MODUL 9
Memprogram Timer Counter

BAGIAN 1
Memprogram Timer Counter

Tujuan Pembelajaran Umum:

1. Mahasiswa trampil memprogram Timer Counter

Tujuan Pembelajaran Khusus:

1. Mahasiswa memahami fasilitas Timer Counter pada mikrokontroler AT89S51
2. Mahasiswa memahami pemrograman Timer Counter mikrokontroler AT89S51

Timer dan Counter dalam AT89S51 (Naskah Asli Tulisan Budhy Sutanto)

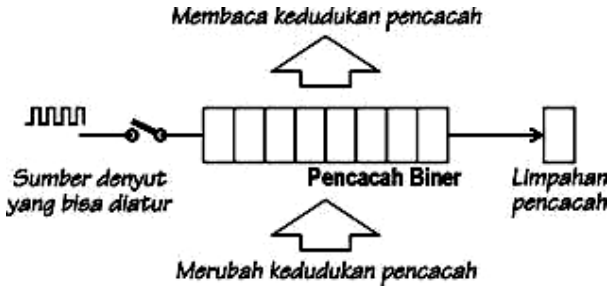
Timer dan Counter merupakan sarana input yang kurang dapat perhatian pemakai mikrokontroler, dengan sarana input ini mikrokontroler dengan mudah bisa dipakai untuk mengukur lebar pulsa, membangkitkan pulsa dengan lebar yang pasti, dipakai dalam pengendalian tegangan secara PWM (Pulse Width Modulation) dan sangat diperlukan untuk aplikasi remote control dengan infra merah.

Pada dasarnya sarana input yang satu ini merupakan seperangkat pencacah biner (binary counter) yang terhubung langsung ke saluran-data mikrokontroler, sehingga mikrokontroler bisa membaca kedudukan pancacah, bila diperlukan mikrokontroler dapat pula merubah kedudukan pencacah tersebut.

Seperti layaknya pencacah biner, bilamana sinyal denyut (clock) yang diumpankan sudah melebihi kapasitas pencacah, maka pada bagian akhir untaian pencacah akan timbul sinyal limpahan, sinyal ini merupakan suatu hal yang penting sekali dalam pemakaian pencacah. Terjadinya limpahan pencacah ini dicatat dalam sebuah flip-flop tersendiri.

Di samping itu, sinyal denyut yang diumpankan ke pencacah harus pula bisa dikendalikan dengan mudah. Hal-hal yang dibicarakan di atas diringkas dalam Gambar 39.

MODUL 9
Memprogram Timer Counter



Gambar 39
Konsep dasar Timer/Counter sebagai sarana input

Sinyal denyut yang diumpankan ke pencacah bisa dibedakan menjadi 2 macam, yang pertama ialah sinyal denyut dengan frekuensi tetap yang sudah diketahui besarnya dan yang kedua adalah sinyal denyut dengan frekuensi tidak tetap.

Jika sebuah pencacah bekerja dengan frekuensi tetap yang sudah diketahui besarnya, dikatakan pencacah tersebut bekerja sebagai *timer*, karena kedudukan pencacah tersebut setara dengan waktu yang bisa ditentukan dengan pasti.

Jika sebuah pencacah bekerja dengan frekuensi yang tidak tetap, dikatakan pencacah tersebut bekerja sebagai *counter*, kedudukan pencacah tersebut hanyalah menyatakan banyaknya pulsa yang sudah diterima pencacah.

Untaian pencacah biner yang dipakai, bisa merupakan *pencacah biner menaik* (*count up binary counter*) atau *pencacah biner menurun* (*count down binary counter*).

Timer/Counter sebagai sarana input banyak dijumpai dalam mikrokontroler, misalnya mikrokontroler keluarga MCS48, keluarga MCS51 ataupun MC68HC11 semuanya memiliki Timer/Counter di dalam chip sebagai sarana input. Di samping itu bisa pula dijumpai chip Timer/Counter yang berdiri sendiri sebagai penunjang kerja mikroprosesor, misalnya 8253/8254 Programmable Interval Timer buatan Intel, atau MC6840 Programmable Counter/Timer buatan Motorola.

Sarana Timer/Counter dalam AT89S51

Keluarga mikrokontroler MCS51, misalnya AT89C51 dan AT89Cx051, dilengkapi dengan dua perangkat Timer/Counter, masing-masing dinamakan sebagai *Timer 0* dan *Timer 1*. Sedangkan untuk jenis yang lebih *besar*, misalnya AT89C52, mempunyai tambahan satu perangkat Timer/Counter lagi yang dinamakan sebagai *Timer 2*.

MODUL 9
Memprogram Timer Counter

Perangkat Timer/Counter tersebut merupakan perangkat keras yang menjadi satu dalam chip mikrokontroler MCS51, bagi pemakai mikrokontroler MCS51 perangkat tersebut dikenal sebagai **SFR** (*Special Function Register*) yang berkedudukan sebagai *memori-data internal*.

Pencacah biner untuk *Timer 0* dibentuk dengan register **TL0** (*Timer 0 Low Byte*, memori-data internal nomor 6AH) dan register **TH0** (*Timer 0 High Byte*, memori-data internal nomor 6CH).

Pencacah biner untuk *Timer 1* dibentuk dengan register **TL1** (*Timer 1 Low Byte*, memori-data internal nomor 6BH) dan register **TH1** (*Timer 1 High Byte*, memori-data internal nomor 6DH).

Pencacah biner pembentuk Timer/Counter MCS51 merupakan *pencacah biner menaik* (*count up binary counter*) yang mencacah dari **0000H** sampai **FFFFH**, saat kedudukan pencacah berubah dari **FFFFH** kembali ke **0000H** akan timbul sinyal limpahan.

