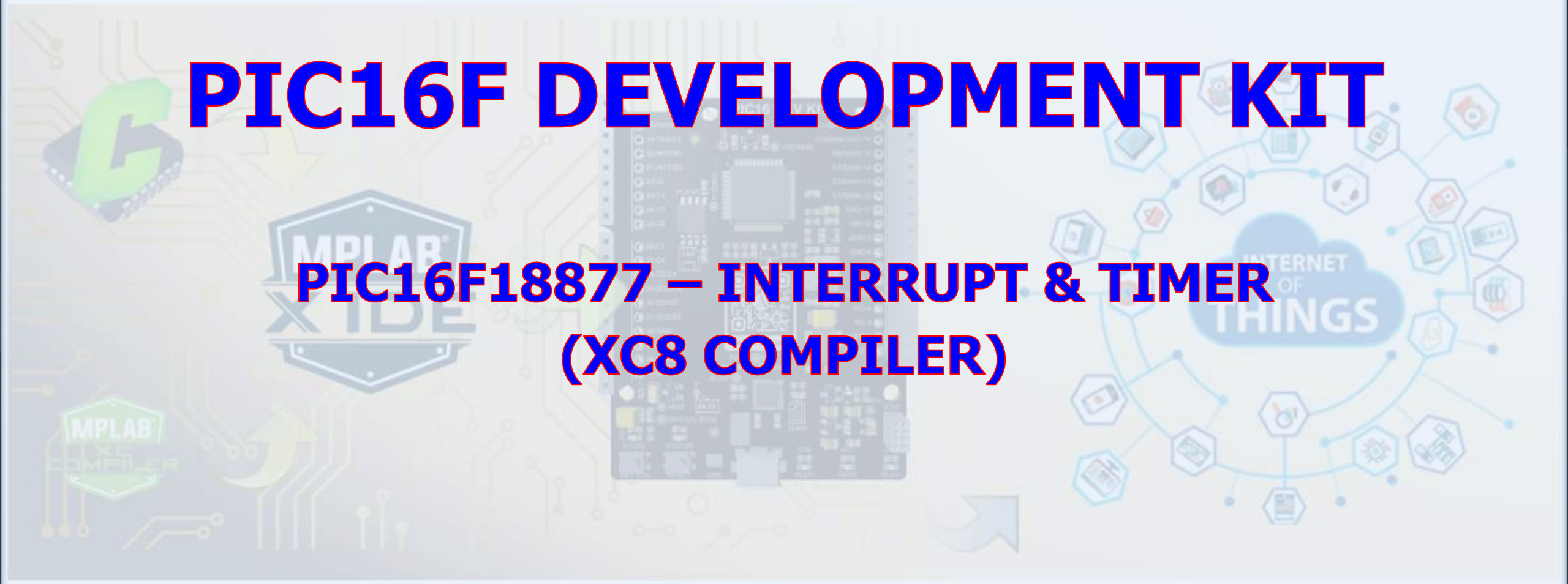
PIC16F DEVELOPMENT KIT

PIC16F18877 – INTERRUPT & TIMER
(XC8 COMPILER)



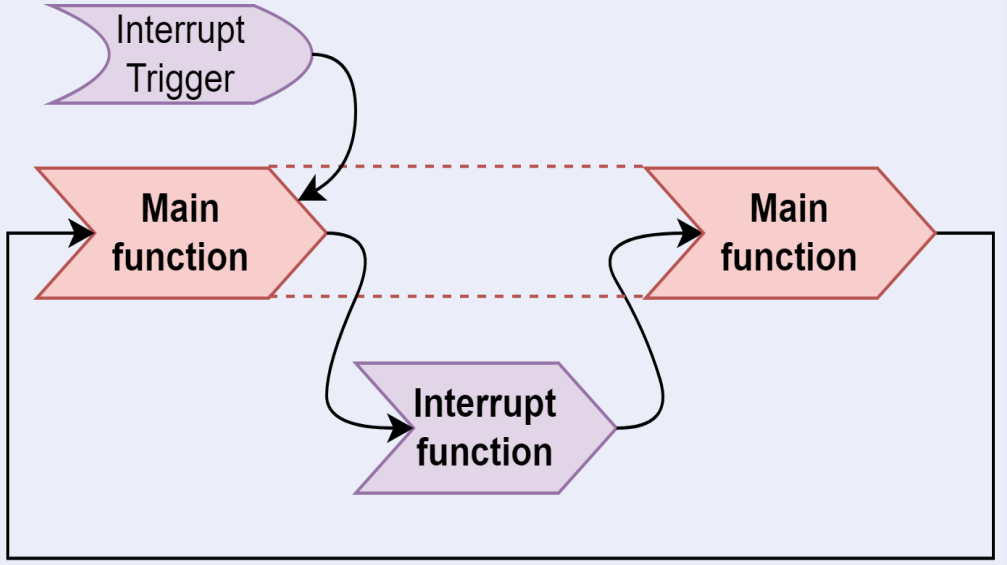
NỘI DUNG

- INTERRUPT
- TIMER

INTERRUPT
1. TỔNG QUAN

INTERRUPT
TIMER

- Ngắt thực thi một chương trình ưu tiên hơn chương trình chính (main function).
- Ngắt xử lý một sự kiện không xác định thời điểm xảy ra.
- Ngắt xử lý sự kiện theo chu kỳ.



