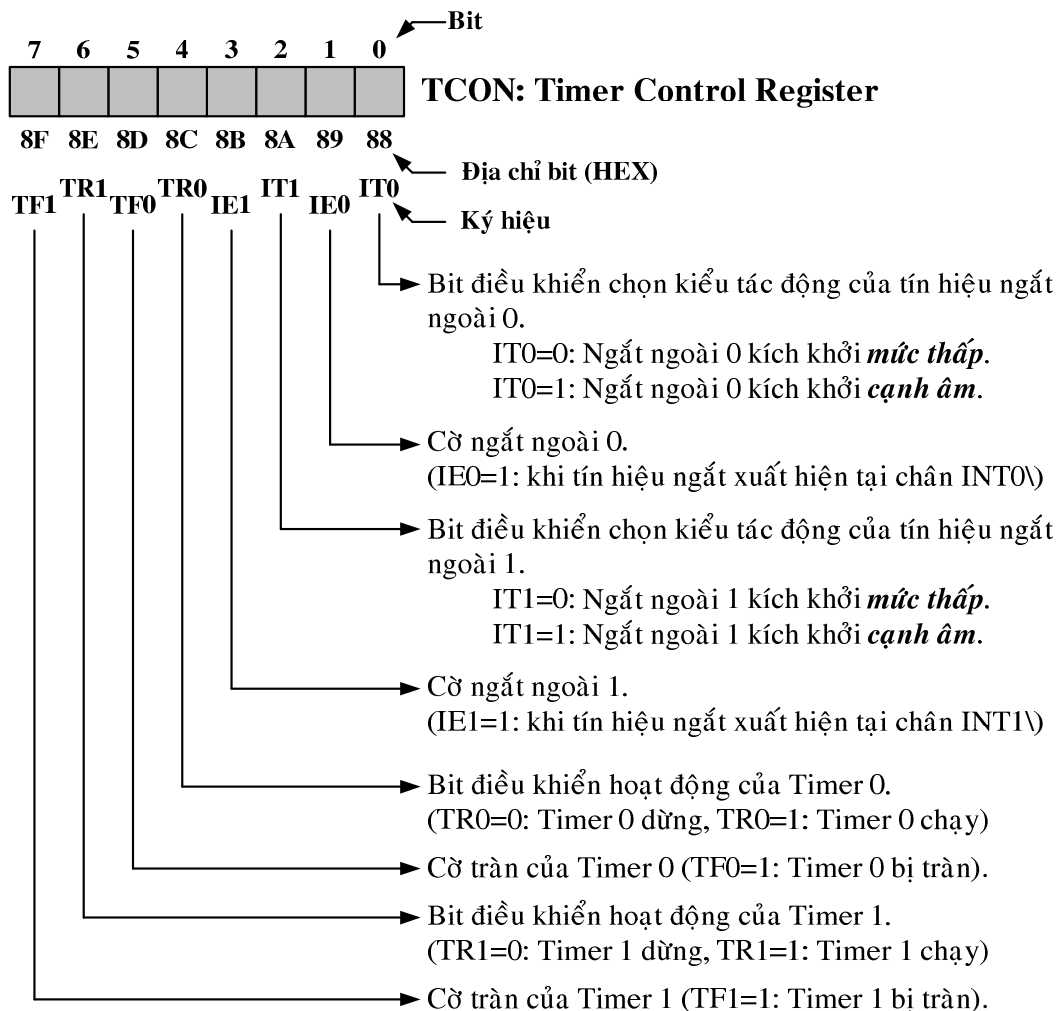
- M1 = 1, M0 = 0. $\Rightarrow$ (4): Chế độ 8 bit tự động nạp lại.
- $C/\overline{T} = 0$. $\Rightarrow$ (5): Bộ định thời gian.
- GATE = 0. $\Rightarrow$ (6): Điều khiển bằng phần mềm.
- M1 = 0, M0 = 1. $\Rightarrow$ (1): Chế độ 16 bit.
- $C/\overline{T} = 0$. $\Rightarrow$ (2): Bộ định thời gian.
- GATE = 0. $\Rightarrow$ (3): Điều khiển bằng phần mềm.

Từ đó ta có:

- Timer 0: là bộ định thời gian 16 bit, được điều khiển bằng phần mềm (*bit TR0*).
- Timer 1: là bộ định thời gian 8 bit tự nạp lại, được điều khiển bằng phần mềm (*bit TR1*).

III. THANH GHI ĐIỀU KHIỂN ĐỊNH THỜI (TCON):

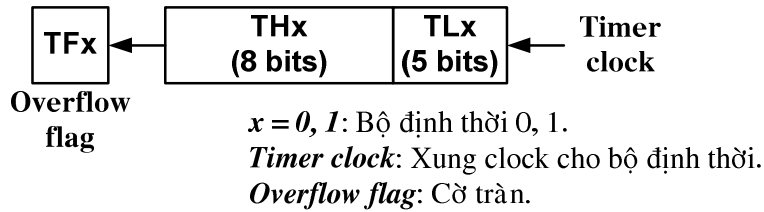
- Thanh ghi TCON (Timer Control Register) chứa các bit dùng để điều khiển và báo trạng thái của bộ định thời 0 và bộ định thời 1.
- Cấu trúc thanh ghi TCON:



- Lưu ý: Các bit IT0, IT1, IE0, IE1 không dùng để điều khiển các bộ định thời. Các bit này được dùng để phát hiện và khởi động các ngắt ngoài. Việc thảo luận các bit này sẽ được trình bày trong “Chương 6: Hoạt động ngắt”.

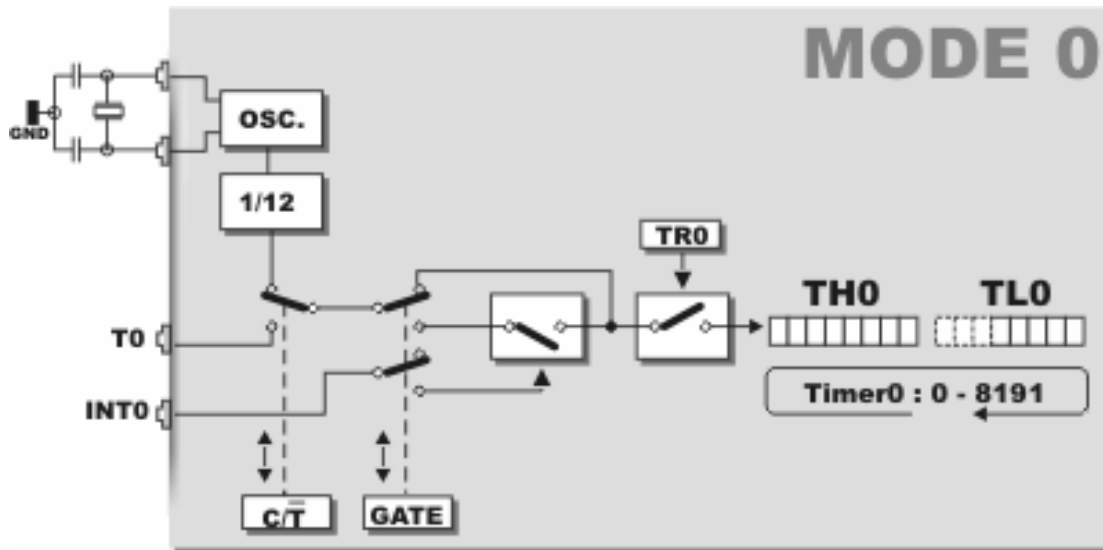
IV. CÁC CHẾ ĐỘ ĐỊNH THỜI VÀ CỜ TRÀN:

1. Chế độ định thời 13 bit (Chế độ 0):



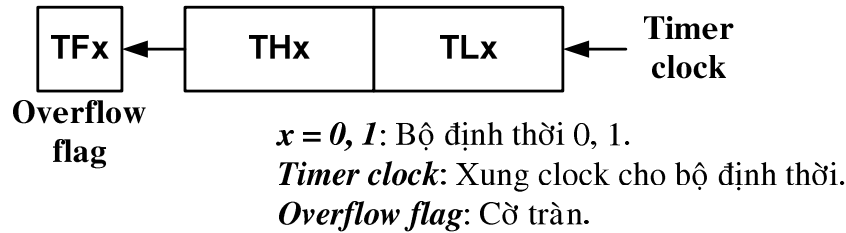
Chế độ 0 (Mode 0):

- Chế độ định thời 13 bit.
- Sử dụng 8 bit của thanh ghi THx và 5 bit thấp của thanh ghi TLx để tạo ra bộ định thời.
- **Số đếm:** 0000H $\rightarrow$ 1FFFH nghĩa là từ 0 $\rightarrow$ 8191. **Thời gian định thời:** từ $1.T_{\text{Timer}} \rightarrow 2^{13}.T_{\text{Timer}}$ nghĩa là từ $1.T_{\text{Timer}} \rightarrow 8192.T_{\text{Timer}}$.
- Thanh ghi THx và TLx chứa giá trị của bộ định thời.
- Khi có xung clock, bộ định thời bắt đầu đếm lên từ giá trị chứa trong THx/TLx.
- Xảy ra tràn (cờ tràn $TFx=1$) khi số đếm chuyển từ 1FFFH sang 0000H và việc đếm sẽ tiếp tục đếm lên từ giá trị 0000H.



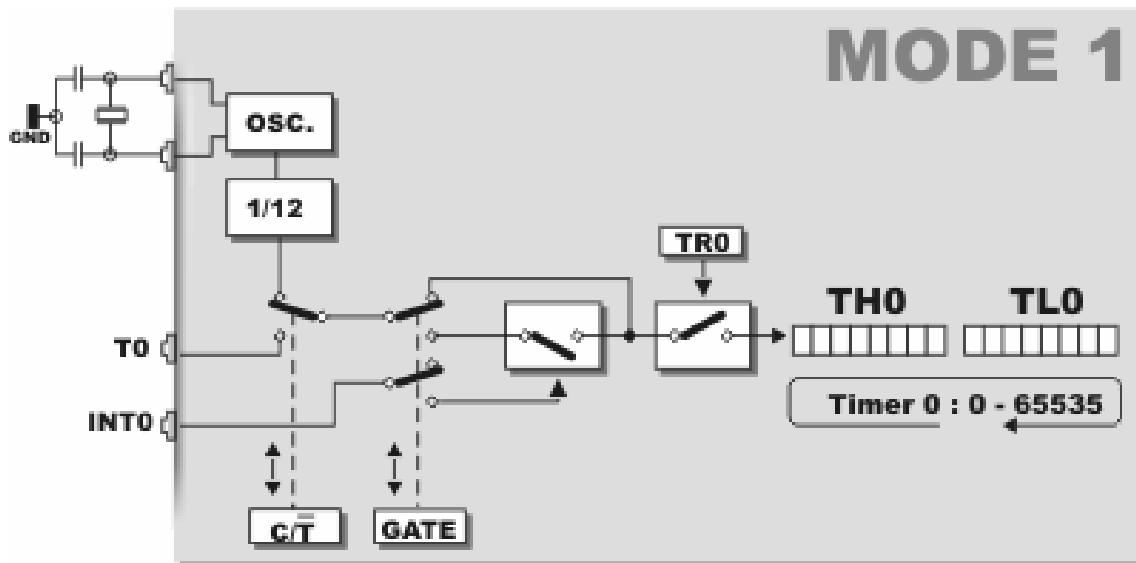
Kiến trúc của Timer 0 ở chế độ 0 (Mode 0).