Untuk mengatur kerja Timer/Counter dipakai 2 register tambahan yang dipakai bersama oleh *Timer 0* dan *Timer 1*. Register tambahan tersebut adalah register **TCON** (*Timer Control Register*, memori-data internal nomor 88H, bisa dialamat secara bit) dan register **TMOD** (*Timer Mode Register*, memori-data internal nomor 89H).

Pencacah biner Timer 0 dan 1

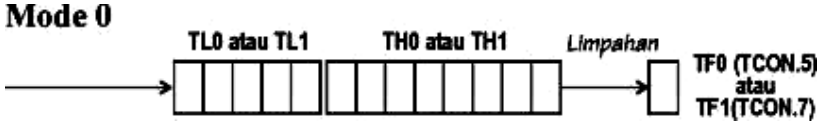
TL0, **TH0**, **TL1** dan **TH1** merupakan **SFR** (*Special Function Register*) yang dipakai untuk membentuk pencacah biner perangkat *Timer 0* dan *Timer 1*. Kapasitas keempat register tersebut masing-masing 8 bit, bisa disusun menjadi 4 macam Mode pencacah biner seperti terlihat dalam Gambar 40a sampai Gambar 40d.

Pada Mode 0, Mode 1 dan Mode 2 *Timer 0* dan *Timer 1* masing-masing bekerja sendiri, artinya bisa dibuat *Timer 0* bekerja pada Mode 1 dan *Timer 1* bekerja pada Mode 2, atau kombinasi mode lainnya sesuai dengan keperluan.

Pada Mode 3 **TL0**, **TH0**, **TL1** dan **TH1** dipakai bersama-sama untuk menyusun sistem timer yang tidak bisa di-kombinasi lain.

Susunan **TL0**, **TH0**, **TL1** dan **TH1** pada masing-masing mode adalah sebagai berikut:

Mode 0 – Pencacah Biner 13 bit

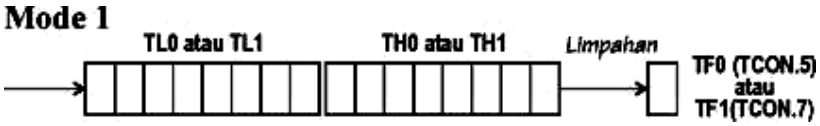


Gambar 40a
Mode 0 - Pencacah Biner 13 Bit

Pencacah biner dibentuk dengan **TLx** (maksudnya bisa **TL0** atau **TL1**) sebagai pencacah biner 5 bit (meskipun kapasitas sesungguhnya 8 bit), limpahan dari pencacah biner 5 bit ini dihubungkan ke **THx** (maksudnya bisa **TH0** atau **TH1**) membentuk sebuah untaian pencacah biner 13 bit, limpahan dari pencacah 13 bit ini ditampung di flip-flop **TFx** (maksudnya bisa **TF0** atau **TF1**) yang berada di dalam register **TCON**.

Mode ini meneruskan sarana Timer yang ada pada mikrokontroler MCS48 (mikrokontroler pendahulu MCS51), dengan maksud rancangan alat yang dibuat dengan MCS48 bisa dengan mudah diadaptasikan ke MCS51. Mode ini tidak banyak dipakai lagi.

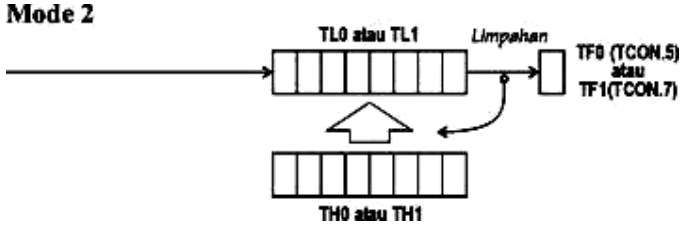
Mode 1 – Pencacah Biner 16 bit



Gambar 40b
Mode 1 - Pencacah Biner 16 Bit

Mode ini sama dengan *Mode 0*, hanya saja register **TLx** dipakai sepenuhnya sebagai pencacah biner 8 bit, sehingga kapasitas pencacah biner yang tersbentuk adalah 16 bit. Seiring dengan sinyal denyut, kedudukan pencacah biner 16 bit ini akan bergerak dari 0000H (biner 0000 0000 0000 0000), 0001H, 0002H … sampai FFFFH (biner 1111 1111 1111 1111), kemudian melimpah kembali menjadi 0000H.

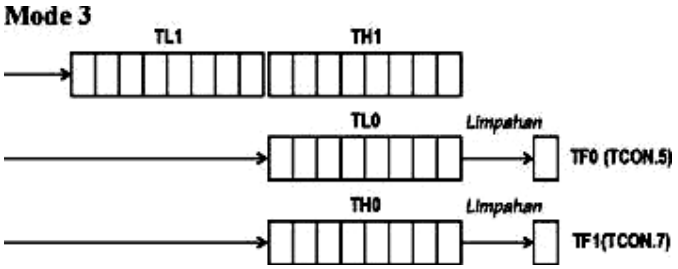
Mode 2 – Pencacah Biner 8 bit dengan Isi Ulang



Gambar 40c
Mode 2 - Pencacah Biner 8 Bit dengan Isi Ulang

TLx dipakai sebagai pencacah biner 8 bit, sedangkan **THx** dipakai untuk menyimpan nilai yang diisikan ulang ke **TLx**, setiap kali kedudukan **TLx** melimpah (berubah dari FFH menjadi 00H). Dengan cara ini bisa didapatkan sinyal limpahan yang frekuensinya ditentukan oleh nilai yang disimpan dalam **TH0**.

