Akhmad Hendriawan

Panduan Praktikum Mikrokontroler

Revisi 3

october 2006

Jurusan teknik elektronika

Politeknik Elektronika Negeri Surabaya

ITS

2006

Lisensi Dokumen

Daftar isi

Bab 1

Pemrograman dasar MCS51

I. Tujuan

Setelah Melakukan praktikum ini yang anda peroleh adalah:

- Dapat menggunakan simulator UMPS sebagai pendukung dalam merancang program.
- Dapat menggunakan resource hardware MCS-51 seperti accumulator, register, port serta register.
- Dapat membuat program sederhana yang di di implementasikan di board MCS-51

II. Pendahuluan

Pada praktikum ini, anda akan mempelajari cara mengembangkan sebuah sistem pada IC Mikrokontroller MCS-51(8951) buatan atmel menggunakan software UMPS dan program downloader buatan innovative electronics. Dimulai dengan menuliskan desain menggunakan bahasa assembler kemudian melakukan simulasi program untuk mengetahui algoritma program sudah benar, Apabila proses telah dilalui tanpa kesalahan (error), file hex yang dihasilkan dikirim pada board MCS-51 melalui kabel serial. Pengiriman file ini digunakan sofware downloader.

III. Gambaran Desain

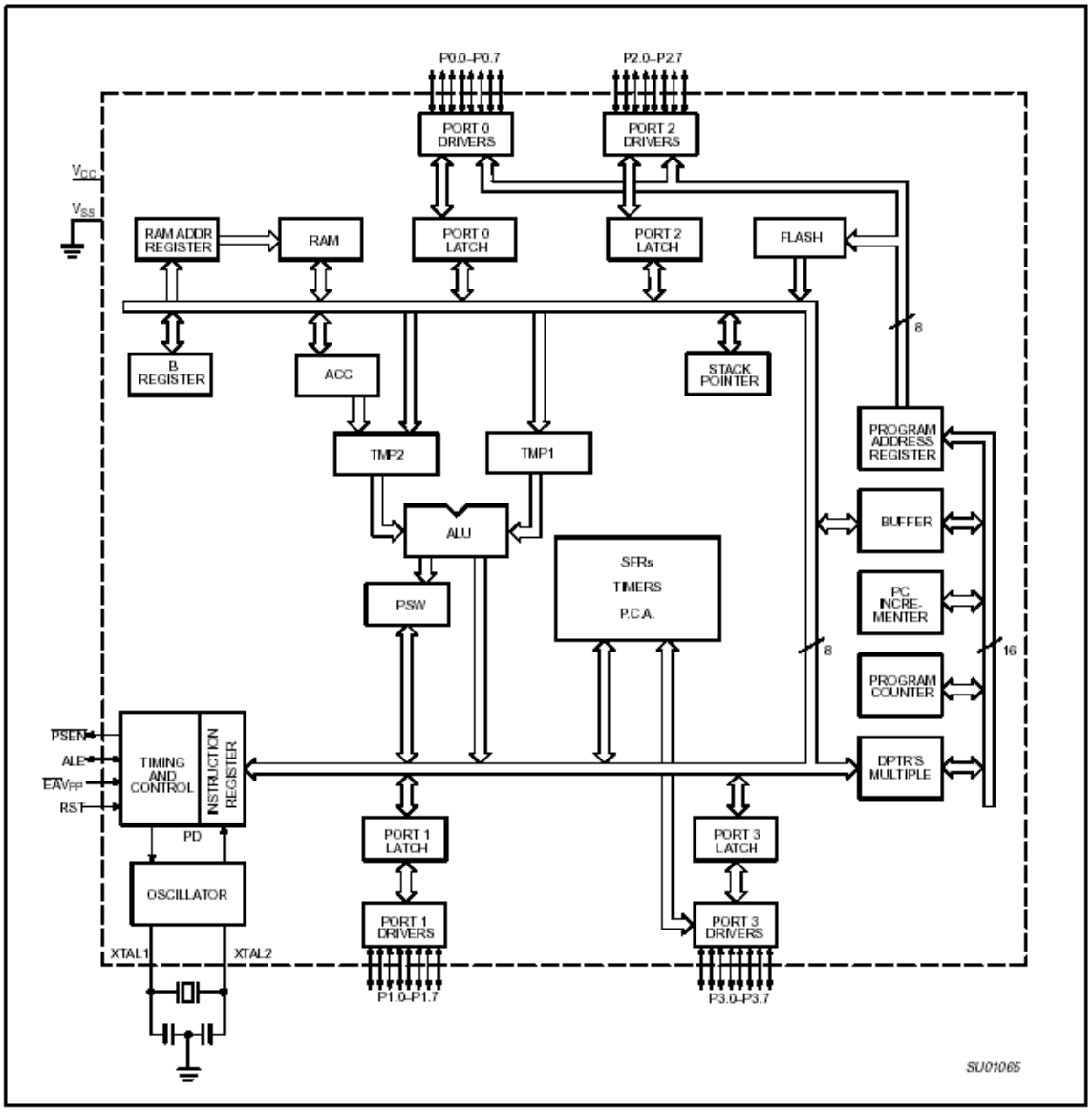
Anda membuat project baru menggunakan mikrokontroller MCS-51 (8051) dengan memanfaatkan fungsi output. Pembuatan project dimulai dengan menulis program dengan bantuan editor text, kemudian melakukan perakitan(compile) program, mensimulasikan program

yang anda buat dan melacak kesalahan program. Apabila anda berhasil membuat project tanpa kesalahan maka langkah selanjutnya adalah menyiapkan kabel penghubung serial antara board dt51 dan komputer tempat anda bekerja. Setelah selesai maka file dengan format intel hexa (mempunyai ekstensi .hex) didownload ke dt51board dengan bantuan program dt51win.
Pada percobaan pertama anda belajar bagaimana membuat program sederhana, mensimulasikannya dengan bantuan umps dan anda belajar bagaimana menulis program ke board development dt51. percobaan kedua anda belajar bagaimana membuat prosedur dalam bahasa assembler 8051. Percobaan ketiga anda dituntun langkah demi langkah untuk membuat project sederhana dengan 8051. Pada percobaan ini anda membuat running led project, yaitu suatu project yang apabila dijalankan membuat nyala led bergerak-gerak seperti halnya gerakan pendulum

IV. Dasar Teori

UMPS adalah simulator mikrokontroller universal yang berjalan pada sistem operasi windows. UMPS dapat mensimulasikan beberapa resource komponen external seperti Panel LCD,Real time I2C clock dan 4 bush button.UMPS mempunyai kemampuan dalam mensimulasikan beberapa macam mikrokontroller dan melakukan proses debugging. Dalam software ini sudah termasuk editor dan assembler/dissambler, tetapi external assembler ataupun compiler yang lain dapat digunakan. UMPS mengijinkan anda untuk melihat source dan variabel

Mikrokontroller MCS-51 adalah mikrokontroller 8 bit (lebar data bus) dengan arsitektur yang fleksibel, banyak keistimewaan dan efisiensi dalam penggunaan bit (yang sangat perlu bagi mikrokontroller).



gambar 1.1 arsitektur internal 8051

Pada gambar 1-1 merupakan arsitektur internal dari 8051. pada gambar ditunjukkan keistimewaan mikrokontroller 8051. keistimewaan dari 8051 adalah

- Internal arsitektur 8051 terdiri dari program memori dan data memori. Program memori digunakan untuk menyimpan code program yang telah di kompile menjadi biner.
- Mendukung interfacing dengan menggunakan port input dan output sebanyak 4 port yang dinamakan port 0, port 1, port 2 dan port 3 dengan besar data yang ditangani pada tiap-tiap port sebanyak delapan bit. Jika jumlah port yang digunakan masih tidak memenuhi kebutuhan port dapat ditambahkan dengan menggunakan device lain seperti PPI 8255 yang dihubungakan dengan prosessor 8051.
- Jumlah input output untuk keperluan interface sebanyak 32 bit yang terdiri dari 4 port.

Masing –masing port dari 8051 bersifat tidak hanya bidectional perbyte tapi juga bidirectional perbit. Sehingga memudahkan untuk pengontrolan dengan banyak input /output dari device lain yang berbeda.

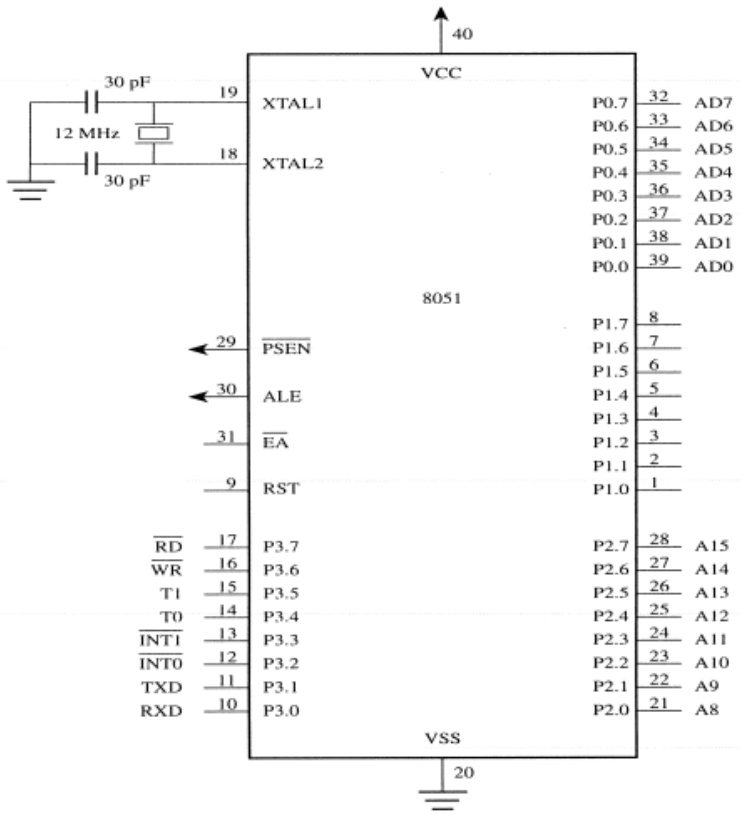
- Terdapat dua timer yaitu timer 1 dan timer 0 yang dapat digunakan secara independen. Input dari timer ini bisa berasal dari frek osilator (berfungsi sebagai timer) atau berasal luar.
- Terdapat dua counter yang yaitu counter T1 dan counter T0 dari port 3 yang dapat diprogram secara independen. Rangkaian banyak digunakan pada aplikasi yang banyak melakukan perhitungan seperti menghitung jumlah orang yang hadir.
- Setiap counter pada 8051 ditangani oleh register 16bit yang diperoleh dari pasangan register 8 bit sebanyak dua buah yaitu register TLx dan register THx, x bisa 0 atau 1 tergantung register yang digunakan.
- Ukuran internal RAM yang dapat digunakan untuk penyimpanan data logging sebesar 128 byte dimulai dari alamat 0 sampai 7F.
- terdapat port serial disediakan untuk melakukan komunikasi secara serial dengan device lain seperti komputer pribadi (PC). Pada komunikasi serial ini dapat digunakan pilihan mode yang digunakan untuk berkomunikasi dengan device lain yaitu sebanyak empat mode dari mode 0, mode 1, mode 2 dan mode 3. mode 0 digunakan untuk komunikasi sinkron, mode 1 digunakan untuk komunikasi standar uart dengan data 8 bit 1 stop bit dan non parity, mode 2 dan mokde 3 digunakan untuk berkomunikasi dengan multiprossor. Komunikasi dengan menggunakan 8051 ini dapat menggunakan komunikasi sinkron maupun komunikasi asinkron tergantung mode komunikasi yang di pilih.
- Interupsi eksternal disediakan sebanyak dua buah pada port 3 yang dapat dipakai untuk menangani device yang memerlukan penanganan segera. Dengan adanya port interrupsi ini program tidak harus menunggu(pooling) sampai device membutuhkan proses dari 8051.
- pada 8051 menyediakan memori program sebanyak 4K byte. Kebanyakan proses pengontrolan membutuhkan memori kurang dari 2K. jika kebutuhan memori sebesar ini masih kurang, dapat ditambahkan rom eksteranal untuk keperluan penyimpanan program sebanyak 64 K byte.
- Berbagai macam mode pengalamatan ke memori didukung oleh prosessor 8051. seperti mode direct addressing, mode inderect addressing, mode immediate constant, indexed