2. Chế độ định thời 16 bit (Chế độ 1):



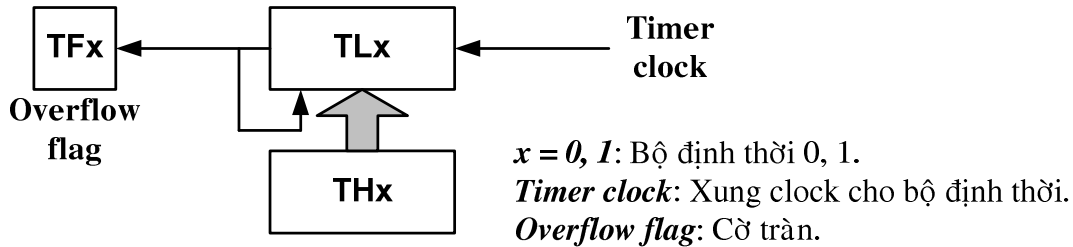
Chế độ 1 (Mode 1):

- Chế độ định thời 16 bit.
- Sử dụng thanh ghi THx và TLx để tạo ra bộ định thời.
- **Số đếm**: 0000H $\rightarrow$ FFFFH nghĩa là từ 0 $\rightarrow$ 65535. **Thời gian định thời**: từ $1.T_{\text{Timer}} \rightarrow 2^{16}.T_{\text{Timer}}$ nghĩa là từ $1.T_{\text{Timer}} \rightarrow 65536.T_{\text{Timer}}$.
- Thanh ghi THx và TLx chứa giá trị của bộ định thời.
- Khi có xung clock, bộ định thời bắt đầu đếm lên từ giá trị chứa trong THx/TLx.
- Xảy ra tràn (cờ tràn $TFx=1$) khi số đếm chuyển từ FFFFH sang 0000H và việc đếm sẽ tiếp tục đếm lên từ giá trị 0000H.



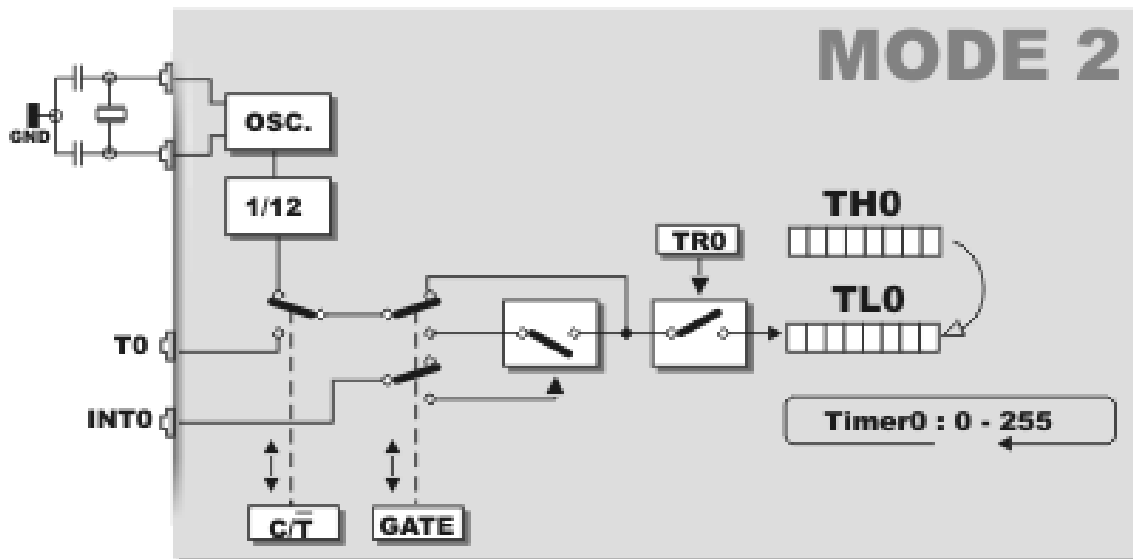
Kiến trúc của Timer 0 ở chế độ 1 (Mode 1).

3. Chế độ định thời 8 bit tự nạp lại (Chế độ 2):



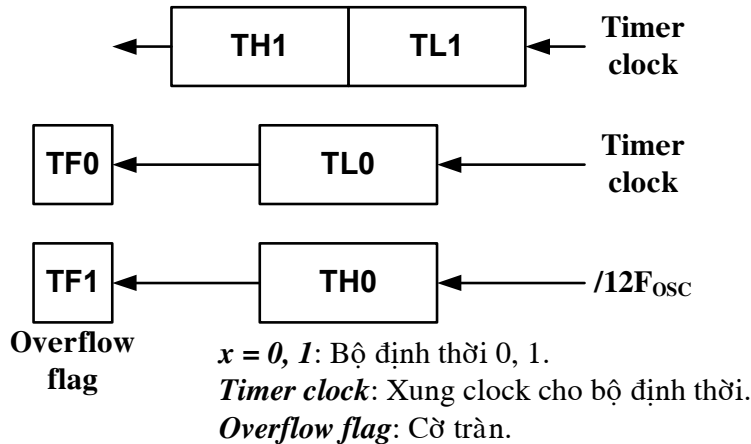
Chế độ 2 (*Mode 2*):

- Chế độ định thời 8 bit tự nạp lại.
- Sử dụng thanh ghi TLx để tạo ra bộ định thời.
- **Số đếm:** 00H $\rightarrow$ FFH nghĩa là từ 0 $\rightarrow$ 255. **Thời gian định thời:** từ $1.T_{\text{Timer}} \rightarrow 2^8.T_{\text{Timer}}$ nghĩa là từ $1.T_{\text{Timer}} \rightarrow 256.T_{\text{Timer}}$.
- Thanh ghi TLx chứa giá trị của bộ định thời và thanh ghi THx chứa giá trị sẽ được dùng để nạp lại cho bộ định thời.
- Khi có xung clock, bộ định thời bắt đầu đếm lên từ giá trị chứa trong TLx (THx không thay đổi giá trị).
- Xảy ra tràn (còn tràn $TFx=1$) khi số đếm chuyển từ FFH sang 00H, đồng thời giá trị trong THx sẽ được nạp vào TLx và việc đếm sẽ tiếp tục đếm lên từ giá trị chứa trong thanh ghi TLx (giá trị này bằng với giá trị của THx).



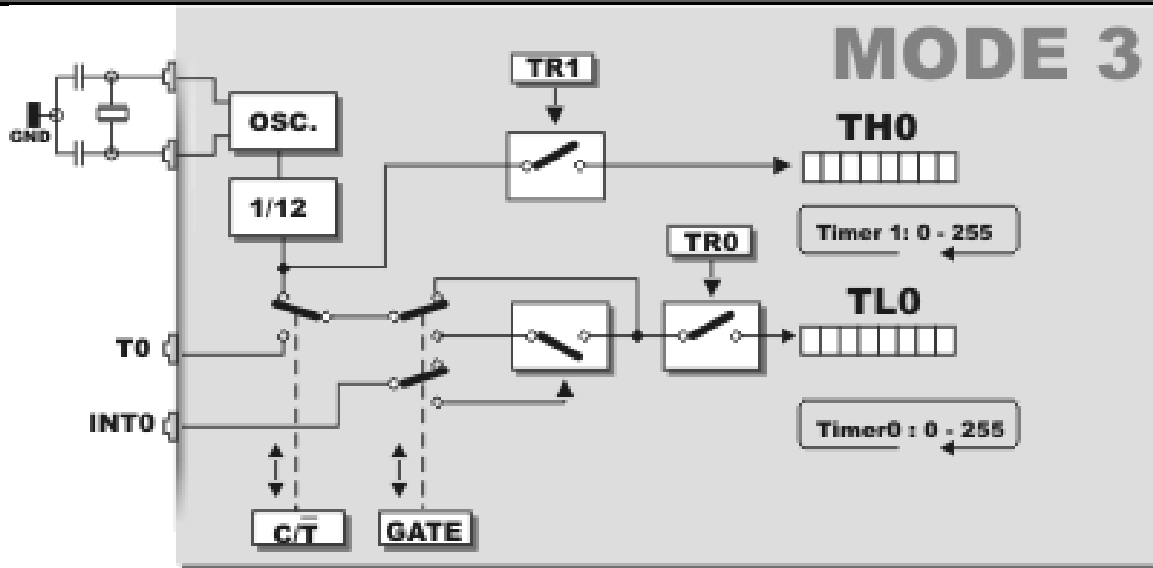
Kiến trúc của Timer 0 ở chế độ 2 (Mode 2).

4. Chế độ định thời chia xẻ (Chế độ 3):



Chế độ 3 (Mode 3) là:

- Chế độ định thời chia xẻ.
- Bộ định thời 0 được chia ra:
 - Bộ định thời 8 bit thứ I:
 - Sử dụng thanh ghi **TL0** để tạo ra bộ định thời.
 - **Số đếm**: 00H → FFH nghĩa là từ 0 → 255. **Thời gian định thời**: từ $1.T_{\text{Timer}} \rightarrow 2^8.T_{\text{Timer}}$ nghĩa là từ $1.T_{\text{Timer}} \rightarrow 256.T_{\text{Timer}}$.
 - Thanh ghi TL0 chứa giá trị của bộ định thời.
 - Khi có xung clock, bộ định thời *bắt đầu đếm lên từ giá trị chứa trong TL0*.
 - Xảy ra tràn (**cờ tràn TF0=1**) khi số đếm chuyển từ FFH sang 00H và việc đếm sẽ tiếp tục đếm lên từ giá trị 00H.
 - Bộ định thời 8 bit thứ II:
 - Sử dụng thanh ghi **TH0** để tạo ra bộ định thời.
 - **Số đếm**: 00H → FFH nghĩa là từ 0 → 255. **Thời gian định thời**: từ $1.T_{\text{Timer}} \rightarrow 2^8.T_{\text{Timer}}$ nghĩa là từ $1.T_{\text{Timer}} \rightarrow 256.T_{\text{Timer}}$.
 - Thanh ghi TH0 chứa giá trị của bộ định thời.
 - Khi có xung clock, bộ định thời *bắt đầu đếm lên từ giá trị chứa trong TH0*.
 - Xảy ra tràn (**cờ tràn TF1=1**) khi số đếm chuyển từ FFH sang 00H và việc đếm sẽ tiếp tục đếm lên từ giá trị 00H.
- Bộ định thời 1:
 - Là bộ định thời 16 bit.
 - Không hoạt động ở chế độ 3 nhưng có thể hoạt động các chế độ khác (chế độ 0, 1, 2).
 - Không có cờ báo tràn như các bộ định thời khác.



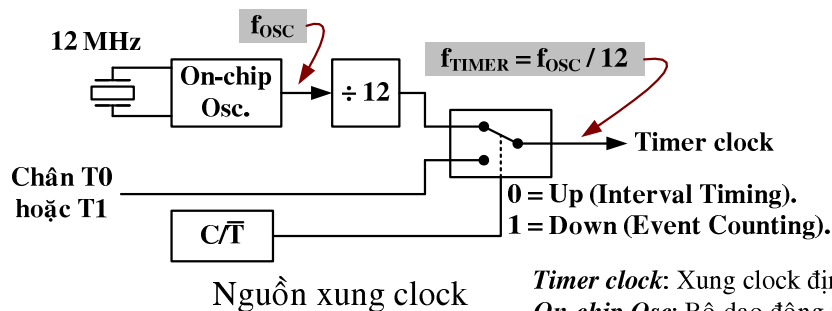
Kiến trúc của Timer 0 ở chế độ 3 (Mode 3).

