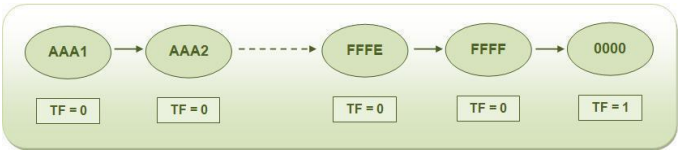
[Timer/Counter] là 1 bộ đếm lên (tự động)

(Trong môn này, mình chỉ dùng Timer làm hàm delay)

1. **Giới thiệu Timer0**
   - Gồm có **2 bộ:** Timer0, Timer1 (Timer2)
   - Một Timer, có 4 mode cấu hình, nhưng chúng ta quan tâm **2 mode**:
     - ⇨ **Mode 8 bit, tự động nạp lại** (Tự động nạp lại giá trị bắt đầu đếm)
     - ⇨ **Mode 16 bit** (Bạn phải nạp lại giá trị bắt đầu, mỗi lần đếm xong)

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
		TH0										**TL0**			

   - Bộ đếm 8 bit: Dùng **TL0 để chứa giá trị đếm lên**. (**TH0 chứa giá trị auto nạp lại**.)
   - Bộ đếm 16 bit: Dùng **2 thanh ghi TH0, TL0 để chứa giá trị đếm**.
2. **Giá trị bắt đầu đếm**: Do bạn chọn tùy ý!
   - Mode 8 bit: chọn từ 0 đến 255
   - Mode 16 bit: chọn từ 0 đến 65535
3. **Timer đếm lên đến khi nào?** Đếm đến "tràn khả năng chứa" của thanh ghi.
   - Mode 8 bit: Tràn khi giá trị đang là 255, và đếm thêm 1 nữa, thành 256 thì tràn!)
   - Mode 16 bit: Tràn khi giá trị đang là 65535 và đếm thêm 1 nữa thành 65536 thì tràn!)
4. **Tốc độ đếm lên:** Ftimer = Fmc = Fcrystal/12 = 1MHz (Giả sử chọn Thạch anh: 12MHz)
   - ⇨ Thời gian đếm lên 1 giá trị: Ttimer = 1/Ftimer = 1us.
   - ⇨ Mode 8 bit: nếu đếm từ 0, thì tốn tổng 256us để "tràn".
   - ⇨ Mode 16 bit: nếu đếm từ 0, thì tốn tổng 65536us để "tràn".
5. **Khi Timer đếm tràn thì chuyện gì xảy ra?**
   - Cờ ngắt của Timer0 (cờ tràn – TF0) lên 1. (mặc định = 0)



   - Chương trình nhảy vào đoạn code xử lý ngắt của Timer0 (interrupt 1)

     (Chỉ khi bạn có cấu hình các bit cho phép ngắt: EA, ET0)
     - ⇨ Cờ Timer tự động được xóa về 0.
     - ⇨ Mode 16b: Cần nạp lại giá trị bắt đầu đếm cho Timer (mode 8b thì không cần)

- Timer bắt đầu đếm lại từ giá trị nạp lại!

6. Vùng bộ nhớ ngắt của Timer:

Interrupt	ROM Location(Hex)	Pin	Flag Clearing	Interrupt no. in C
Reset	0000	9	Auto	--
External HW Interrupt 0 (INT0)	0003	P3.2(12)	Auto	0
Timer 0 Interrupt(TF0)	000B	-	Auto	1
External HW Interrupt 1 (INT1)	0013	P3.3(13)	Auto	2
Timer 1 Interrupt(TF1)	001B	-	Auto	3
Serial Com Interrupt(RI and TI)	0023	-	Program SW	4