addressing, regiter specific instructions, register instruction.

- Interface ke device lain mudah dilakukan. Untuk interface digunakan address bus yaitu p0 sebagai lower address bus (A0-A7) dan p2 sebagai address bus untuk upper address bus (A8-A15). Control bus untuk interface menggunakan pin RD (read), pin WR(write), pin ALE (Adress Latch Enable), pin EA (external access) dan PSEN(Program Strobe Enable) untuk berhubungan dengan ROM external (gambar 1-1).

Arstitektur 8051 terdiri dari :

- Delapan bit CPU dengan register A (accumulator) dan register B
- 16 bit program counter (PC) dan16 bit data pointer (DPTR
- 8 bit program status word (PSW)
- 8 bit stack pointer (SP)
- Internal RAM sebanyak 128 byte terdiri dari
  - 4 bank register, dengan setiap bank terdiri dari 8 register
  - 16 byte memori yang dapat dialamati perbit
  - 8 byte data memori untuk keperluan umum
- 32 ping input/output yang tiap-tiap port terdiri dari 8 bit, port P0-P3
- 16 bit timer/counter; T0 dan T1
- Full duplex serial data receiver/transmitter menggunakan register SBUF



Gambar 1-2 susunan pin 8051

- Sebanyak 5 register control yaitu TCON, TMOD, SCON, PCON, IP dan IE
- Dua eksternal interrupt dan tiga sumber interrupt internal
- Osilator dan rangkaian clock .

Hampir semua register dalam prosessor 8051 mempunyai fungsi khusus. Setiap register kecuali program counter mempunyai internal address sebesar 1 byte. Beberapa register pada 8051 datanya bisa diakses per byte maupun perbit seperti register a (accumulator).

Ketika dialamati per bit menjadi acc.0 , acc.1 sampai acc.7. Instruksi keregister yang bersangkutan dapat menggunakan simbolic name, address atau kedua-duanya. Pada gambar 1-2 terlihat bahwa susunan pin dari 8051 sebanyak 40 pin DIP. Perlu diketahui bahwa tidak semua pin dapat digunakan untuk lebih dari satu fungsi. Pemrograman instruksi atau hubungan pin secara fisik menentukan fungsi dari pin tersebut. Sebagai contoh jika port 3 bit 0 (atau p3.0) mungkin digunakan untuk fungsi umum keperluan input output, atau sebagai pin RXD untuk keperluan komunikasi data secara serial yang akan diterima oleh register SBUF. Disainer memutuskan apakah pin p3.0 digunakan untuk serial atau digunakan untuk fungsi yang lain.

V. Peralatan yang dibutuhkan

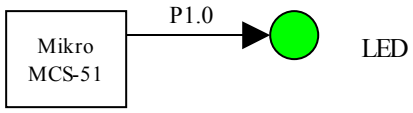
- Editor ascii (UMPS)
- Simulator MCS-51 (UMPS)
- Cross Assembler MCS-51 (UMPS)
- Program downloader(dt51win)
- IBM PC kompatibel
- Development board MCS-51
- Kabel serial untuk download program

VI. Prosedur Praktikum

Percobaan 1- LED on

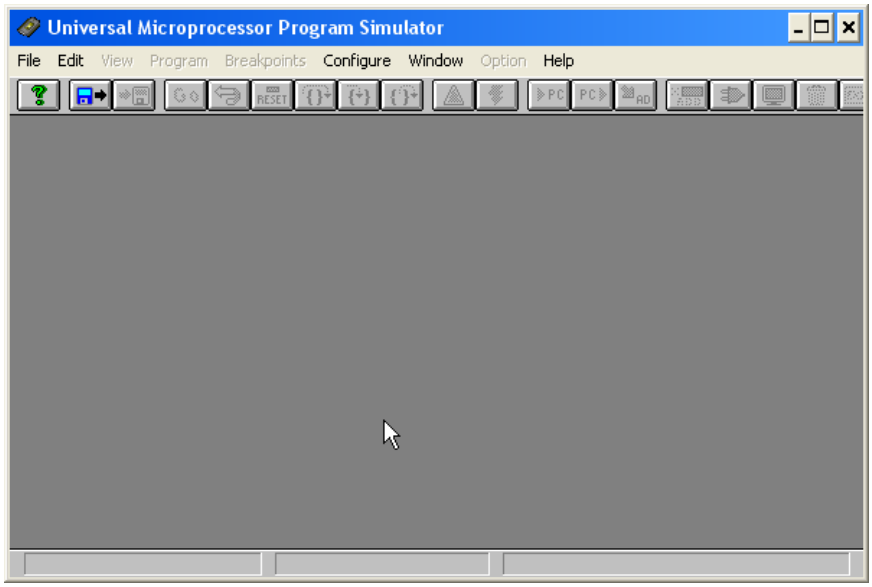
Memulai desain dengan UMPS

Pada percobaan pertama anda belajar bagaimana membuat program sederhana dan mensimulasikannya dengan bantuan umps Pada percobaan ini dibuat disain mikrokontroller untuk mengontrol nyala led pada p1.0 yang dibuat sebagai berikut:



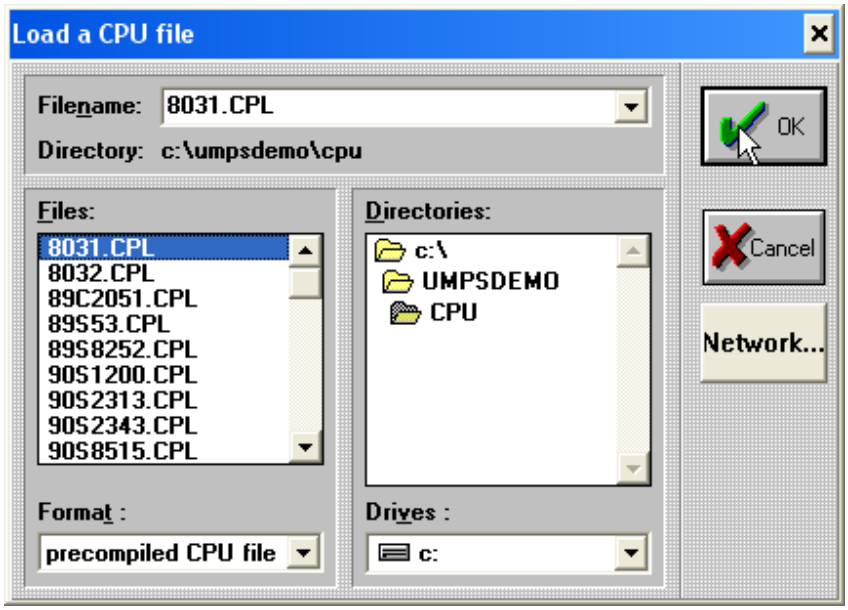
Gambar 1-3 Interfacing mikro ke LED

1. Jalankan program umps.exe



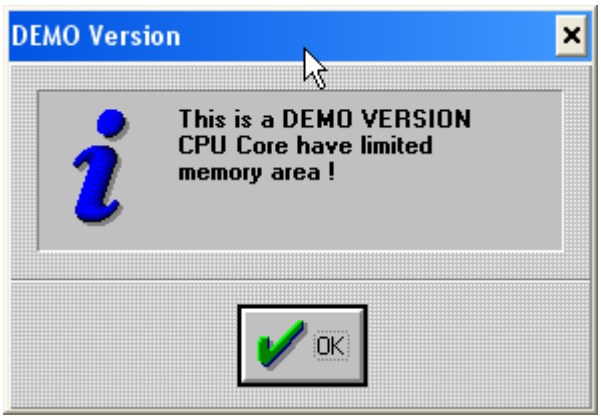
Gambar 1-4

2. pilih jenis cpu yang hendak di simulasikan dengan cara
   - Pada Menu pilih configure
   - pilih load cpu -> pilih 8031.cpl. tampilan tampak pada gambar 1-5. Kemudian klik ok.



