Microcontrollers

PWM

Dr. Jafar Jallad

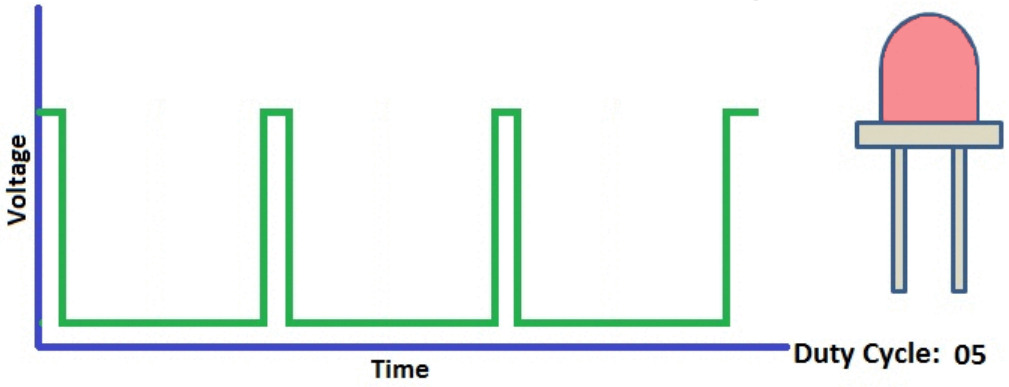
Palestine Technical University – Kadoorie