Mode 3 – Gabungan Pencacah Biner 16 bit dan 8 Bit

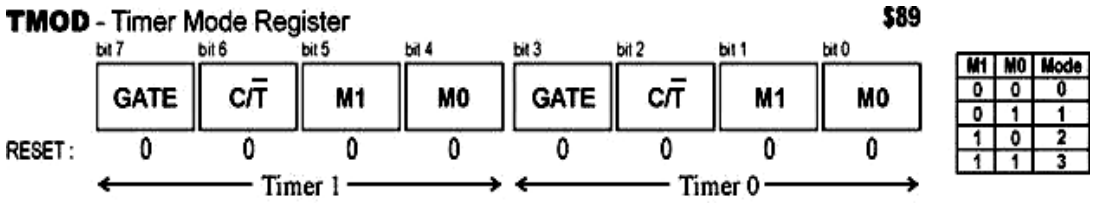


Gambar 40d
Mode 3 – Gabungan Pencacah Biner 16 Bit dan 8 Bit

Pada Mode 3 **TL0**, **TH0**, **TL1** dan **TH1** dipakai untuk membentuk 3 untaian pencacah, yang pertama adalah untaian pencacah biner 16 bit tanpa fasiltas pemantau sinyal limpahan yang dibentuk dengan **TL1** dan **TH1**. Yang kedua adalah **TL0** yang dipakai sebagai pencacah biner 8 bit dengan **TF0** sebagai sarana pemantau limpahan. Pencacah biner ketiga adalah **TH0** yang dipakai sebagai pencacah biner 8 bit dengan **TF1** sebagai sarana pemantau limpahan.

Register Pengatur Timer

Register **TMOD** dan register **TCON** merupakan register pembantu untuk mengatur kerja *Timer 0* dan *Timer 1*, kedua register ini dipakai bersama oleh *Timer 0* dan *Timer 1*.



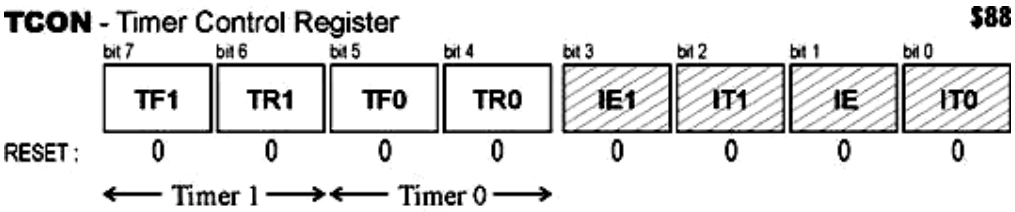
Gambar 41a
Denah susunan bit dalam register TMOD

Register **TMOD** dibagi menjadi 2 bagian secara simitris, bit 0 sampai 3 register **TMOD** (**TMOD** bit 0 .. **TMOD** bit 3) dipakai untuk mengatur *Timer 0*, bit 4 sampai 7 register **TMODE** (**TMOD** bit 4 .. **TMOD** bit 7) dipakai untuk mengatur *Timer 1*, pemakaiannya sebagai berikut :

- Bit **M0/M1** dipakai untuk menentukan Mode Timer seperti yang terlihat dalam Tabel di Gambar 41a.
- Bit **C/T*** dipakai untuk mengatur sumber sinyal denyut yang diumpankan ke pencacah biner. Jika **C/T***=0 sinyal denyut diperoleh dari osilator kristal yang frekuensinya sudah dibagi 12,

sedangkan jika **C/T***=1 maka sinyal denyut diperoleh dari kaki **T0** (untuk *Timer 0*) atau kaki **T1** (untuk *Timer 1*).

- Bit **GATE** merupakan bit pengatur saluran sinyal denyut. Bila bit **GATE**=0 saluran sinyal denyut hanya diatur oleh bit **TRx** (maksudnya adalah **TR0** atau **TR1** pada register **TCON**). Bila bit **GATE**=1 kaki **INT0** (untuk *Timer 0*) atau kaki **INT1** (untuk *Timer 1*) dipakai juga untuk mengatur saluran sinyal denyut (lihat Gambar 42).



Gambar 41b
Denah susunan bit dalam register TCON

Register **TCON** dibagi menjadi 2 bagian, 4 bit pertama (bit 0 .. bit 3, bagian yang diarsir dalam Gambar 41b) dipakai untuk keperluan mengatur kaki **INT0** dan **INT1**, ke-empat bit ini dibahas dibagian lain.

Sisa 4 bit dari register **TCON** (bit 4..bit 7) dibagi menjadi 2 bagian secara simitris yang dipakai untuk mengatur *Timer0*/*Timer 1*, sebagai berikut:

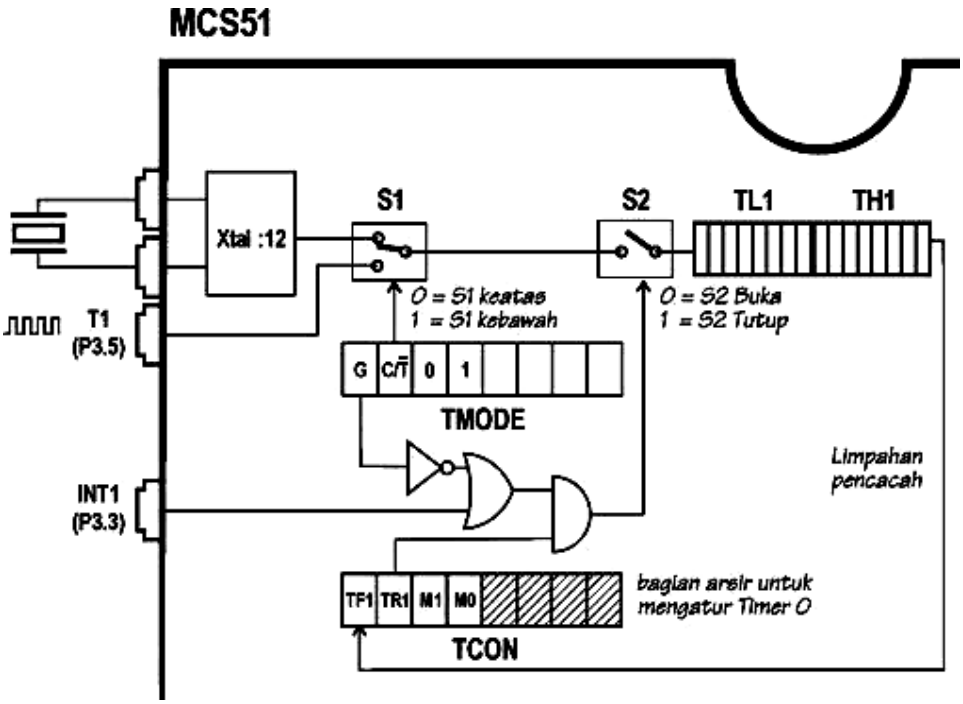
- Bit **TFx** (maksudnya adalah **TF0** atau **TF1)** merupakan bit penampung limpahan (lihat Gambar 40), **TFx** akan menjadi '**1**' setiap kali pencacah biner yang terhubung padanya melimpah (kedudukan pencacah berubah dari FFFFH kembali menjadi 0000H). Bit **TFx** di-nol-kan dengan istruksi **CLR TF0** atau **CLR TF1**. Jika sarana interupsi dari *Timer 0*/*Timer 1* dipakai, **TRx** di-nol-kan saat AT89S51 menjalankan *rutin layanan interupsi* (**ISR** – *Interupt Service Routine*).
- Bit **TRx** (maksudnya adalah **TR0** atau **TR1)** merupakan bit pengatur saluran sinyal denyut, bila bit ini =**0** sinyal denyut tidak disalurkan ke pencacah biner sehingga pencacah berhenti mencacah. Bila bit **GATE** pada register **TMOD** =**1**, maka saluran sinyal denyut ini diatur bersama oleh **TRx** dan sinyal pada kaki **INT0**/**INT1** (lihat Gambar 42).