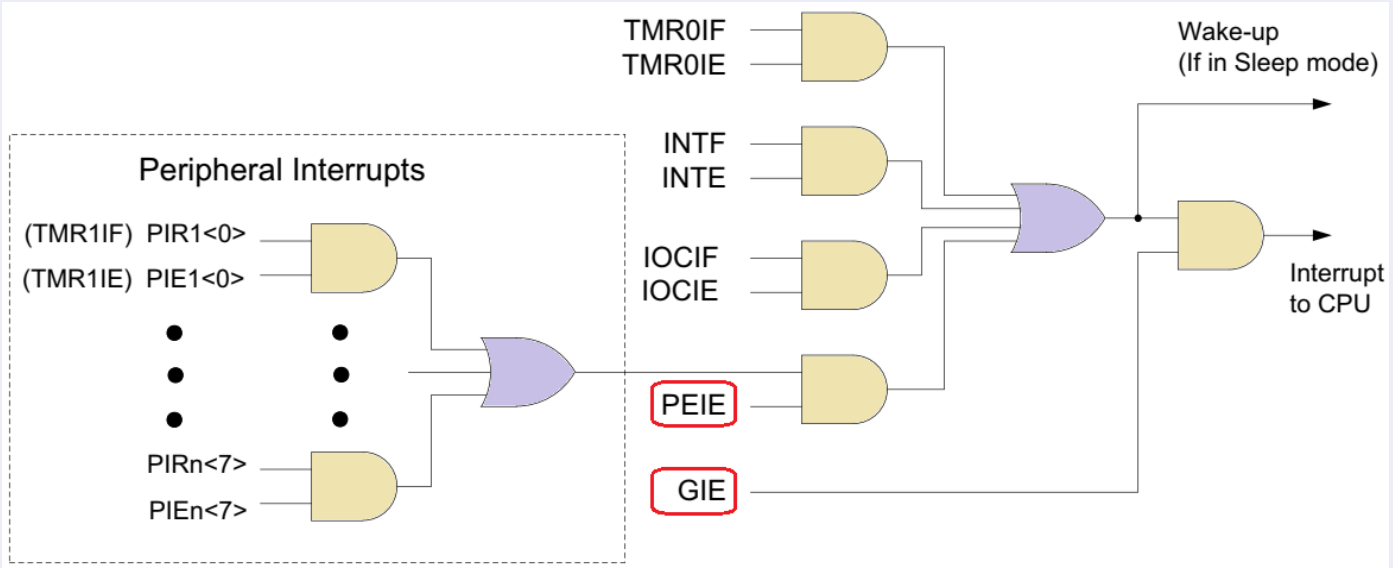
INTERRUPT
2. CÁC NGUỒN NGẮT

INTERRUPT
TIMER

❖ PIC16F18877 có 1 vector ngắt tại địa chỉ 0x0004.

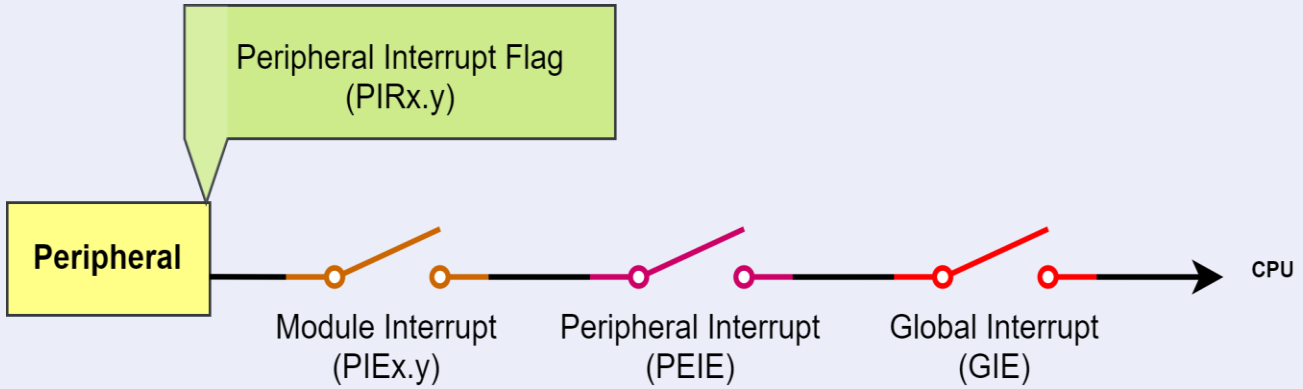
❖ Có 2 nhóm ngắt:

- Ngắt ngoại vi (peripheral interrupts), điều khiển bởi bit PEIE.
- Ngắt ngoài ngoại vi, Điều khiển bởi bit GIE. Bit này cũng đồng thời điều khiển tất cả các nguồn ngắt khác.



INTERRUPT
3. THANH GHI NGẮT

- Thanh ghi điều khiển ngắt toàn cục INTCON.
- Thanh ghi điều khiển ngắt ngoại vi PIEx.
- Thanh ghi cờ ngắt PIRx.
- Thanh ghi điều khiển module ngoại vi cần ngắt.



INTERRUPT
3. THANH GHI NGẮT

INTERRUPT
TIMER

❖ Thanh ghi điều khiển ngắt INTCON:

INTCON			
7	6	5:1	0
GIE	PEIE	RFU	INTEDG

Interrupt Edge Select bit
1 = Interrupt on rising edge of INT pin
0 = Interrupt on falling edge of INT pin

Peripheral Interrupt Enable bit
1 = Enables all active peripheral interrupts
0 = Disables all peripheral interrupts

Global Interrupt Enable bit
1 = Enables all active interrupts
0 = Disables all interrupts

INTERRUPT
4. CHƯƠNG TRÌNH NGẮT

INTERRUPT

TIMER

- Chương trình ngắt đặt bất kỳ vị trí nào trong file source.
- Cấu trúc:

```
void __interrupt() FuncName(void)
{
    if(INCONbits.PEIE) // Ngắt ngoại vi
    {}
    else // Ngắt ngoài ngoại vi
    {}
}
```

```
3   void __interrupt() InterruptFunction(void)
4   {
5       if(INTCONbits.PEIE) // Peripheral interrupt
6       {
7           if(PIE0bits.INTE) // External interrupt enable bit
8           {
9               if(PIR0bits.INTF) // External interrupt flag bit
10              {
11                  PIR0bits.INTF=0; // Clear interrupt flag bit
12                  // User's callback function
13              }
14          }
15          else {}// Other interrupts
16      }
17      else // Other interrupts
18      {
19          if(PIE0bits.TMR0IE) // TMR0 interrupt enable bit
20          {
21              if(PIR0bits.TMR0IF) // TMR0 interrupt flag bit
22              {
23                  PIR0bits.TMR0IF=0; // Clear TMR0 interrupt flag bit
24                  // User's callback function
25              }
26          }
27          else {} // Other interrupts
28      }
29  }
```

INTERRUPT
4. CHƯƠNG TRÌNH NGẮT

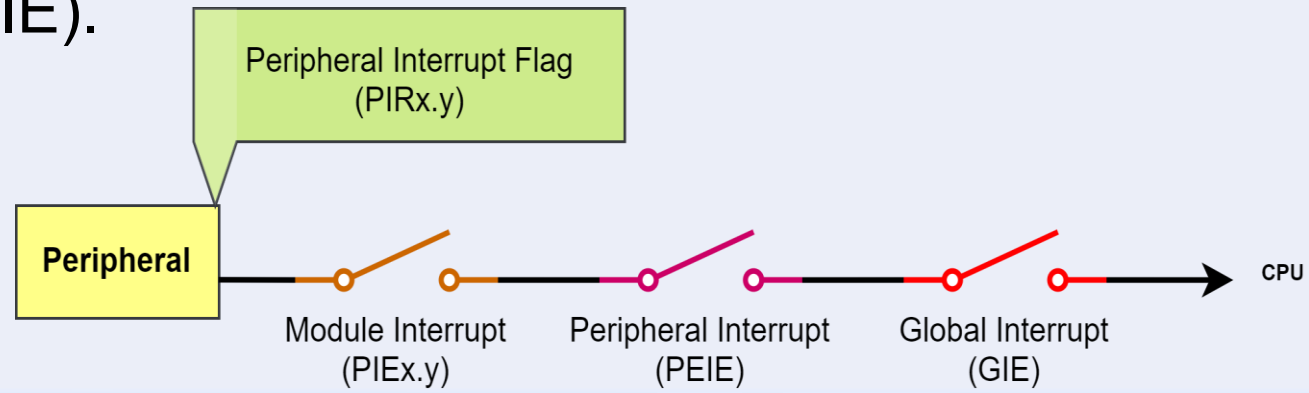
INTERRUPT

TIMER

❖ Cấu hình ngắt:

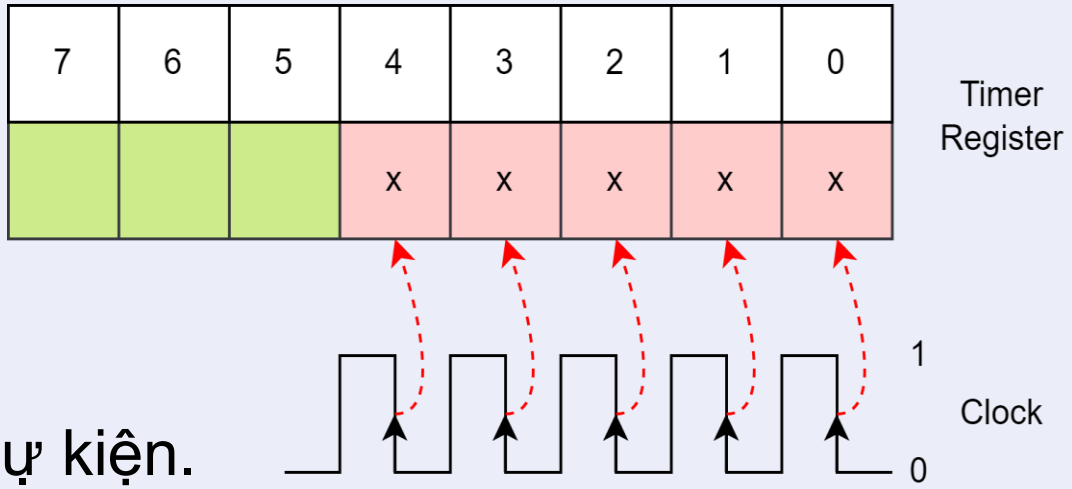
- Cấu hình module ngoại vi.
- Xoá cờ ngắt module ngoại vi.
- Cho phép ngắt module ngoại vi.
- Cho phép ngắt ngoại vi (PEIE).
- Cho phép ngắt toàn cục (GIE).

```
31  void main(void)
32  {
33      // Peripheral setup
34      PIR0bits.INTF=0; // Clear interrupt flag bit
35      PIR0bits.TMR0IF=0; // Clear TMR0 interrupt flag bit
36      PIE0bits.INTE=1; // Enable external interrupt
37      PIE0bits.TMR0IE=1; // Enable TMR0 interrupt
38      INTCONbits.PEIE=1; // Enable peripheral interrupts
39      INTCONbits.GIE=1; // Enable global interrupt
40
41      while(1);
42  }
```



TIMER
1. TỔNG QUAN

- Timer/counter thực tế là một bộ đếm với giá trị đếm tăng 1 đơn vị tương ứng với mỗi xung clock cấp vào.
- Timer/counter bản chất là 1 module, chúng chỉ khác nhau về nguồn clock.
  - Timer: Xung clock xác định được tần số, sử dụng để đếm thời gian.
  - Counter: Xung clock không xác định tần số, sử dụng để đếm sự kiện.