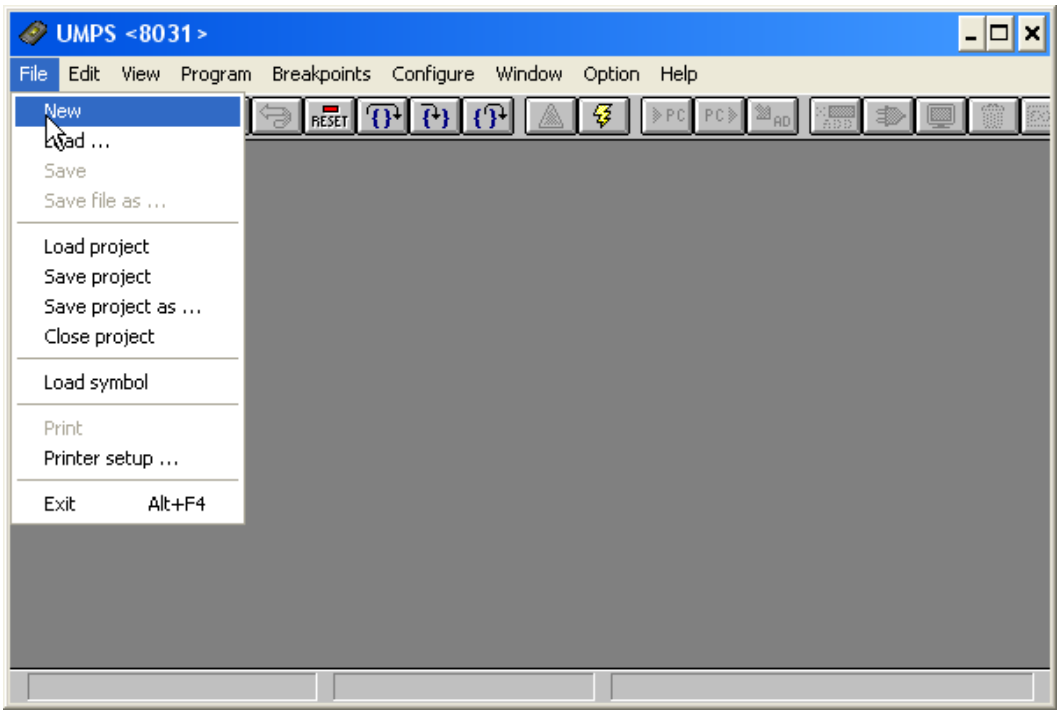
gambar 1-5

- Program yang anda pakai adalah versi shareware sehingga bila muncul popup klik ok



Gambar 1-6

3. Kemudian buat file baru dengan cara:
   - pilih menu-> pilih file-> pilih new



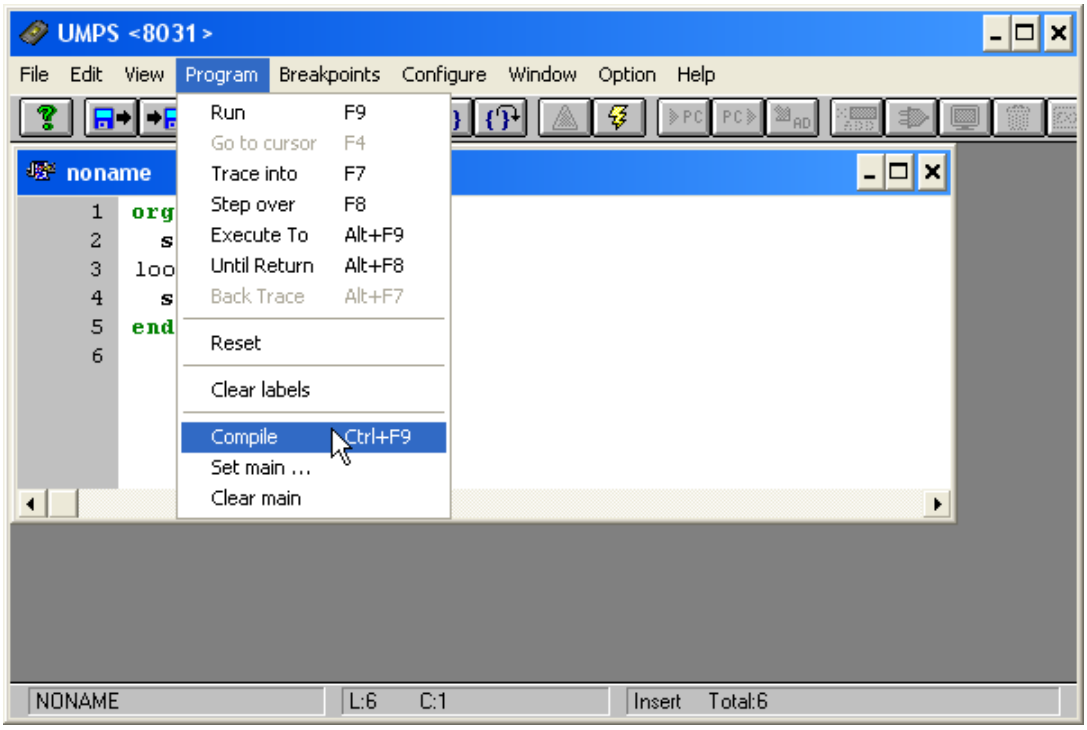
Gambar 1-7

   - ketik program dibawah ini

```
org 0
  setb p1.0
loop:
  sjmp loop
end
```

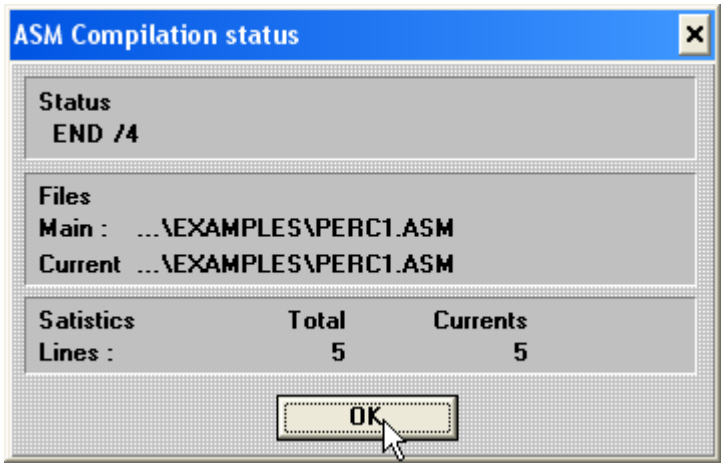
4. simpan dengan nama perc1.asm

5. compile program dengan cara pilih menu program kemudian pilih compile.



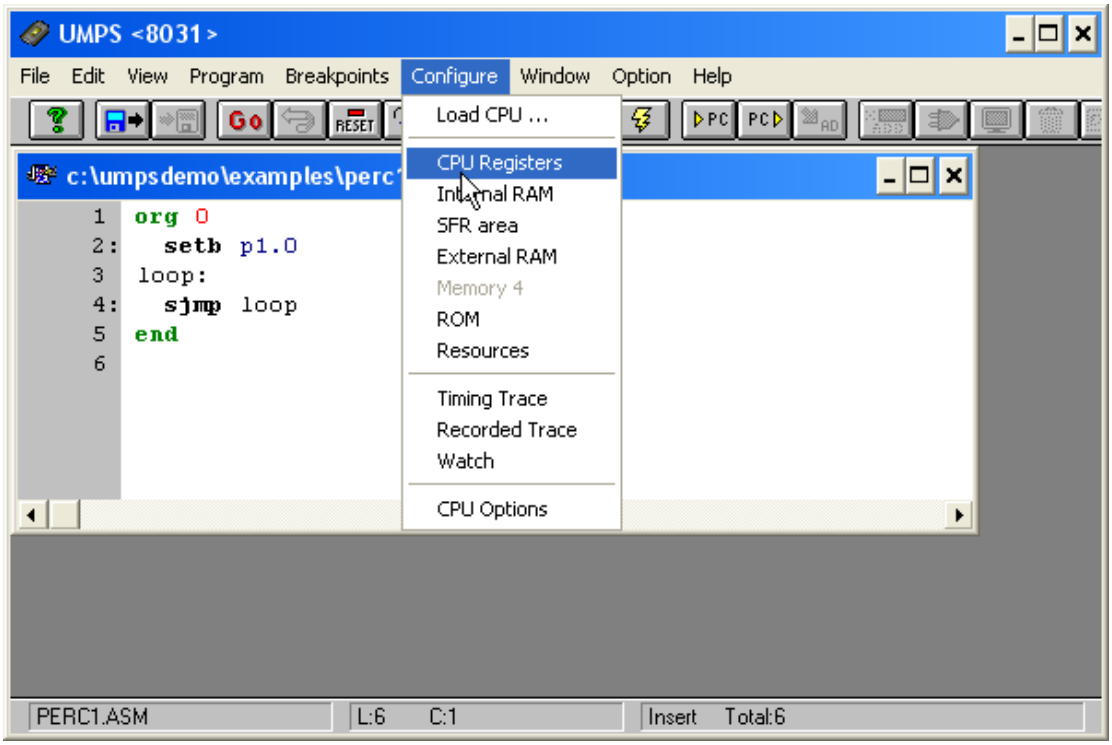
Gambar 1-8

6. Pilih file yang akan di kompile. Dalam hal ini pilih perc1.asm. bila tidak ada kesalahan tampilan akan seperti gambar 1-9



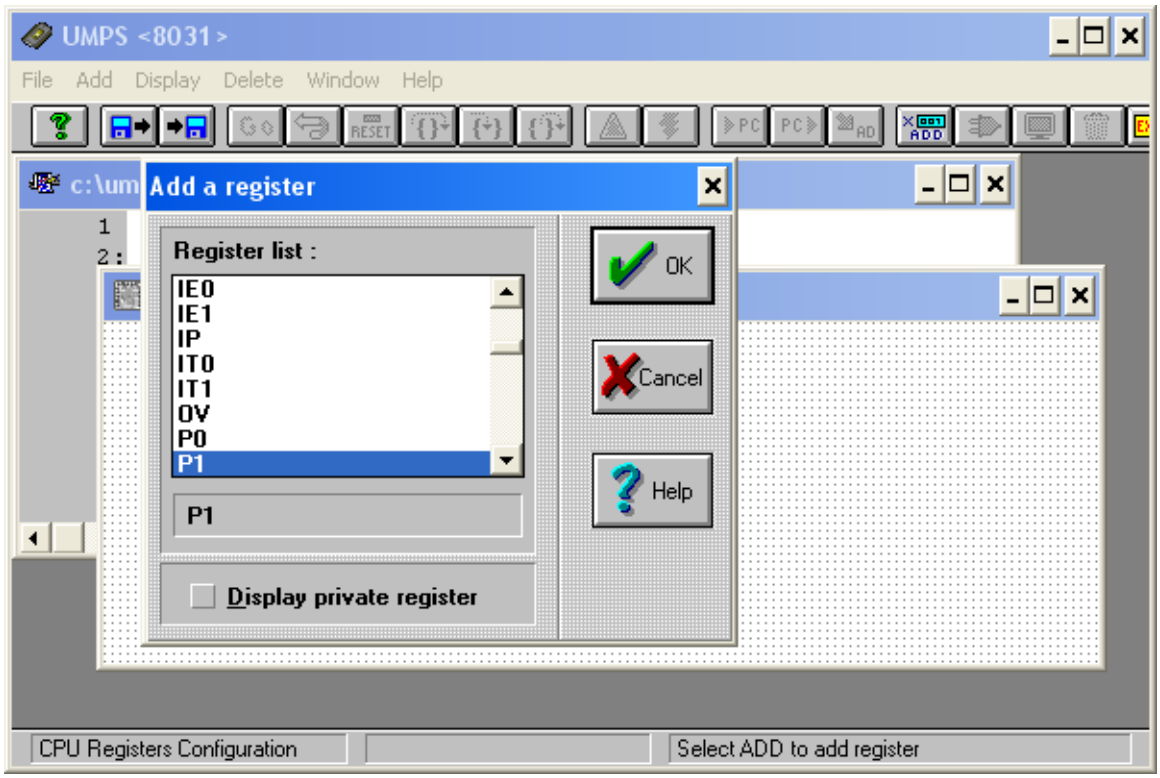
gambar 1-9 status kompile program

7. Langkah pembuatan program sampai compile program telah selesai. Langkah berikutnya adalah simulasi program. Simulasi ini digunakan untuk melihat kebenaran dari program yang telah dibuat, salah satunya dengan melihat kondisi port 1. Langkahnya:
   - Pada Menu pilih configure-> pilih CPU Register