MODUL 9
Memprogram Timer Counter

Mengatur Timer

Gambar 42 merupakan bagan susunan rangkaian yang bisa terjadi pada *Timer 1* secara lengkap, digambarkan pula hubungan-hubungan semua register pembentuk dan pengatur *Timer 1*. Gambar ini berlaku pula untuk *Timer 0.*

Dalam pemakaian sesungguhnya, rangkaian yang dipakai hanya sebagian dari rangkaian lengkap tersebut, sesuai dengan keperluan sistem yang dibangun. Rangkaian yang dikehendaki dibentuk dengan mengatur register **TMODE**, sedangkan kerja dari Timer dikendalikan lewat register **TCON**.



Gambar 42
Skema lengkap Timer 1 dalam Mode 1

Setelah MCS51 di-reset register **TMOD** bernilai 00H, hal ini berarti :

- bit **C/T*** =’**0**’, menurut Gambar 42 keadaan ini membuat saklar **S1** ke posisi atas, sumber sinyal denyut berasal dari osilator kristal yang frekuensinya sudah dibagi 12, pencacah biner yang dibentuk dengan **TL1** dan **TH1** berfungsi sebagai ***timer***. Jika sistem yang dirancang memang menghendaki *Timer 1* bekerja sebagai ***timer*** maka bit **C/T*** tidak perlu diatur lagi.

Tapi jika sistem yang dirancang menghendaki agar *Timer 1* bekerja sebagai ***counter*** untuk menghitung pulsa yang masuk lewat kakai **T1** (**P3.5**), maka posisi saklar **S1** harus dikebawahkan dengan membuat bit **C/T*** menjadi '**1**'.

- bit **GATE**='**0**', hal ini membuat output gerbang OR selalu '**1**' tidak dipengaruhi keadaan '**0**' atau '**1**' pada kaki **INT1** (**P3.3**). Dalam keadaan semacam ini, saklar **S2** hanya dikendalikan lewat bit **TR1** dalam register **TCON**. Jika **TR1**='**1**' saklar **S2** tertutup sehingga sinyal denyut dari **S1** disalurkan ke sistem pencacah biner, aliran sinyal denyut akan dihentikan jika **TR='0'**. Sebaliknya jika bit **GATE**='**1**', output gerbang OR akan mengikuti keadaan kaki **INT1**, saat **INT1**='**0**' apa pun keadaan bit **TR1** output gerbang AND selalu ='**0**' dan saklar **S1** selalu terbuka, agar saklar **S1** bisa tertutup kaki **INT1** dan bit **TR1** harus ='**1**' secara bersamaan.

Jika sistem yang dirancang menghendaki kerja dari timer/counter dikendalikan dari sinyal yang berasal dari luar chip, maka bit **GATE** harus dibuat menjadi '**1**' bit **M1** dan **M0**='**0**', berarti **TL1** dan **TH1** disusun menjadi pencacah biner 13 bit (Mode 0), jika dikehendaki *Timer 1* bekerja pada mode 1 seperti terlihat dalam Gambar 4, maka bit **M1** harus dibuat menjadi '**0**' dan bit **M0** menjadi '**1'**.

Pengetahuan di atas dipakai sebagai dasar untuk mengatur dan mengendalikan Timer seperti terlihat dalam contoh-contoh berikut :

Setelah reset **TMOD** bernilai **00H**, berarti *Timer 1* bekerja sebagai pencacah biner 13 bit, sumber sinyal denyut dari osilator kristal atau *Timer 1* bekerja sebagai 'timer', bit **GATE** ='**0**' berarti kaki **INT1** tidak berpengaruh pada rangkaian sehingga *Timer 1* hanya dikendalikan dari bit **TR1**.

Dalam pemakaian biasanya dipakai pencacah biner 16 bit, untuk keperluan itu instruksi yang diperlukan untuk mengatur **TMOD** adalah :

MOV TMOD,#00010000B

Catatan dalam instruksi di atas tanda '**#**' menyatakan bagian di belakangnya adalah bilangan konstan yang akan diisikan ke **TMOD**, '**%**' merupakan awalan yang menandakan bahwa bilangan di belakangnya adalah bilangan biner. Penulisan dengan bilangan biner semacam ini, memudahkan untuk mengenali dengan cepat bit-bit apa saja yang diisikan ke **TMOD**.