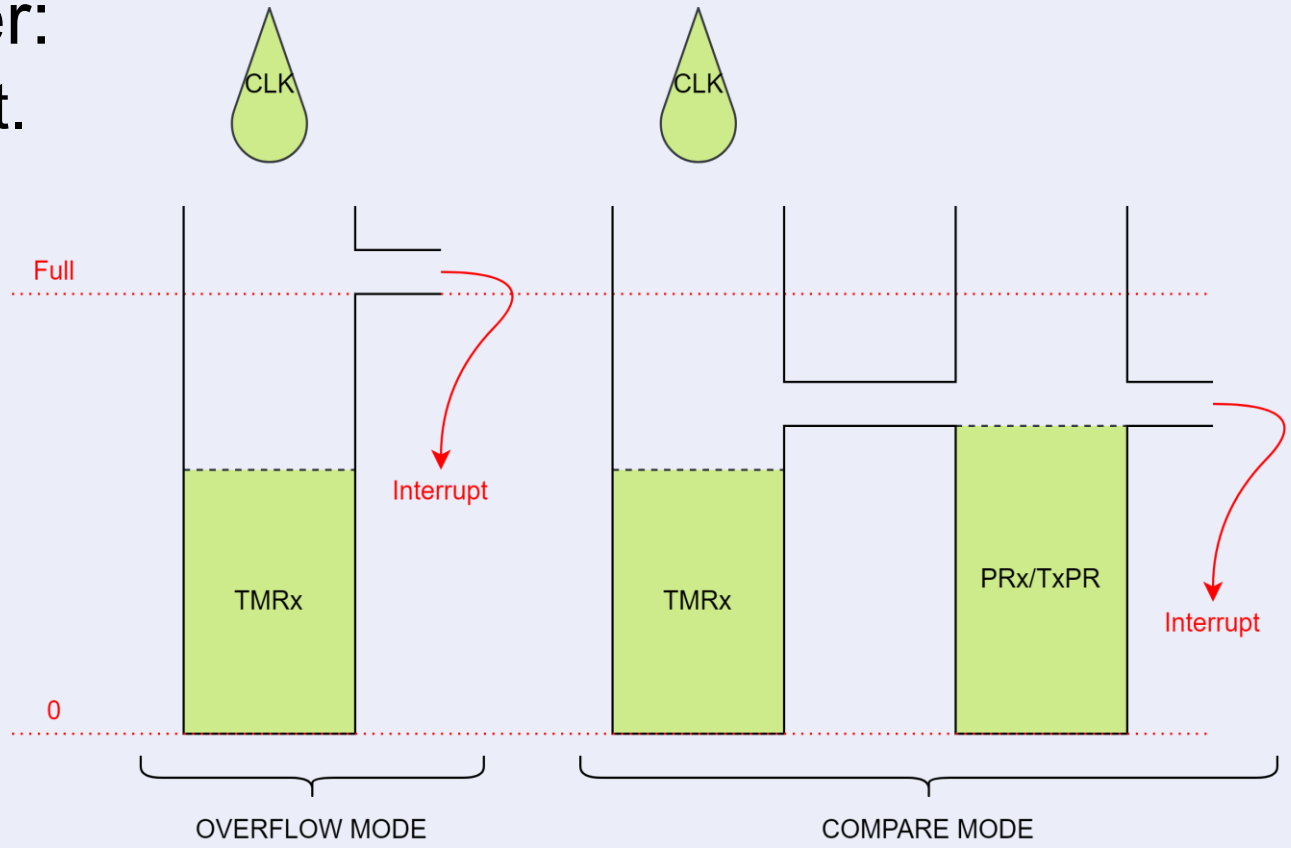


TIMER
1. TỔNG QUAN

INTERRUPT

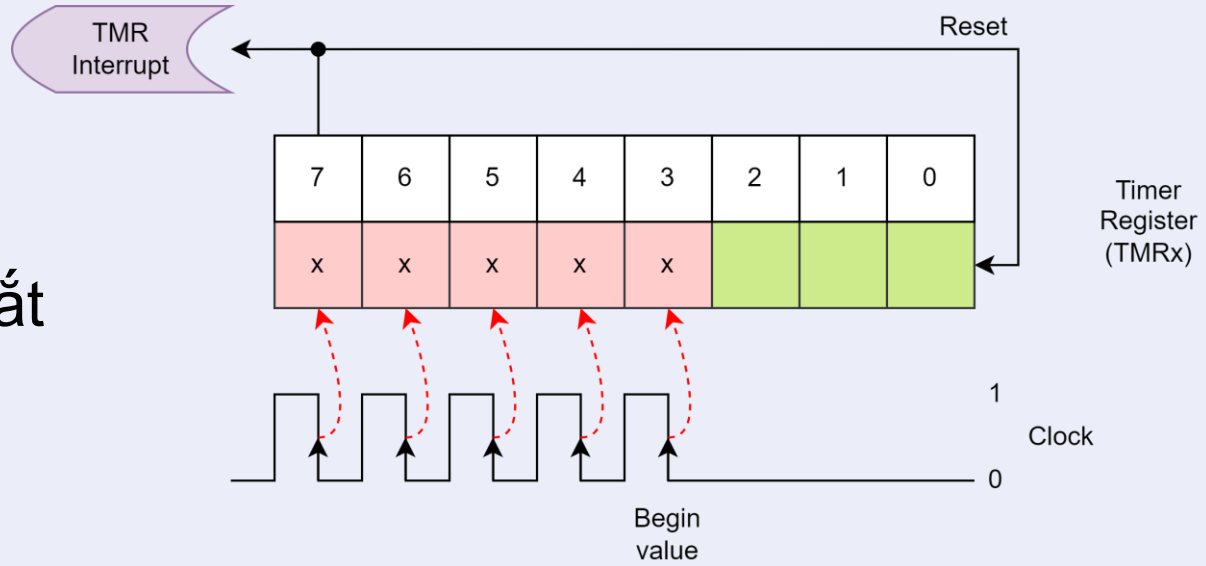
TIMER

- PIC16F18877 có 7 timer:
  - Timer 0: 8-bit hoặc 16-bit.
  - Timer 1/3/5: 16-bit.
  - Timer 2/4/6: 8-bit.
- Hai chế độ hoạt động:
  - Tràn (Overflow).
  - So sánh (Compare).