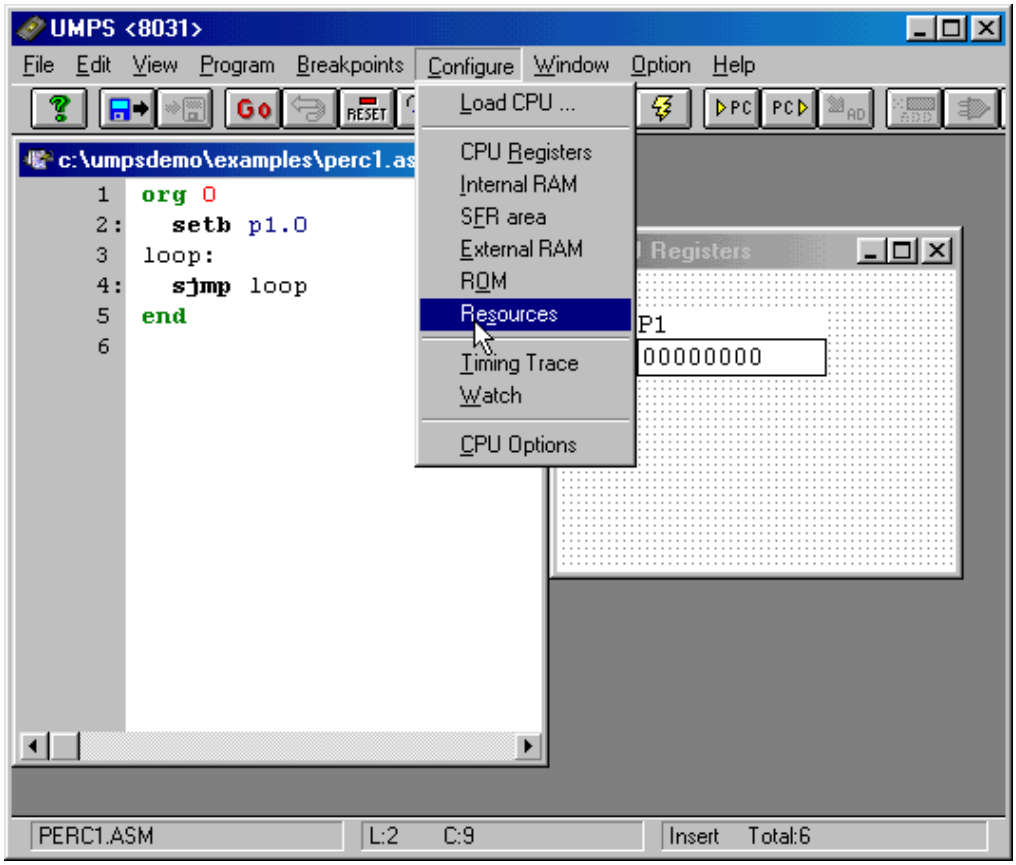
Gambar 1-10

- selanjutnya tambahkan port p1 pada cpu register dengan cara pilih toolbar add pada option cpu register. Bisa juga pilih menu add pada main menu. Setelah itu window CPU dengan cara melakukan *drag* mouse pada tepi *window*.



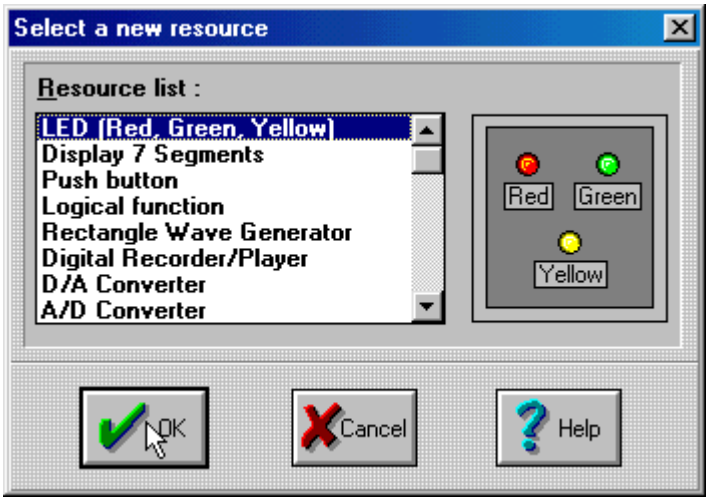
Gambar 1-11

8. Port p1 bit ke 0 nantinya dihubungkan dengan LED. Untuk memastikan bahwa program yang kita buat bisa berfungsi dengan baik apabila dihubungkan dengan LED, maka di perlu dibuat device LED virtual yang dihubungankan dengan port 1 bit ke nol sebagai simulator UMPS Untuk menghubungkan resource LED pilih pada file menu **configure->resource**



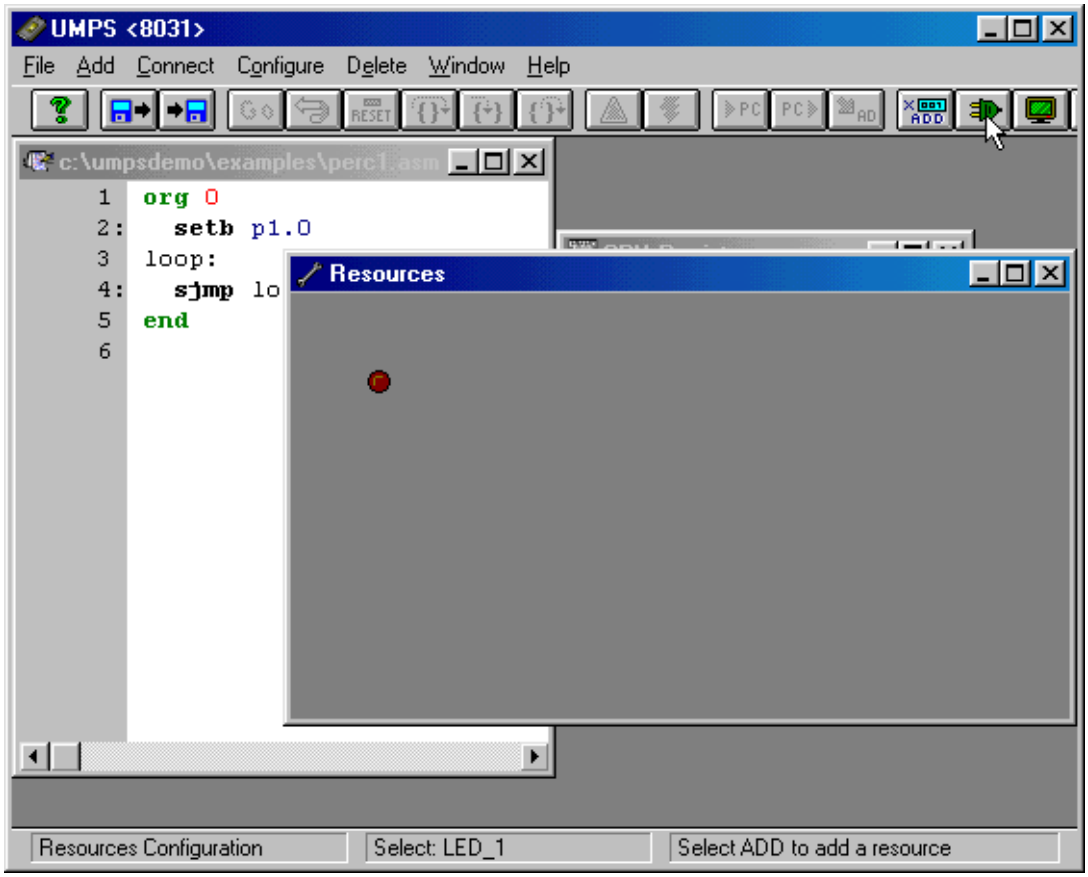
Gambar 1-12

9. Pilih ADD pada main menu-> pilih LED.



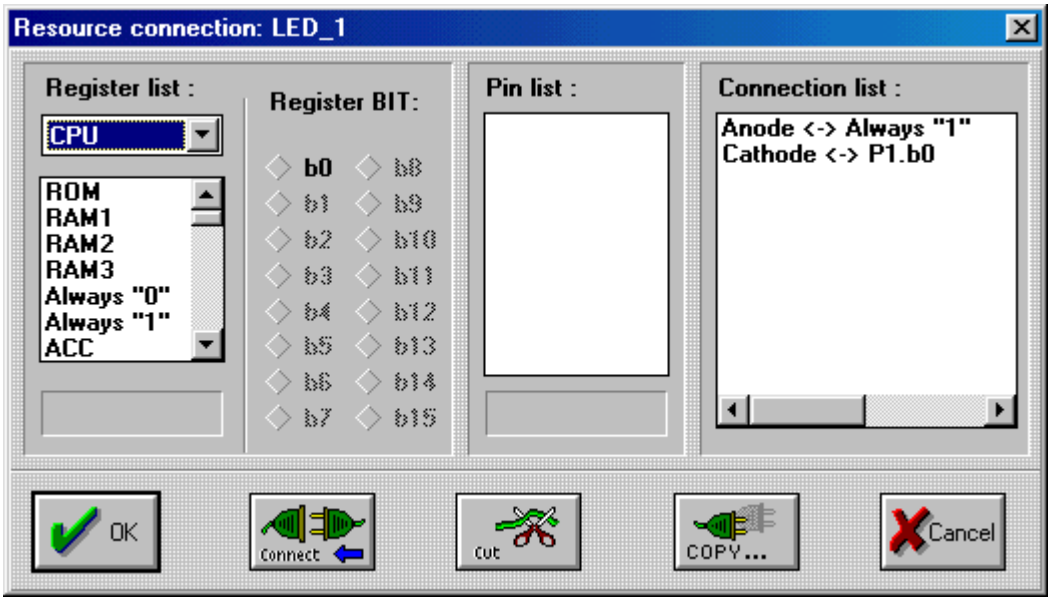
Gambar 1-13

10. Selanjutnya hubungkan anode LED dengan vcc dan katoda LED dengan p1.0 dengan cara klik kanan resouce LED atau pilih gambar properties seperti yang ditunjukkan oleh mouse pada gambar 1-14.



