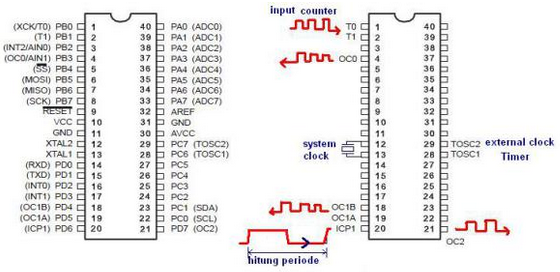
**MODUL IV**

**MIKROKONTROLER**

**(TIMER, COUNTER DAN PWM)**

1. **TUJUAN**
   * 1. Mengetahui dan memahami sistem pewaktu (timer) pada mikrokontroler.
     2. Mengetahui dan memahami sistem pencacah (counter) pada mikrokontroler.
     3. Dapat membuat sinyal PWM dengan memanfaatkan fitur timer pada mikrokontroler.
2. **DASAR TEORI**
3. **TIIMER**

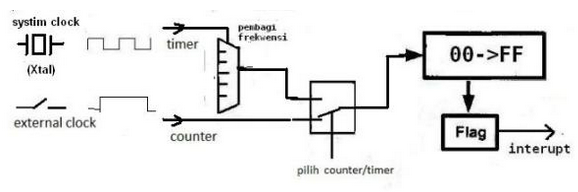
Timer dan Counter merupakan fitur yang di mikrokontroler AVR yang memiliki fungsi sebagai pewaktu dan pencacah. Fungsi pewaktu yang dimaksud disini adalah penentuan kapan program tersebut dijalankan, tidak hanya itu saja fungsi timer yang lainnya adalah untuk menghasilkan sinyal PWM. Prinsip kerja timer dengan cara membagi frekuensi (prescaler) pada clock yang terdapat pada mikrokontroler sehingga timer dapat berjalan sesuai dengan frekuensi yang di kehendaki.



Gambar 4.1 Pin Timer/Counter Pada ATmega 16

Pada gambar diatas dapat dilihat pin-pin yang juga memiliki fungsi sebagai input/output untuk fitur timer/counter antara lain adalah : PD4, PB0, PB1, PB3, PD6 dan PD7.

Timer merupakan fungsi waktu yang sumber clocknya berasal dari clock internal. Ketika sumber clock  dari   system clock  (kristal) maka ia  berfungsi sbg timer. Ketika sumber clock dari external maka ia berlaku sebagai counter. pemilihan sumber clock ada pada bit  CS pada register TCCR seperti pada skema dibawah

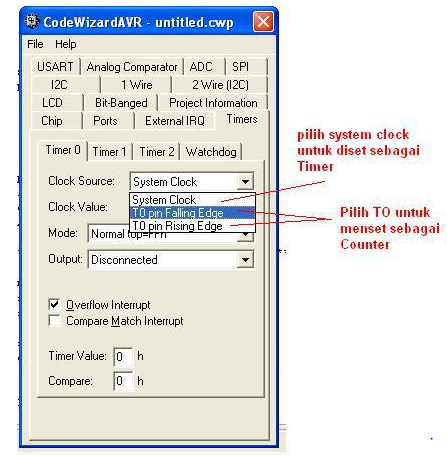


Gambar 4.2 Skema Pemilihan Timer/Counter

Pada ATmega16 terdapat 3 buah timer, yaitu Timer0 (8 bit), Timer1 (16 bit) dan Timer2 (8 bit). Komponen utama Timer/Counter adalah sebuah register  yg tugasnya hanya berhitung dari 0 sampai batas maximumnya, register ini pada AVR disebut  register TCNT, secara umum nilai timer akan bertambah setiap masukan clock, dan akan disimpan dalam register TCNTx, apabila nilai TCNTx sudah penuh, maka masukan clock berikutnya akan menyebabkan nilai TCNTx kembali ke nol. Kondisi ini dapat dimanfaatkan sebagai sumber interupsi. Secara umum kondisi ini dinamakan timer overflow, ditandai dengan terjadiya set pada bii TOVx di register TIFR. Misalnya  sebuah register TCNT pada AVR adalah 8 bit,  maka nilai maksimunya adalah 255. Register register pada ATmega16 yang digunakan untuk timer maupun counter antara lain sebagai berikut:

* Register TCNT = register pencacah dari 0 sampai nilai maximum yg kit tentukan.
* Register TCCR =Untuk pengaturan mode  operasi  Timer/Counter
* Register TIMSK =Untuk  memilih Timer Counter  mana yg aktif.
* Register TIFR = Untuk mengetahui adanya interupsi akibat operasi Counter Timer .
* Register OC (output compare) = untuk menyimpan nilai pembanding dgn nilai pd register  TCNT.

