MODUL 10

Memprogram Interupsi

BAGIAN 1
Memprogram Interupsi AT89S51

Tujuan Pembelajaran Umum:

1. Mahasiswa trampil memprogram interupsi

Tujuan Pembelajaran Khusus:

1. Mahasiswa memahami dasar-dasar interupsi Mikrokontroler AT89S51
2. Mahasiswa memahami pemrograman interupsi Mikrokontroler AT89S51

Sistem Interupsi AT89S51 (Naskah Asli oleh Budhy Sutanto)

Interupsi adalah selaan yang diberikan oleh alat luar atau bagian dalam dari sebuah mikrokontroler untuk menjalankan suatu rutin program tertentu. Pengetahuan mengenai interupsi sangat membantu mengatasi masalah pemrograman mikroprosesor/mikrokontroler dalam hal menangani banyak peralatan input/output. Pendalaman mengenai interupsi mencakup pendalaman teori dan pendalaman praktek melalui pengembangan program aplikasi konkrit.

Saat kaki **RESET** pada IC mikroprosesor/mikrokontroler menerima sinyal reset (pada AT89S51 sinyal tersebut berupa sinyal '**1**' sesaat, pada prosesor lain umumnya merupakan sinyal '**0**' sesaat), *Program Counter* diisi dengan sebuah nilai. Nilai tersebut dinamakan sebagai vektor reset (*reset vector*), merupakan nomor awal memori-program yang menampung program yang harus dijalankan.

Pembahasan di atas memberi gambaran bahwa proses reset merupakan peristiwa perangkat keras (sinyal reset diumpankan ke kaki Reset) yang dipakai untuk mengatur kerja dari perangkat lunak, yakni menentukan aliran program prosesor (mengisi Program Counter dengan vektor reset).Program yang dijalankan dengan cara reset, merupakan program utama bagi prosesor. Peristiwa perangkat keras yang dipakai untuk mengatur kerja dari perangkat lunak, tidak hanya terjadi pada proses reset, tapi terjadi pula dalam proses interupsi.

Dalam proses interupsi, terjadinya sesuatu pada perangkat keras tertentu dicatat dalam flip-flop khusus, flip-flop tersebut sering disebut sebagai 'petanda' (flag), catatan dalam petanda tersebut diatur sedemikian rupa sehingga bisa merupakan sinyal permintaan interupsi pada prosesor. Jika

permintaan interupsi ini dilayani prosesor, *Program Counter* akan diisi dengan sebuah nilai. Nilai tersebut dinamakan sebagai vektor interupsi (*interrupt vector*), yang merupakan nomor awal memori-program yang menampung program yang dipakai untuk melayani permintaan interupsi tersebut.

Program yang dijalankan dengan cara interupsi, dinamakan sebagai program layanan interupsi (**ISR** - *Interrupt Service Routine*). Saat prosesor menjalankan **ISR**, pekerjaan yang sedang dikerjakan pada program utama sementara ditinggalkan, selesai menjalankan **ISR** prosesor kembali menjalankan program utama, seperti yang digambarkan dalam Gambar 45.

Gambar 45 Bagan kerja prosesor melayani interupsi

Sebuah prosesor bisa mempunyai beberapa perangkat keras yang merupakan sumber sinyal permintaan interupsi, masing-masing sumber interupsi dilayani dengan **ISR** berlainan, dengan demikian prosesor mempunyai beberapa vektor interupsi untuk memilih **ISR** mana yang dipakai melayani permintaan interupsi dari berbagai sumber. Kadang kala sebuah vektor interupsi dipakai oleh lebih dari satu sumber interupsi yang sejenis, dalam hal semacam ini **ISR** bersangkutan harus menentukan sendiri sumber interupsi mana yang harus dilayani saat itu.

Jika pada saat yang sama terjadi lebih dari satu permintaan interupsi, prosesor akan melayani permintaan interupsi tersebut menurut perioritas yang sudah ditentukan, selesai melayani permintaan interupsi perioritas yang lebih tinggi, prosesor melayani permintaan interupsi berikutnya, baru setelah itu kembali mengerjakan program utama.

Saat prosesor sedang mengerjakan **ISR**, bisa jadi terjadi permintaan interupsi lain, jika permintaan interupsi yang datang belakangan ini mempunyai perioritas lebih tinggi, **ISR** yang sedang dikerjakan ditinggal dulu, prosesor melayani permintaan yang perioritas lebih tinggi, selesai melayani interupsi perioritas tinggi prosesor meneruskan **ISR** semula, baru setelah itu kembali mengerjakan program utama. Hal ini dikatakan sebagai interupsi bertingkat (*nested interrupt*), tapi tidak semua prosesor mempunyai kemampuan melayani interupsi secara ini.