Gambar 1-14

11. Selanjutnya kita harus menghubungkan anode dengan vcc sementara katode dengan port 1 bit ke 0. Pada register list pilih Always ”1” dan pada pin list pilih anode kemudian klik **connect**. Untuk katode hubungkan dengan port1 bit ke nol . Setelah selesai tampilan akan seperti gambar 1-15



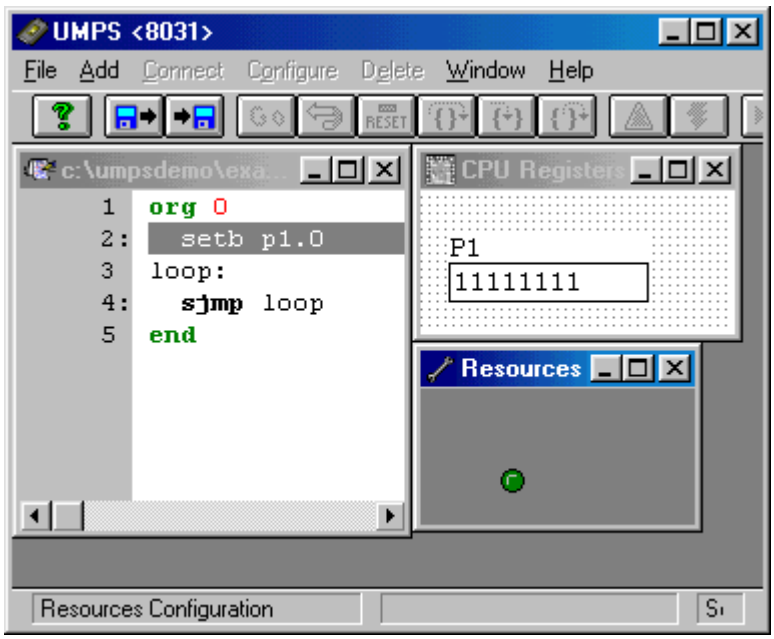
gambar 1-15 resource connection

untuk melihat jalannya program dapat digunakan mode simulasi degan beberapa pilihan yaitu:

- **Run (F9),** Menjalankan program dengan alamat awal register PC
- **Step over (F8),** menjalankan program step by step dengan tidak melakukan proses pada procedure
- **Trace into (F7),** Menjalankan program step by step
- Dan beberapa pilihan lain seperti until return (ALT –F8), execute to (ALT- F9) dan backtrace (ALT –F7). Lihat gambar 1-16

gambar 1-16 toolbar simulasi

12. pilih mode run dengan cara klik toolbar go atau tekan tombol F9. program akan aktif. Tampilan ditunjukkan pada gambar 1-17



gambar 1-17 Tampilan register dan kode program

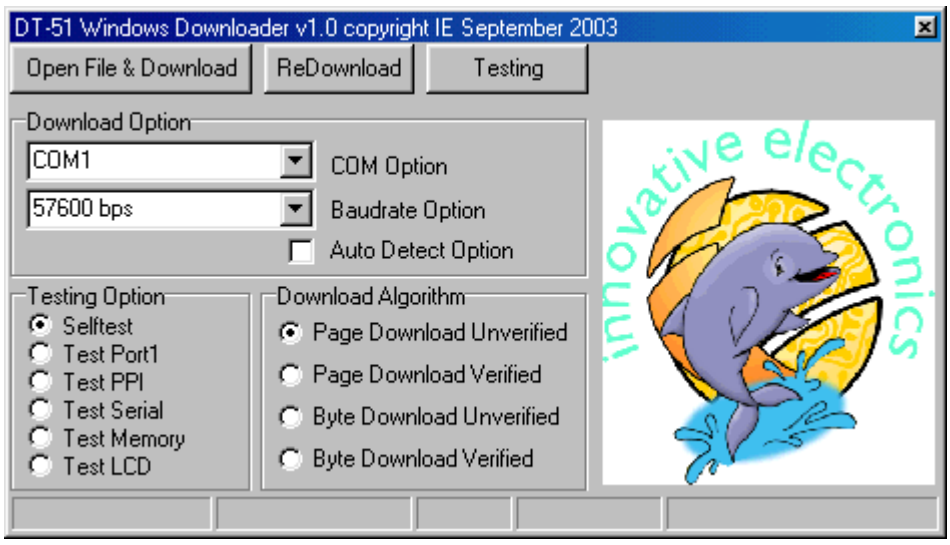
13. Stop dan reset program dari menu program. Ubah listing baris kedua dari setb p1.0 menjadi clr p1.0 tekan reset dan kemudian jalankan lagi program dengan menekan tombol F9. amati program yang berjalan dan hasil tampilan led

Download ke device MCS-51

Setelah program disimulasikan dan berhasil langkah selanjutnya adalah melakukan test ke hardware. Pada software UMPS ketika di kompile akan menghasilkan file dengan extensi hex dan lst. File berekstensi hex mempunyai format intel hexa. File dengan ekstensi hex ini dimasukkan ke development board MCS-51. sementara file berekstensi lst berisi listing dan kode mneumonic dari program yang dibuat.

Urutan download ke device MCS-51 adalah sebagi berikut:

1. load program dt51Lwin. Apabila berhasil tampilan akan seperti pada gambar 1-18



Gambar 1-18 Tampilan awal dt51l versi windows

2. konfigurasi port serial dan baud rate, bisa juga dipilih auto detect option supaya lebih mudah.
3. pilih open file and download, dan pilih file program yang dibuat dengan ekstensi hex. Pada langkah ini pastikan dt51 sudah terhubung ke port serial dan board sudah dalam keadaan on.

STOP	• Awal program untuk development board dt51 mulai alamat 4000h. Pastikan program awal anda di alamat 4000h dengan mengganti org 0 dengan org 4000h dan melakukan kompile ulang lagi. • Pastikan posisi jumper berada pada 1 dan 2

4. lihat tampilan pada p1.0 kemudian ganti instruksi pada perc1.asm pada baris ke 2 dari setb p1.0 menjadi clr p1.0 amati yang terjadi.

Percobaan 2 – flashing LED

Penggunaan prosedur dalam program

Disain kedua mempunyai rancangan hardware seperti dalam percobaan pertama (gambar 1-1), yang ditambahkan prosedur delay untuk membuat led pada p1.0 berkedip. Lama waktu LED dari on ke off diatur oleh perulangan(loop) dengan tiga register yaitu regsiter r0, r1 dan r2. Perulangan terjadi sebesar 200 ribu kali didapatkan dari nilai pada register ro dikalikan nilai pada register r1 dikalikan nilai pada register r2.

Urutannya sebagai berikut:

- Buka file baru dengan nama file perc2.asm
- Pastikan alamat awal 4000h
- ketik program dibawah ini

```
org 4000h
  ljmp start
delay:
  mov r2,#100
loop3:
  mov r1,#100
loop2:
  mov r0,#20
  loop1: djnz r0,loop1
  djnz r1,loop2
  djnz r2,loop3
  ret
;mulai program
org 4100h
start:
  setb p1.0
  lcall delay
  clr p1.0
  lcall delay
  ljmp start
end
```

- compile program yang dibuat (ketik F9)
- jalankan program downloader dt51lwin, buka file perc2.hex, download ke development

board dt51

Percobaan 3 – Running

Running LED 1

Disain running LED 1 adalah menjalankan LED yang bergerak dari kiri kekanan searah jarum jam. Untuk itu langkah yang diperhatikan adalah:

- Buka file baru dengan nama file perc3.asm
- Pastikan alamat awal 4000h
- Ketikkan program dibawah ini

```
; case 1
; running led on port 1
; program ini menggunakan simulator MCS-51
; program ini Menjelaskan Penggunaan prosedur
; akhmad hendriawan
; selasa 22 sept 2004
; Hardware Notes:

;Input Definition

;Output Definition

;define segment
code_seg equ 4000h

org code_seg
    ljmp start
org code_seg+100h
;------------------------
; prosedur delay detik
;------------------------
delay:
  push 0
  push 1
  push 2
  mov r2,#100
loop3:
  mov r1,#100
loop2:
```

```
  mov r0,#1
  loop1: djnz r0,loop1
  djnz r1,loop2
  djnz r2,loop3
  pop  2
  pop  1
  pop  0
  ret

;=================
; main program   =
;=================
start:
; inisisalisasi
     mov  a,#0FFh
     cpl  a
     mov r0,#08
     clr psw.5
ulang:
     jb   psw.5,set_kanan
     setb C
     rrc  a
     ljmp  proses
set_kanan:
     clr C
     rlc  a
proses:
     mov  p1,a
     lcall delay
     djnz r0,ulang
     cpl  psw.5
     mov  r0,#09
     ljmp  ulang
end;
```

- compile program yang dibuat (ketik F9)
- jalankan program downloader dt51lwin, buka file perc3.hex, download ke development board dt51
- Lihat hasilnya

VII. Laporan Sementara

1. Jelaskan fungsi ret pada listing program di percobaan 2
2. Hitung waktu kasar yang diperlukan pada percobaan 2 untuk membuat LED on dan off jika Frek kristal yang digunakan sebesar 11.0592 Mhz.
3. Analisa program pada percobaan 3
4. Buat rancangan hardware yang menampilkan bilangan biner yang nilainya naik dari 0-9 kemudian turun dari 9-0. jelaskan algoritma
5. Buat rangkaian counter 8 bit yang menghasilkan cacahan output dengan urutan bilangan prima dari 2,3,5,7,11,13. Jelaskan algoritma yang dipakai

VIII. Laporan Resmi

Sebagai analisa pada laporan resmi, gambarkan dan jelaskan secara detail dengan diagram alir pada tugas yang anda kerjakan pada no 4 dan 5 (Laporan resmi)

IX. Tambahan

Berikan saran atau komentar guna pengembangan lebih lanjut praktikum ini

Bab 2

Interfacing Mikrokontroler ke PPI

I. Tujuan

Setelah Melakukan praktikum ini yang anda peroleh adalah:

- Dapat menjelaskan kegunaan *peripheral interface (PPI) 8255*
- Dapat membuat projec baru dengan menggunakan PPI 8255
- Dapat membuat program sederhana Pembacaan input menggunakan.