V. NGUỒN XUNG CLOCK CHO BỘ ĐỊNH THỜI:

Nguồn xung cho bộ định thời được tạo ra từ:

- Mạch dao động trên chip → dùng cho tính năng định thời gian.
- Xung kích thích bên ngoài → dùng cho tính năng đếm sự kiện.

1. Trường hợp định thời gian:



Timer clock: Xung clock định thời.

On-chip Osc: Bộ dao động trên chip.

Up (Interval Timing): Vị trí trên (định thời gian).

Down (Event Counting): Vị trí dưới (đếm sự kiện).

Nếu $C/T=0$ thì:

- Bộ định thời được dùng để định thời gian (Timer).
- Nguồn xung clock định thời được lấy từ mạch dao động trên chip.

Lưu ý:

- Tần số xung clock cung cấp cho bộ định thời bằng $1/12$ tần số của mạch dao động trên chip 8051.
- **Thời gian định thời** là khoảng thời gian được tính từ lúc bộ định thời bắt đầu đếm lên (từ giá trị chứa trong các thanh ghi THx/TLx) cho đến lúc bộ định thời bắt đầu tràn (thời gian này phụ thuộc vào giá trị ban đầu được nạp cho các thanh ghi THx và TLx).

- Ví dụ: Tìm tần số xung clock và chu kỳ của bộ định thời đối với trường hợp các hệ thống vi điều khiển xây dựng trên chip 8051 với tần số thạch anh như sau: 11,0592 MHz, 12 MHz và 16 MHz.

Giải

Gọi f_{TIMER} là tần số xung clock của bộ định thời, f_{OSC} là tần số xung clock của thạch anh. Theo như trên đã trình bày, ta có:

$$f_{OSC} = 11,0592(MHz) \rightarrow f_{TIMER} = \frac{f_{OSC}}{12} = \frac{11,0592(MHz)}{12} = 921,6(KHz)$$

$$T_{TIMER} = \frac{1}{f_{TIMER}} = \frac{1}{921,6(KHz)} = 1,085(\mu s)$$

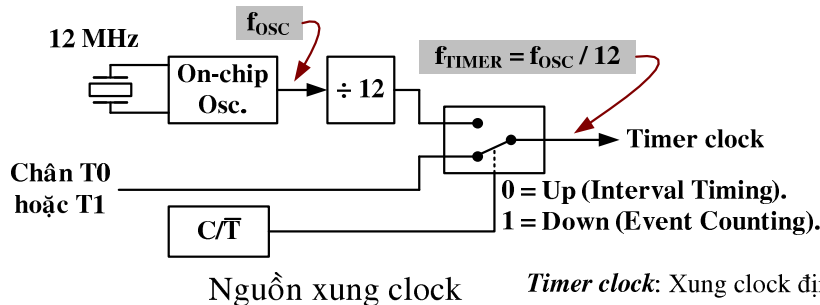
$$f_{OSC} = 12(MHz) \rightarrow f_{TIMER} = \frac{f_{OSC}}{12} = \frac{12(MHz)}{12} = 1(MHz)$$

$$T_{TIMER} = \frac{1}{f_{TIMER}} = \frac{1}{1(MHz)} = 1(\mu s)$$

$$f_{OSC} = 16(MHz) \rightarrow f_{TIMER} = \frac{f_{OSC}}{12} = \frac{16(MHz)}{12} = 1,333(MHz)$$

$$T_{TIMER} = \frac{1}{f_{TIMER}} = \frac{1}{1,333(MHz)} = 0,75(\mu s)$$

2. Trường hợp đếm sự kiện:



Timer clock: Xung clock định thời.

On-chip Osc: Bộ dao động trên chip.

Up (Interval Timing): Vị trí trên (định thời gian).

Down (Event Counting): Vị trí dưới (đếm sự kiện).

Nếu $C/T=1$ thì:

- Bộ định thời được dùng để đếm sự kiện (Counter).
- Nguồn xung clock định thời được lấy từ xung kích thích bên ngoài tại hai chân T0 và T1 của chip 8051.

Lưu ý:

- Tần số kích thích **tối đa cho phép** tại chân T0 và T1:

$$f_{T0,T1(MAX)} = \frac{f_{TIMER}}{2}$$

f_{TIMER} : tần số xung clock định thời.

$f_{T0,T1(MAX)}$: tần số kích thích tối đa cho phép tại T0 và T1.

- Ví dụ: Tính tần số kích thích tối đa cho phép tại chân T0 và T1 đối với trường hợp các hệ thống vi điều khiển xây dựng trên chip 8051 với tần số thạch anh như sau: 11,0592 MHz, 12 MHz và 16 MHz.

Giải

$$f_{OSC} = 11,0592(\text{MHz}) \rightarrow f_{T0,T1(\text{MAX})} = \frac{f_{TIMER}}{2} = \frac{921,6(\text{KHz})}{2} = 460,8(\text{KHz})$$

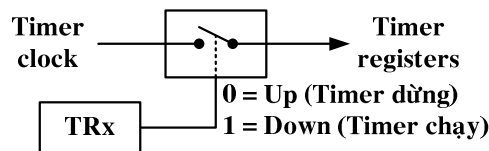
$$f_{OSC} = 12(\text{MHz}) \rightarrow f_{T0,T1(\text{MAX})} = \frac{f_{TIMER}}{2} = \frac{1(\text{MHz})}{2} = 500(\text{KHz})$$

$$f_{OSC} = 16(\text{MHz}) \rightarrow f_{T0,T1(\text{MAX})} = \frac{f_{TIMER}}{2} = \frac{1,333(\text{MHz})}{2} = 666,5(\text{KHz})$$

○ **Số lượng sự kiện (số xung)** mà bộ định thời đếm được sẽ được chứa trong các thanh ghi THx/TLx, giá trị trong các thanh ghi này sẽ tăng theo mỗi xung kích thích bên ngoài tại T0 và T1 của chip 8051.

○ **Một kích thích** được gọi là một sự kiện (một xung) khi xảy ra **sự chuyển trạng thái từ 1 xuống 0** ở chân T0 hoặc T1.

VI. KHỞI ĐỘNG, DỪNG VÀ ĐIỀU KHIỂN CÁC BỘ ĐỊNH THỜI:



Bắt đầu và dừng các bộ định thời.

- Cách 1: (thường được dùng để định thời gian).

Điều kiện sử dụng: bit GATE = 0

(Phương pháp điều khiển bằng phần mềm)

⇒ Bộ định thời x chạy khi bit TRx = 1.

⇒ Bộ định thời x dừng khi bit TRx = 0.

Ví dụ: Để khởi động bộ định thời 0 ta dùng lệnh: **SETB TR0**

Để dừng bộ định thời 0 ta dùng lệnh: **CLR TR0**

- Cách 2: (thường được dùng để đo độ rộng xung tại chân INTx\).

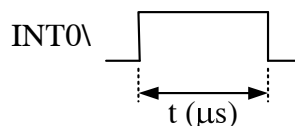
Điều kiện sử dụng: bit GATE = 1 và bit /TRx = 1

(Phương pháp điều khiển bằng phần cứng)

⇒ Bộ định thời x chạy khi chân INTx\ = 1.

⇒ Bộ định thời x dừng khi chân INTx\ = 0.

Ví dụ: Đo độ rộng xung (tính bằng μs) tại chân INT0, với $f_{OSC} = 12\text{MHz}$.



Giải

Ta khởi động bộ định thời 0 như sau:

- Chế độ định thời 16 bit (*chế độ 1*).
- Giá trị trong TH0/TL0 là 0000H.
- GATE = 1 và TR0 = 1 (*điều khiển hoạt động của Timer 0 bằng phần cứng, tức điều khiển bằng tín hiệu tại chân INT0*).

Khi INT0 $\uparrow$ $\Rightarrow$ Bộ định thời 0 được mở cổng và nhận xung clock 1 MHz

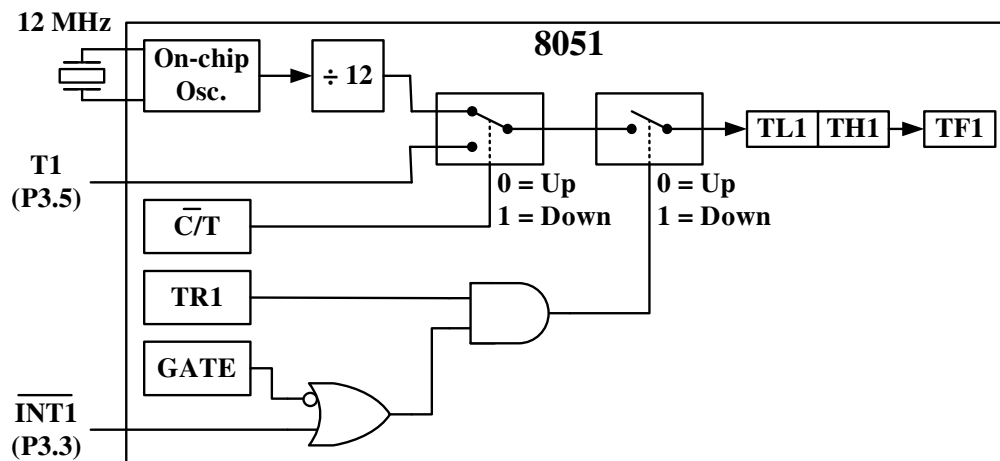
Bộ định thời bắt đầu đếm lên từ giá trị chứa trong TH0/TL0

Khi INT0 $\downarrow$ $\Rightarrow$ Bộ định thời 0 bị khoá cổng và không nhận xung clock 1 MHz

Bộ định thời dừng

$\Rightarrow$ **Độ rộng xung (tính bằng μ s) = Số đếm chứa trong TH0/TL0.**

Hình minh họa Timer 1 hoạt động ở chế độ 1 (Timer 16 bit):



Hoạt động ở chế độ 1 của Timer 1

VII. KHỞI ĐỘNG VÀ TRUY XUẤT THANH GHI ĐỊNH THỜI:

Trước khi các bộ định thời hoạt động cần phải:

- Qui định chế độ của bộ định thời $\Rightarrow$ **thanh ghi TMOD.**
- Qui định điểm bắt đầu đếm của bộ định thời (*khoảng thời gian định thời*) $\Rightarrow$ **thanh ghi THx/TLx.**

○ Ví dụ 1: Khởi động bộ định thời 1 hoạt động ở chế độ 16 bit, xung clock được lấy từ mạch dao động trên chip (*nghĩa là bộ định được dùng để định thời một khoảng thời gian*), được khởi động bằng bit TR1 (*điều khiển bằng phần mềm*).

Giải

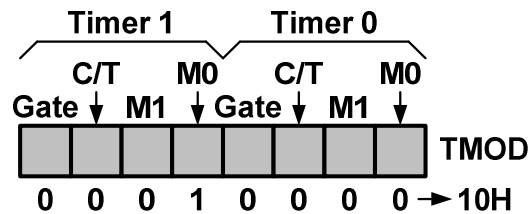
Ta dùng lệnh:

MOV TMOD, #10H

hoặc

MOV TMOD, #00010000B

Giải thích:



- GATE = 0 → điều khiển bằng phần mềm (bit *TRI*).
- C/T = 0 → sử dụng mạch dao động trên chip (dùng để định một khoảng thời gian).
- M1 = 0, M1 = 1 → TIMER1 hoạt động ở chế độ 1 (chế độ định thời 16 bit).