Bilangan biner **00010000B** diisikan ke **TMOD**, berakibat bit 7 **TMOD** (bit **GATE**) bernilai '**0**', bit 6 (bit **C/T***) bernilai '**0**', bit 5 dan 4 (bit **M1** dan **M0**) bernilai '**01'**, ke-empat bit ini

dipakai untuk mengatur *Timer 1*, sehingga Timer 1 bekerja sebagai *timer dengan pencacah biner 16 bit yang dikendalikan hanya dengan TR1*.

Jika dikehendaki pencacah biner dipakai sebagai counter untuk mencacah jumlah pulsa yang masuk lewat kaki **T1** (**P3.5**), instruksinya menjadi :

MOV TMOD,#01010000B

Perbedaannya dengan instruksi di atas adalah dalam instruksi ini bit 6 (bit **C/T***) bernilai '**1**'.

Selanjutnya jika diinginkan sinyal dari perangkat keras di luar chip MCS51 bisa ikut mengendalikan *Timer 1*, instruksi pengatur *Timer 1* akan menjadi :

MOV TMOD,#11010000B

Dalam hal ini bit 7 (bit **GATE**) bernilai '**1**'.

Setelah mengatur konfigurasi *Timer 0* seperti di atas, pencacah biner belum mulai mencacah sebelum diperintah dengan instruksi :

SETB TR1

Perlu diingatkan jika bit **GATE** = '**1**', selama kaki **INT1** bernilai '**0**' pencacah biner belum akan mencacah. Untuk menghentikan proses pencacahan, dipakai instruksi

CLR TR1

Di atas hanya dibahas *Timer 1* saja, tata canya untuk *Timer 0* persis sama. Yang perlu diperhatikan adalah register **TMOD** dipakai untuk mengatur *Timer 0* dan juga *Timer 1*, sedangkan **TMOD** *tidak bisa dialamati secara bit* (*non bit addressable*) sehingga jika jika kedua Timer dipakai, pengisian bit-bit dalam register **TMOD** harus dipikirkan sekali gus untuk *Timer 0* dan *Timer 1*.

Bit **TR1** dan **TR0** yang dipakai untuk mengendalikan proses pencacahan, terletak di dalam register **TCON** (memori-data internal nomor $88) yang *bisa dialamati secara bit* (*bit addressable*). Sehingga **TR0** dan **TR1** bisa diatur secara terpisah (dengan perintah **SETB** atau **CLR**), tidak seperti mengatur **TMOD** yang harus dilakukan secara bersamaan.

Demikian pula *bit penampung limpahan pencacah biner* **TF0** dan **TF1**, juga terletak dalam register **TCON** yang masing-masing bisa di-monitor sendiri.

Pemakaian waktu tunda

Waktu tunda banyak dipakai dalam pemerograman mikronkontroler untuk membangkitkan pulsa, membangkitkan sinyal periodik dengan frekuensi tertentu, untuk menghilangkan effek getar dari skalar dalam membuat key pad (keyboard sederhana) dan lain sebagainya.

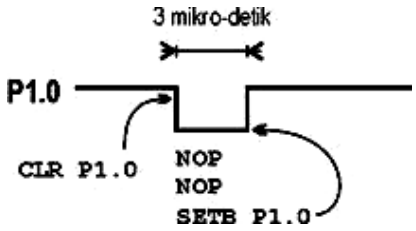
MODUL 9
Memprogram Timer Counter

Waktu tunda bisa dibangkitkan secara sederhana dengan menjalankan instruksi-instruksi yang waktu pelaksanaanya bisa diperhitungkan dengan tepat. Untuk mendapatkan waktu tunda yang panjang, tidak dipakai cara di atas tapi pakai Timer. Waktu tunda yang dibentuk dengan kedua cara tersebut sangat tergantung pada frekuensi kerja mikrokontroler, dalam contoh-contoh berikut dianggap mikrokontroler bekerja pada frekuensi 12 MHz.

Instruksi-instruksi berikut ini bisa dipakai untuk membangkitkan pulsa '**0**' dengan lebar 3 mikro-detik pada kaki **P1.0**

```
01:   CLR P1.0
02:   NOP          ; 1 mikro-detik
03:   NOP          ; 1 mikro-detik
04:   SETB P1.0    ; 1 mikro-detik
```

Instruksi baris pertama membuat **P1.0** yang mula-mula '**1**' menjadi '**0**', pelaksanaan instruksi **NOP** memerlukan waktu 1 mikro-detik (jika MCS51 bekerja pada frekuensi 12 MHz), instruksi **SETB P1.0** juga memerlukan waktu 1 mikro-detik, total waktu sebelum **P1.0** kembali menjadi '**1**' adalah 3 mikro-detik (baris 2 3 dan 4). Dengan demikian terjadilah pulsa dengan lebar 3 mikro-detik pada kaki **P1.0** seperti terlihat pada Gambar 5.



Gambar 43
Pulsa 3 mikro-detik pada P1.0

Dengan sedikit perubahan instruksi-instruksi di atas bisa membangkitkan sinyal dengan frekuensi 100 KHz pada kaki **P1.0** :

```
01: Sinyal 100KHz:
02:   CPL  P1.0       ; 1 mikro-detik
03:   NOP             ; 1 mikro-detik
04:   NOP             ; 1 mikro-detik
05:   SJMP Sinyal100KHz ; 2 mikro-detik
```