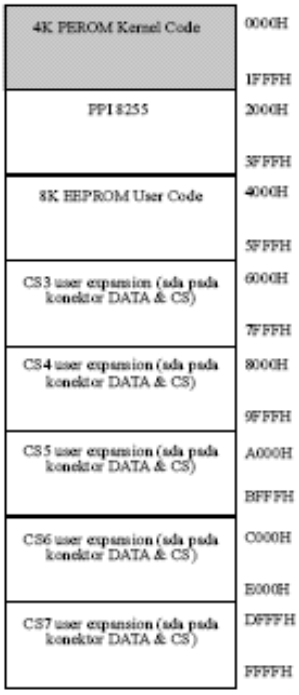
II. Pendahuluan

Pada praktikum ini anda akan mencoba memanfaatkan fasilitas komunikasi pararel antara mikrokontroller MCS-51 dengan PPI 8255. Modul sebelumnya adalah mengeluarkan output dari Port 1 ke LED, menghidupkan dan mematikan LED dan membuat modul running LED. Pada praktikum ini anda membaca input dari port PPI dan mengeluarkan hasilnya ke LED melalui Port P1. Kemudian dilanjutkan dengan desain line following robot sederhana menggunakan algoritma sederhana

III. Gambaran Desain

Anda membuat Project baru yang memanfaatkan fasilitas output dan input. Fasilitas output berasal dari port 1, fasiltas input diperoleh dengan menggunakan port C device 8255 yang dihubungkan dengan 89c51. inisialisasi diperlukan agar port C 8255 berfungsi sebagai input. Inisialisasi dilakukan dengan menuliskan data pada port CW(control word) PPI 8255 di alamat 2003h. Sedang port C mempunyai alamat 2002h. Pemilihan alamat ini diperoleh dengan melihat

mapping memori pada board dt51 yang ditunjukkan pada gambar 2-1



gambar 2-1 mapping memori board dt51

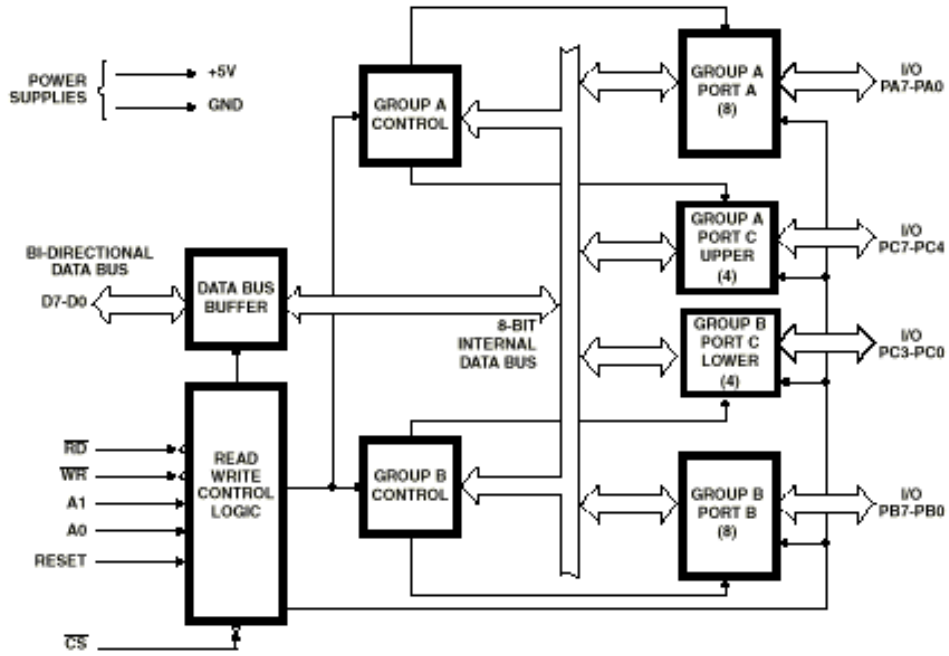
Pada mapping memori yang ditunjukkan pada gambar 2-1 terlihat bahwa:

- 8Kb pertama (0000H - 1FFFH) digunakan untuk internal 4K PEROM yang berisi kernel code, sedangkan 4K sisanya reserved.
- 8Kb kedua (2000H - 3FFFH) digunakan untukPPI 8255 dan hanya terpakai 4 alamat :
    - 2000H - Port A
    - 2001H - Port B
    - 2002H - Port C
    - 2003H - Control Word Register
- 8Kb ketiga (4000H - 5FFFH) digunakan oleh EEPROM untuk menyimpan User Code.
- CS3-CS7 (6000H - FFFFH) disediakan untukekspansi.

IV. Dasar Teori

PPI 8255 adalah programable I/O device yang didesain untuk digunakan dengan intel mikroprosessor. Pada device ini terdapat sebanyak 24 pin I/O port yang dapat diprogram secara individu dalam dua group yaitu group a dan group b dan dapat di program dalam tiga mode yaitu mode 0 mode 1 dan mode 2. Mode 0 dapat disamakan dengan simple I/O mode. Mode 1

sebagai single handshake protokol dan mode 2 sebagai double handshake protokol.



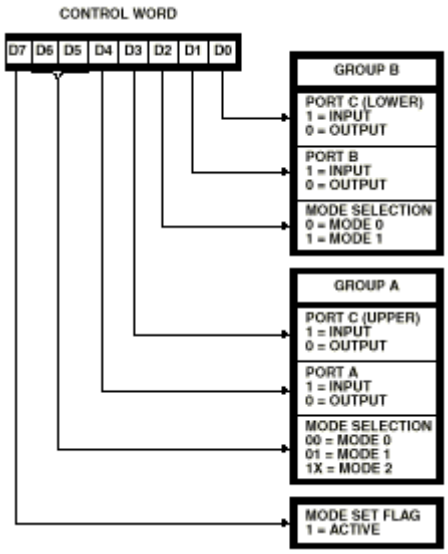
Gambar 2-2

Port pada 8255 dibagi menjadi dua kelompok yaitu kelompok A terdiri dari port A (A0..A7) dan port C upper(C4..C7) sedang kelompok B terdiri dari port B (B0..B7) dan Port C lower (C0..C3). nisialisasi fungsi port ditangani oleh register control CW. Pemberian data ke Control word (CW). Mempengaruhi fungsi Port pada MCS-51. Untuk menentukan hubungan data bus dan port digunakan sinyal A0 dan A1 seperti pada gambar 2-3

A1	A0	RD	WR	CS	INPUT OPERATION (READ)
0	0	0	1	0	Port A → Data Bus
0	1	0	1	0	Port B → Data Bus
1	0	0	1	0	Port C → Data Bus
1	1	0	1	0	Control Word → Data Bus
					OUTPUT OPERATION (WRITE)
0	0	1	0	0	Data Bus → Port A
0	1	1	0	0	Data Bus → Port B
1	0	1	0	0	Data Bus → Port C
1	1	1	0	0	Data Bus → Control
					DISABLE FUNCTION
X	X	X	X	1	Data Bus → Three-State
X	X	1	1	0	Data Bus → Three-State

Gambar 2-3

Konfigurasi setiap port dapat diprogram dengan melakukan inisialisasi terhadap control word. Control Word ini menggandung informasi seperti mode,bit set dan bit reset yang ditunjukkan pada gambar 2-4



Gambar 2-4

V. Peralatan yang dibutuhkan

1 Modul board DT 51

2 Simulator MCS-51 (UMPS)

3 Cross Assembler MCS-51 (UMPS)

4 IBM PC kompatibel

5 LED modul.

VI. Prosedur Percobaan

Simple I/O

- Buka file baru dengan nama file perc3.asm
- Pastikan alamat awal 4000h
- Ketikkan program dibawah ini

```
;*********************************************************************
;Switch reader
;ABSTRACT :program baca switch hasil biner tampil di LED
;AUTHOR   :core_fpga
; PORT A  :output 2000H
; PORT B  :output 2001H
; PORT C  :input 2002H
; CW      :kontrol 2003H
; P1      :output LED
;*********************************************************************
porta   equ 2000h
portb   equ 2001h
portc   equ 2002h
cw      equ 2003h
datacw  equ %10001001
org 4000h
;*****************************************
; awal program
;*****************************************
 start:
; inisialisasi sp
   mov  sp,#20h
; inisialisasi ppi
   mov  dptr,#cw
   mov  a,#datacw
   movx @dptr,a
ulang:
   mov  dptr,#portC
   movx a,@dptr
   mov  p1,a
   ljmp ulang
```

```
end
```

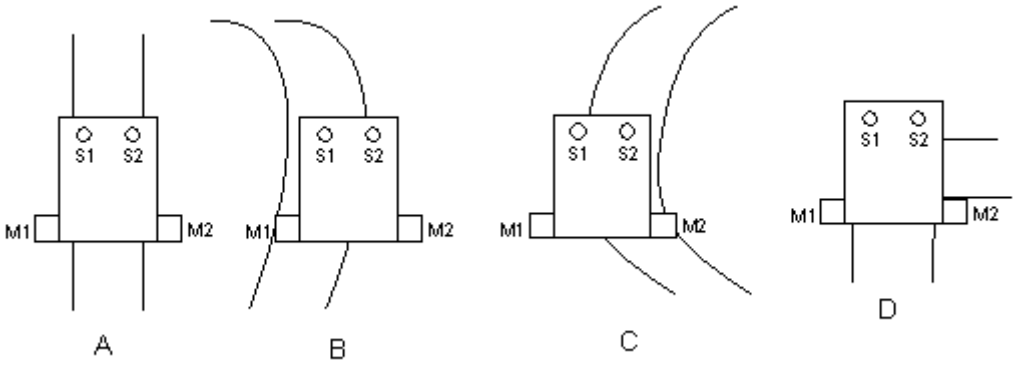
STOP	• awal program untuk dt51 mulai alamat 4000h. Pastikan program awal ada dialamat 4000h dengan mengganti org 0 dengan org 4000h dan melakukan kompile ulang lagi. • Pastikan posisi jumper berada pada 1 dan 2

- simpan program dengan nama p2-1.asm
- compile program.
- download program ke board dt51
- lihat hasil pada display