Untuk pemilihan fitur timer/counter dengan menggunakan CodeVision AVR pengturan dan pemilihan register dapat dilakukan dengan mudah melalui CodeWizard seperti pada gambar dibawah.



Gambar 4.3 Pemilihan Timer/Counter di codewizard

Perhitungan untuk Timero dan Timer1 adalah sebagai berikut:

**Ttimer0 = Tosc\*(256-TCNT0)\*N**   → (8 bit = 256)

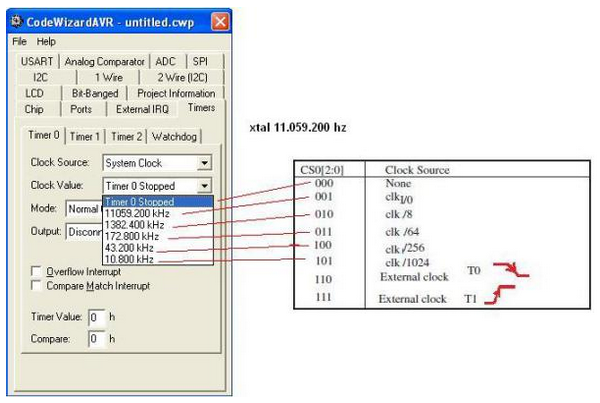
**Ttimer1 = Tosc\*(65536-TCNT1)\*N**   → (16 bit = 65536)

**Tosc = 1/Fosc**

Dimana:  
Ttimer0 = lamanya periode Timer0  
Ttimer1 = lamanya periode Timer1  
TCNT0 = Register Timer0  
TCNT1 = Register Timer1  
N = Skala clock/prescaler (mempunyai nilai 1, 8, 64, 256 dan 1024)  
Tosc = periode  clock  
Fosc = frekuensi clock Kristal

Pada dasarnya Timer hanya menghitung pulsa clock. Frekuensi pulsa clock yang dihitung tersebut bisa sama dengan frekuensi crystal yang digunakan atau dapat diperlambat menggunakan prescaler dengan faktor 8, 64, 256 atau 1024.

Pemilihan skala clock/prescaler pada CodeVision AVR yang dapat dilihat pada menu CodeWizard pada tab timer sudah berupa pilihan  frekwensi clock timer, sebagai contoh cala clock untuk xtal 11.059.200 hz dapat dilihat pada gambar dibawah ini.



Gambar 4.4 Pemilihan Prescaler Pada Code Vision

1. **COUNTER**

Sebagai counter/pencacah, sumber clock yang digunakan adalah clock eksternal atau berasal dari luar mikrokontroler (eksternal). Sebagai contoh counter, jika kita ingin menghitung banyaknya barang yang lewat pada konveyor, maka sumber clocknya berasal dari sensor yang mendeteksi barang saat melewatinya.

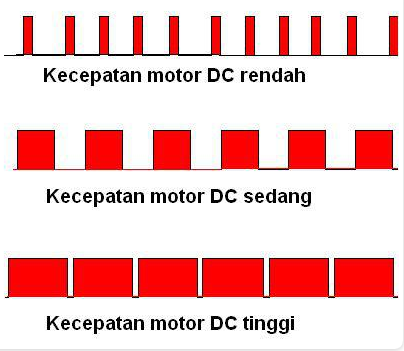
Ada dua mode untuk clock eksternal, yang pertama adalah Positive going Transition (PGT) atau biasa disebut Rising Edge pada CodeVision AVR, yaitu pencacah akan naik saat kondisi clock berubah dari kondisi Low ke High. Mode yang kedua sebaliknya, yaitu Negative Going Transition (NGT) atau pada CodeVision AVR disebut Falling Edge yang merupakan kebalikan dari Rising Edge.

Untuk penggunaan fungsi counter pada mikrokontroler lebih mudah jika dibandingkan dengan fungsi timer, karena tidak memerlukan perhitungan untuk penginputan nilai ke register TCNT. Register TCNT akan secara otomatis akan mencacah jika ada input yang masuk, input yang masuk contohnya dapat berupa push button.