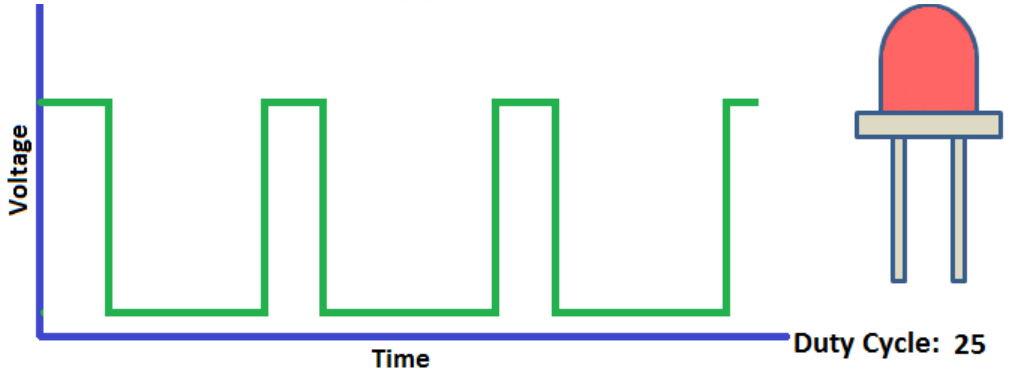
Second semester

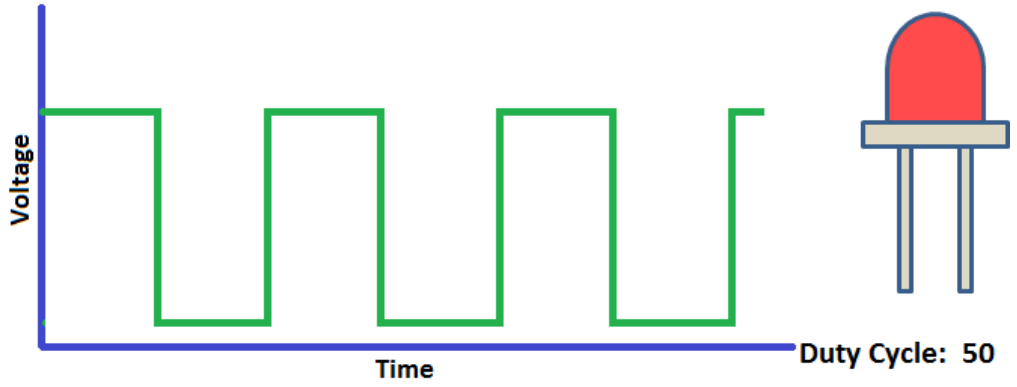
2019-2020

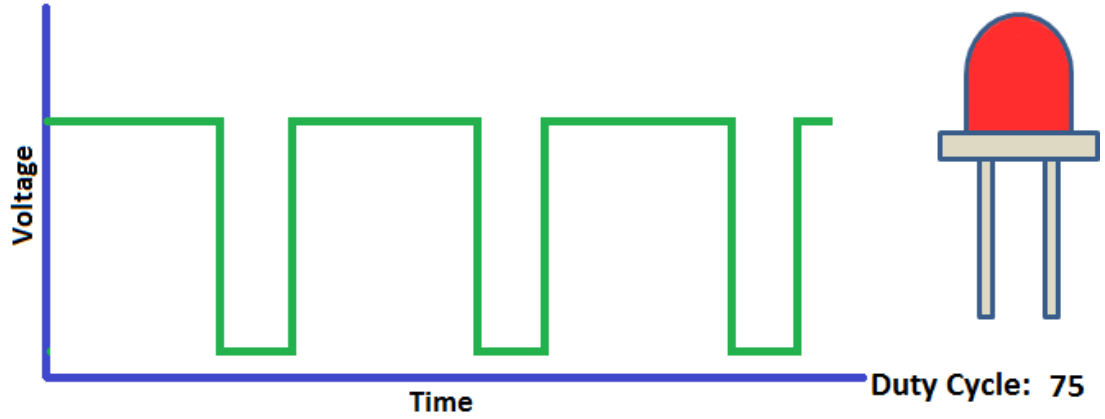
PWM module of PIC16F877A

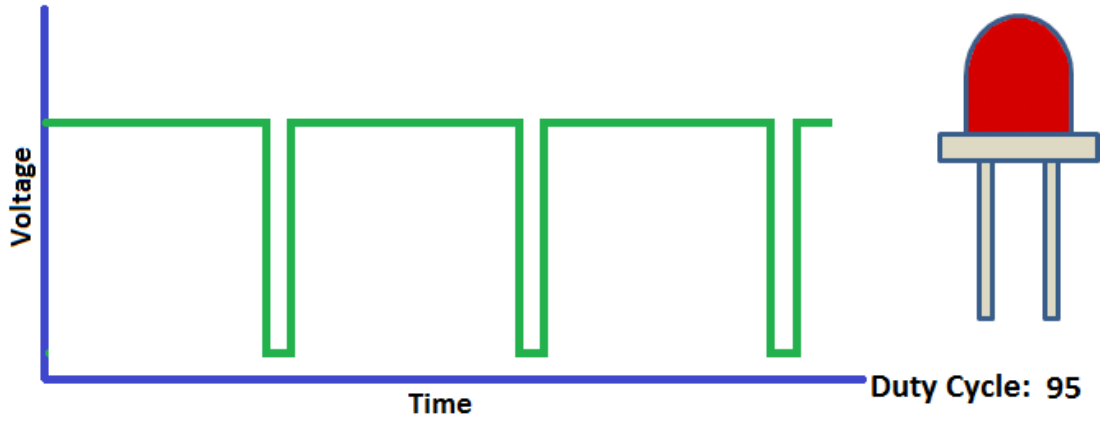
First we will see how to configure the PWM registers to generate signals of required PWM