VII. Laporan Sementara

Tugas berikut dikerjakan sebagai laporan sementara

1. Sebuah line following robot dikontrol oleh dua buah sensor s1 dan s2. Sensor ini terletak didepan dan berada dalam lintasan. Sementara itu robot digerakkan oleh dua buah motor yang dipasang dibelakang (lihat gambar 2-5)



Gambar 2-5

Line following robot mempunyai ketentuan sebagai berikut:

- Kedua sensor (S1 dan S2) akan berlogik '1' jika berada dalam lintasan
- Motor akan hidup jika driver motor berlogik 1
- Sensor S1 dihubungkan dengan port C.0 dari PPI 8255
- Sensor S2 dihubungkan dengan port C.1 dari PPI 8255
- Motor 1 dihubungkan dengan port P1.0 dari MCS-51

- Motor 2 dihubungkan dengan port P1.1 dari MCS-51

Posisi	Aksi		Keterangan
	M1	M2	
A	On	on	Lurus
B	Off	on	kiri
C	On	off	Kanan
D	Off	off	Diam

2. Desain line following robot (rancangan pada gambar 2-5) yang terdiri dari dua buah sensor dan dua buah motor, sensor terletak didepan sementara motor terletak di belakang. Setiap motor dikontrol dua pin dari MCS-51 yang mempunyai tabel sebagai berikut:

Motor		Aksi
MxA	MxB	
0	0	Mati
0	1	Forward
1	0	Reverse
1	1	Berhenti Mendadak
Note: X=1 motor 1 X=0 motor 2		

Pin koneksi dari motor sebagai berikut:

Alat	Koneksi Port
M1A	Port P1.0
M1B	Port P1.1
M2A	Port P1.2
M2B	Port P1.3
S1	Port C.0
S2	Port C.1

Robot mempunyai ketentuan sebagai berikut:

- Sensor akan berlogik ’1’ jika berada dalam lintasan

Posisi	Aksi		Keterangan
	M1	M2	
A	forward	forward	Lurus
B	Off	forward	kiri
C	forward	off	Kanan
D	Reverse	reverse	Mundur

VIII. Laporan Resmi

Sebagai analisa pada laporan resmi, gambarkan dan jelaskan metode yang lain yang dapat anda buat (termasuk rancangan hardware) untuk membuat line following robot

IX. Tambahan

Berikan saran atau komentar guna pengembangan praktikum ini lebih lanjut

Bab 3

Pemanfaatan timer

I. Tujuan

Setelah Melakukan praktikum ini yang anda peroleh adalah:

- Dapat menjelaskan berbagai mode timer dalam MCS-51
- Dapat memanfatkan fasilitas timer dalam MCS-51
- Dapat memanfaatkan fasilitas counter dalam MCS-51

II. Pendahuluan

Pada praktikum ini anda akan mempelajari dan memanfaatkan fasilitas timer pada mikrokontroller 8951. modul sebelumnya anda mempelajari dasar interfacing 8051 ke PPI 8255, membuat program simple input output dengan masukan dari switch dan keluaran menuju LED,menerapkan lebih lanjut dengan cara membuat program untuk line following robot. Praktikum Modul sekarang anda akan membuat timer hardware yang dapat diprogram untuk keperluan delay, kemudian membuat lite function generator (versi kecil function generator yang dapat mengeluarkan sinyal kotak) diakhiri dengan membuat desain traffic light controller serta beberapa project kecil.

III. Gambaran Desain

Pada membuat desain baru pada 8051 dengan memanfaatkan fasilitas timer, fasilitas timer ini diaktifkan oleh beberapa register timer pada 8051. register TMOD digunakan untuk mengubah mode timer. Ada tiga mode timer yang dapat di fungsikan yaitu mode 0 (13 bit), mode 1(16 bit),

mode 2 (8 bit auto reload) dan mode 3. Register lain yang pentin untuk mengaktifkan programmable timer pada 8051 adalah register TCON. Register ini digunakan untuk menstart timer dan melihat hasil dari timer yang telah diprogram.

IV. Dasar Teori

Banyak aplikasi dari mikrokontroller memerlukan perhitungan untuk beberapa kejadian seperti jumlah orang yang hadir, pengukuran frekuensi, pengukuran duty cycle dan pembangkitan frekuensi. Dengan menggunakan timer bisa didapatkan waktu yang lebih presisi. Register 16 bit T0 dan T1 digunakan oleh programmer untuk membangkitkan fungsi timer. Setiap counter dapat di program untuk menggunakan sumber clock eksternal (berfungsi sebagai counter) atau sumber clock eksternal (berfungsi sebagai timer).

Timer 1				Timer 0			
Gate	C/T'	M1	M0	Gate	C/T'	M1	M0

Keterangan

Gate : 0 Interupsi enable jika TRx pada TCON diset 1

1 Interupsi enable jika TRx pada TCON diset 1 dan pin int high

C/T': Set untuk operasi counter, reset untuk operasi timer

M1	M0	Keterangan
0	0	8048 Emulator (13 bit)
0	1	16 bit counter/timer
1	0	8 bit auto reload
1	1	2 timer 8 bit: TL0 8 bit timer dikontrol timer 0 standar TH0 8 bit yang dikontrol timer bit timer 1

Gambar 3 -1 register timer mode

Counter dibagi dalam 2 register 8 bit yaitu timer rendah (TL0,TL1) dan timer tinggi (TH0,TH1). Semua counter dikontrol oleh bit state dalam register mode timer (TMOD), register timer/control (TCON) dan instruksi program. Konfigurasi register tmod ditunjukkan pada gambar 3-2, sementara konfigurasi register TCON ditunjukkan pada gambar 3-2

Simbol	Posisi	Fungsi
TF1	TCON.7	Timer 1 overflow flag. Di set oleh hardware pada saat timer/counter overflow. Di clear oleh hardware pada saat menjalankan rutin interupsi.
TR1	TCON.6	Timer 1 Run control bit. Di set / clear oleh software. Digunakan untuk mengaktifkan/nonaktifkan timer/counter
TF0	TCON.5	Timer 0 overflow flag. Di set oleh hardware pada saat timer/counter overflow. Di clear oleh hardware pada saat menjalankan rutin interupsi.
TR0	TCON.4	Timer 0 Run control bit. Di Set / clear oleh software. Digunakan untuk mengaktifkan/nonaktifkan timer/counter
IE1	TCON.3	Interrupt 1 Edge flag. Di set oleh hardware ketika interupsi eksternal mendeteksi adanya edge. Di clear ketika prroses interupsi
IT1	TCON.2	Interrupt 1 Type control bit. Di set / clear oleh software untuk menentukan pen-triger-an interrupsi eksternal pada transisi turun / low level
IE0	TCON.1	Interrupt 0 Edge flag. Di set oleh hardware ketika interupsi eksternal mendeteksi adanya edge. Di clear ketika prroses interupsi
IT0	TCON.0	Interrupt 0 Type control bit. Di set / clear oleh software untuk menentukan pen-triger-an interrupsi eksternal pada transisi turun / low level

Gambar 3-2 konfigurasi bit dari register TCON

Pada saat sebagai Timer, register naik satu *(increment*) setiap satu *cycl*e. Jika digunakan osilator 12 Mhz, maka satu *cycle* sama dengan 1/12 frekuensi osilator = 1us. Pada saat sebagai counter, register naik satu *(increment*) pada saat transisi 1 ke 0 dari input eksternal , T0 atauT1.

Apabila periode tertentu telah dilampaui, Timer/Counter segera menginterupsi mikrokontroler untuk memberitahukan bahwa perhitungan periode waktu telah selesai dilaksanakan. Periode waktu Timer/Counter secara umum ditentukan oleh persamaan berikut:

a. Sebagai T/C 8 bit

T = (256 – TLx) * 1 siklus mesin

Dimana TLx adalah isi register TL0 atau TL1.

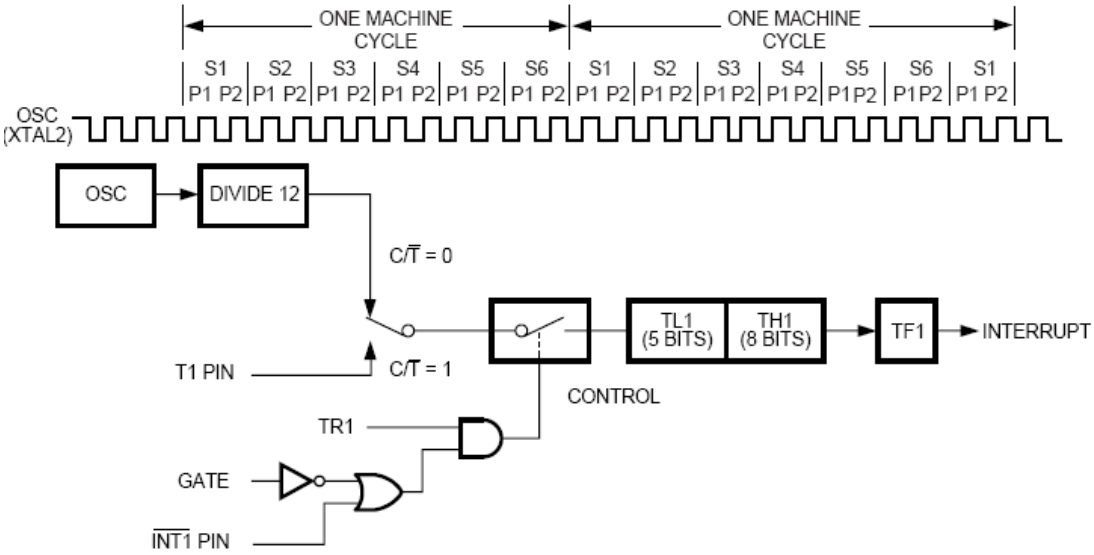
b. Sebagai T/C 16 bit

T = (65536 – THx TLx) * 1 siklus mesin

THx = isi register TH0 atau TH1

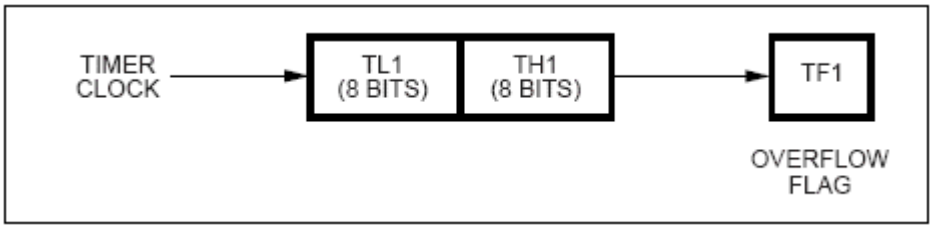
TLx = isi register TL0 atau TL1

Konfigurasi untuk timer 0 ditunjukkan pada gambar 3-3. pada gambar terlihata bahwa mode timer yang digunakan adalah mode timer 13 bit yang dapat mengemulasikan 8048 timer. Pada mode timer 13 bit ini, timer (TL0 dan TH0) digunakan sebagai counter 5 bit, sementara TH0 atau TH1 digunakan sebagai prescaller.