MODUL 10
Memprogram Interupsi

Sumber interupsi AT89S51

Seperti terlihat dalam Gambar 46, AT89S51 mempunyai 6 sumber interupsi, yakni Interupsi External (*External Interrupt*) yang berasal dari kaki **INT0** dan **INT1**, Interupsi Timer (*Timer Interrupt*) yang berasal dari **Timer 0** maupun **Timer 1**, Interupsi Port Seri (*Serial Port Interrupt*) yang berasal dari bagian penerima dan bagian pengirim Port Seri.

Di samping itu AT89C52 mempunyai 2 sumber interupsi lain, yakni Interupsi Timer 2 bersumber dari **Timer 2** yang memang tidak ada pada AT89S51.

Bit **IE0** (atau bit **IE1**) dalam **TCON** merupakan petanda (flag) yang menandakan adanya permintaan Interupsi Eksternal. Ada 2 keadaan yang bisa meng-aktip-kan petanda ini, yang pertama karena level tegangan '**0**' pada kaki **INT0** (atau **INT1**), yang kedua karena terjadi transisi sinyal '**1**' menjadi '**0**' pada kaki **INT0** (atau **INT1**). Pilihan bentuk sinyal ini ditentukan lewat bit **IT0** (atau bit **IT1**) yang terdapat dalam register **TCON**.

- Kalau bit **IT0** (atau **IT1**) ='**0**' maka bit **IE0** (atau **IE1**) dalam **TCON** menjadi '**1**' saat kaki **INT0**='**0**'.
- Kalau bit **IT0** (atau **IT1**) ='**1**' maka bit **IE0** (atau **IE1**) dalam **TCON** menjadi '**1**' saat terjadi transisi sinyal '**1**' menjadi '**0**' pada kaki **INT0**.

Menjelang prosesor menjalankan **ISR** dari Interupsi Eksternal, bit **IE0** (atau bit **IE1**) dikembalikan menjadi '**0**', menandakan permintaan Interupsi Eksternal sudah dilayani. Namun jika permintaan Interupsi Ekternal terjadi karena level tegangan '**0**' pada kaki **IT0** (atau **IT1**), dan level tegangan pada kaki tersebut saat itu masih ='**0**' maka bit **IE0** (atau bit **IE1**) akan segera menjadi '**1**' lagi.

Bit **TF0** (atau bit **TF1**) dalam **TCON** merupakan petanda (flag) yang menandakan adanya permintaan Interupsi Timer, bit **TF0** (atau bit **TF1**) menjadi '**1**' pada saat terjadi limpahan pada pencacah biner Timer 0 (atau Timer 1). Menjelang prosesor menjalankan **ISR** dari Interupsi Timer, bit **TF0** (atau bit **TF1**) dikembalikan menjadi '**0**', menandakan permintaan Interupsi Timer sudah dilayani.

Interupsi port seri terjadi karena dua hal, yang pertama terjadi setelah port seri selesai mengirim data 1 byte, permintaan interupsi semacam ini ditandai dengan petanda (flag) **TI**='**1**'. Yang kedua terjadi saat port seri telah menerima data 1 byte secara lengkap, permintaan interupsi semacam ini ditandai dengan petanda (flag) **RI**='**1**'.

Petanda di atas tidak dikembalikan menjadi '**0**' menjelang prosesor menjalankan **ISR** dari Interupsi port seri, karena petanda tersebut masih diperlukan **ISR** untuk menentukan sumber interupsi

berasal dari **TI** atau **RI**. Agar port seri bisa dipakai kembali setelah mengirim atau menerima data, petanda-petanda tadi harus di-nol-kan lewat program.

Petanda permintaan interupsi (**IE0**, **TF0**, **IE1**, **TF1**, **RI** dan **TI**) semuanya bisa di-nol-kan atau di-satu-kan lewat instruksi, pengaruhnya sama persis kalau perubahan itu dilakukan oleh perangkat keras. Artinya permintaan interupsi bisa diajukan lewat pemrograman, misalnya permintaan interupsi eksternal **IT0** bisa diajukan dengan instruksi **SETB IE0**.

Mengaktipkan Interupsi

Semua sumber permintaan interupsi yang di bahas di atas, masing-masing bisa di-aktip-kan atau di-nonaktip-kan secara tersendiri lewat bit-bit yang ada dalam register **IE** (*Interrupt Enable Register*).

Bit **EX0** dan **EX1** untuk mengatur interupsi eksternal **INT0** dan **INT1**, bit **ET0** dan **ET1** untuk mengatur interupsi timer 0 dan timer 1, bit **ES** untuk mengatur interupsi port seri, seperti yang digambarkan dalam Gambar 46. Di samping itu ada pula bit **EA** yang bisa dipakai untuk mengatur semua sumber interupsi sekali gus.