○ Ví dụ 2: Dùng bộ định thời 1 ở ví dụ trên để định một khoảng thời gian là 100 μ s. Giả sử vi điều khiển sử dụng thạch anh 12 MHz.

Giải

Ta dùng lệnh:

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

hoặc

```
MOV TL1, #LOW(-100)
MOV TH1, #HIGH(-100)
```

Giải thích:

$$f_{OSC} = 12(MHz) \rightarrow f_{TIMER} = \frac{f_{OSC}}{12} = \frac{12}{12} = 1(MHz)$$

$$\rightarrow T_{TIMER} = \frac{1}{f_{TIMER}} = \frac{1}{1(MHz)} = 1(\mu s)$$

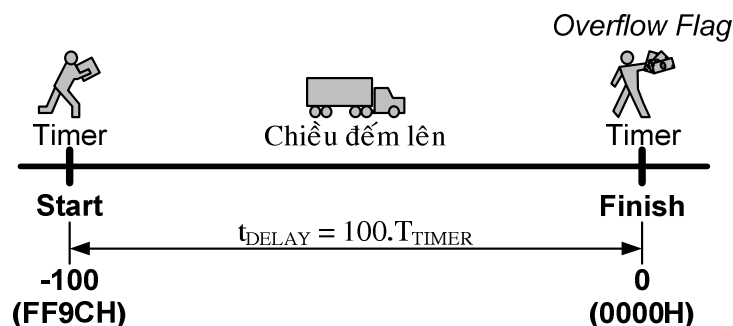
Trong đó: f_{OSC} : tần số thạch anh.
 f_{TIMER} : tần số xung clock định thời.
 T_{TIMER} : chu kỳ xung clock định thời.

⇒ Vậy cứ mỗi 1 μ s (tức là sau mỗi chu kỳ của xung clock định thời) thì bộ định thời sẽ tăng giá trị một lần.

Mà ta đã biết: thời gian định thời là khoảng thời gian được tính từ lúc bộ định thời bắt đầu đếm lên cho đến lúc bộ định thời bắt đầu tràn.

⇒ Vậy để bộ định thời tràn sẽ tràn sau khoảng thời gian 100 μ s thì ta phải khởi động bộ định thời tại thời điểm cách điểm tràn (theo chiều âm – vì bộ định thời chỉ đếm lên) 100 chu kỳ xung clock định thời.

⇒ Vì điểm tràn có giá trị là 0 cho nên giá trị cần nạp cho các thanh ghi TH1/TL1 là **-100** (hay **FF9CH**).



Tổng quát, ta có *công thức tính giá trị cần nạp cho bộ định thời* để có thời gian định thời như mong muốn là:

$$N = -\frac{f_{OSC}}{12} t_{DELAY}$$

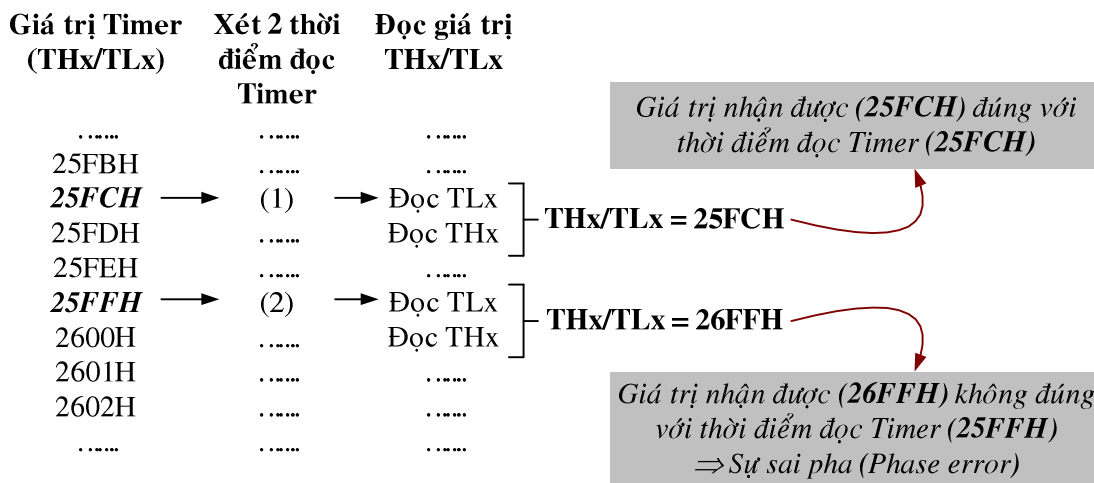
Trong đó: N: giá trị cần nạp cho bộ định thời.
 f_{OSC} (MHz): tần số thạch anh.
 t_{DELAY} (μs): thời gian cần định thời.

2. Truy xuất giá trị của bộ định thời đang hoạt động:

Trong các ứng dụng thực tế, ta cần phải đọc giá trị (*nội dung*) chứa trong các thanh ghi định thời THx/TLx trong khi bộ định thời vẫn đang hoạt động. Do giá trị của bộ định thời được chứa trong cả hai thanh ghi THx/TLx. Cho nên ta phải đọc hai thanh ghi này bằng hai dòng lệnh liên tiếp nhau (*do không có lệnh nào có thể đọc đồng thời cả hai thanh ghi định thời này*). Một sự sai pha (*phase error*) có thể xuất hiện nếu có sự tràn từ byte thấp chuyển sang byte cao giữa hai lần đọc và do vậy ta không thể đọc đúng được giá trị cần đọc.

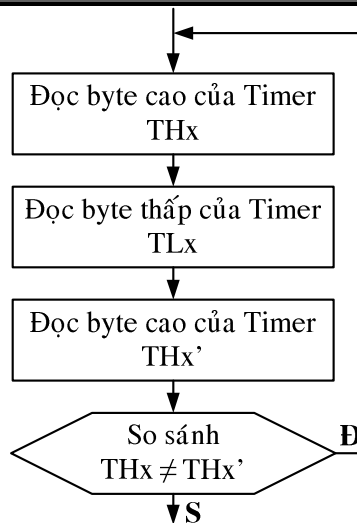
• *Ví dụ:* Minh họa về sự sai pha (*phase error*) có thể xuất hiện nếu có sự tràn từ byte thấp chuyển sang byte cao giữa hai lần đọc giá trị làm cho ta không thể đọc đúng được giá trị cần đọc của THx/TLx trong khi bộ định thời đang hoạt động.

Giải



Minh họa trường hợp đọc giá trị chứa trong các thanh ghi định thời đang hoạt động.

Giải pháp đưa ra là trước tiên ta phải đọc byte cao, kế đến đọc byte thấp và rồi đọc byte thấp lần nữa. Nếu byte cao thay đổi giá trị, ta lặp lại thao tác đọc vừa nêu. Lưu đồ giải thuật dùng để đọc chính xác giá trị (*nội dung*) chứa trong các thanh ghi định thời THx/TLx của *bộ định thời đang hoạt động*:



- Ví dụ: Đọc nội dung của các thanh ghi TH1/TL1 trong khi bộ định thời 1 đang hoạt động. Nội dung sau khi đọc của thanh ghi TH1 chứa trong R7, của thanh ghi TL1 chứa trong R6.

AGAIN:

```

MOV A, TH1
MOV R6, TL1
CJNE A, TH1, AGAIN
MOV R7, A
    
```

VIII. CÁC KHOẢNG THỜI GIAN ĐỊNH THỜI:

Khảo sát trường hợp 8051 dùng thạch anh **12 MHz**:

- Khoảng thời gian định thời **ngắn nhất** (μs): **1**
- Khoảng thời gian định thời **dài nhất** (μs):
 - ≈ 10 $\Rightarrow$ Dùng các lệnh.
 - ≤ 256 $\Rightarrow$ Dùng bộ định thời 8 bit tự động nạp lại.
 - ≤ 65536 $\Rightarrow$ Dùng bộ định thời 16 bit.
 - **Không giới hạn** $\Rightarrow$ Dùng bộ định thời 16 bit + các vòng lặp.

Khảo sát trường hợp tổng quát:

- Khoảng thời gian định thời **ngắn nhất**: **1.T_{TIMER}**
- Khoảng thời gian định thời **dài nhất**:
 - $\approx 10.T_{TIMER}$ $\Rightarrow$ Dùng các lệnh.
 - $\leq 256.T_{TIMER}$ $\Rightarrow$ Dùng bộ định thời 8 bit tự động nạp lại.
 - $\leq 65536.T_{TIMER}$ $\Rightarrow$ Dùng bộ định thời 16 bit.
 - **Không giới hạn** $\Rightarrow$ Dùng bộ định thời 16 bit + các vòng lặp.

với $T_{TIMER} = \frac{12}{f_{OSC}}$ $T_{TIMER}(\mu s)$: chu kỳ xung clock định thời.
 f_{OSC} (MHz): tần số thạch anh.

IX. CÁC BƯỚC CƠ BẢN KHỞI ĐỘNG TIMER VÀ COUNTER:**1. Các bước cơ bản để khởi động Timer:**

- Chọn chế độ hoạt động cho Timer, cho bit GATE=0 và C/T=0:

MOV TMOD, #...(1)...

- Chọn giá trị thích hợp (*khoảng thời gian định thời*) cho Timer:

MOV THx, #...(2)...

MOV TLx, #...(3)...

- Cho Timer chạy:

SETB TRx

- Kiểm tra cờ báo tràn (*kiểm tra đủ thời gian định thời*):

JNB TFx, \$

hoặc

WAIT: JNB TFx, WAIT

- Xóa cờ báo tràn (*chuẩn bị cho lần định thời tiếp theo*):

CLR TFx

- Dừng Timer (*sau khi đã hoàn tất quá trình định thời*):

CLR TRx

Lưu ý:

- ✚ x: Số thứ tự của Timer sử dụng.

- ✚ (1): Giá trị này phụ thuộc vào Timer được chọn và chế độ hoạt động của Timer đó.

✚ (2), (3): Giá trị này phụ thuộc vào khoảng thời gian cần định thời. Cũng cần lưu ý thêm, việc chọn giá trị cho không phải lúc nào ta cũng phải chọn giá trị cho cả hai thanh ghi này mà nó tùy thuộc vào từng chế độ hoạt động của Timer (*Mode 0: THx/TLx, Mode 1: THx/TLx, Mode 2: THx, Mode 3: THx hoặc TLx*).

- ✚ Các giá trị trên phải thỏa mãn điều kiện sau:

- **Chế độ 8 bit:** giá trị trong khoảng từ -255 đến -1 (*tương ứng từ 255.T_{TIMER} đến 1.T_{TIMER}*).

Ví dụ: **MOV TH1, #(-255)** → định thời 255.T_{TIMER}

- **Chế độ 13 bit:** giá trị trong khoảng từ -8191 đến -1 (*tương ứng từ 8191.T_{TIMER} đến 1.T_{TIMER}*).

Ví dụ: **MOV TL1, #LOW(-8000)** → định thời 8000.T_{TIMER}
 MOV TH1, #HIGH(-8000)

- **Chế độ 16 bit:** giá trị trong khoảng từ -65535 đến -1 (*tương ứng từ 65535.T_{TIMER} đến 1.T_{TIMER}*).

Ví dụ: **MOV TL1, #LOW(-10000)** → định thời 10000.T_{TIMER}
 MOV TH1, #HIGH(-10000)

Trường hợp đặc biệt nếu giá trị (N) nạp vào thanh ghi THx/TLx là **giá trị 0** thì thời gian định thời sẽ là lớn nhất cho từng chế độ.