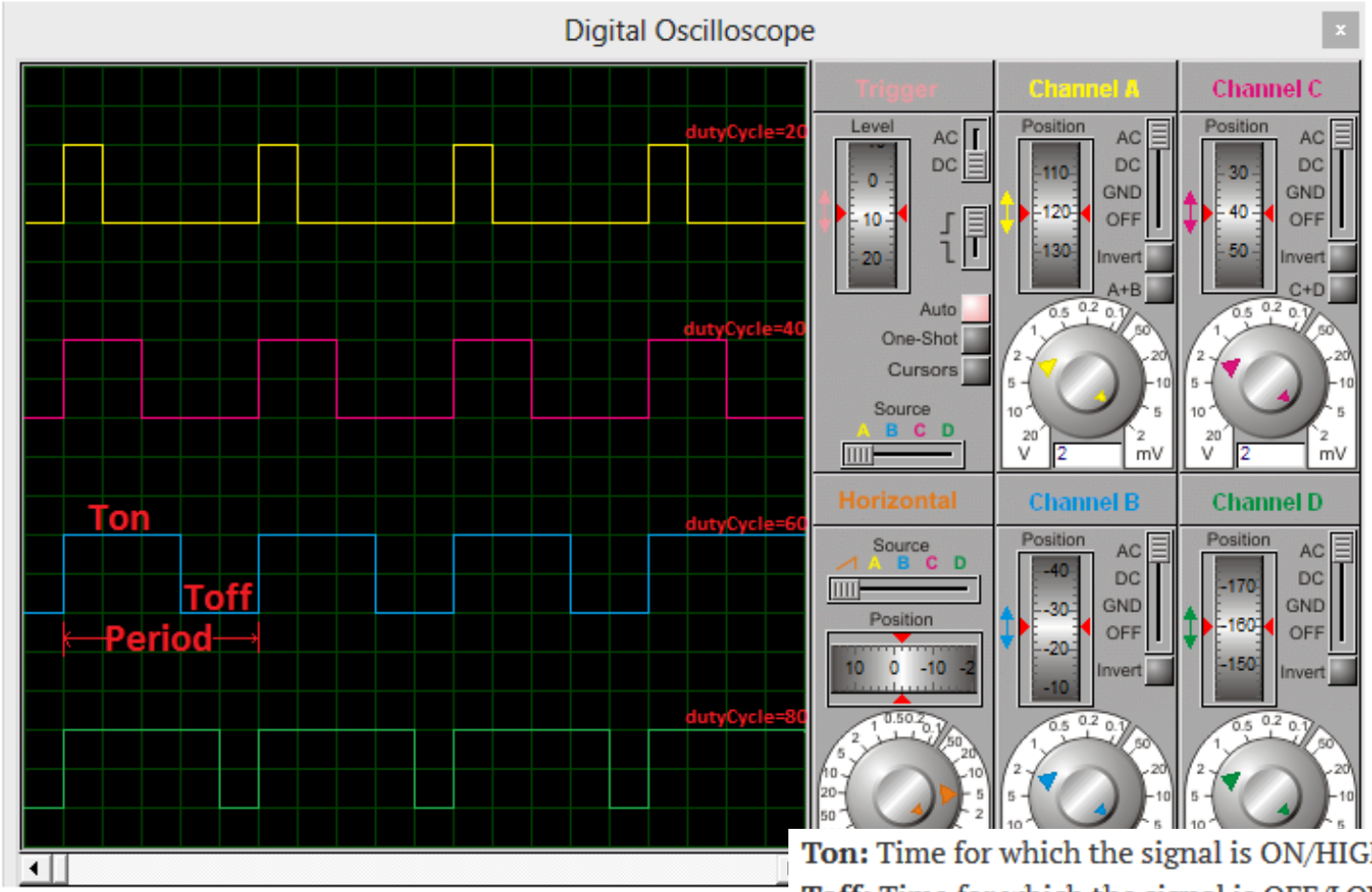






PWM Working

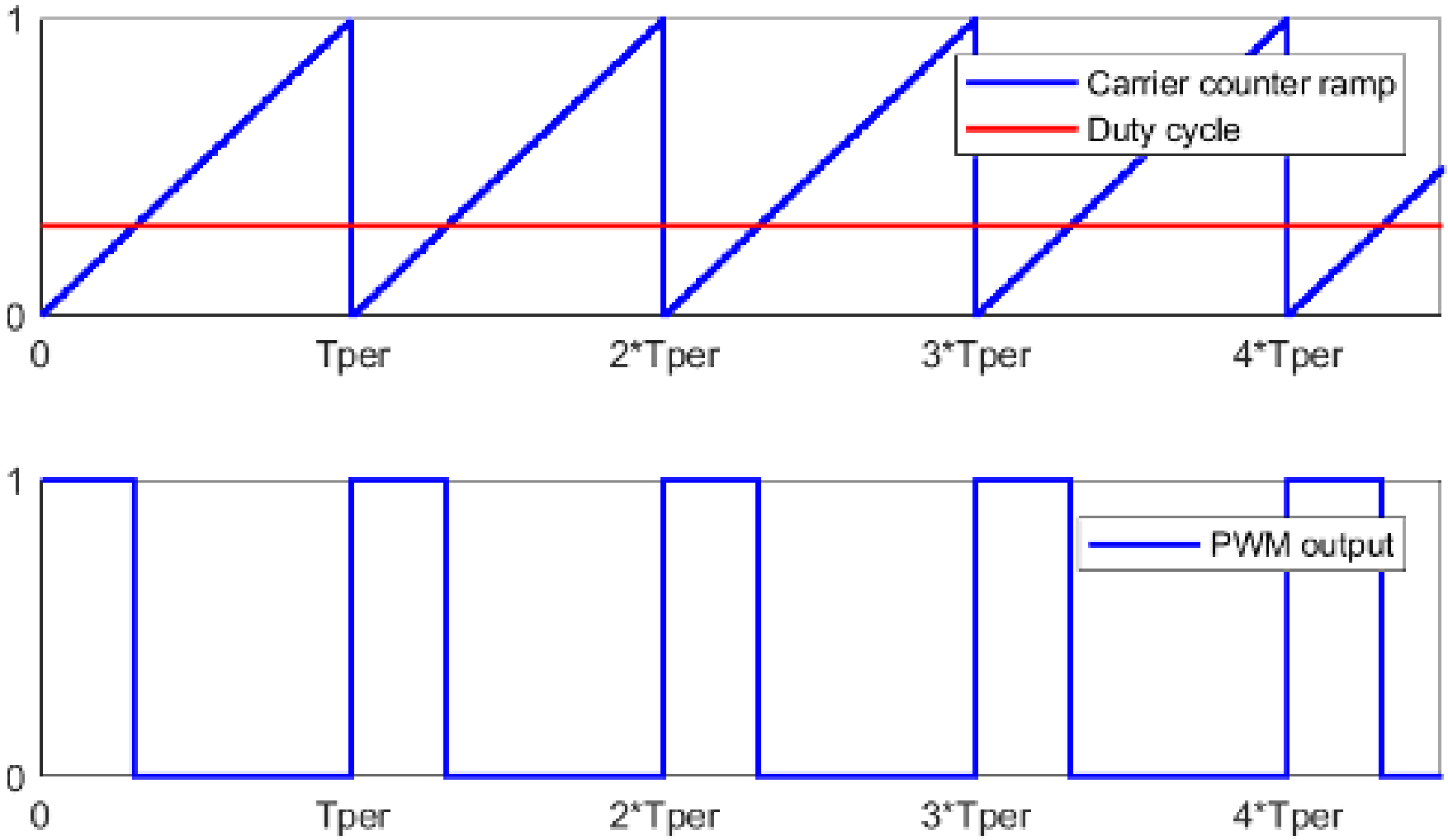
Before going to the PIC PWM module lets understand few terminologies wrt below image:

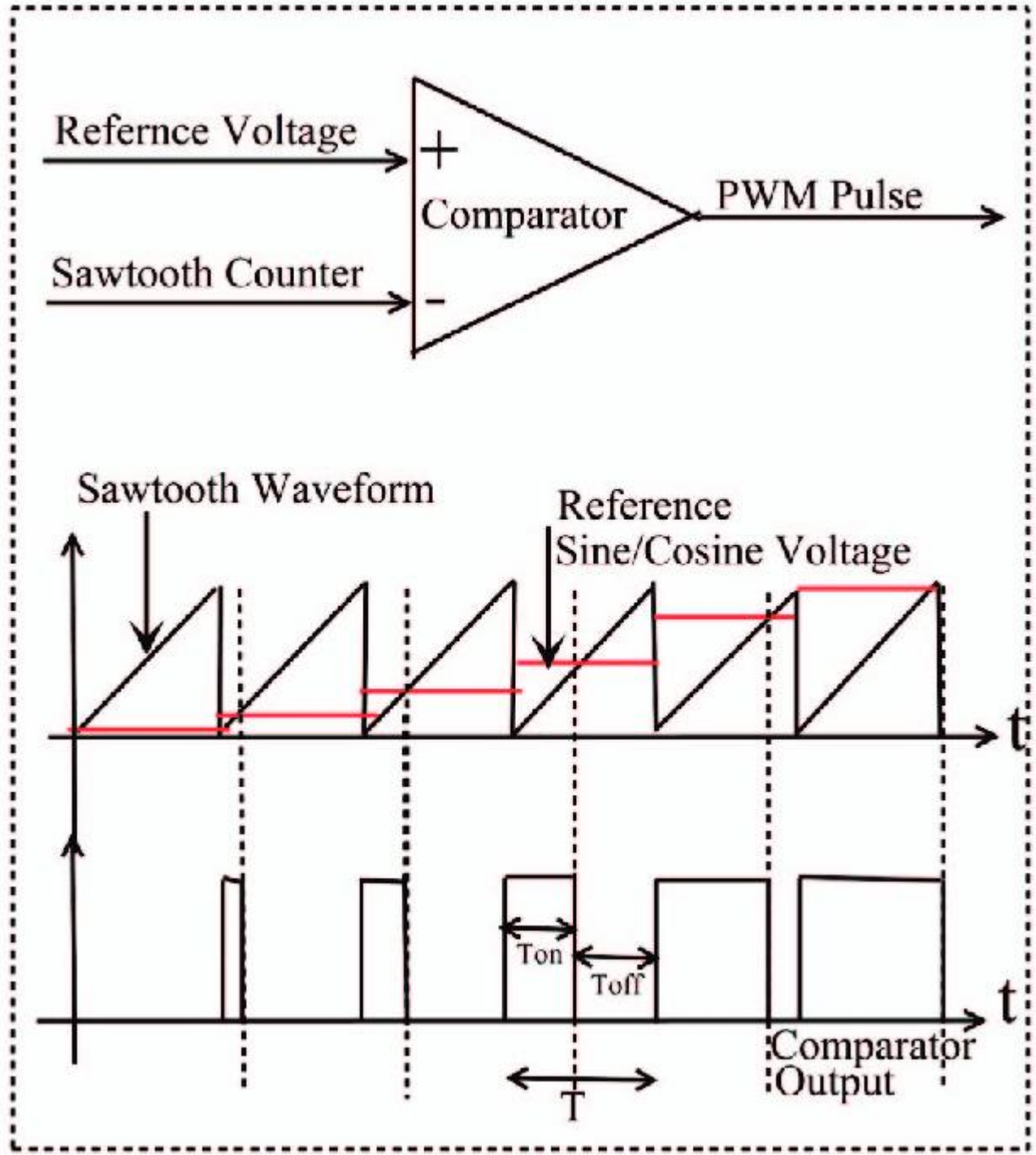


Ton: Time for which the signal is ON/HIGH.
Toff: Time for which the signal is OFF/LOW.
Period: The cycle time of the signal which is nothing but the sum of Ton and Toff period.
Period= Ton+Toff
Freq of Pwm signal = 1/Period.