Setelah reset, semua bit dalam register **IE** bernilai '**0**', artinya sistem interupsi dalam keadaan non-aktip. Untuk mengaktipkan salah satu sistem interupsi, bit pengatur interupsi bersangkutan diaktipkan dan juga **EA** yang mengatur semua sumber interupsi. Misalnya instruksi yang dipakai untuk mengaktipkan interupsi ekternal **INT0** adalah **SETB EX0** disusul dengan **SETB EA**.

Vektor Interupsi

Saat AT89S51 menanggapi permintaan interupsi, Program Counter diisi dengan sebuah nilai yang dinamakan sebagai vektor interupsi, yang merupakan nomor awal dari memori-program yang menampung **ISR** untuk melayani permintaan interupsi tersebut. Vektor interupsi itu dipakai untuk melaksanakan inststuksi **LCALL** yang diaktipkan secara perangkat keras.

Vektor interupsi untuk interupsi eksternal **INT0** adalah **0003H**, untuk interupsi timer 0 adalah **000BH**, untuk interupsi ekternal **INT1** adalah **0013H**, untuk interupsi timer 1 adalah **001BH** dan untuk interupsi port seri adalah **0023H**. Jarak vektor interupsi satu dengan lainnya sebesar 8, atau hanya tersedia 8 byte untuk setiap **ISR**. Jika sebuah **ISR** memang hanya pendek saja, tidak lebih dari 8 byte, maka **ISR** tersebut bisa langsung ditulis pada memori-program yang disediakan untuknya. **ISR** yang lebih panjang dari 8 byte ditulis ditempat lain, tapi pada memori-program yang ditunjuk oleh vektor interupsi diisikan instruksi **JUMP** ke arah **ISR** bersangkutan.

MODUL 10

Memprogram Interupsi

Tingkatan Perioritas

Masing-masing sumber interupsi bisa ditempatkan pada dua tingkatan perioritas yang berbeda. Pengaturan tingkatan perioritas isi dilakukan dengan bit-bit yang ada dalam register **IP** (Interrupt Priority).

Bit **PX0** dan **PX1** untuk mengatur tingkatan perioritas interupsi eksternal **INT0** dan **INT1**, bit **PT0** dan **PT1** untuk mengatur interupsi timer 0 dan timer 1, bit **PS** untuk mengatur interupsi port seri, seperti yang digambarkan dalam Gambar 46.

Setelah reset, semua bit dalam register **IP** bernilai '**0**', artinya semua sumber interupsi ditempatkan pada tingkatan tanpa perioritas. Masing-masing sumber interupsi bisa ditempatkan pada tingkatan perioritas utama dengan cara men-'satu'-kan bit pengaturnya. Misalnya interupsi timer 0 bisa ditempatkan pada tingkatan perioritas utama dengan instruksi **SETB PT1**.

Sebuah ISR untuk interupsi tanpa perioritas bisa diinterupsi oleh sumber interupsi yang berada dalam tingkatan perioritas utama. Tapi interupsi yang berada pada tingkatan perioritas yang sama, tidak dapat saling meng-interupsi.

Jika 2 permintaan interupsi terjadi pada saat yang bersamaan, sedangkan kedua interupsi tersebut terletak pada tingkatan perioritas yang berlainan, maka interupsi yang berada pada tingkatan perioritas utama akan dilayani terlebih dulu, setelah itu baru melayani interupsi pada tingkatan tanpa perioritas.

Jika kedua permintaan tersebut bertempat pada tingkatan perioritas yang sama, perioritas akan ditentukan dengan urutan sebagai berikut : interupsi eksternal **INT0**, interupsi timer 0, interupsi ekternal **INT1**, interupsi timer 1 dan terakhir adalah interupsi port seri.