Pada Counter 0, input berasal dari T0 atau PORT B 0 yang mampu mencacah input hingga 256 (8 Bit) sedangkan Pada Counter 1, input berasal dari T1 atau PORT B 1 yang mampu mencacah hingga 65536 (16 Bit). Untuk konfigurasi counter 0 & 1 pada Codevision AVR sama seperti konfigurasi Timer 0 dan 1.

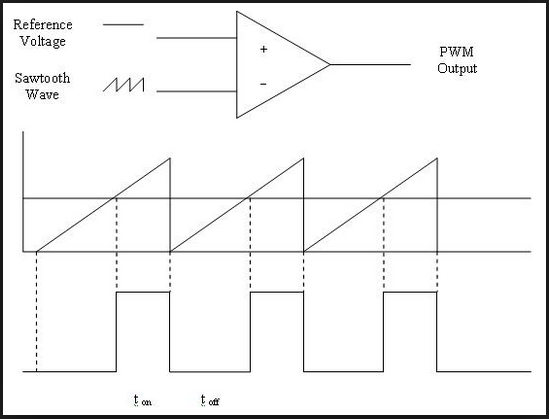
1. **PWM**

PWM (*Pulse Width Modulation*) atau modulasi lebar pulsa adalah salah satu keunggulan *Timer/Counter* yangterdapat pada Atmega16. Ketiga jenis *Timer/Counter* pada Atmega16 antara laian timer0, timer1 dan timer2, dapat menghasilkan pulsa PWM. Timer 1 menyediakan 2 buah sumber PWM dengan akurasi yang paling baik (resolusi maksimal 16 bit). Pin keluaran PWM untuk timer 0 adalah PB3 (OC0), tuntuk timer 1 adalah PD4 (OC1B) dan PD5 (OC1A), sedangkan untuk timer 2 adalah PD7 (OC2). Pulsa PWM berfungsi mengatur kecepatan motor DC, mengatur gelap terang LED dan aplikasi lainnya.



Gambar 4.5 Aplikasi PWM Untuk Motor DC

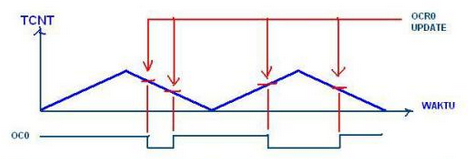
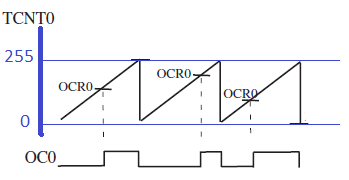
*Timer/Counter* 0 hanya memiliki PWM 8 bit, sedangkan pada *Timer/Counter* 1 memiliki 9 bit dan PWM 10 bit, dan *Timer/Counter* 2 memiliki PWM 8 bit. PWM merupakan kendali pengganti tegangan analog. Pengaturan level tegangan analog (dalam volt) tiganti dengan pengaturan waktu hidup (dalam % yang disebut duty cycle) suatu sinyal digital dengan frekwensi tertentu. Sinyal PWM terjadi dari hasil komparasi sinyal refrensi dengan sinyal segitiga seperti pada gambar 4.5.



Gambar 4.6 Mekanisme Pembangkitan PWM

Pada mikrokontroler ACR, sinyal segitiga didapatkan dari register pencacah timer TCNTx, sedangkan sinyal refrensi diset melalui register OCRx, apabila nilai OCRx lebih besar dari TCNTx maka keluaran komparator akan high, sedangkan apabila nilai OCRx lebih kecil maka keluaran komparator akan low.

Mode PWM ada dua yaitu PWM phase correct dan Fast PWM.  perbedananya yaitu pada PWM phase corect counter  (TCNT) bergulir/mencacah naik dan turun. sedangkan pada Fast PWM counter  (TCNT) hanya bergulir naik saja, selain itu pada PWM mode phase correct pembentukan sinyal PWM berupa double slope dan pada mode fast correct sinyal segitiga berupa gigi gergaji (saw tooth).