Duty Cycle: The percentage of time for which the PWM signal will be ON/HIGH. $$DutyCycle = (Ton/(Ton+Toff))*100 = (Ton/Period) * 100 $$

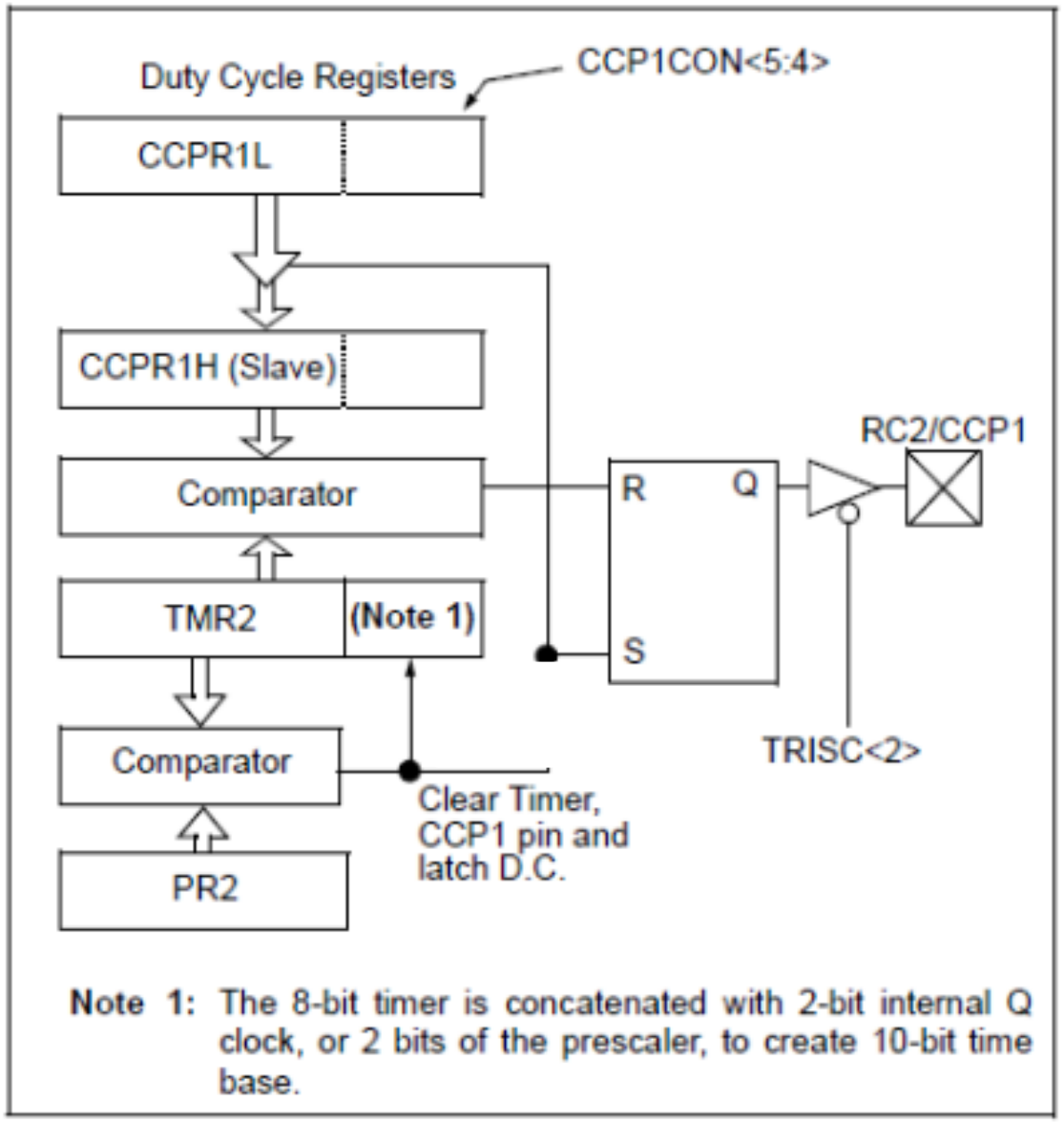
PWM Pulse Width Modulation; Basics and Some Advanced Concepts