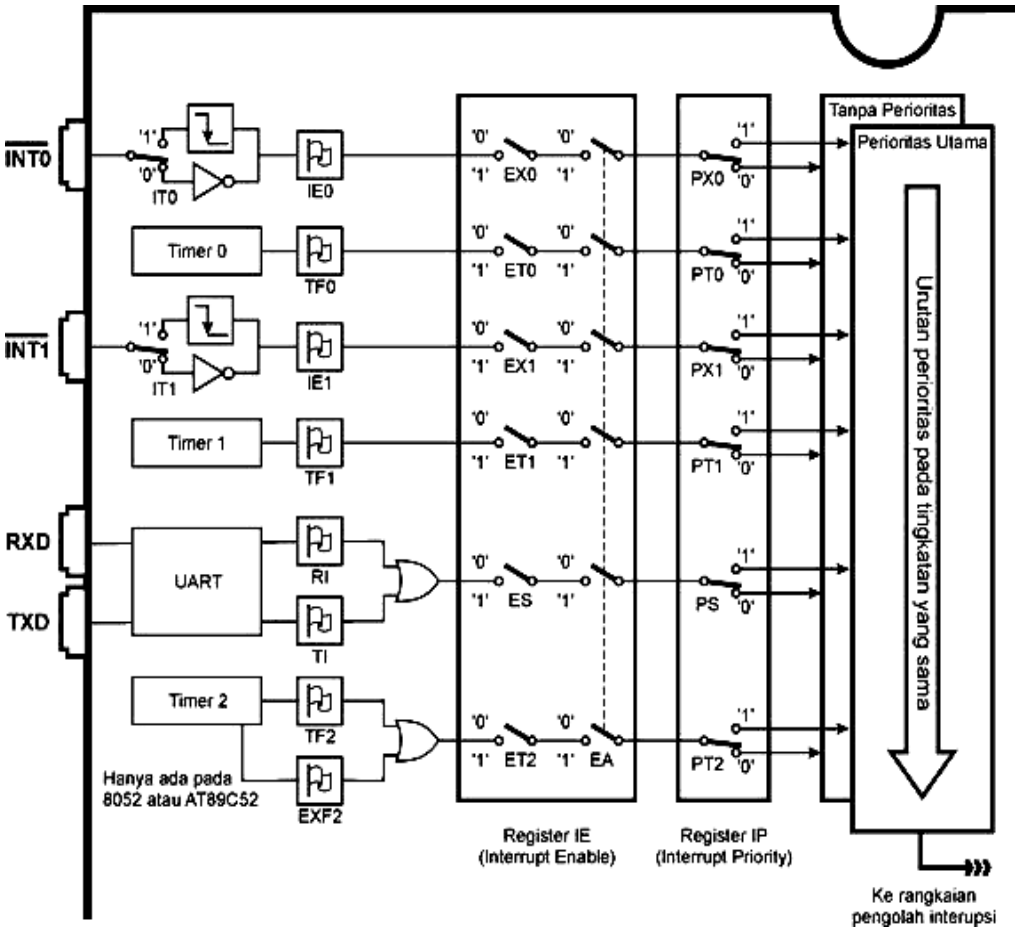
Bagan Lengkap Sistem Interupsi AT89S51

Meskipun sistem interupsi AT89S51 termasuk sederhana dibandingkan dengan sistem interupsi MC68HC11 buatan Motorola, tapi karena menyangkut 5 sumber interupsi yang masing-masing harus diatur secara tersendiri, tidak mudah untuk mengingat semua masalah tersebut, terutama pada saat membuat program sering dirasakan sangat merepotkan membolak-balik buku untuk mengatur masing-masing sumber interupsi tersebut.

Gambar 46 menggambarkan sistem interupsi AT89S51 selangkapnya, berikut dengan masing-masing bit dalam register-register **SFR** (*Special Function Register*) yang dipakai untuk mengatur masing-masing sumber interupsi.

Saklar yang digambarkan dalam Gambar 2 mewakili bit dalam register yang harus diatur untuk mengendalikan sumber interupsi, kotak bergambar bendera kecil merupakan flag (petanda) dalam register yang mencatat adanya permintaan interupsi dari masing-masing sumber interupsi. Kedudukan saklar dalam gambar tersebut menggambarkan kedudukan awal setelah AT89S51 di-reset.

Gambar ini sangat membantu saat penulisan program menyangkut interupsi AT89S51.



Gambar 46 Bagan Lengkap Sistem Interupsi AT89S51

MODUL 10

Memprogram Interupsi

Pedoman pembuatan ISR

Layanan interupsi oleh AT89S51 dilakukan dengan menjalankan program setara dengan **LCALL** mulai dari memori-program yang ditunjuk oleh vektor interupsi, sampai AT89S51 menjumpai instruksi **RETI** (*Return from Interrupt*). Intstruksi **RETI** memberitahu AT89S51 bahwa **ISR** sudah selesai dikerjakan, kemudian mengambil kembali isi *Program Counter* yang sebelumnya disimpan ke dalam *Stack*. Dengan demikian AT89S51 akan melanjutkan kembali pekerjaan di program utama yang ditinggal untuk melayani interupsi.