- Chế độ 8 bit: $N = 0 \rightarrow t_{\text{DELAY}} = 256.T_{\text{TIMER}}$.
- Chế độ 13 bit: $N = 0 \rightarrow t_{\text{DELAY}} = 8192.T_{\text{TIMER}}$.
- Chế độ 16 bit: $N = 0 \rightarrow t_{\text{DELAY}} = 65536.T_{\text{TIMER}}$.

2. Các bước cơ bản để khởi động Counter:

- Chọn chế độ hoạt động cho Counter, cho bit GATE=0 và C/T=1:
MOV TMOD, #...(1)...
- Xoá các giá trị chứa trong thanh ghi THx và TLx (nghĩa là cho số xung ban đầu bằng 0):
MOV THx, #00H
MOV TLx, #00H
- Cho Counter chạy:
SETB TRx
- Kiểm tra cờ báo tràn (kiểm tra số đếm bị tràn) để xử lý trường hợp số đếm bị tràn.
- Xóa cờ báo tràn (sau khi đã xử lý cho trường hợp số đếm bị tràn):
CLR TFx
- Dừng Counter (sau khi đã hoàn tất quá trình đếm xung):
CLR TRx
- Đọc số xung đếm được trong thanh ghi THx và TLx.

Lưu ý:

 x: Số thứ tự của Counter sử dụng.

 (1): Giá trị này phụ thuộc vào Counter được chọn và chế độ hoạt động của Counter. Giá trị này phải thoả mãn điều kiện sau:

- **Chế độ 8 bit:** số lượng xung tối đa mà Counter đếm được từ 0 đến 255.
- **Chế độ 13 bit:** số lượng xung tối đa mà Counter đếm được từ 0 đến 8191.
- **Chế độ 16 bit:** số lượng xung tối đa mà Counter đếm được từ 0 đến 65535.

Trong quá trình đọc số xung đếm được chứa trong các thanh ghi THx/TLx ta phải chú ý đến trường hợp Counter bị tràn. Vì khi đó giá trị trong thanh ghi THx/TLx (nơi chứa số xung đếm được) sẽ trở về 0. Cho nên nếu ta không có biện pháp xử lý cho trường hợp này thì kết quả là số xung mà ta nhận được sẽ bị sai. Vì thế, nếu ta giả sử ban đầu Counter được khởi động với giá trị là 0 thì cứ mỗi lần Counter bị tràn thì ta phải cộng thêm vào số xung đọc về 256 xung (trường hợp 8 bit) hoặc 8192 xung (trường hợp 13 bit) hoặc 65536 xung (trường hợp 16 bit).

X. CÁC VÍ DỤ MINH HỌA:

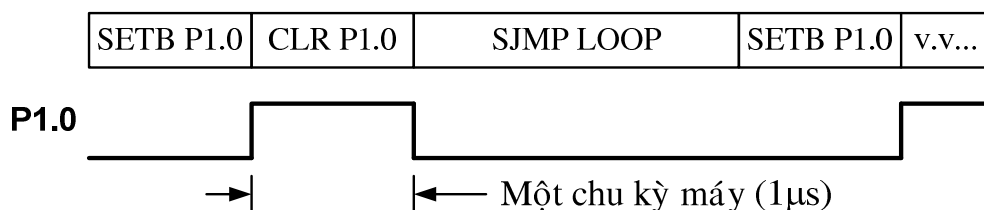
1. Ví dụ 1: (Tạo dạng xung)

Viết chương trình tạo dạng xung tuần hoàn trên chân P1.0 có tần số cao nhất có thể có. Tần số và chu kỳ nhiệm vụ của dạng xung này là bao nhiêu?

Giải

```

ORG 8100H
LOOP:  SETB P1.0           ;1 chu kỳ máy
        CLR P1.0           ;1 chu kỳ máy
        SJMP LOOP          ;2 chu kỳ máy
        END
    
```



- Chu kỳ xung: $4 \mu s \rightarrow$ Tần số xung: 250 KHz.
- Thời gian mức cao: $1 \mu s$.
- Thời gian mức thấp: $3 \mu s$.
- Chu kỳ nhiệm vụ: 25%.

2. Ví dụ 2: (Tạo thời gian trễ)

Viết chương trình con tạo thời gian trễ t_{Delay} , sử dụng phương pháp dùng các lệnh (không dùng Timer). Biết rằng tần số thạch anh là 12 MHz.

- Thời gian trễ $t_{Delay} = 100 \mu s$.
- Thời gian trễ $t_{Delay} = 10 ms$.
- Thời gian trễ $t_{Delay} = 1 s$.

Giải

• Phương pháp:

Phương pháp thực hiện là sử dụng lệnh vòng lặp DJNZ để tạo thời gian trễ t_{Delay} như mong muốn. Ví dụ mẫu:

```

DELAY:                                ; $t_{Delay} = 10 \times 20 \times 30 \times 2 \cdot T_{Timer}$ 
      MOV R0, #10
BBB:   MOV R1, #20
AAA:   MOV R2, #30
      DJNZ R2, $                        ;lệnh 2.  $T_{Timer}$ 
      DJNZ R1, AAA
      DJNZ R0, BBB
      RET
    
```

Tổng quát, ta có công thức tính thời gian trễ t_{Delay} như sau:

$$t_{Delay} = [Rn] \times [Rm] \times \dots \times [Rv] \times 2 \cdot \frac{12}{f_{Osc}} \quad (1)$$

Trong đó: t_{Delay} (μs): thời gian trễ.
 f_{Osc} (MHz): tần số thạch anh.

$$T_{Timer} (\mu s): \text{chu kỳ Timer} \left(T_{Timer} = \frac{12}{f_{Osc}} \right).$$

$[Rn], [Rm], \dots, [Rv]$: giá trị của các vòng lặp (số lần lặp lại lệnh DJNZ).

Lưu ý: Giá trị của các vòng lặp phải thỏa điều kiện $0 \leq [Rn] \leq 255$. Đặc biệt nếu chọn $[Rn] = 0$ thì điều này sẽ tương đương với trường hợp ta chọn $[Rn] = 256$ (tương tự cho các $[Rm], \dots, [Rv]$ khác).

- Tính toán: Tìm giá trị cần nạp cho các thanh ghi vòng lặp:

$$\text{Ta có: } t_{Delay} = [Rn] \times [Rm] \times \dots \times [Rv] \times 2 \cdot \frac{12}{f_{Osc}}$$

Với $t_{Delay} = 100 \mu s$, $f_{Osc} = 12 \text{ MHz}$ thì ta chọn:

$$[Rn] = 50 \\ \Leftrightarrow t_{Delay} = 50 \times 2 \cdot \frac{12}{12} = 100 \mu s$$

Với $t_{Delay} = 10 \text{ ms}$, $f_{Osc} = 12 \text{ MHz}$ thì ta chọn:

$$[Rn] = 20 \text{ và } [Rm] = 250$$

$$\Leftrightarrow t_{Delay} = 20 \times 250 \times 2 \cdot \frac{12}{12} = 10000 \mu s$$

Với $t_{Delay} = 1 \text{ s}$, $f_{Osc} = 12 \text{ MHz}$ thì ta chọn:

$$[Rn] = 10, [Rm] = 200 \text{ và } [Ro] = 250$$

$$\Leftrightarrow t_{Delay} = 10 \times 200 \times 250 \times 2 \cdot \frac{12}{12} = 1000000 \mu s$$

- **Chương trình:** Dựa vào những tính toán trên, ta có:

Với $t_{Delay} = 100 \mu s$:

DELAY:

MOV R0, #50

DJNZ R0, \$

RET

$t_{Delay} = 50 \times 2 \cdot T_{Timer}$

;lệnh 1. T_{Timer}

;lệnh 2. T_{Timer}

;lệnh 2. T_{Timer}

Với $t_{Delay} = 10 \text{ ms}$:

DELAY:

MOV R0, #20

AAA:

MOV R1, #250

DJNZ R1, \$

DJNZ R0, AAA

RET

$t_{Delay} = 20 \times 250 \times 2 \cdot T_{Timer}$

;lệnh 1. T_{Timer}

;lệnh 1. T_{Timer}

;lệnh 2. T_{Timer}

;lệnh 2. T_{Timer}

;lệnh 2. T_{Timer}

;lệnh 2. T_{Timer}

Với $t_{Delay} = 1 \text{ s}$:

DELAY:

MOV R0, #10

BBB:

MOV R1, #200

AAA:

MOV R2, #250

DJNZ R2, \$

DJNZ R1, AAA

DJNZ R0, BBB

RET

$t_{Delay} = 10 \times 200 \times 250 \times 2 \cdot T_{Timer}$

;lệnh 1. T_{Timer}

;lệnh 1. T_{Timer}

;lệnh 1. T_{Timer}

;lệnh 1. T_{Timer}

;lệnh 1. T_{Timer}

;lệnh 2. T_{Timer}

;lệnh 2. T_{Timer}

;lệnh 2. T_{Timer}

;lệnh 2. T_{Timer}

• **Lưu ý về độ chính xác của t_{Delay} :** Khi sử dụng phương pháp tạo thời gian trễ như trên (phương pháp dùng lệnh, không dùng Timer) thì việc định thời gian thường có một sai số nhất định. Vì ở đây, đề đơn giản trong việc tính toán mà ta đã bỏ qua không tính đến thời gian cần thiết để thực hiện từng lệnh trong chương trình, chỉ quan tâm đến thời gian thực hiện của lệnh **DJNZ** ($2T_{Timer}$) và số lần thực hiện của nó. Công thức trình bày ở trên (1) chỉ là công thức tính thời gian t_{Delay} có độ chính xác tương đối, muốn tính thời gian t_{Delay} chính xác thì ta cần phải tính tổng thời gian thực hiện (nghĩa là tính số chu kỳ máy hay số chu kỳ Timer) của tất cả các lệnh có trong chương trình. Độ chính xác của chương trình tạo thời gian trễ theo phương pháp tính tương đối này phụ thuộc vào số lần lặp lại và số vòng lặp.

Với $t_{Delay} = 100 \mu s$, $f_{Osc} = 12 \text{ MHz}$ thì t_{Delay} chính xác là:

$$t_{Delay(cx)} = 1 \cdot T_{Timer} + 2 \cdot T_{Timer} \times 50 + 2 \cdot T_{Timer} = 103 \cdot T_{Timer}$$

Với $t_{\text{Delay}} = 10 \text{ ms}$, $f_{\text{Osc}} = 12 \text{ MHz}$ thì t_{Delay} chính xác là:

$$t_{\text{Delay}(cx)} = 1.T_{\text{Timer}} + \left(1.T_{\text{Timer}} + 2.T_{\text{Timer}} \times 250 + 2.T_{\text{Timer}} \right) \times 20 + 2.T_{\text{Timer}} = 10065.T_{\text{Timer}}$$

Với $t_{\text{Delay}} = 1 \text{ s}$, $f_{\text{Osc}} = 12 \text{ MHz}$ thì t_{Delay} chính xác là:

$$t_{\text{Delay}(cx)} = 1.T_{\text{Timer}} + \left(1.T_{\text{Timer}} + \left(1.T_{\text{Timer}} + 2.T_{\text{Timer}} \times 250 + 2.T_{\text{Timer}} \right) \times 200 + 2.T_{\text{Timer}} \right) \times 10 + 2.T_{\text{Timer}} = 1006033.T_{\text{Timer}}$$

3. Ví dụ 3: (Tạo thời gian trễ)

Viết chương trình con tạo thời gian trễ $100 \mu\text{s}$ dùng Timer 0. Biết rằng tần số thạch anh là 12 MHz .