Refernce Voltage
+
Comparator
PWM Pulse
Sawtooth Counter
-
Sawtooth Waveform
Reference
Sine/Cosine Voltage
t
Ton
Toff
t
Comparator
Output
T

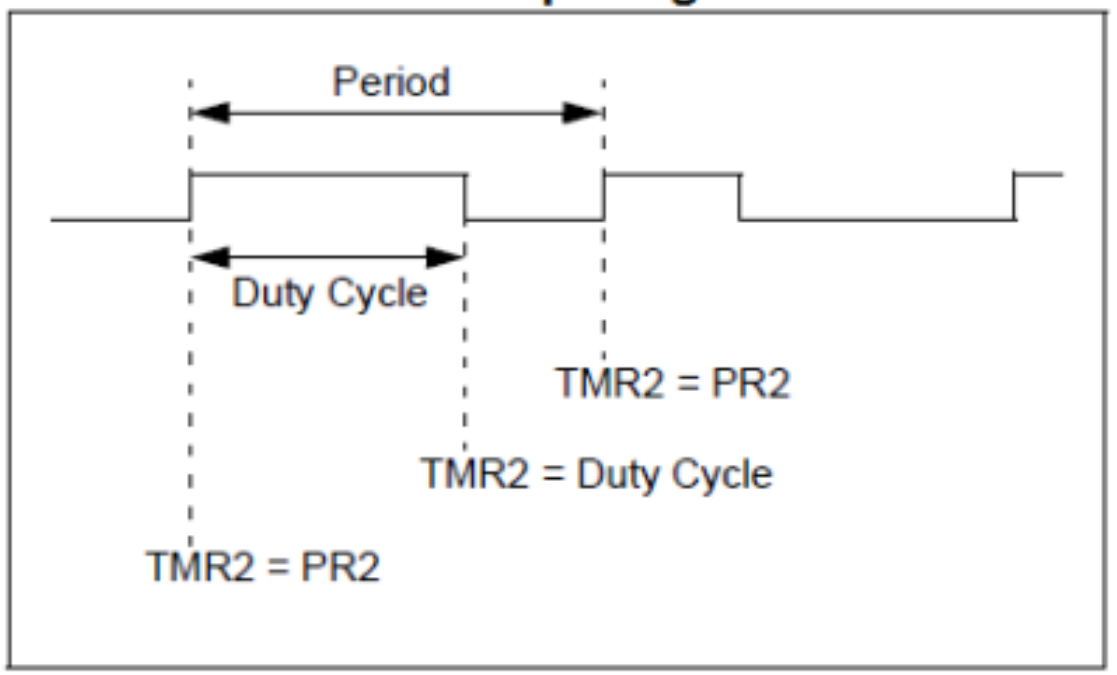
Below diagram shows the PIC16f877a PWM block diagram and output signal generation.

PWM Block Diagram



Note 1: The 8-bit timer is concatenated with 2-bit internal Q clock, or 2 bits of the prescaler, to create 10-bit time base.

PWM Output Signal



PIC16F877A PWM Module

PIC16F877A microcontroller has two independent CCP(Capture/Compare/PWM) modules, named as CCP1 and CCP2. Each CCP module has two 8-bit resistors(CCPxH,CCPxL) that can be use as:

- 16 bit Capture Register
- 16 bit Compare Register
- 10-bit PWM Register.

In this tutorial we will be discussing only the PWM part of CCP. PIC has 2PWM module with a resolution of 10-bits. The 8-Msb bits are stored in CCPRxL and remeining 2-bits in CCPxCON register. Below tables shows the PWM module of PIC.