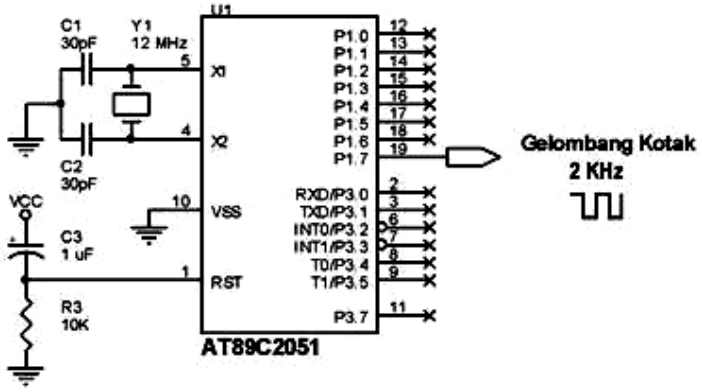
Instruksi **RET** memang bisa dipakai untuk meninggalkan **ISR** dan melanjutkan kerja program utama. Tapi **RET** tidak bisa menghentikan proses interupsi, akibatnya sebelum menjumpai instruksi **RETI**, AT89S51 tidak akan melayani permintaan interupsi yang lain.

Untuk menjelaskan pembuatan program interupsi, diperlukan program lengkap yang bisa berfungsi penuh, tapi ukuran program tersebut harus kecil sehingga bisa memberi gambaran jelas tentang interupsi. Untuk keperluan penjelasan ini, dipakai rangkaian AT89C2051 pada Gambar 47. Dengan program yang akan dibahas rangkaian ini bisa membangkitkan gelombang kotak pada kaki **P1.7** (kaki nomor 19), dalam contoh kedua frekuensi gelombang kotak yang dibangkitkan bisa ditentukan dengan mengatur lebar pulsa

Pembangkit gelombang kotak 2000 Hz

Pembangkit gelombang kotak 2000 Hz ini merupakan contoh pembuatan sistem interupsi yang sangat sederhana, hanya menggunakan rangkaian baku AT89C2051 seperti terlihat dalam Gambar 3. Yakni hanya memakai **U1** AT89C2051 berikut dengan rangkaian osilator kristal yang dibentuk dengan **C1**, **C2** dan **Y1** (kristal 12 MHz), serta rangkaian reset yang terdiri dari **C3** dan **R3**. Sinyal gelombang kotak dibangkitkan pada kaki **P1.7** (kaki nomor 19).

MODUL 10
Memprogram Interupsi



Gambar 47 Pembentuk Gelombang Kotak

Program 1 yang hanya 18 baris merupakan program lengkap pembangkit gelombang kotak 2000 Hz ini. Setelah program di-assembly, program-objek hasil kerja assembler diisikan ke Flash PEROM AT89C2051 dengan menggunakan AT89C2051 Flash PEROM Programmer .

Program ini memakai *interupsi timer 0* yang diatur sedemikian rupa, sehingga *timer 0* menginterupsi AT89C2051 setiap 250 mikro-detik. Saat terjadi interupsi, nilai pada **P1.7** dibalik, jika **P1.7** bernilai '**0**' akan dibalik menjadi '**1**' dan sebaliknya jika bernilai '**1**' dibalik menjadi '**0**'. Dengan demikian pada kaki **P1.7** akan timbul gelombang kotak yang periodenya 2x250 = 500 mikro-detik (atau frekuensinya sama dengan 1/500 mikro-detik = 2000 Hz).

Program 1 : Pembangkit gelombang kotak 2000 Hz

```
01: Clock bit P1.7          ; Clock dihasilkan pada P1.7
02: ;
03:     ORG  0000H          ; Vektor Reset
04:     LJMP Start
05: ;
06:     ORG 000BH           ; Vektor interupsi Timer 0
07:     CPL  Clock          ; membalik nilai P1.7
08:     RETI
09: ;
10: Start:
11:     MOV  TMOD,#02H      ; Timer 0 bekerja pada mode 2 Mode Isi Ulang
12:     MOV  TH0,#-250      ; setiap 250 mikro-detik interupsi sekali
13:     SETB ET0            ; Aktipkan sistem interupsi Timer 0
14:     SETB EA             ; Aktipkan sistem interupsi AT89S51
15:     SETB TR0            ; aktipkan Timer 0
16: TanpaHenti:
17:     SJMP TanpaHenti ; berputar tanpa henti di sini
18:     END
```

Baris 01 pada program menentukan yang dinamakan sebagai **Clock** (yakni tempat gelombang kotak dibangkitkan) diletakkan pada **P1.7**, cara semacam ini merupakan cara umum dan praktis dipakai untuk memberi nama lain pada kaki-kaki port parallel AT89S51, agar program yang ditulis lebih enak dibaca, seperti terlihat pada baris 07.