Gambar 4.7 PWM phase correct dan Fast PWM

Perumusan perhitungan unuk frekwensi output PWM bagi phase correct dan mode fast PWM yang dihasilkan oleh timer 0 dan timer 1 adalah sebagai berikut

* **Timer0**
* **Mode Phase Correct PWM**  
  Foc0 = Fosc/(N\*512)  
  D = (OCR0/255)\*100%
* **Mode Fast PWM**  
  Foc0 = Fosc/(N\*256)  
  D = (OCR0/255)\*100%

Dimana:  
Foc0 = Frekuensi output OC0  
N = Skala clock (mempunyai nilai 1, 8, 64, 256 dan 1024)  
D = Duty cycle  
Fosc = Frekuensi clock kristal yang digunakan

* **Timer1**
* **Mode Phase Correct PWM**  
  Foc1a = Fosc/(2\*N\*TOP)  
  Foc1b = Fosc/(2\*N\*TOP)  
  D = (OCR1X/TOP)\*100%
* **Mode Fast PWM**  
  Foc1a = Fosc/(N\*(1+TOP))  
  Foc1b = Fosc/(N\*(1+TOP))  
  D = (OCR1X/TOP)\*100%

Dimana:  
Foc1a = Frekuensi output OC1A  
Foc1b = Frekuensi output OC1B  
N = Skala clock (mempunyai nilai 1, 8, 64, 256 dan 1024)  
D = Duty cycle  
Fosc = Frekuensi clock kristal yang digunakan  
TOP = nilai maksimum counter (TCNT1), TOP mempunyai 3 buah nilai untuk kedua mode tersebut yaitu 8 bit (FF), 9 bit (1FF) dan 10 bit (3FF)

Nilai TOP menyatakan resolusi PWM. Nilai TOP untuk timer 0 dan timer 2 dapat diatur sebesar 8-bit, sedangkan untuk timer 1 sebesar 8-bit, 9-bit, atau 10-bit. Nilai TOP ini dapat ditentukan secara manual melalui register ICRx.

**C. PERALATAN**

1. Breadboard

2. Minimum Sistem ATmega 16

3. Downloader

4. Power Supply

5. Kabel jumper

7. LCD

8. Potentiometer/trimpot

9. Push Button

10. Resistor

11. Motor DC

12. Trnasistor NPN 2N2369

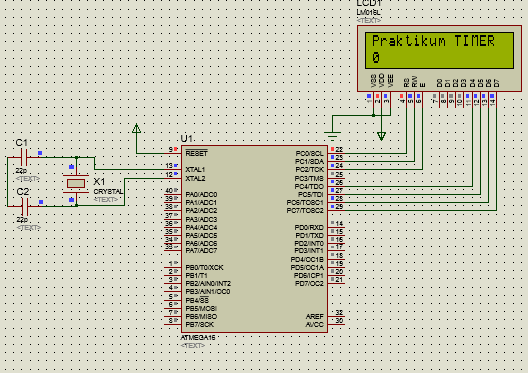
13. Osiloskop

1. **PROSEDUR PERCOBAAN**

**TIMER**

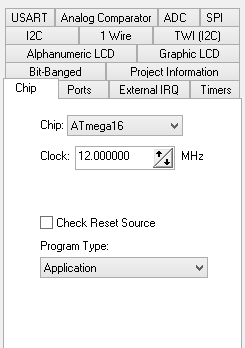
Pada praktikum ini akan akan dibuat sebuah aplikasi timer dengan menggunakan timer1 pada Atmega16 yang ditampilkan pada LCD, cara kerjanya pada LCD akan menampilkan nilai awal yaitu 0 kemudian setelah 1 detik (menggunakan timer) nilai tersebut akan naik menjadi 1, kemudian 2 dan seterusnya, jika sudah sampai 10 maka akan diset kembali menjadi 0. Langkah-langkahnya adalah sebagai berikut:

* 1. Buat rangkaian seperti gambar dibawah.



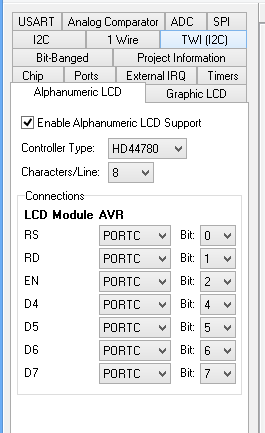
Gambar 4.8 Rangkaian LED dan push button

* 1. Buka CodeVision AVR, buat project baru
  2. Pada CodeWizard, pada tab Chip, pilih Atmega 16, dan isikan frekwensi clock sebesar 12.000000 Mhz



Gambar 4.9 Tab Chip

* 1. Pilih Tab Alphanumeric LCD, setting port c sebagai port output untuk LCD seperti pada gambar berikut.



