**Tạo xung PWM với PIC16F877A**

**I, PWM là gì?**

PWM (Pulse Width Modulation) là tín hiệu số được sủ dụng dụng phổ biến trong các mạch điện tử, mạch điều khiển. Tín hiệu này được setting ở mức cao là 5V, mức thấp là 0V trong 1 khoảng thời gian và tốc độ được xác định trước.

Thời gian tín hiệu duy trì ở mức cao được gọi là “on time”, thời gian ở mức thấp được gọi là “off time”. Có 2 thông số quan trọng của PWM sẽ được trình bày ở dưới đây.

**1, Duty Cycle của PWM**

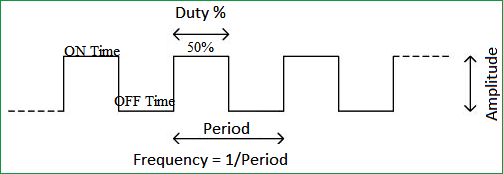
Tỷ lệ phần trăm thời gian của “on time” trong 1 chu kỳ được gọi là “Duty Cycle”.

**Duty Cycle =**

**2, Tần số của PWM**

Tần số của tín hiệu PWM cho ta biết mức độ nhanh chóng hoàn thành 1 period.

Một Period hoàn thành On và Off tín hiệu PWM như trong hình dưới đây:



**II, PWM với PIC16F877A**

PWM có thể được tạo trong vi điều khiển PIC bằng cách sử dụng module CCP (Compare Capture PWM).

Độ phân giải của tín hiệu PWM trong PIC16F877A là 10bits. Điều này có nghĩa là với duty cycle 100% sẽ có giá trị là 210 = 1024, duty cycle 0% sẽ có giá trị bằng 0.

Có 2 module CCP trong PIC16F877A MCU (CCP1 và CCP2), đồng nghĩa với việc chúng ta có thể tạo ra 2 tín hiệu PWM khác nhau trên 2 chân khác nhau (chân PC1 và PC2). Trong bài viết này chúng ta sẽ sử dụng CCP1.

Các thanh ghi quan trọng để tạo PWM với PIC MCU:

1. CCP1CON (CCP1 control Register)
2. T2CON (Timer 2 Control Register)
3. PR2 (Timer 2 modules Period Register)
4. CCPR1L (CCP Register 1 Low)

**III, Lập trình PIC16F877A để tạo tín hiệu PWM**

Để lập trình tạo được tín hiệu PWM, ta cần thực hiện các bước sau:

* Set PWM Period bằng cách viết giá trị lên thanh ghi PR2
* Set PWM Duty Cycle bằng cách viết giá trị lên thanh ghi CCPR1L và 2 bit CCP1CON<5:4>
* Cấu hình chân CCP1 ở chế độ Output
* Set giá trị Timer 2 Prescale và enable Timer 2
* Cấu hình cho Module CCP1 ở chế độ PWM

Có 2 hàm quan trọng khi khởi tạo tín hiệu PWM, đó là hàm khởi tạo PWM và set duty cycle cho PWM.

**1, Hàm khởi tạo PWM\_Initialize(void)**

float PR2\_Value**(**float PWM\_Freg**)**

**{**

float value **=** **(**XTAL\_Freq**/(**TMR2\_Prescale**\***4**\***PWM\_Freg**))-**1**;**

**return** value**;**

**}**

void PWM\_Initialize**(**void**)**

**{**

PR2 **=** PR2\_Value**(**5000**);** //(8000000/(16\*4\*5000))-1;

CCP1CON**.**CCP1M3 **=** 1**;** // PWM mode

CCP1CON**.**CCP1M2 **=** 1**;** // PWM mode

T2CON**.**TMR2ON **=** 1**;** // TIMER2 ON

T2CON**.**T2CKPS1 **=** 1**;**

T2CON**.**T2CKPS0 **=** 0**;** // Prescaler is 16

**}**

Trong bài này chúng ta sẽ tạo 1 xung PWM tần số 5 KHz, thạch anh sử dụng là 8MHz (XTAL\_Freq), setting giá trị Prescale cho Timer 2 bằng 16.

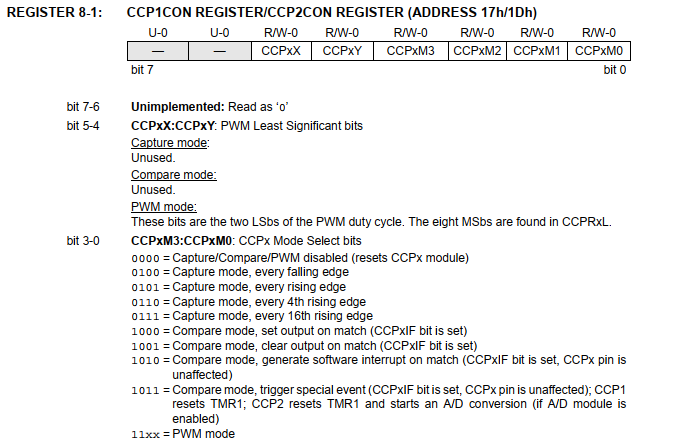
Hàm **PR2\_Value(float PWM\_Freq)** sẽ thực hiện tính toán để trả về giá trị cho biến ‘value’ để gán vào thanh ghi PR2. Ta sẽ cùng tìm hiểu xem tại sao lại có công thức **value =** **(XTAL\_Freq/(TMR2\_Prescale\*4\*PWM\_Freg)) – 1**