Giải

- Tính toán:** Tìm giá trị cần nạp cho bộ định thời và chế độ hoạt động của bộ định thời này:
Theo đề bài ta có: $t_{\text{Delay}} = 100(\mu\text{s})$ và $f_{\text{Osc}} = 12(\text{MHz})$

Giá trị cần nạp cho bộ định thời được tính theo công thức:

$$N = -\frac{f_{\text{Osc}}}{12} \times t_{\text{Delay}} = -\frac{12.10^6}{12} \times 100.10^{-6} = -100$$

Vậy: $N = -100$ hoặc $N = 9\text{CH}$.

Ta có: $t_{\text{Delay}} = 100(\mu\text{s})$

$$T_{\text{Timer}} = \frac{1}{f_{\text{Timer}}} = \frac{12}{f_{\text{Osc}}} = \frac{12}{12.10^6} = 10^{-6}(\text{s}) = 1(\mu\text{s})$$

Vì $t_{\text{Delay}} \leq 256.T_{\text{Timer}}$ (hay N nằm trong khoảng từ -255 đến -1) nên ta có thể chọn Timer ở chế độ 1 (chế độ 16 bit) hoặc chế độ 2 (chế độ 8 bit tự động nạp lại).

- Chương trình:** Dựa vào những tính toán trên, ta có:

⇒ Chương trình con hoàn chỉnh khi sử dụng Timer 0 ở chế độ 2:

DELAY:

```
MOV TMOD, #02H
MOV TH0, #(-100)   hoặc   MOV TH0, #9CH
SETB TR0
JNB TF0, $
CLR TF0
CLR TR0
RET
```

⇒ Chương trình con hoàn chỉnh khi sử dụng Timer 0 ở chế độ 1:

DELAY:

```
MOV TMOD, #01H
MOV TH0, #HIGH(-100)   hoặc   MOV TH0, #0FFH
MOV TL0, #LOW(-100)    hoặc   MOV TL0, #9CH
SETB TR0
JNB TF0, $
CLR TF0
CLR TR0
RET
```

• **Độ chính xác (xét về mặt thời gian) của chương trình:** Khi sử dụng phương pháp tạo thời gian trễ như trên (phương pháp dùng Timer) thì việc định thời gian cũng xuất hiện một sai số. Vì ở đây, để đơn giản trong việc tính toán mà ta đã bỏ qua không tính đến thời gian cần thiết để thực hiện từng lệnh trong chương trình, chỉ quan tâm đến giá trị cần phải nạp cho bộ định thời sao cho Timer định được khoảng thời gian mà ta yêu cầu. Độ chính xác của chương trình định thời khi sử dụng phương pháp này không phụ thuộc vào giá trị cần nạp cho Timer (N) mà nó chỉ phụ thuộc vào số lượng lệnh sử dụng trong chương trình.

Với dạng thứ nhất (chế độ 2) thì t_{Delay} chính xác là: $t_{Delay(cx)} = 11.T_{Timer} + t_{Delay} = 111(\mu s)$

Với dạng thứ hai (chế độ 1) thì t_{Delay} chính xác là: $t_{Delay(cx)} = 13.T_{Timer} + t_{Delay} = 113(\mu s)$

4. Ví dụ 4: (Tạo thời gian trễ)

Viết chương trình con tạo thời gian trễ 10 ms dùng Timer 1. Biết rằng tần số thạch anh là 12 MHz.

Giải

- **Tính toán:** Tìm giá trị cần nạp cho bộ định thời và chế độ hoạt động của bộ định thời này:
Theo đề bài ta có: $t_{Delay} = 10(ms)$ và $f_{Osc} = 12(MHz)$

Giá trị cần nạp cho bộ định thời được tính theo công thức:

$$N = -\frac{f_{Osc}}{12} \times t_{Delay} = -\frac{12.10^6}{12} \times 10.10^{-3} = -10000$$

Vậy: $N = -10000$ hoặc $N = D8F0H$.

Ta có: $t_{Delay} = 10(ms) = 10000(\mu s)$

$$T_{Timer} = \frac{1}{f_{Timer}} = \frac{12}{f_{Osc}} = \frac{12}{12.10^6} = 10^{-6}(s) = 1(\mu s)$$

Vì $t_{Delay} \leq 65536.T_{Timer}$ (hay N nằm trong khoảng từ -65535 đến -1) nên ta chọn Timer ở chế độ 1 (chế độ 16 bit).

- **Chương trình:** Dựa vào những tính toán trên, ta có:

⇒ Chương trình con hoàn chỉnh khi sử dụng Timer 1 ở chế độ 1:

DELAY:

```
MOV TMOD, #10H
MOV TH1, #HIGH(-10000) hoặc MOV TH1, #0D8H
MOV TL1, #LOW(-10000) hoặc MOV TL1, #0F0H
SETB TR1
JNB TF1, $
CLR TF1
CLR TR1
RET
```

- **Độ chính xác (xét về mặt thời gian) của chương trình:**

Với ví dụ trên (chế độ 1) thì t_{Delay} chính xác là: $t_{Delay(cx)} = 13.T_{Timer} + t_{Delay} = 10013(\mu s)$

5. Ví dụ 5: (Tạo thời gian trễ)

Viết chương trình con tạo thời gian trễ 1s dùng Timer 1. Biết rằng tần số thạch anh là 12 MHz.

Giải

- **Tính toán:** Tìm giá trị cần nạp cho bộ định thời và chế độ hoạt động của bộ định thời này:
Theo đề bài ta có:

$$t_{Delay} = 1(s) \text{ và } f_{Osc} = 12(MHz)$$

Giá trị cần nạp cho bộ định thời được tính theo công thức:

$$N = -\frac{f_{Osc}}{12} \times t_{Delay} = -\frac{12.10^6}{12} \times 1 = -1000000$$

Vậy: $N = -1000000$ (giá trị quá lớn không thể nạp trực tiếp vào các thanh ghi THx/TLx).

Ta có: $t_{Delay} = 1(s) = 1000000(\mu s)$

$$T_{Timer} = \frac{1}{f_{Timer}} = \frac{12}{f_{Osc}} = \frac{12}{12.10^6} = 10^{-6}(s) = 1(\mu s)$$

Vì $t_{Delay} > 65536.T_{Timer}$ nên ta chọn phải Timer ở **chế độ 1** (chế độ 16 bit) kết hợp với các thanh ghi để tạo vòng lặp.

Gọi: N' là giá trị cần nạp cho các thanh ghi định thời.

$[Rn]$ là giá trị cần nạp cho thanh ghi kết hợp (vòng lặp).

$$\Rightarrow N = [Rn] \times N'$$

Ta tự chọn: $N' = -10000 \rightarrow [Rn] = 100$

○ Lưu ý:

- N' được chọn sao cho phù hợp với qui định chọn giá trị cần nạp cho các thanh ghi định thời ở chế độ 1.
- $[Rn] \leq 255$, đặc biệt nếu chọn $[Rn] = 0$ thì điều này sẽ tương đương với trường hợp ta chọn $[Rn] = 256$.

$\Rightarrow$ Giá trị cần nạp cho các thanh ghi định thời là **-10000** và giá trị cần nạp cho thanh ghi kết hợp là **100**.

- **Chương trình:** Dựa vào những tính toán trên, ta có:

$\Rightarrow$ Chương trình con hoàn chỉnh khi sử dụng Timer 1 ở chế độ 1:

DELAY:

```
PUSH 00H
MOV TMOD, #10H
MOV R0, #100
```

AAA:

```
MOV TH1, #HIGH(-10000) hoặc MOV TH1, #0D8H
MOV TL1, #LOW(-10000)  hoặc MOV TL1, #0F0H
SETB TR1
JNB TF1, $
CLR TF1
CLR TR1
DJNZ R0, AAA
POP 00H
RET
```

• **Độ chính xác (xét về mặt thời gian) của chương trình:** Trường hợp này cũng tương tự như các trường hợp định thời sử dụng Timer đã nêu ở các ví dụ trên. Tuy nhiên ở đây, độ chính xác của chương trình định thời khi sử dụng phương pháp này không phụ thuộc vào giá trị cần nạp cho Timer (N) mà nó phụ thuộc vào số lượng lệnh sử dụng trong chương trình, số lần lặp lại và số vòng lặp.

Với ví dụ trên (chế độ 1 + vòng lặp) thì t_{Delay} chính xác là:

$$t_{Delay(cx)} = 5.T_{Timer} + \left(11.T_{Timer} + t_{Delay(Timer)} \right) \times 100 + 4.T_{Timer}$$

$$= 1001109(\mu s) = 1,001109(s)$$

Với $t_{Delay(Timer)}$: thời gian định thời của Timer ($10000 \mu s$).

6. Ví dụ 6: (Tạo thời gian trễ)

Viết chương trình con tạo thời gian trễ 60s dùng Timer 0. Biết rằng tần số thạch anh là 12 MHz.

Giải

- **Tính toán:** Tìm giá trị cần nạp cho bộ định thời và chế độ hoạt động của bộ định thời này:
Theo đề bài ta có:

$$t_{Delay} = 60(s) \text{ và } f_{Osc} = 12(MHz)$$

Giá trị cần nạp cho bộ định thời được tính theo công thức:

$$N = -\frac{f_{Osc}}{12} \times t_{Delay} = -\frac{12.10^6}{12} \times 60 = -600000000$$

Vậy: $N = -600000000$ (giá trị quá lớn không thể nạp trực tiếp vào các thanh ghi THx/TLx).

Ta có: $t_{Delay} = 60(s) = 60000000(\mu s)$

$$T_{Timer} = \frac{1}{f_{Timer}} = \frac{12}{f_{Osc}} = \frac{12}{12.10^6} = 10^{-6}(s) = 1(\mu s)$$

Vì $t_{Delay} > 65536.T_{Timer}$ nên ta chọn phải Timer ở chế độ 1 (chế độ 16 bit) kết hợp với các thanh ghi để tạo vòng lặp.

Gọi: N' là giá trị cần nạp cho các thanh ghi định thời.

[Rn] là giá trị cần nạp cho thanh ghi kết hợp (vòng lặp 1).

[Rm] là giá trị cần nạp cho thanh ghi kết hợp (vòng lặp 2).

$$\Rightarrow N = [Rn] \times [Rm] \times N'$$

Ta tự chọn: $N' = -10000 \rightarrow [Rm] = 100 \rightarrow [Rn] = 60$

○ Lưu ý:

- N' được chọn sao cho phù hợp với qui định chọn giá trị cần nạp cho các thanh ghi định thời ở chế độ 1.
- $[Rn], [Rm] \leq 255$, đặc biệt nếu chọn $[Rn], [Rm] = 0$ thì điều này sẽ tương đương với trường hợp ta chọn $[Rn], [Rm] = 256$.

$\Rightarrow$ Giá trị cần nạp cho các thanh ghi định thời là **-10000** và giá trị cần nạp cho các thanh ghi kết hợp là **60** (cho vòng lặp 1), **100** (cho vòng lặp 2).