Gambar 4.10 Setting Tab LCD

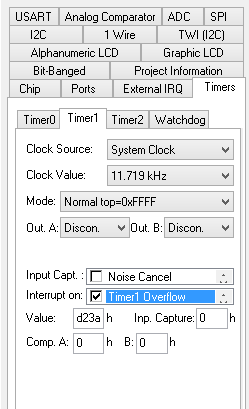
* 1. Pilih Tab Timer, Pada aplikasi ini diinginkan lamanya timer adalah 1 detik  (Ttimer1 = 1 detik) dan frekwensi Kristal yang digunakan adalah sebesar 12 MHz serta menggunakan skala clock N = 1024, maka dengan perhitungan menggunakan rumus timer1 didapat:

Tosc = 1/12Mhz = 0,0000000833 detik

Ttimer1 = Tosc\*(65536-TCNT1)\*N

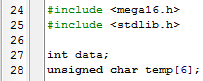
1 = 0,0000000833\*(65536-TCNT1)\*1024  
TCNT1= 53818 = D23A (dalam hexa)

Dari perhitungan diatas didapat nilai 53818 atau dalam hexa D23A, nilai tersebut harus diisikan pada register TCNT1 agar Timer 1 bernilai 1 detik. Sehingga setiingan yang harus dilakukan pada tab timer1 adalah sebagai berikut.



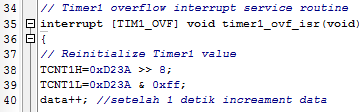
Gambar 4.11 Settingan Tab Timer1

* 1. Generate program, deklarasikan variable yang bernama “data”, sera tambahkan library #include <stdlib.h>, seperti pada gambar dibawah ini.



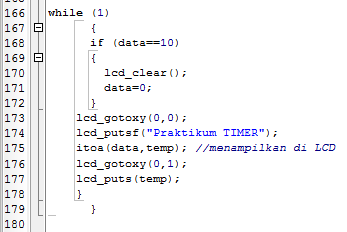
Gambar 4.12 Deklarasi Variabel dan Penambahan Library

* 1. Tambahkan increament data++ pada bagian overflow interrupt service routine, seperti pada baris 40 gambar berikut.



Gambar 4.13 Penambahan Increament

* 1. Tambahkan script dibawah ini pada program utama.



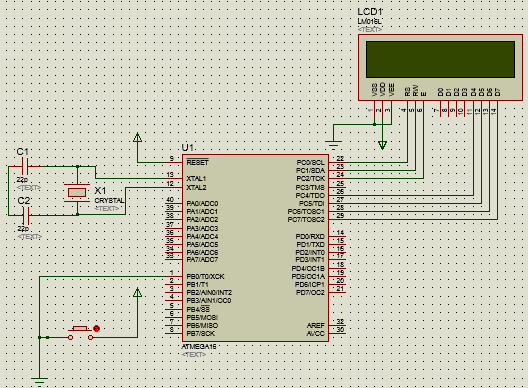
Gambar 4.14 Program Utama

* 1. Compile program tersebut dengan meng klik “compile the project”, Setelah itu masukkan file “\*.hex” yang berada pada folder “exe” ke mikrokontroller dengan menggunakan software khazama.
  2. Buat aplikasi timer yang sama seperti percobaan diatas tetapi dengan menggunakan timer0. Analisa perbedaan nya.

**COUNTER**

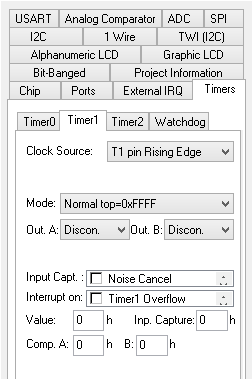
Pada praktikum counter kali ini akan dibuat aplikasi counter untuk menghitung/mencacah input yang berasal dari pushbutton dan ditampilkan pada LCD, langkah-langkahnya adalah sebagai berikut:

* 1. Buat rangkaian seperti gambar dibawah.



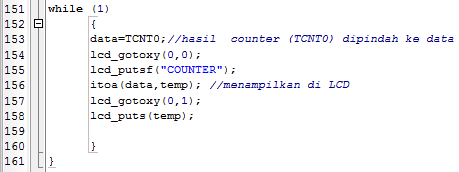
Gambar 4.15 Rangkaian Percobaan Counter

* 1. Buka CodeVision AVR, buat project baru
  2. Pada CodeWizard, pada tab Chip, pilih Atmega 16, dan isikan frekwensi clock sebesar 12.000000 Mhz
  3. Pilih Tab Alphanumeric LCD, setting port c sebagai port output untuk LCD.
  4. Pilih tab timer, pilih timer 1, buat timer 1 sebagai counter dengan, memilih sumber clock T0 dengan mode rising edge, seperti pada gambar dibawah.