Baris 03 menentukan agar program yang dibangkitkan assembler diletakkan pada memori-program nomor **0000H**, yakni lokasi pada memori-program yang ditunjuk oleh vektor reset.

Instruksi pada memori-program nomor **0000H** adalah **LJMP Start** (baris 04), yakni mengalihkan program yang harus dikerjakan pada saat reset ke memori-program yang ditandai dengan label **Start** (baris 10). Baris 06 menentukan memori-program berikutnya yang dipakai adalah nomor **000BH**, yakni lokasi pada memori-program yang ditunjuk oleh vektor interupsi Timer *0*.

Program Layanan Interupsi (**ISR** – *Interrupt Service Routine*) untuk interupsi Timer 0 ini sangat sederhana, hanya sekedar membalik nilai **Clock** (**P1.7)** pada baris 07 dan kemudian mengakhiri **ISR** dengan istruksi **RETI** pada baris 08.

Baris 10 dan berikutnya merupakan program utama. Baris 11 sampai 15 mempersiapkan kerja dari AT89C2051, setelah itu AT89C2051 hanya berputar-putar terus pada baris 16 dan 17, tanpa mengerjakan hal yang lain.

Selama AT89C2051 berputar-putar pada baris 16 dan 17, setiap 250 mikro-detik sekali Timer 0 menginterupsi AT89C2051, sehingga AT89C2051 menjalankan **ISR** pada baris 07 dan 08 untuk membangkitkan gelombang kotak, dan kembali berputar lagi pada baris 16 dan 17.

Baris 16 dan 17 yang berputar tanpa maksud tersebut, memang dibuat agar contoh program ini sederhana, untuk mempertegas pembicaraan tentang mekanisme interupsi. Dalam sistem yang sesungguhnya, baris 16 dan 17 bisa diganti dengan program apa saja yang berdaya-guna.

Baris 11 mengatur kerja dari Timer 0, instruksi **MOV TMODE,#02H** mengatur Timer 0 bekerja dalam Mode 2, yakni Pencacah Biner 8 bit dengan isi ulang. Dalam hal ini Timer 0 hanya dibentuk dengan pencacah **TL0** yang 8 bit, setiap kali pencacah tersebut melimpah maka **TL0** akan diisi ulang dengan nilai yang disimpan dalam **TH0**.

Baris 12 mengisi **TH0** dengan nilai –250, nilai ini diisikan ke **TL0** setiap kali pencacah **TL0** melimpah. Pencacah **TL0** merupakan pancacah naik (count up counter), karena nilai awal dari **TL0** adalah –250 maka setelah menerima 250 pulsa sinyal denyut (clock) pencacah **TL0** akan melimpah. Dengan kristal 12 MHz, pencacah **TL0** mencacah sekali setiap 1 mikro-detik, dengan demikian Timer 0 akan menginterupsi AT89C2051 sekali setiap 250 mikro-detik.

Sesuai dengan Gambar 2, baris 13 (**SETB ET0**) mengaktipkan permintaan interupsi timer 0 dan baris 14 (**SETB EA**) mengaktipkan sistem interupsi AT89C2051 secara keseluruhan. Setelah

kedua instruksi ini, Timer 0 akan meng-interupsi AT89C2051 setiap kali flag (petanda) **TF0** menjadi '**1**' karena pencacah **TL0** melimpah. Baris 15 (**SETB TR0**) mengaktipkan Timer 0, yakni menghubungkan sumber sinyal denyut (clock) ke Timer 0, setelah instruksi ini Timer 0 akan mulai bekerja dan 250 mikro-detik kemudian meng-interupsi AT89C2051.

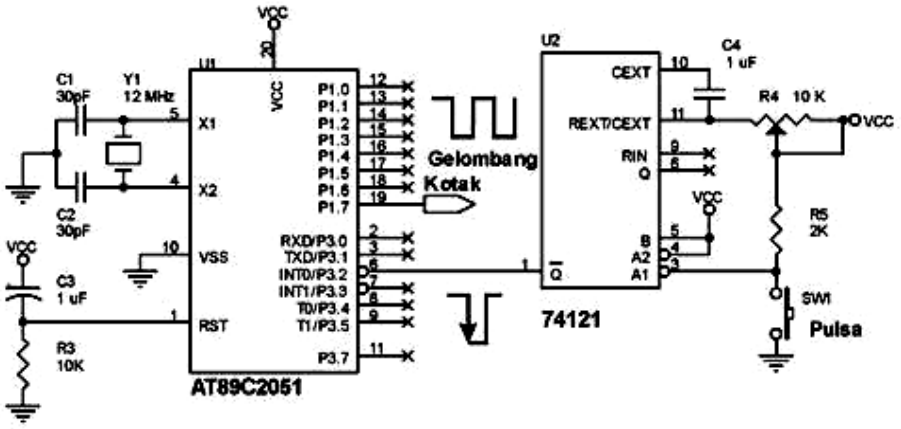
Gelombang kotak dengan frekuensi variabel

Dengan menambahkan **U2** 74121 (monostable multivibrator) pada rangkaian Gambar 47, bisa dibuat pembangkit gelombang kotak dengan frekuensi variabel, seperti terlihat dalam Gambar 48.