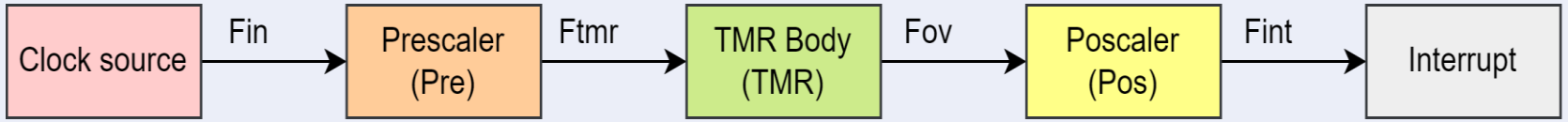
TIMER
2. CHẾ ĐỘ TRÀN

- Áp dụng cho timer 16-bit (TMR0/1/3/5).
- Hoạt động:
  - Ứng với mỗi xung cạnh lên, bộ đếm sẽ tăng 1 giá trị, khi đạt giá trị tối đa sẽ sinh ngắt và reset timer về 0.
  - Giá trị bắt đầu là tuỳ chọn.



TIMER
2. CHẾ ĐỘ TRÀN

❖ Tính toán:



Tần số xung cấp vào timer: $Ftmr = \frac{Fin}{Pre}$

Khoảng định thời: từ $\frac{1000}{Ftmr}$ đến $\frac{1000}{Ftmr} \times 65536$ (ms)

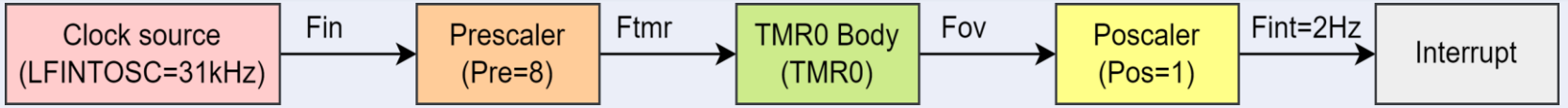
Tần suất tràn: $Fov = \frac{Ftmr}{65536-TMR} = \frac{Fin}{Pre \times (65536-TMR)}$

Tần suất ngắt (TMR=0, Pos>1): $Fint = \frac{Fov}{Pos} = \frac{Fin}{Pre \times Pos \times 65536}$

Tần suất ngắt (Pos=1): $Fint = Fov = \frac{Fin}{Pre \times (65536-TMR)} \Rightarrow TMR = 65536 - \frac{Fin}{Fint \times Pre}$

TIMER
2. CHẾ ĐỘ TRÀN

❖ Ví dụ:



Tần số xung cấp vào timer: $Ftmr = \frac{Fin}{Pre} = \frac{31000}{8} = 3875Hz$

Khoảng định thời: từ $\frac{1000}{3875} = 0.258ms$ đến $\frac{1000}{3875} \times 65536 = 16912.516ms$

Giả sử ta muốn ngắt sau mỗi 500ms (Pos=1 ⇒ Fov=Fint=2Hz):

Giá trị TMR theo Fint: $TMR = 65536 - \frac{31000}{2\times8} = 63599.$

MCU 8-bit nên thanh ghi 16-bit sẽ tách thành 2 thanh ghi 8bit:

8-bit thấp: $TMRL = TMR\%256 = 111$

8-bit cao: $TMRH = \frac{TMR}{256} = 248$

TIMER
2. CHẾ ĐỘ TRÀN

❖ Cấu hình TMR trên MPLAB:

- ➢ Bật module timer.
- ➢ Cấu hình Fin, Pre, Pos.
- ➢ Cấu hình giá trị timer.
- ➢ Khởi động timer.
- ➢ Cấu hình interrupt.

```
73      // Timer 0 configure
74      PMD1bits.TMR0MD=0; // TMR0 module is enabled
75      T0CON0=0b00010000; // T0EN disabled; T016BIT 16-bit; T0OUTPS 1:1;
76      T0CON1=0b10010011; // T0CS LFINTOSC; T0ASYNC not sync; T0CKPS 1:8;
77      // TMR0 initialize value for 500ms overflow;
78      TMR0L=111;
79      TMR0H=248;
80      PIR0bits.TMR0IF=0; // Clear Interrupt flag
81      PIE0bits.TMR0IE=1; // Enabling TMR0 interrupt
82      T0CON0bits.T0EN=1; // enable TMR0
83      INTCONbits.GIE=1; // enable global interrupt
```