7. Tính toán giá trị ban đầu để tạo bộ đếm tràn, T = 100us

- Mode 8 bit: Công thức $T = T_{Timer} * (256 – TL0)$
  - ⇨ $100us = 1us * (256 – TL0)$
  - ⇨ $TL0 = 256 – 100 = 156$. (Bạn nạp 1 lần duy nhất, TL0=TH0=156)
- Mode 16 bit: Công thức $T = T_{Timer} * (65536 – TH0TL0)$
  - ⇨ $100us = 1us * (65536 – TH0TL0)$
  - ⇨ $TH0TL0 = 65536 – 100 = 65436 = 0xFF9C$
  - ⇨ TH0 = 0xFF, TL0 = 0x9C (Nhớ nạp lại 2 con số này cho TH0, TL0 sau mỗi lần đếm tràn)

8. Tính toán giá trị ban đầu để tạo ra bộ đếm tràn, T= 50ms

- Mode 8 bit: Công thức $T = T_{Timer} * (256 – TL0)$
  - ⇨ $50000us = 1us * (256 – TL0)$
  - ⇨ $TL0 = 256 – 50000 < 0$ (Không thể!)
- Mode 16 bit: Công thức $T = T_{Timer} * (65536 – TH0TL0)$
  - ⇨ $50000us = 1us * (65536 – TH0TL0)$
  - ⇨ $TH0TL0 = 65536 – 50000 = 15536 = 0x3CB0$
  - ⇨ TH0 = 0x3C, TL0 = 0xB0

9. Chương trình tạo hàm ngắt delay 1ms sử dụng Timer0

- Đặc điểm:
  - Timer0 luôn chạy,
  - Mỗi lần Timer0 đếm tràn: thì sẽ được 1ms, và chương trình sẽ nhảy vào hàm Ngắt để tăng biến dcount lên 1.
  - Hàm delayMS(): bị treo tại chỗ cho đến khi dcount đếm lên bằng với thời gian t.

```
1   #include <REGX52.H>
2   sbit led    = P2^1;
3   unsigned int dcount=0;
4   void ISR_timer0() interrupt 1       //Timer0 always runs!!!
5   {
6       TH0  = 0xFC;
7       TL0  = 0x18;
8       dcount++;
9   }
10
11  void Timer0_1ms_init(void)
12  {
13      TMOD = 0x01;    //Timer 0, Mode 1(16bit )
14      TH0  = 0xFC;    //THTL = 0xFC18 : overflow 1ms
15      TL0  = 0x18;
16      EA   = 1;           //Enable Global Interrupt
17      ET0  = 1;           //Enable Timer0 Interrupt
18      TR0  = 1;       //Start Timer0
19  }
20
21  void delayMS(unsigned int t)
22  {
23      dcount = 0;
24      while (dcount<t);
25  }
26  void main(void) {
27      Timer0_1ms_init();
28      while(1)
29      {
30          led = !led;
31          delayMS(500);
32      }
33  }
```

10. Các thanh ghi SFR liên quan Timer

[TMOD] Chọn chế độ đếm cho Timer 0

7	6	5	4	3	2	1	0
GATE1	C/T1#	M11	M01	GATE0	C/T0#	M10	M00

Bit Number	Bit Mnemonic	Description
7	GATE1	**Timer 1 Gating Control Bit** Clear to enable Timer 1 whenever TR1 bit is set. Set to enable Timer 1 only while INT1# pin is high and TR1 bit is set.
6	C/T1#	**Timer 1 Counter/Timer Select Bit** Clear for Timer operation: Timer 1 counts the divided-down system clock. Set for Counter operation: Timer 1 counts negative transitions on external pin T1.
5	M11	**Timer 1 Mode Select Bits** M11 M01 Operating mode 0 0 Mode 0: 8-bit Timer/Counter (TH1) with 5-bit prescaler (TL1). 0 1 Mode 1: 16-bit Timer/Counter. 1 0 Mode 2: 8-bit auto-reload Timer/Counter (TL1) (1) 1 1 Mode 3: Timer 1 halted. Retains count
4	M01	
3	GATE0	**Timer 0 Gating Control Bit** Clear to enable Timer 0 whenever TR0 bit is set. Set to enable Timer/Counter 0 only while INT0# pin is high and TR0 bit is set.
2	C/T0#	**Timer 0 Counter/Timer Select Bit** Clear for Timer operation: Timer 0 counts the divided-down system clock. Set for Counter operation: Timer 0 counts negative transitions on external pin T0.
1	M10	**Timer 0 Mode Select Bit** M10 M00 Operating mode 0 0 Mode 0: 8-bit Timer/Counter (TH0) with 5-bit prescaler (TL0). 0 1 Mode 1: 16-bit Timer/Counter. 1 0 Mode 2: 8-bit auto-reload Timer/Counter (TL0) (2) 1 1 Mode 3: TL0 is an 8-bit Timer/Counter TH0 is an 8-bit Timer using Timer 1's TR0 and TF0 bits.
0	M00	