Pembangkit gelombang kotak dengan frekuensi variabel ini mendaya-gunakan 2 sistem interupsi AT89S51, masing-masing adalah interupsi Timer 0 seperti contoh di atas, ditambah dengan interupsi Ekternal 0.

Setiap kali tombol SW1 ditekan, Q1 dari **U2** 74121 akan mengeluarkan pulsa nol sesaat, lebar pulsa tersebut ditentukan oleh nilai kapasitor C4 dan resistor R4 yang nilainya bisa di-rubah. Pulsa nol dari **U2** 74121 dipakai untuk menginterupsi **U1** AT89C2051 lewat **INT0** (**P3.2**, kaki nomor 6).

Permintaan interupsi terjadi pada saat kaki **INT0** berubah dari '1' menjadi '0'. Program Layanan Interupsi (**ISR** – *Interrupt Service Routine*) untuk Interupsi Eksternal 0 bertugas mengukur periode waktu kal



Gambar 48 Pembentuk Gelombang Kotak

Program 2 : Gelombang kotak dengan frekuensi variabel

```
01: Clock bit P1.7
02: Pulse bit P3.2
03: ;
04: ; Vektor Reset
05:       ORG 0000H
06:       LJMP Start
07: ;
08: ; Vektor interupsi Eksternal 0
09:       ORG  0003H
10:       LJMP MengukurPulsa
11: ;
12: ; Vektor interupsi Timer 0
13:       ORG  000BH
14:       LJMP MembuatClock
15: ;
16: Start:
17:       MOV  TMODE,#01H
18:       SETB IT0
19:       SETB EX0
20:       SETB EA
21: TanpaHenti:
22:       SJMP TanpaHenti
23: ;
24: ; ISR interupsi eksternal
25: MengukurPulsa:
26:       MOV  TH0,#0
27:       MOV  TL0,#1
28:       SETB TR0
29:       JNB  Pulse,$
30:       CLR  TR0
31:       XRL  TH0,#0FFH
32:       XRL  TL0,#0FFH
33:       MOV  A,#1
34:       ADD  A,TL0
35:       MOV  R2,A
36:       MOV  A,#0
37:       ADDC A,TH0
38:       MOV  R3,A
39:       SETB ET0
40:       SETB TF0
41:       RETI
42: ;
43: ; ISR interupsi timer 0
44: MembuatClock:
45:       CLR  TR0
46:       MOV  TH0,R3
47:       MOV  TL0,R2
48:       CPL  Clock
49:       SETB TR0
50:       RETI
51: ;
52:       END
```

Program utama terletak pada baris 16 sampai 22. Baris 17 mengatur Timer 0 bekerja pada Mode 1 - Pencacah Biner 16 bit, yakni menghubungkan **TL0** dan **TH0** menjadi satu. Baris 18 menentukan yang dianggap sebagai sinyal interupsi adalan perubahan tegangan pada kaki **INT0** dari '**1**' menjadi '**0**'. Baris 19 mengaktipkan sistem interupsi eksternal 0 dan baris 20 mengaktipkan sistem interupsi AT89C2051 secara keseluruhan. Tapi saat ini interupsi Timer 0 belum diaktipkan.

Seperti halnya contoh program pertama, AT89C2051 akan berputar-putar pada baris 21 dan 22.Saat tegangan pada kaki **INT0** berubah dari '**1**' menjadi '**0**', bit **IE0** menjadi '**1**' yang merupakan sinyal permintaan interupsi eksternal *0*, hal ini mengakibatkan AT89C2051 mengerjakan rutin **MengukurPulsa** yang merupakan **ISR** interupsi eksternal 0 (baris 25 sampai 41).

Dalam **ISR** ini, lebar pulsa pada kaki **INT0** dengan Timer 0. Baris 26 dan 27 membuat **TH0** dan **TL0** bernilai 0. Baris 28 membuat Timer 0 mulai mencacah, dan baris 29 menunggu tegangan pada kaki **INT0** kembali menjadi '**1**', dan selama itu Timer 0 terus mencacah. Baris 30 menghentikan Timer 0, nilai yang tersimpan dalam **TH0** dan **TL0** saat itu merupakan panjang pulsa dalam satuan mikro-detik.