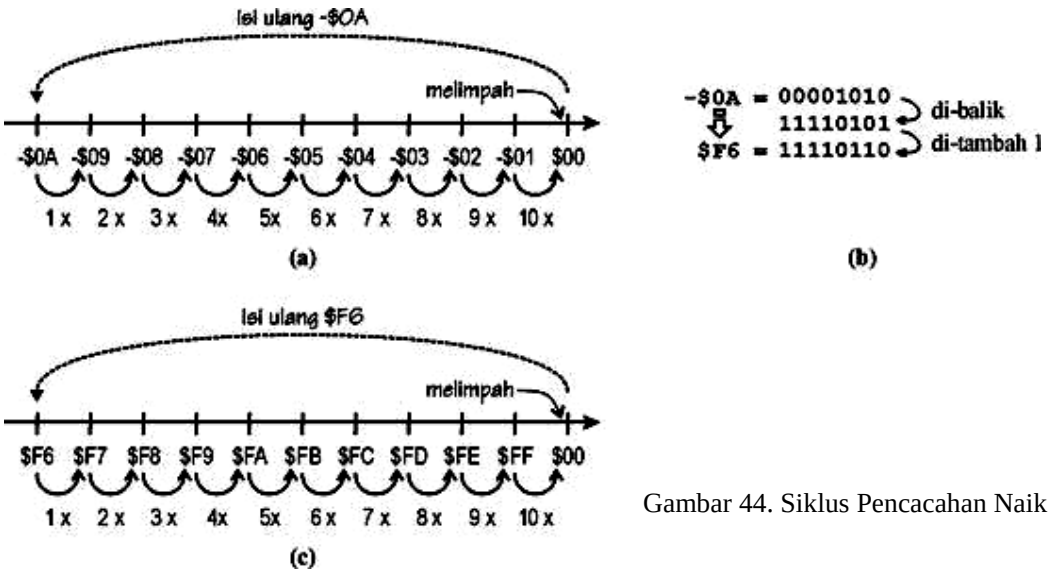
Instruksi **CPL P1.0** pada baris 1 membalik keadaan pada **P1.0**, bila mula-mula **P1.0** bernilai '**1**' akan dirubah menjadi '**0**', sebaliknya bila mula-mula '**0**' akan dirubah menjadi '**1**'. Total waktu tunda ke-empat baris di atas adalah 5 mikro-detik, sehingga yang terjadi adalah **P1.0** bernilai '**0**'

selama 5 mikro-detik dan bernilai '**1**' selama 5 mikro-detik berulang terus tanpa henti, dengan frekuensi sebesar 1/10 mikro-detik = 100.000 Hertz.

Program di atas bisa pula dibuat dengan memakai *Timer 1* sebagai pengatur waktu tunda sebagai berikut :

```
01:    MOV    TMOD,#00100000B    ; Timer 1 bekerja pada Mode 2
02:    MOV    TH1,#F6H           ; Nilai pengisi ulang TL1
03:    SET    TR1                ; Timer 1 mulai mencacah
04: Ulangi:
05:    BIT    TF1,$              ; Tunggu sampai melimpah
06:    CPL    P1.0               ; Keadaan pada P1.0 di-balik
07:    CLR    TF1                ; Hapus limpahan pencacah
08:    SJMP   Ulangi             ; Ulangi terus tiada henti…
```

Instruksi baris pertama mempersiapkan *Timer 0* bekerja pada Mode 2 – *Pencacah Biner 8 bit dengan Isi Ulang*, bilangan pengisi ulang ditentukan sebesar **F0H** yang disimpan ke register **TH1** pada baris 2, instruksi berikutnya memerintahkan pencacah biner mulai mencacah.



Gambar 44. Siklus Pencacahan Naik

Pencacah biner yang dibentuk dengan register **TL1** akan mencacah naik seirama dengan sikluas sinyal denyut, mulai dari **F6H** sampai **FFH**, saat pencacah melimpah dari **FFH** ke **00H** bit **TR1** pada register **TCON** akan menjadi '**1**' dan **TL1** secara otomatis di isi ulang dengan bilangan **F0H** yang tersimpan pada register **TH0**. Hal ini akan terjadi terus menerus dan berulang setiap 10 siklus sinyal denyut (**F6H**, **F7H**, **F8H**, **F9H**, **FAH**, **FBH**, **FCH**, **FDH**, **FEH**, **FFH** kembali ke **00H**, total 10 siklus)

MODUL 9
Memprogram Timer Counter

Instruksi **BIT TR1,$** menunggu bit **TR1** menjadi '**1**', yakni saat pecacah biner melimpah dari **FFH** ke **00H** yang dibahas di atas. Lepas dari penantian tersebut, **P1.0** dibalik keadaanya dengan instruksi **CPL P1.0**, **TR1** dikembalikan menjadi 0 (harus dikembalikan sendiri dengan instruksi ini), agar bisa ditunggu lagi sampai menjadi '**1**' kembali setelah instruksi **SJMP Ulangi**.

Frekuensi dari sinyal di P1.0 sebesar 1 / 16 mikro-detik = 31,25 KHz.

Nilai awal yang diisikan ke pencacah biner di atas, adalah nilai negatip dari faktor pembagi yang dikehendaki. Dalam contoh di atas, bilangan pembaginya adalah 16 (atau **10H** heksadesimal), nilai negatip dari **10H** adalah **F0H** (bilangan negatip komplemen 2 diturunkan dengan cara bilangan asal dibalik/di-not-kan kemudian ditambah 1, dalam hal ini **10H** di-not-kan menjadi **EFH** dan ditambah 1 menjadi **F0H**).

Cara menentukan bilangan negatip di atas cukup merepotkan, tapi hal ini mudah diselesaikan dengan bantuan assembler, instruksi **MOV TH1,#F0H** di atas bisa digantikan dengan **MOV TH1,#-10H**, assembler yang akan menghitung **-10H** menjadi **F0H**.

Contoh pemakaian waktu tunda berikutnya adalah untuk mengatasi getaran saklar dalam membuat key pad (keyboard sederhana).

Pada saat saklar mekanis di tekan atau pada saat tekanan pada saklar dilepas, selama lebih kurang 30 mili-detik saklar tersebut akan bergetar atau sebelum mencapai keadaan stabil saklar tersebut akan on/off selama waktu lebih kurang 30 detik, mengakibatkan program mikrokontroler merasakan tombol berulang-ulang ditekan, meskipun sesungguhnya hanya ditekan satu kali saja.