Công thức ban đầu được cung cấp trong Datasheet như sau:

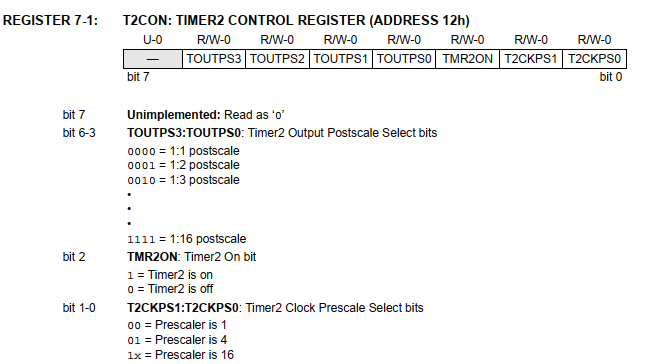
**PWM Period = [(PR2) + 1] \* 4 \* TOSC \* (TMR2 Prescale Value)**

Chính vì thế, sau khi biến đổi, ta sẽ được công thức tính toán giá trị cho thanh ghi PR2.

Trong thanh ghi **CCP1CON**, chúng ta cần cấu hình cho 2 bit CCP1M3 và CCP1M2 ở mức 1, để chọn chế độ PWM cho CCP1 Module



Trong thanh ghi **T2CON**, chúng ta cần enable cho Time 2 bằng cách set bit TMR2ON ở mức 1. Hai bit T2KPS1 và T2KPS0 sẽ giúp chúng ta chọn giá trị cho Prescale của Timer. Ở đây chúng ta set T2KPS1 = 1, T2KPS0 = 0, để giá trị Prescale bằng 16.



**2, Hàm set duty cycle PWM\_Duty(unsigned int duty)**

void PWM\_Duty**(**unsigned int duty**) //duty = 0~100**

**{**

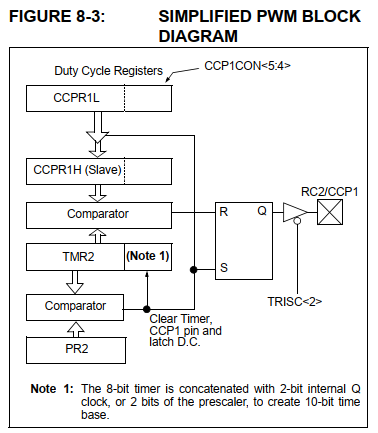
CCP1CON**.**CCP1X **=** duty **&** 2**;** //Store the 1st bit

CCP1CON**.**CCP1Y **=** duty **&** 1**;** //Store the 0th bit

CCPR1L **=** duty**>>**2**;**// Store the remining 8 bit

**}**

Trước hết chúng ta sẽ quan sát Block Diagram của PWM trong PIC16F877A



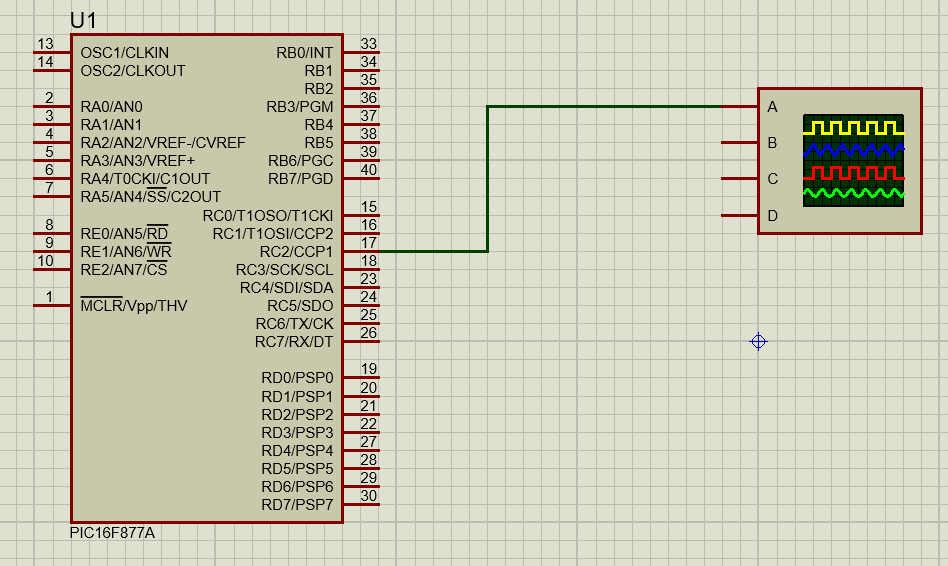
Chúng ta có thể dễ dàng nhận ra thanh ghi **CCPR1L** (thanh ghi 8bits) và 2bits **CCP1CON<5:4>** sẽ chứa giá trị của Duty Cycle. Chính vì thế, trong hàm PWM\_Duty, cần phải truyền giá trị của bit thứ 0 và thứ 1 của ‘duty’ vào cho 2bits **CCP1CON<5:4>**, truyền 8 giá trị còn lại vào thanh ghi **CCP1RL**.

Giá trị của biến ‘duty’ từ 0 đến 100.

**Lưu ý:** trong hàm main, chúng ta phải cấu hình cho chân CCP1 ở chế độ output.

**III, Kết quả mô phỏng**

Sơ đồ nguyên lý:



Kết quả đo được trên mô phỏng (thực hiên với tần số 5KHz và Duty = 85%)