PWM Channel	Port Pin	Control Register	Duty Cycle Register	Period Register
PWM1	PC.2	CCP1CON	CCPR1L	PR2
PWM2	PC.1	CCP2CON	CCPR2L	PR2

PIC uses TIMER2 for generating the PWM signals. Lets relate the above concepts with the PIC registers.

Period:PR2 register is used to specify the PWM Period. The PWM period can be calculated using the following formula.

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

DutyCycle:CCPR1L and CCP1CON<5:4> are used to specify the DutyCycle.The CCPR1L contains the eight MSbs and the CCP1CON<5:4> contains the two LSbs.

PWM Signal Generation: Once the PWM is configured and Timer2 is enabled, TMR2 starts incrementing depending on the prescalar. Once the **TMR2** value is equal to dutyCycle**(CCPR1L+CCP1CON<5:4>)** the PWM pin will be pulled LOW. The timer still continues to increment till it matches with the period **PR2**. After which the PWM pin will be pulled HIGH and TMR2 is reset for next cycle.

Example: We use 20 MHz clock and the O/P frequency required is 5KHz

Here PWM period = 1/ Frequency (that will be 1/5000 = 0.0002)

0 .0002 = (PR2 + 1) x (1 / 20000000) x 16

PR2 + 1 = (0.0002 x 20000000) / 16 =250

PR2 = 249

PR2 = 0xF9

PIC16F877A PWM Registers

The below table shows the registers associated with PIC16F877A PWM.

Register	Description
CCPxCON	This register is used to Configure the CCP module for Capture/Compare/PWM opertaion.
CCPRxL	This register holds the 8-Msb bits of PWM, lower 2-bits will be part of CCPxCON register.
TMR2	Free running counter which will be compared with CCPR1L and PR2 for generating the PWM output.

Register Configuration

The below table shows the registers associated with PIC16f877a PWM.

CCPxCON							
7	6	5	4	3	2	1	0
—	—	CCPxX	CCPxY	CCPxM3	CCPxM2	CCPxM1	CCPxM0

CCPxX:CCPxY: PWM Least Significant bits
These bits are the two LSbs of the PWM duty cycle. The eight MSbs are found in CCPRxL.

CCPxM3:CCPxM0: CCPx Mode Select bits
0000 - Capture/Compare/PWM disabled (resets CCPx module)
0100 - Capture mode, every falling edge
0101 - Capture mode, every rising edge
0110 - Capture mode, every 4th rising edge
0111 - Capture mode, every 16th rising edge
1000 - Compare mode, set output on match (CCPxIF bit is set)
1001 - Compare mode, clear output on match (CCPxIF bit is set)
1010 - Compare mode, generate software interrupt on match (CCPxIF bit is set, CCPx pin is unaffected)
1011 - Compare mode, trigger special event (CCPxIF bit is set, CCPx pin is unaffected);
11xx - PWM mode

T2CON							
7	6	5	4	3	2	1	0
—	TOUTPS3	TOUTPS2	TOUTPS1	TOUTPS0	TMR2ON	T2CKPS1	T2CKPS0

TOUTPS3:TOUTPS0: Timer2 Output Postscale Select bits
0000 = 1:1 postscale
0001 = 1:2 postscale
0010 = 1:3 postscale
•
•
•
1111 = 1:16 postscale

TMR2ON: Timer2 On bit
1-Timer2 is on
0-Timer2 is off

T2CKPS1:T2CKPS0: Timer2 Clock Prescale Select bits
00 = Prescaler is 1
01 = Prescaler is 4
1x = Prescaler is 16