- **Chương trình:** Dựa vào những tính toán trên, ta có:

⇒ Chương trình con hoàn chỉnh khi sử dụng Timer 0 ở chế độ 1:

DELAY:

```
PUSH 00H
PUSH 01H
MOV TMOD, #01H
MOV R0, #60
```

AAA:

```
MOV R1, #100
```

BBB:

```
MOV TH0, #HIGH(-10000) hoặc MOV TH0, #0D8H
MOV TL0, #LOW(-10000)  hoặc MOV TL0, #0F0H
SETB TR0
JNB TF0, $
CLR TF0
CLR TR0
DJNZ R1, BBB
DJNZ R0, AAA
POP 01H
POP 00H
RET
```

• **Độ chính xác (xét về mặt thời gian) của chương trình:** Trường hợp này cũng tương tự như các trường hợp định thời sử dụng Timer đã nêu ở các ví dụ trên. Tuy nhiên ở đây, độ chính xác của chương trình định thời khi sử dụng phương pháp này không phụ thuộc vào giá trị cần nạp cho Timer (N) mà nó phụ thuộc vào số lượng lệnh sử dụng trong chương trình, số lần lặp lại và số vòng lặp.

Với ví dụ trên (chế độ 1 + vòng lặp) thì t_{Delay} chính xác là:

$$t_{\text{Delay}(cx)} = 7.T_{\text{Timer}} + \left(1.T_{\text{Timer}} + \left(11.T_{\text{Timer}} + t_{\text{Delay}(\text{Timer})} \right) \times 100 + 2.T_{\text{Timer}} \right) \times 60 + 6.T_{\text{Timer}}$$

$$= 60066193(\mu s) = 60,066193(s)$$

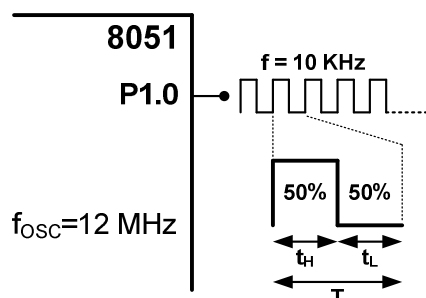
Với $t_{\text{Delay}(\text{Timer})}$: thời gian định thời của Timer (10000 μs).

7. Ví dụ 7: (Tạo sóng vuông)

Viết chương trình tạo sóng vuông có tần số 10 KHz ở ngõ ra P1.0 và có chu kỳ làm việc D=50%. Biết rằng tần số thạch anh là 12 MHz và sử dụng bộ định thời 0.

Giải

- **Tính toán:**



Theo đề bài, ta có chu kỳ làm việc $D=50\%$ cho nên:

$$t_H = 50\% \times T = 50\% \times \frac{1}{f} = 0,5 \times \frac{1}{10.10^3} = 5.10^{-5}(s) = 50(\mu s)$$

$\Rightarrow t_H = 50(\mu s)$ và $t_L = 50(\mu s)$.

Vậy ở đây ta phải dùng Timer 0 để tạo thời gian trễ $50(\mu s)$ cho thời gian sáng ở mức cao và $50(\mu s)$ cho thời gian sáng ở mức thấp.

Theo như trên, ta có:

$$t_{Delay} = 50(\mu s) \text{ và } f_{Osc} = 12(MHz)$$

Giá trị cần nạp cho bộ định thời được tính theo công thức:

$$N = -\frac{f_{Osc}}{12} \times t_{Delay} = -\frac{12.10^6}{12} \times 50.10^{-6} = -50$$

Vậy: $N = -50$ hoặc $N = CEH$.

Ta có: $t_{Delay} = 50(\mu s)$

$$T_{Timer} = \frac{1}{f_{Timer}} = \frac{12}{f_{Osc}} = \frac{12}{12.10^6} = 10^{-6}(s) = 1(\mu s)$$

Vì $t_{Delay} \leq 256.T_{Timer}$ (hay N nằm trong khoảng từ -255 đến -1) nên ta có thể chọn Timer ở chế độ 1 (chế độ 16 bit) hoặc chế độ 2 (chế độ 8 bit tự động nạp lại).

- **Chương trình:** Dựa vào những tính toán trên, ta có:

MAIN:

```
SETB P1.0
ACALL DELAY50US
CLR P1.0
ACALL DELAY50US
SJMP MAIN
```

DELAY50US:

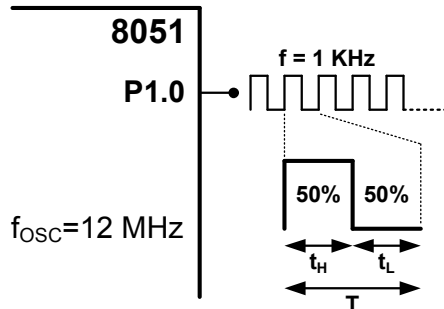
```
MOV TMOD, #02H
MOV TH0, #(-50) hoặc MOV TH0, #0CEH
SETB TR0
JNB TF0, $
CLR TR0
CLR TF0
RET
END
```

8. Ví dụ 8: (Tạo sóng vuông)

Viết chương trình tạo sóng vuông có tần số 1 KHz ở ngõ ra P1.0 và có chu kỳ làm việc $D=50\%$. Biết rằng tần số thạch anh là 12 MHz và sử dụng bộ định thời 0.

Giải

• Tính toán:



Theo đề bài, ta có chu kỳ làm việc $D=50\%$ cho nên:

$$t_H = 50\% \times T = 50\% \times \frac{1}{f} = 0,5 \times \frac{1}{1.10^3} = 5.10^{-4}(s) = 500(\mu s)$$

$\Rightarrow t_H = 500 (\mu s)$ và $t_L = 500 (\mu s)$.

Vậy ở đây ta phải dùng Timer 0 để tạo thời gian trễ 500(μs) cho thời gian sóng ở mức cao và 500(μs) cho thời gian sóng ở mức thấp.

Theo như trên, ta có:

$$t_{Delay} = 500(\mu s) \text{ và } f_{Osc} = 12(MHz)$$

Giá trị cần nạp cho bộ định thời được tính theo công thức:

$$N = -\frac{f_{Osc}}{12} \times t_{Delay} = -\frac{12.10^6}{12} \times 500.10^{-6} = -500$$

Vậy: $N = -500$ hoặc $N = FE0CH$.

Ta có: $t_{Delay} = 500(\mu s)$

$$T_{Timer} = \frac{1}{f_{Timer}} = \frac{12}{f_{Osc}} = \frac{12}{12.10^6} = 10^{-6}(s) = 1(\mu s)$$

Vì $t_{Delay} \leq 65536.T_{Timer}$ (hay N nằm trong khoảng từ -65535 đến -1) nên ta chọn Timer ở chế độ 1 (chế độ 16 bit).

• Chương trình: Dựa vào những tính toán trên, ta có:

MAIN:

```
SETB P1.0
ACALL DELAY500US
CLR P1.0
ACALL DELAY500US
SJMP MAIN
```

DELAY500US:

```
MOV TMOD, #01H
MOV TH0, #HIGH(-500) hoặc MOV TH0, #0FEH
MOV TL0, #LOW(-500) hoặc MOV TL0, #0CH
```

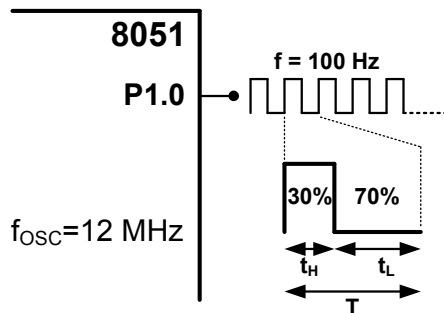
```

SETB TR0
JNB TF0, $
CLR TR0
CLR TF0
RET
END

```

9. Ví dụ 9: (Tạo sóng vuông)

Viết chương trình tạo sóng vuông có tần số 100 Hz ở ngõ ra P1.0 và có chu kỳ làm việc $D=30\%$. Biết rằng tần số thạch anh là 12 MHz và sử dụng bộ định thời 0.

Giải• Tính toán:

Theo đề bài, ta có chu kỳ làm việc $D=30\%$ cho nên:

$$t_H = 30\% \times T = 30\% \times \frac{1}{f} = 0,3 \times \frac{1}{100} = 3 \cdot 10^{-3} (s) = 3000 (\mu s)$$

$\Rightarrow t_H = 3000 (\mu s)$ và $t_L = 7000 (\mu s)$.

Vậy ở đây ta phải dùng Timer 0 để tạo thời gian trễ 3000(μs) cho thời gian sóng ở mức cao và 7000(μs) cho thời gian sóng ở mức thấp.

Theo như trên, ta có (xét trường hợp t_H):

$$t_{Delay} = 3000 (\mu s) \text{ và } f_{Osc} = 12 (MHz)$$

Giá trị cần nạp cho bộ định thời được tính theo công thức:

$$N = -\frac{f_{Osc}}{12} \times t_{Delay} = -\frac{12 \cdot 10^6}{12} \times 3000 \cdot 10^{-6} = -3000$$

Vậy: $N = -3000$ hoặc $N = F448H$.

Ta có: $t_{Delay} = 3000 (\mu s)$

$$T_{Timer} = \frac{1}{f_{Timer}} = \frac{12}{f_{Osc}} = \frac{12}{12 \cdot 10^6} = 10^{-6} (s) = 1 (\mu s)$$

Vì $t_{Delay} \leq 65536 \cdot T_{Timer}$ (hay N nằm trong khoảng từ -65535 đến -1) nên ta chọn Timer ở chế độ 1 (chế độ 16 bit).

Tương tự như trên, ta có (xét trường hợp t_L):

$$t_{Delay} = 7000 (\mu s) \text{ và } f_{Osc} = 12 (MHz)$$

Giá trị cần nạp cho bộ định thời được tính theo công thức:

$$N = -\frac{f_{Osc}}{12} \times t_{Delay} = -\frac{12.10^6}{12} \times 7000.10^{-6} = -7000$$

Vậy: $N = -7000$ hoặc $N = E4A8H$.

Ta có: $t_{Delay} = 7000(\mu s)$

$$T_{Timer} = \frac{1}{f_{Timer}} = \frac{12}{f_{Osc}} = \frac{12}{12.10^6} = 10^{-6}(s) = 1(\mu s)$$

Vì $t_{Delay} \leq 65536.T_{Timer}$ (hay N nằm trong khoảng từ -65535 đến -1) nên ta chọn Timer ở chế độ 1 (chế độ 16 bit).

- **Chương trình:** Dựa vào những tính toán trên, ta có:

MAIN:

```
SETB P1.0
ACALL DELAY3000US
CLR P1.0
ACALL DELAY7000US
SJMP MAIN
```

DELAY3000US:

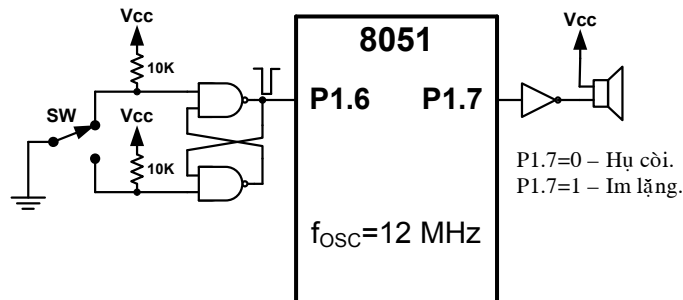
```
MOV TMOD, #01H
MOV TH0, #HIGH(-3000) hoặc MOV TH0, #0F4H
MOV TL0, #LOW(-3000) hoặc MOV TL0, #48H
SETB TR0
JNB TF0, $
CLR TR0
CLR TF0
RET
```

DELAY7000US:

```
MOV TMOD, #01H
MOV TH0, #HIGH(-7000) hoặc MOV TH0, #0E4H
MOV TL0, #LOW(-7000) hoặc MOV TL0, #0A8H
SETB TR0
JNB TF0, $
CLR TR0
CLR TF0
RET
END
```

10. Ví dụ 10: (Giao tiếp với thiết bị ngoại vi)

Một còi được nối với chân P1.7 và một chuyển mạch (có chống dội) được nối với chân P1.6 của chip 8051 (xem trong hình vẽ). Viết chương trình điều khiển đọc mức logic cung cấp bởi chuyển mạch (khi chuyển mạch thay đổi từ vị trí trên xuống vị trí dưới thì một xung mức thấp được tạo ra tại chân P1.6) và hụ còi trong thời gian 1sec sau mỗi lần phát hiện sự chuyển trạng thái từ 1 xuống 0 tại chân P1.6.