Hal ini bisa diatasi dengan cara menunda waktu sekitar 50 mili-detik setelah program merasakan sebuah tombol ditekan, seperti berikut:

```
01: BacaTombol:
02:     MOV     TMOD,#00010000B     ; Timer 1 bekerja pada mode 1
03:     JB      T1,$                ; tunggu di sini sampai tombol ditekan
04:     MOV     TL1,#-50000/256     ; siapkan waktu tunda 50 mili-detik
05:     MOV     TH1,#-50000
06:     CLR     TF1                 ; me-nol-kan bit limpahan
07:     SETB    TR1                 ; timer mulai bekerja
08:     JNB     TF1,$               ; tunggu di sini sampai melimpah
09:     CLR     TR1                 ; timer berhenti kerja
10:     RET
```

MODUL 9
Memprogram Timer Counter

Dalam program di atas, saklar dipasangkan antara kaki **T1** dan Ground, pencacah biner yang dipakai adalah pencacah biner 16 yang sudah ditentukan dengan instruksi **MOV TMOD,#00010000B**. Pada baris berikut program menunggu saklar yang terhubung pada **T1** ditekan, selama saklar belum ditekan mikrokontroler akan tertahan pada instruksi **JB T1,$**.

Dua baris berikutnya adalah cara mengisikan nilai –**50000** yang terdiri dari 2 byte ke register **TL1** dan **TH1**. Bit **TF1** di-nol-kan dan akan kembali menjadi '**1**' setelah **Tl1/TH1** mencacah dari –**50000** kembali menjadi **0000h**. Waktu tunda dimulai segera setelah instruksi **SETB TR1**, dan selesainya waktu tunda ditunggu dengan instruksi **JNB TF1**,**$**. Jika MCS51 bekerja dengan kristal 12 MHz, waktu tunda selama 50000 siklus sama dengan 50.000 x 1 mikro-detik = 50 mili-detik,

Setelah menunggu selama 50 mili-detik, pencacah biner kembali di-non-aktipkan dengan **CLR TR1** dan berikutnya meninggalkan sub-rutin ini dengan instruksi **RET**. Dengan demikian mikrokontroler akan menungu saklar stabil, selama saklar masih bergetar mikrokontroler masih tertahan di dalam su-rutin **BacaTombol**.

Untuk mendalami pemanfaatan Timer Counter berikut disajikan satu contoh sebagai bahan kajian. Dengan menggunakan pasilitas timer counter dapat dihasilkan clock pulsa 1 Khz di Port P0.0

Contoh Program 9-1

```
;-------------------------------------------------------------------------------------
; Program Interupsi Timer
; Membangkitkan Pulsa 2000 HZ : Perioda  T = 250x2  = 500 Mikrodetik
; Vektor Interupsi INT0 ORG 000BH , Vektor RESET ORG 00H
; Output pada P0.0
;-------------------------------------------------------------------------------------

Clock   Bit P0.0            ; Output Clock 2000 HZ di Port P0.0

        ORG 00H             ; Vektor Reset beralamat  0000H
        LJMP START          ; Lompat ke Main Program

        ORG 000BH           ; Vektor Interupsi INT0 beralamat 000BH
        CPL Clock           ; Membalik nilai P0.0 dai logika "1" ke logika "0" dan sebaliknya
        RETI                ; Kembali dari interupsi
```

```
;----------------------
;Main Program
;----------------------
START:
        MOV TMOD,#002H    ; Timer 0 bekerja pada Mode 2 atau mode 8 bit isi ulang
        MOV TH0,#-250D    ; Setiap 250 mikro detik Interupsi Aktif
        SETB ET0          ; Aktifkan interupsi Timer0
        SETB EA           ; Aktifkan sistim interupsi
        SETB TR0          ; Aktifkan TIMER 0

TanpaHenti:
        SJMP TanpaHenti

END
```

Pada main Program MOV TMOD,#002H membentuk fungsi pada Register TMOD sebagai berikut:

G	C/T*	M1	M0	G	C/T*	M1	M0
0	0	0	0	0	0	1	0

TIMER/COUNTER 0 berfungsi sebagai timer memanfaatkan frekuensi clock internal dari kristal karena C/T* = 0; Mode kerja Mode 2 atau mode 8 bit isi ulang karena M1 = 1 dan M0 = 0; Kemudian karena G = 0 maka aktifnya timer dikendalikan secara internal oleh bit TR0 pada register TCON (lihat Gambar 42).

Pada main program MOV TH0,#-250D atau 06H membentuk nilai konstanta cacahan sebanyak 250 kali pulsa frekuensi clock. Tanda negatif diberikan karena counter bekerja sebagai pencacah naik mulai dari 06H sampai FFH sebanyak 250 kali. Dengan frekuensi kristal 12 MHZ dibagi 12 maka dihasilkan clock cacahan 1 MHZ atau perioda 1 mikrosekon. Jika setiap 250 mikrodetik pada Port P0.0 terjadi logika “1” dan selama 250 mikrodetik pada Port P0.0 terjadi logika “0” berarti ada pulsa berulang dengan perioda pulsa 2 x 250 = 500 mikrodetik. Pulsa dengan perioda 500 mikrodetik sama dengan pulsa clock dengan frekuensi 2000 HZ yang terbangkit pada port P0.0. Pelajari jika menggunakan mode 2 untuk membangkitkan pulsa, berapa nilai frekuensi maksimum dan frekuensi minimum yang dapat dibangkitkan. Disamping nilai konstanta pada TH dan TL frekuensi kristal juga ikut menentukan hasil pewaktuan. Jika nilai konstanta pada register TH = FFH maka satu kali cacahan input akan mengakibatkan terjadi limpahan. Perioda pewaktuan akan menjadi 24/Frekuensi Kristal. Untuk kristal dengan frekuensi 12 MHZ maka nilai pewaktuan minimal yang dapat dihasilkan adalah 24/12 MHZ = 2 mikrodetik. Dalam dimensi frekuensi maksimal yang dapat dihasilkan adalah 500 KHZ. Untuk mendapatkan nilai pewaktuan yang lebih besar misalnya untuk basis waktu 1 detik, 10 mili detik diperlukan penyetingan nilai register timer dan pemilihan mode yang tepat.