Lebar pulsa yang didapat dari pengukuran ini akan dipakai untuk menentukan frekuensi gelombang kotak yang dibangkitkan. Lebar pulsa ini dinegatipkan dan disimpan ke dalam **R2** dan **R3** pada baris 31 sampai 38.

Baris 39 mengaktipkan permintaan interupsi dari Timer 0, dan baris 40 (**SETB TF0**) membangkitkan permintaan interupsi timer 0 yang pertama kali lewat program.

Rutin **MembuatClock** (baris 44 sampai 50) merupakan **ISR** untuk interupsi Timer 0. Dalam rutin ini mula-mula sumber sinyal denyut (clock) untuk Timer 0 dihentikan (baris 45), setelah itu **TH0** dan **TL0** diisi ulang dengan nilai hasil pengukuran pulsa yang sudah disimpan dalam **R2** dan **R2** (baris 46 dan 47). Level tegangan pada **P1.7** dibalik pada baris 48, setelah itu sumber sinyal denyut Timer 0 diaktipkan kembali pada baris 49.

MODUL 10
Memprogram Interupsi

BAGIAN 2
PETUNJUK KERJA

A. PETUNJUK PRE-TEST

1. Kerjakan soal pre-test yang ada pada Modul 10 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

B. PETUNJUK POST-TEST

I. UMUM

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

1. Membuat program pengendalian interupsi

II. KHUSUS

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

MODUL 10

Memprogram Interupsi

BAGIAN 3
PRE-TEST

Subkompetensi	Pernyataan	Saya memiliki kompetensi ini	
		Tidak	Ya
10. Memprogram Interupsi	10.1 Apakah saudara memahami dasar-dasar interupsi		
	10.2. Apakah saudara memahami pemrograman interupsi		

BAGIAN 4
POST-TEST

1. Buatlah program yang bekerja jika ada interupsi dari INT 0 Lampu LED pada Port 1 menyajikan nyala belah tengah, jika ada interupsi dari INT 1 Lampu LED pada Port 1 menyajikan nyala berkedip. Dalam keadaan tanpa interupsi Lampu LED nyala berputar satu digit.

BAGIAN 5
KUNCI JAWABAN

Jawaban Soal no 1

```
;-------------------------------------------------------------------
;Program Interupsi Port Output
;Interupsi lewat INT0 P3.2
;Vektor Interupsi INT0 ORG 0003H
;Main Program Geser LED 1 bit
;Interupsi Ke Program Belah Tengah
;Port 0 aktif LOW
;-------------------------------------------------------------------

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

        ORG 0003H       ;Vektor Interupsi INT0
        LJMP Belah      ;Lompat ke Sub Rutin Belah

        ORG 0013H       ;Vektor Interupsi INT1
        LJMP Kedip      ;Lompat ke Sub Rutin Kedip

;----------------------
;Main Program
;----------------------
START:
        SETB EX0
        SETB EA
        SETB IT0
        SETB EX1
        SETB EA
        SETB IT1
        MOV R3,#0FEH
PUTAR:
        MOV P1,R3
        ACALL DELAY
        MOV A,R3
        RL A
        MOV R3,A
        SJMP PUTAR

;-------------------------------------------------------------------
;Interrupt Service Routine Belah Aktif oleh Interupsi INT 0
;-------------------------------------------------------------------
Belah:
Mulai:  MOV R2,#1AH
        MOV DPTR,#Angka
```

```
Geser:
        CLR A
        MOVC A,@A+DPTR
        INC DPTR
        MOV P1,A
        ACALL DELAY
        DJNZ R2,Geser
        RETI
;-------------------------------------------------------------------------------
;Interrupt Service Routine Kedip  diaktifkan oleh Interupsi INT1
;-------------------------------------------------------------------------------
Kedip:
        MOV P1,#00H
        ACALL DELAY
        MOV P1,#0FFH
        ACALL DELAY
        MOV P1,#00H
        ACALL DELAY
        MOV P1,#0FFH
        ACALL DELAY
        RETI

;----------------------
;Tunda Waktu
;----------------------
DELAY :         MOV R0,#003H

Delay2: MOV R4,#0FFH
Delay3: DJNZ R4,Delay3
        DJNZ R1,Delay3
        DJNZ R0,DELAY2
        RET

Angka: DB 0FFH,0E7H,0DBH,0BDH,07EH,0BDH,0DBH,0E7H
        DB 0FFH,0FFH,000H,000H,0FFH,0FFH,00FH,00FH
        DB 0F0H,0F0H,0FFH,0FFH,000H,000H,00FH,00FH
        DB 0FFH,0FFH

END
```