Steps to Configure PWM

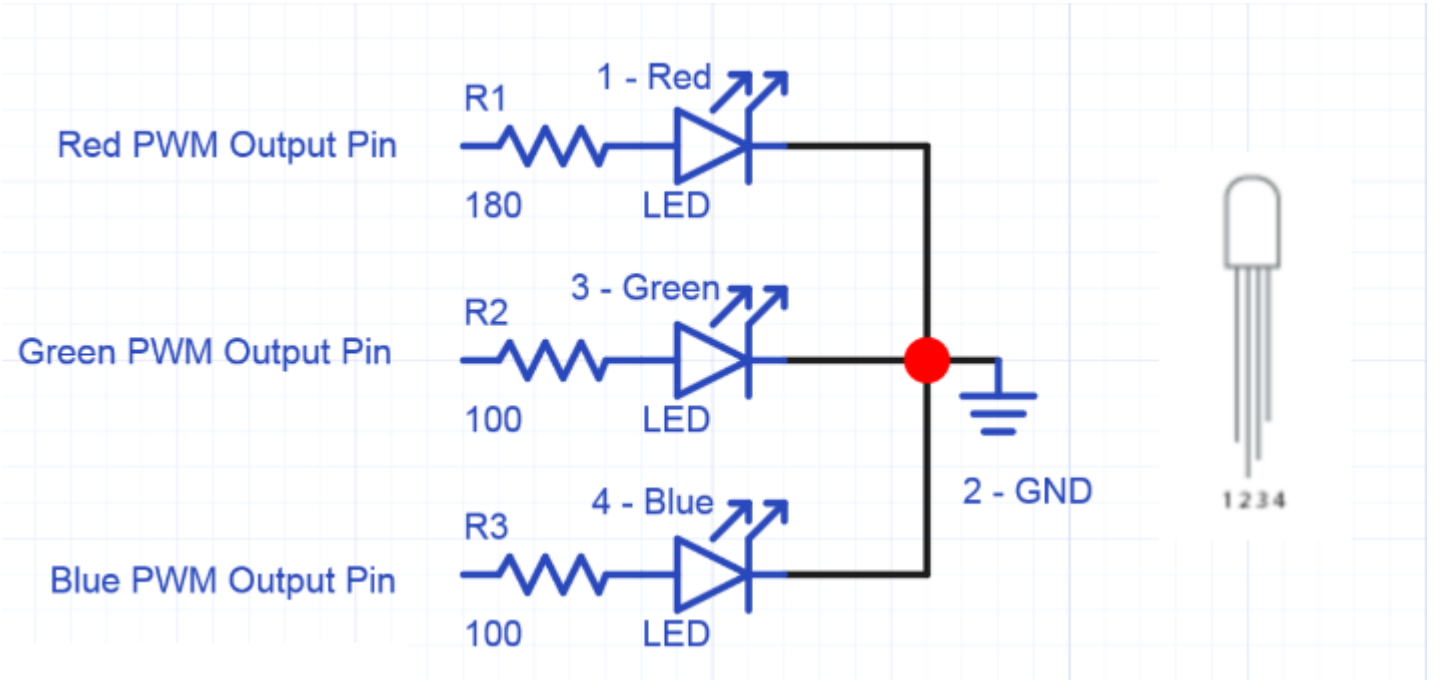
1. Configure the CCP1 module for PWM operation.
2. Set the PWM period by writing to the PR2 register.
3. Set the PWM duty cycle by writing to the CCPR1L register and CCP1CON<5:4> bits.
4. Make the CCP1 pin an output by clearing the TRISC<2> bit.
5. Set the TMR2 prescale value and enable Timer2 by writing to T2CON.

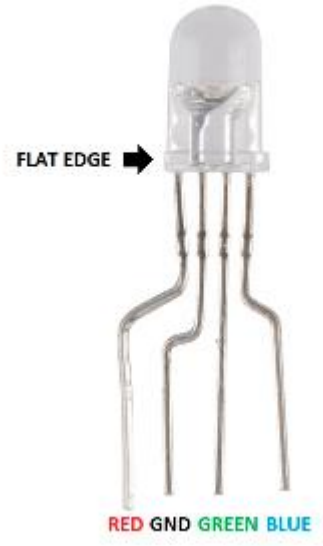
```
      Bank0
              movlw b'00001100'
              movwf CCP1CON
              movlw b'00001100'
              movwf CCP2CON
       Bank1
              MOVLW  .249
              MOVWF  PR2
       Bank0
              MOVLW .20
              MOVWF CCPR1L
              MOVLW .100
              MOVWF CCPR2L
      Bank1
              BCF  TRISC,2
              BCF  TRISC,1
     Bank0
              MOVLW  B'00000101'
              MOVWF T2CON
    START1

              GOTO START1
END
```

Fosc=20MHz

RGB (red green blue) LED

PWM used to control LED brightness





Period= 2ms, Freq=500Hz
D=0% -100%

https://rgbcolorcode.com/color/floral-white

```
PGM   code
Start
        Bank0
        movlw b'00001100'
         movwf CCP1CON
         movlw b'00001100'
         movwf CCP2CON
        Bank1
       MOVLW  .124
       MOVWF  PR2
       Bank0
      MOVLW .74
      MOVWF CCPR1L
      MOVLW .62
      MOVWF CCPR2L
     Bank1
     BCF  TRISC,2
     BCF  TRISC,1

     Bank0
     MOVLW  B'00000111'
     MOVWF T2CON
    START1

    GOTO START1

;=======================
    END
```

Fosc=4MHz


Thank
you!