Giải

```

HUNDRED      EQU 100                ; Khai báo biến
COUNT       EQU -10000
ORG 0000H

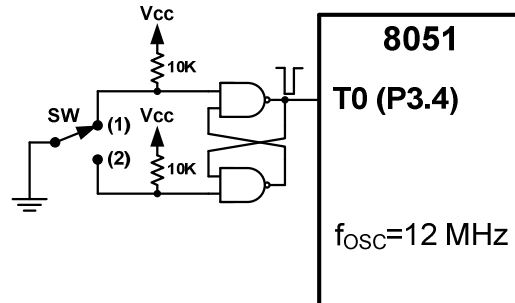
MAIN:
    JNB P1.6, $      ;Chờ logic 1 ở ngõ vào P1.6.
    JB P1.6, $       ;Chờ logic 0 ở ngõ vào P1.6
    SETB P1.7        ;Còi hụ.
    ACALL DELAY       ;Thời gian 1 giây.
    CLR P1.7         ;Tắt còi.
    SJMP MAIN

DELAY:
    PUSH 00H
    MOV TMOD, #10H
    MOV R0, # HUNDRED

AAA:
    MOV TH1, #HIGH(COUNT)
    MOV TL1, #LOW(COUNT)
    SETB TR1
    JNB TF1, $
    CLR TF1
    CLR TR1
    DJNZ R0, AAA
    POP 00H
    RET
END
    
```

11. Ví dụ 11: (Đếm xung)

Một chuyển mạch (có chống dội) được nối với chân T0 (P3.4) của chip 8051. Viết chương trình điều khiển đếm số lượng xung được tạo ra bởi chuyển mạch (khi chuyển mạch thay đổi từ vị trí (1) sang vị trí (2) thì một xung mức thấp được tạo ra tại chân T0). Số xung đếm được sẽ chứa trong RAM nội (dạng số HEX) tại các ô nhớ có địa chỉ bắt đầu tại 40H. Biết rằng số lượng xung tạo ra được không chế nằm trong khoảng 0 – 255 xung.



Giải

- **Tính toán:**

Theo yêu cầu của đề bài:

- Viết chương trình đếm xung → Cấu hình cho Timer 0 là một bộ đếm xung (Counter).
- Số xung nằm trong khoảng 0 – 255 xung → Chọn chế độ 8 bit (chế độ 2).

- **Chương trình:** Dựa vào những tính toán trên, ta có:

MAIN:

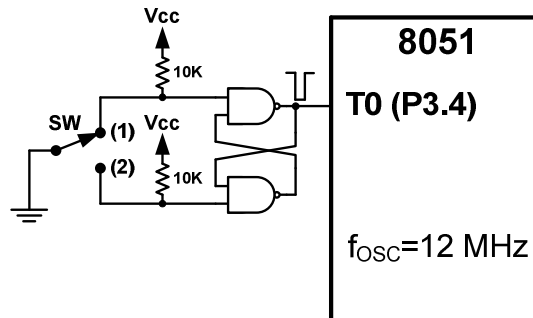
MOV	TMOD, #06H	;Chế độ Counter 8 bit (chế độ 2).
MOV	TH0, #00H	;Giá trị ban đầu của bộ đếm.
SETB	P3.4	;Cấu hình P3.4 là ngõ vào.
SETB	TR0	;Khởi động bộ đếm.

LOOP:

MOV	A, TL0	;Đọc số xung từ bộ đếm.
MOV	40H, A	;Cất số xung đếm được vào ô 40H.
JNB	TF0, LOOP	;Tiếp tục quá trình đếm xung nếu bộ đếm chưa bị tràn.
CLR	TF0	;Xoá cờ tràn.
CLR	TR0	;Dừng bộ đếm.
END		;Kết thúc chương trình.

12. Ví dụ 12: (Đếm xung)

Một chuyển mạch (có chống dội) được nối với chân T0 (P3.4) của chip 8051. Viết chương trình điều khiển đếm số lượng xung được tạo ra bởi chuyển mạch (khi chuyển mạch thay đổi từ vị trí (1) sang vị trí (2) thì một xung mức thấp được tạo ra tại chân T0). Số xung đếm được sẽ chứa trong RAM nội (dạng số HEX) tại các ô nhớ có địa chỉ bắt đầu tại 50H. Biết rằng số lượng xung tạo ra được không chế nằm trong khoảng 0 – 65535 xung.



Giải

- Tính toán:**

Theo yêu cầu của đề bài:

- Viết chương trình đếm xung → Cấu hình cho Timer 0 là một bộ đếm xung (Counter).
- Số xung nằm trong khoảng 0 – 65535 xung → Chọn chế độ 16 bit (chế độ 1).

- Chương trình:** Dựa vào những tính toán trên, ta có:

MAIN:

MOV	TMOD, #05H	;Chế độ Counter 16 bit (chế độ 1).
MOV	TH0, #00H	;Giá trị ban đầu của bộ đếm (cao).
MOV	TL0, #00H	;Giá trị ban đầu của bộ đếm (thấp).
SETB	P3.4	;Cấu hình P3.4 là ngõ vào.
SETB	TR0	;Khởi động bộ đếm.

LOOP:

MOV	A, TH0	;Đọc giá trị của bộ đếm đang hoạt động.
MOV	50H, TL0	;Đọc số xung đếm được (phần cao).
CJNE	A, TH0, LOOP	;Cắt số xung đếm được (phần thấp).
		;Đọc số xung đếm được (phần cao)
		;lần nữa để kiểm tra.
MOV	51H, A	;Cắt số xung đếm được (phần cao).
JNB	TF0, LOOP	;Tiếp tục quá trình đếm xung nếu
		;bộ đếm chưa bị tràn.
CLR	TF0	;Xóa cờ tràn.
CLR	TR0	;Dừng bộ đếm.
END		;Kết thúc chương trình.

XI. PHẦN BÀI TẬP:• **Tạo trễ:**

Bài 1: Viết chương trình con mang tên DELAY500US có nhiệm vụ tạo trễ 0,5ms dùng Timer. ($f_{OSC}=6MHz$).

Bài 2: Viết chương trình con mang tên DELAY10MS có nhiệm vụ tạo trễ 10ms dùng Timer. ($f_{OSC}=12MHz$).

Bài 3: Viết chương trình con mang tên DELAY10S có nhiệm vụ tạo trễ 10s dùng Timer. ($f_{OSC}=12MHz$).

Bài 4: Viết chương trình con mang tên DELAY1S có nhiệm vụ tạo trễ 1s dùng Timer. ($f_{OSC}=11,0592MHz$).

Bài 5: Viết chương trình con delay 100s, biết rằng f_{OSC} dùng trong hệ thống là:

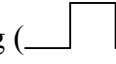
- 6 MHz.
- 11,0592 MHz.
- 12 MHz.
- 24 MHz.

Bài 6: Viết chương trình con delay 100ms, biết rằng f_{OSC} dùng trong hệ thống là:

- 6 MHz.
- 11,0592 MHz.
- 12 MHz.
- 24 MHz.

Bài 7: Viết chương trình con delay 1s, biết rằng f_{OSC} dùng trong hệ thống là:

- 6 MHz.
- 11,0592 MHz.
- 12 MHz.
- 24 MHz.

Bài 8: Viết đoạn lệnh tạo một xung dương () tại chân P1.0 với độ rộng xung 1ms, biết rằng $f_{OSC}=12MHz$.

• **Tạo xung:**

Bài 1: Dùng chương trình con DELAY500US (*Bài 1 phần tạo trễ*) để viết đoạn lệnh tạo sóng vuông $f=1KHz$ tại P1.0.

Bài 2: Dùng chương trình con DELAY10MS (*Bài 2 phần tạo trễ*) để viết đoạn lệnh tạo sóng vuông $f=50Hz$ tại P1.1.

Bài 3: Dùng chương trình con DELAY500US (*Bài 1 phần tạo trễ*) để viết đoạn lệnh tạo sóng vuông $f=500Hz$ ($D=25\%$) tại P1.2.

Bài 4: Dùng chương trình con DELAY10MS (*Bài 2 phần tạo trễ*) để viết đoạn lệnh tạo sóng vuông $f=20Hz$ ($D=20\%$) tại P1.3.

Bài 5: Viết đoạn lệnh tạo chuỗi xung vuông có $f=100KHz$ tại chân P1.1 ($f_{OSC}=12MHz$).

Bài 6: Viết đoạn lệnh tạo chuỗi xung vuông có $f=100KHz$ và có chu kỳ làm việc $D=40\%$ tại chân P1.2 ($f_{OSC}=12MHz$).

Bài 7: Viết đoạn lệnh tạo chuỗi xung vuông có $f=10KHz$ tại chân P1.3 ($f_{OSC}=24MHz$).

Bài 8: Viết đoạn lệnh tạo chuỗi xung vuông có $f = 10 \text{ KHz}$ và có chu kỳ làm việc $D = 30\%$ tại chân P1.3 ($f_{OSC} = 24 \text{ MHz}$).

Bài 9: Viết đoạn lệnh tạo chuỗi xung vuông có $f = 10 \text{ Hz}$ tại chân P1.4 ($f_{OSC} = 12 \text{ MHz}$).

Bài 10: Viết đoạn lệnh tạo chuỗi xung vuông có $f = 10 \text{ Hz}$ và có chu kỳ làm việc $D = 25\%$ tại chân P1.5 ($f_{OSC} = 11,0592 \text{ MHz}$).

Bài 11: Viết đoạn lệnh dùng Timer tạo sóng vuông $f = 500 \text{ Hz}$ tại P1.4. ($f_{OSC} = 12 \text{ MHz}$).

Bài 12: Viết đoạn lệnh dùng Timer tạo sóng vuông $f = 20 \text{ KHz}$ tại P1.5. ($f_{OSC} = 24 \text{ MHz}$).

Bài 13: Viết đoạn lệnh dùng Timer tạo 2 sóng vuông có cùng $f = 1 \text{ KHz}$ tại P1.6 và P1.7. Biết rằng sóng vuông tại P1.7 chậm pha hơn sóng vuông tại P1.6 là $100 \mu\text{s}$. ($f_{OSC} = 12 \text{ MHz}$).

Bài 14: Viết đoạn lệnh dùng Timer điều khiển đèn giao thông tại một giao lộ. Cho biết rằng:

Đèn	Bit điều khiển	Thời gian
Xanh 1	P1.0	25s
Vàng 1	P1.1	3s
Đỏ 1	P1.2	
Xanh 2	P1.3	33s
Vàng 2	P1.4	3s
Đỏ 2	P1.5	

Đèn sáng khi bit điều khiển bằng 0 ($f_{OSC} = 12 \text{ MHz}$).