TIMER
2. CHẾ ĐỘ TRÀN

INTERRUPT

TIMER

❖ Ngắt TMR trên MPLAB:

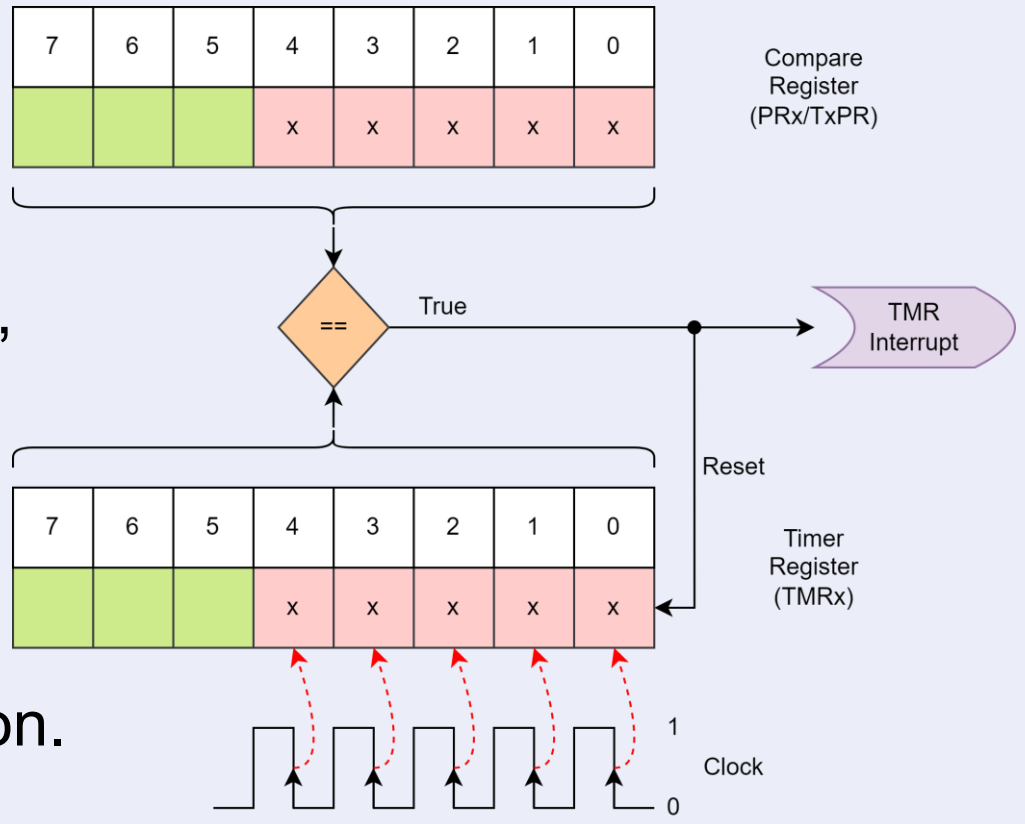
- ➢ Kiểm tra bit điều khiển ngắt.
- ➢ Kiểm tra cờ ngắt.
- ➢ Xoá cờ ngắt.
- ➢ Đặt lại giá trị bắt đầu của TMR.
- ➢ Thực hiện chương trình người dùng.

```
49  void __interrupt() INTERRUPT_InterruptManager(void)
50  {
51      if((PIE0bits.TMR0IE==1)&&(PIR0bits.TMR0IF==1))
52      {
53          PIR0bits.TMR0IF=0; // clear the TMR0 interrupt flag
54          // Reinitialize TMR0 value for 500ms overflow;
55          TMR0L=111;
56          TMR0H=248;
57          // Do something
58          LATAbits.LATA1^=1; // toggle A1
59      }
60  }
```

→ Lưu ý: Tránh sử dụng biến hoặc function trùng với chương trình chính để tránh sự xung đột.

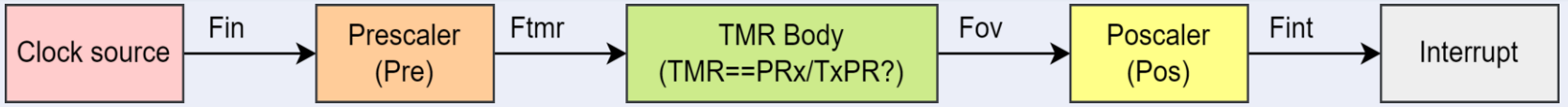
TIMER
3. CHẾ ĐỘ SO SÁNH

- Áp dụng cho timer 8-bit (TMR0/2/4/6).
- Hoạt động:
  - Ứng với mỗi xung (cạnh tuỳ chọn), bộ đếm sẽ tăng 1 giá trị, khi đạt giá trị của thanh ghi so sánh sẽ sinh ngắt và reset timer về 0.
  - Giá trị TMR bắt đầu là 0.
  - Giá trị thanh ghi so sánh là tuỳ chọn.