Gambar 4.16 Setingan Tab Timer

* 1. Generate program, deklarasikan variable yang bernama “data”, sera tambahkan library #include <stdlib.h>, seperti pada percobaan sebelumnya.
  2. Tambahakan script berikut pada program utama.



Gambar 4.17 Program Utama Counter

* 1. Compile program tersebut dengan meng klik “compile the project”, Setelah itu masukkan file “\*.hex” yang berada pada folder “exe” ke mikrokontroller dengan menggunakan software khazama.
  2. Buat aplikasi Counter yang sama seperti percobaan diatas tetapi dengan menggunakan metode falling edge. Analisa perbedaan nya.

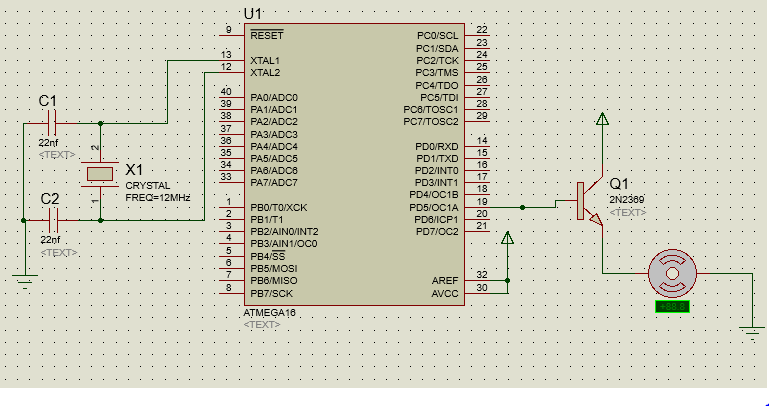
**PWM**

Pada Percobaan kali ini akan dibuat aplikasi membangkitkan sinyal PWM dengan periode 20 ms dengan duty cycle 75%**,** menggunakan Timer1 10 Bit Mode Fast PWM. Dengan menggunakan kristal 12 Mhz, prescaler/N = 256 dan TOP = 10 bit = 3FF = 1023

Maka akan didapat frekuensi output (Foc1x) sebesar 45,77 Hz atau jika diubah kedalam peroide 21,8 ms ≈ 20 ms

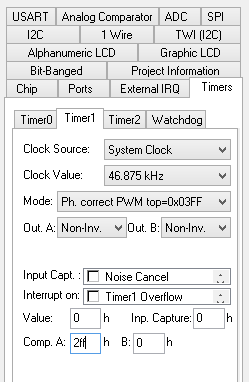
Untuk Duty cycle:  
D = (OCR1X/TOP)\*100%  
75% = (OCR1X/1023)\*100%  
OCR1X = 767 = 2FF (dalam hexa) > perhitungan ini kemudian dimasukkan sebagai input settingan pada tab Timer1 di CodeWizard yang akan di jelaskan pada langkah-langkah berikut ini:

1. Buat rangkaian dengan LED dan push button seperti pada gambar dibawah, lalu hubungkan dengan minimum sistem atmega sesuai dengan dengan port yang sesuai dengan gambar dibawah.



Gambar 4.18 Rangkaian Percobaan PWM

1. Buka CodeVision AVR, buat project baru
2. Pada CodeWizard, pada tab Chip, pilih Atmega 16, dan isikan frekwensi clock sebesar 12.000000 Mhz
3. Pilih tab Timer, Pilih timer 1, masukkan hasil perhitungan register OCR1X = 2FF untuk menghasilkan dutycycle sebesar 75%, Clock Value bernilai 46.875 berasal dari Fosc/N atau 12 MHz/256.  dan lakukan settingan pada tab timer1 seperti pada gambar berikut.



Gambar 4.19 Setting Tab Timer1

1. Generate program.
2. Compile program tersebut dengan meng klik “compile the project”, Setelah itu masukkan file “\*.hex” yang berada pada folder “exe” ke mikrokontroller dengan menggunakan software khazama.
3. Buat aplikasi PWM seperti pada praktikum diatas namun dengan duty cycle sebesar 50 % dengan periode yang sama. Sambungkan mikrokontroler dengan osiloskop. Amati perbedaan sinyal keluarannya, hubungkan dengan kecepatan motor DC yang dihasilkan kemudian simpulkan.