1. Reloaded from TH1 at overflow.
2. Reloaded from TH0 at overflow.

Reset Value = 0000 0000b

[IE] Cấu hình cho phép ngắt Timer 0

7	6	5	4	3	2	1	0
EA	EC	ET2	ES	ET1	EX1	ET0	EX0

Bit Number	Bit Mnemonic	Description
7	EA	**Enable All Interrupt bit** Clear to disable all interrupts. Set to enable all interrupts. If EA=1, each interrupt source is individually enabled or disabled by setting or clearing its interrupt enable bit.
3	ET1	**Timer 1 Overflow Interrupt Enable bit** Clear to disable timer 1 overflow interrupt. Set to enable timer 1 overflow interrupt.
1	ET0	**Timer 0 Overflow Interrupt Enable bit** Clear to disable timer 0 overflow interrupt. Set to enable timer 0 overflow interrupt.

[TCON] Thanh ghi cho phép Timer 0 bắt đầu, và chứa cờ báo tràn

7	6	5	4	3	2	1	0
TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0

Bit Number	Bit Mnemonic	Description
7	TF1	**Timer 1 Overflow Flag** Cleared by hardware when processor vectors to interrupt routine. Set by hardware on Timer/Counter overflow, when Timer 1 register overflows.
6	TR1	**Timer 1 Run Control Bit** Clear to turn off Timer/Counter 1. Set to turn on Timer/Counter 1.
5	TF0	**Timer 0 Overflow Flag** Cleared by hardware when processor vectors to interrupt routine. Set by hardware on Timer/Counter overflow, when Timer 0 register overflows.
4	TR0	**Timer 0 Run Control Bit** Clear to turn off Timer/Counter 0. Set to turn on Timer/Counter 0.

11. Câu hỏi

1. Gọi lên bảng, chỉ 1 câu lệnh trong chương trình mẫu và yêu cầu giải thích.

12. Bài tập áp dụng

1. Viết chạy lại bài ở mục 9. Gắn OSC (dao động ký) để xem dạng sóng (Proteus).
2. Tạo xung PWM, T=100us (Ton=Toff=50us). Xuất ra pin P1.1. Gắn OSC xem dạng sóng
   Yêu cầu:
   - Tính TH,TL để Timer tràn đúng bằng 50us.
   - Đảo trạng thái pin P1.1 trong hàm Ngắt của Timer.
   - Vòng while(1){} trong main để trống (lặp tại chỗ, không làm gì).
3. Viết chương trình dùng 2 nút nhấn để điều khiển 1 động cơ DC, với yêu cầu sau:
   a. Nhấn nút START: Động cơ DC chạy 5s rồi dừng, led 7 đoạn đếm từ 5 về 0.
      *Lưu ý: Khi động cơ đang chạy, thì nhấn nút START không có tác dụng.
   b. Nhấn nút STOP: Động cơ nếu đang chạy thì bị dừng tức thì.
4. [Không bắt buộc] Viết chương trình với các yêu cầu sau:
   a. hiệu ứng 8 led đơn sáng dần từ trái sang phải, chu kỳ 1s/lần
   b. sử dụng hàm ngắt của Timer0: để cứ 1s lại cập nhật trạng thái mới cho led.
   c. vòng lặp while(1) trong main để trống (lặp tại chỗ, không làm gì)