MODUL 9
Memprogram Timer Counter

BAGIAN 2
PETUNJUK KERJA

A. PETUNJUK PRE-TEST

1. Kerjakan soal pre-test yang ada pada Modul 9 dengan mengisi tanda cek.
2. Isi dengan sebenarnya sesuai keadaan saudara
3. Jika saudara telah memiliki kompetensi seperti yang dinyatakan dalam pre test kerjakan soal-soal Post-Test
4. Jika saudara belum memiliki kompetensi seperti yang dinyatakan dalam pre test pelajari materi pada bagian satu dari Modul ini
5. Ujicoba contoh program 9-1 dan lakukan pengamatan hasil menggunakan CRO apakah betul frekuensi pulsa sama dengan 2000 HZ.

B. PETUNJUK POST-TEST

I. UMUM

Dalam tugas ini, pada akhirnya saudara akan memiliki kompetensi terkait dengan :

1. Pemahaman fasilitas Timer Counter pada mikrokontroler AT89S51
2. Pemrograman Timer Counter pada mikrokontroler AT89S51

II. KHUSUS

1. Kerjakan kasus-kasus program pada bagian post test sampai pada pengujian hasilnya pada down loader atau in system programming.

MODUL 9

Memprogram Timer Counter

BAGIAN 3
PRE-TEST

Subkompetensi	Pernyataan	Saya memiliki kompetensi ini	
		Tidak	Ya
9. Memprogram Timer Counter AT89S51	9.1 Apakah saudara memahami dasar-dasar timer counter mikrokontroler At89S51		
	9.2. Apakah saudara memahami pemrograman timer counter mikrokontroler At89S51		

BAGIAN 4
POST-TEST

1. Buatlah rancangan program yang menghasilkan clock dengan frekuensi 1 KHZ
2. Buatlah rancangan program yang menghasilkan pewaktuan 10 mili detik untuk keperluan pengembangan jam digital atau stopwatch digital dengan ketelitian 0,01 detik.
3. Buat program untuk mencacah masukan pada input T1

BAGIAN 5
KUNCI JAWABAN

Jawaban Soal no 1

```
; ---------------------------------------------------------------------------------------------
; Program 9-1 Pembangkit pulsa clock 1 KHZ di port 1.7
; Nama File Modul92.asm
; ---------------------------------------------------------------------------------------------

Clock bit P1.7                  ; Clock dihasilkan pada P1.7

        ORG 00H                 ; Vektor Reset
        LJMP Start              ; lompat ke sub-rutin start

        ORG  000BH              ; Vektor interupsi Timer 0 berada pada alamat 000BH
        CPL  Clock              ; membalik nilai P1.7
        MOV TH0,#0FEH           ; Nilai FE0CH sama dengan nilai -500
        MOV TL0,#00CH           ; membangun interupsi setiap 500 mikrodetik
        RETI                    ; Kembali setelah ISR selesai

Start:
        MOV  TMOD,#001H         ; Timer 0 bekerja pada mode 1 mode 16 bit
        MOV TH0,#0FEH           ;
        MOV TL0,#00CH           ; setiap 500 mikro-detik interupsi sekali
        SETB ET0                ; Aktipkan sistem interupsi Timer 0
        SETB EA                 ; Aktipkan sistem interupsi AT89C51
        SETB TR0                ; aktipkan Timer 0

TanpaHenti:
        SJMP TanpaHenti ; berputar tanpa henti di sini
    END
```

Jawaban Soal No. 2

```
;-------------------------------------------------------------------------
;Program Timer/Pewaktu 10 mili detik
;Vektor Interupsi INT0 ORG 0003H
;
;-------------------------------------------------------------------------


        ORG 0H              ;Vektor Reset
        LJMP START          ;Lompat ke Main Program

        ORG 000BH           ;Vektor Interupsi INT0 alamat 000BH
        INC R0              ; Isi Register  R0 ditambar 1 setiap 10 mili detik
        RETI


;--------------------------------------
;Main Program
;--------------------------------------
START:
        MOV     R0,#00H
        MOV     TMOD,#001H      ; Timer 0 bekerja pada mode 1 mode 16 bit
        MOV     TH0,#027H       ; 10 mili detik = 10.000 mikrodetik
        MOV     TL0,#010H       ; setiap 500 mikro-detik interupsi sekali
        SETB    ET0             ; Aktifkan interupsi Timer0
        SETB    EA              ; Aktifkan sistim interupsi
        SETB    TR0             ; Aktifkan TIMER 0

TanpaHenti:
        SJMP TanpaHenti

END
```

Jawaban Soal No. 3

```
;---------------------------------------------------------------------
;Program Pencacah input T1
;Vektor Interupsi INT0 ORG 0003H
;
;---------------------------------------------------------------------


        ORG 0H              ;Vektor Reset
        LJMP START          ;Lompat ke Main Program

        ORG 001BH           ;Vektor Interupsi INT1
        INC R0              ;Menambahkan 1 ke R0
        RETI


;-------------------------------------
;Main Program
;-------------------------------------
START:

        MOV     TMOD,#060H      ; Timer 1 bekerja pada mode 2 mode isi ulang
        MOV     TH0,#0FFH       ; 10 mili detik = 10.000 mikrodetik
        SETB    ET1             ; Aktifkan interupsi Timer0
        SETB    EA              ; Aktifkan sistim interupsi
        SETB    TR1             ; Aktifkan TIMER 0

TanpaHenti:
        SJMP TanpaHenti

END
```