TIMER
3. CHẾ ĐỘ SO SÁNH

❖ Tính toán:



Tần số xung cấp vào timer: $Ftmr = \frac{Fin}{Pre}$

Khoảng định thời: từ $\frac{1000}{Ftmr}$ đến $\frac{1000}{Ftmr} \times 256$ (ms)

Tần suất tràn: $Fov = \frac{Ftmr}{TMR+1} = \frac{Fin}{Pre \times (TxPR+1)} \Rightarrow TxPR = \frac{Fin}{Fov \times Pre} - 1$

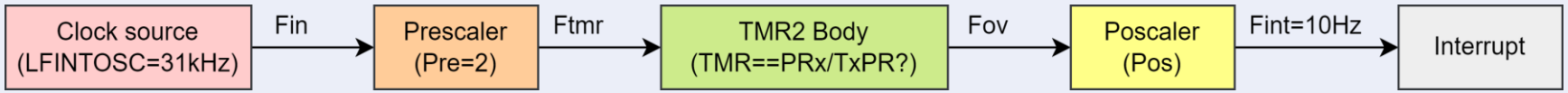
Tần suất ngắt: $Fint = \frac{Fov}{Pos} = \frac{Fin}{Pre \times Pos \times (TxPR+1)} \Rightarrow TxPR = \frac{Fin}{Fint \times Pre \times Pos} - 1$

TIMER
3. CHẾ ĐỘ SO SÁNH

INTERRUPT

TIMER

❖ Ví dụ:



Tần số xung cấp vào timer: $Ftmr = \frac{Fin}{Pre} = \frac{31000}{2} = 15500Hz$

Khoảng định thời: từ $\frac{1000}{15500} \approx 0.065ms$ đến $\frac{1000}{15500} \times 256 \approx 16.516ms$

Giả sử ta muốn ngắt sau mỗi 100ms (>16.516ms), ta có $Fint = 10Hz$, chọn $\mathrm{Pos} = 10$

$\Rightarrow Fov = Fint \times Pos = 10 \times 10 = 100\mathrm{Hz}$

Giá trị $TxPR$ theo Fov: $TxPR = \frac{31000}{100 \times 2} - 1 = 154.$

TIMER
3. CHẾ ĐỘ SO SÁNH

INTERRUPT

TIMER

❖ Cấu hình TMR trên MPLAB:

- ➢ Bật module timer.
- ➢ Cấu hình Fin, Pre, Pos.
- ➢ Cấu hình cạnh xung clock.
- ➢ Cấu hình chế độ hoạt động.
- ➢ Cấu hình giá trị timer.
- ➢ Khởi động timer.
- ➢ Cấu hình interrupt.

```
69      // Timer 2 configure
70      PMD1bits.TMR2MD=0; // TMR2 module is enabled
71      T2CON=0b00011001; // Disable TMR2; CKPS 1:2; OUTPS 1:10;
72      T2CLKCON=0b00000100; // CS LFINTOSC;
73      T2HLT=0b01000000; // CKPOL falling edge; Free running period
74      // TMR2 initialize value for 100ms overflow;
75      TMR2=0;
76      T2PR=154;
77      PIR4bits.TMR2IF=0; // Clear Interrupt flag
78      PIE4bits.TMR2IE=1; // Enabling TMR0 interrupt
79      T2CONbits.ON=1; // enable TMR2
80      INTCONbits.PEIE=1; // enable peripheral interrupt
81      INTCONbits.GIE=1; // enable global interrupt
```

TIMER
3. CHẾ ĐỘ SO SÁNH

INTERRUPT

TIMER

❖ Ngắt TMR trên MPLAB:

```
49    void __interrupt() INTERRUPT_InterruptManager(void)
50    {
51        if((INTCONbits.PEIE==1)&&(PIE4bits.TMR2IE==1)&&(PIR4bits.TMR2IF==1))
52        {
53            PIR4bits.TMR2IF=0; // clear the TMR2 interrupt flag
54            // Do something
55            LATAbits.LATA1^=1; // toggle A1
56        }
57    }
```

➢ Kiểm tra bit điều khiển ngắt.

➢ Kiểm tra cờ ngắt.

➢ Xoá cờ ngắt.

➢ Đặt lại giá trị bắt đầu của TMR.

➢ Thực hiện chương trình người dùng.

→ Lưu ý: Tránh sử dụng biến hoặc function trùng với chương trình chính để tránh sự xung đột.