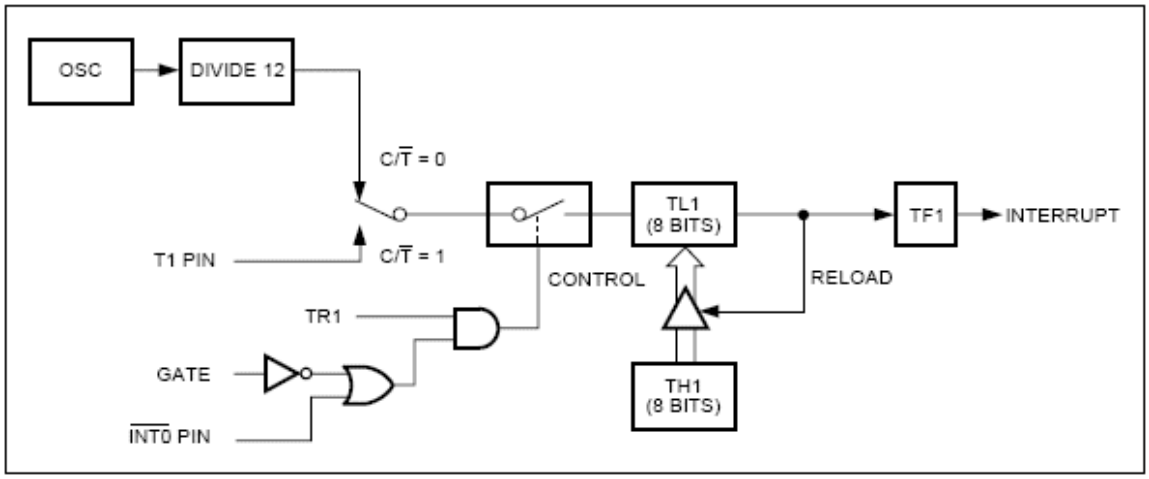
Gambar 3-3 timer/counter control logic timer 0 dan 1

Konfigurasi timer mode 1 ditunjukkan pada gambar 3-4. mode 1 sama halnya dengan mode 0, kecuali timer register yang digunakan sebesar 16 bit. Register 16 bit ini di peroleh dari gabungan register 8 bit THx dan TLx (x =0 atau 1 tergantung timer yang digunakan). Sekali pulsa diterima maka counter akan menghitung naik, overflow akan terjadi apabila counter menghitung dari FFFFh ke 00000h, timer tetap akan melanjutkan hitungan. Overflow bit di register TCON kemudian dapat dibaca oleh user.



Gambar 3-4 timer mode 1

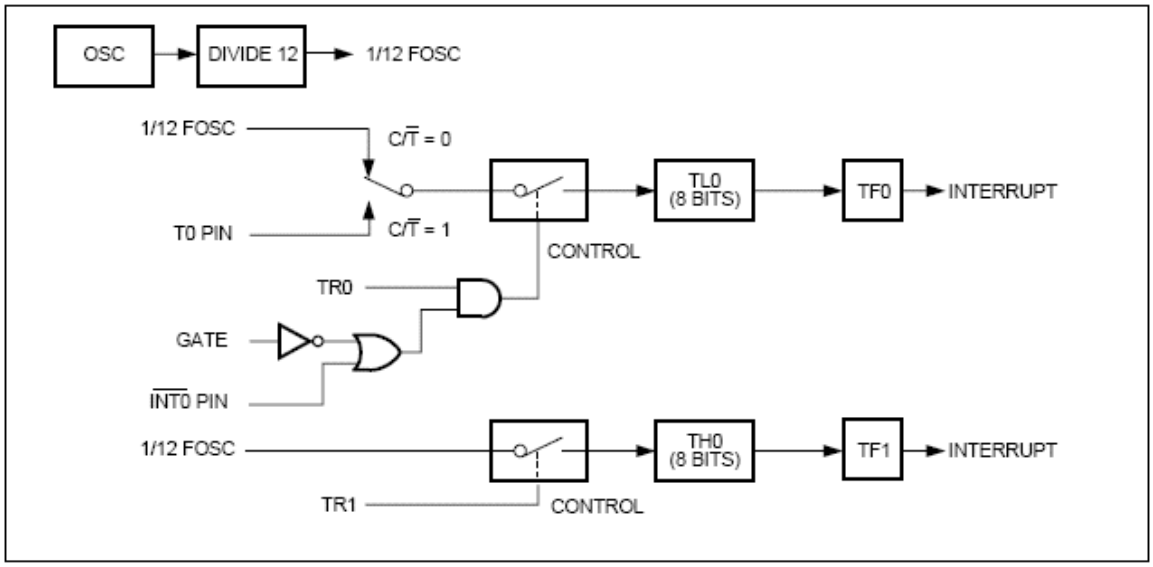
Konfigurasi untuk timer mode 2 dapat dilihat pada gambar 3-5. pada mode ini register timer yang digunakan sebesar 8 (TL). Apabila counter telah mencapai hitungan maksimum, overflow akan dibangkitkan. Bersamaan dengan itu data yang ada pada register th secara otomatis di dipindahkan ke counter TL. Pemindahan data ini tidak mengakibatkan perubahan data pada register TH.



Gambar 3-5 timer mode 2

Timer 1 pada mode 3 menahan hitungan, yang mempunyai efek yang sama apabila kita setting TR1=0. Timer 0 pada mode 3 terpisah menjadi dua timer (TL0 dan TH0). Blok diagaram dari timer mode 3 ditunjukkan pada gambar 3-6. TL0 yang digunakan pada timer 0 mempunyai control bit C/T, Gate, TR0, INT0 dan TF0. untuk TH0 mempunyai control bit TR1 dan TF1. TH0 sekarang berfungsi untuk mengontrol timer 1 interrupt. Dengan timer 0 pada mode 3 8051 dapat mempunyai timer sebanyak 3 buah. Ketika timer 0 diaktifkan pada mode 3, timer 1 dapat dihibupkan dan dimatikan dengan dalam mode 3, yang berarti timer 1 dalam mode ini masih dapat digunakan untuk keperluar serial port sebagai generator baudrate atau aplikasi yang tidak membutuhkan interrupsi

.



Gambar 3-6 timer mode 2

V. Peralatan yang dibutuhkan:

1 Modul board DT 51 + serial kabel
2 Development board 8051
3 IBM PC kompatibel
4 Crimson Editor
5 cross Compiler Metalink MCS-51
6 Databook MCS-51

VI. Prosedur Percobaan

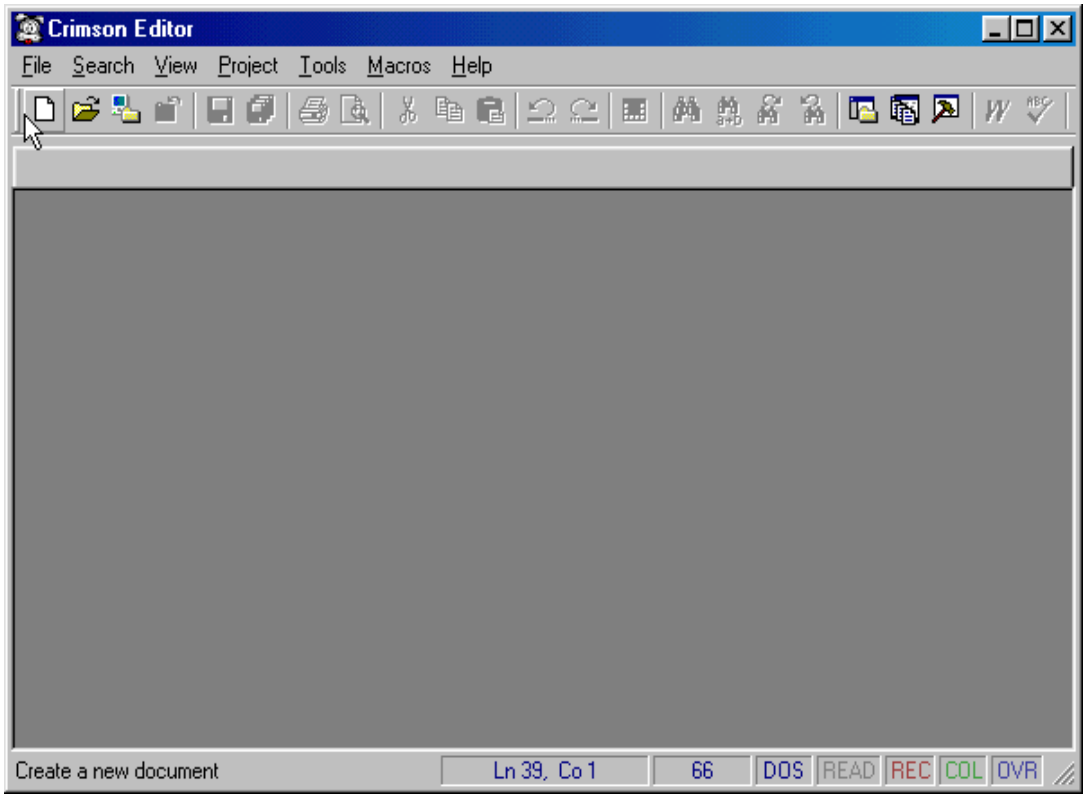
Percobaan 1 – Delay hardware

Pada percobaan ini anda diminta untuk membangkitkan delay 1 dt jika MCS-51 bekerja dengan menggunakan kristal 11.0592 Mhz. Gunakan timer mode 1 dengan pewaktuan sebesar 50ms. Output berupa nyala dan mati Led selama 1 detik.

Pada praktikum sebelumnya kita melakukan programming dengan menggunakan versi shareware edition dari UMPS. Versi ini mempunyai keterbatasan dalam jumlah baris yang dapat dikompile. Pada praktikum ini anda akan mencoba pendekatan baru dalam melakukan programming. Untuk praktikum ini dan seterusnya akan menggunakan crimson editor untuk editor dan software compiler metalink.

Memulai project baru

- Buka file baru



gambar 3-7

- Pastikan alamat awal 4000h
- Ketikkan program dibawah ini

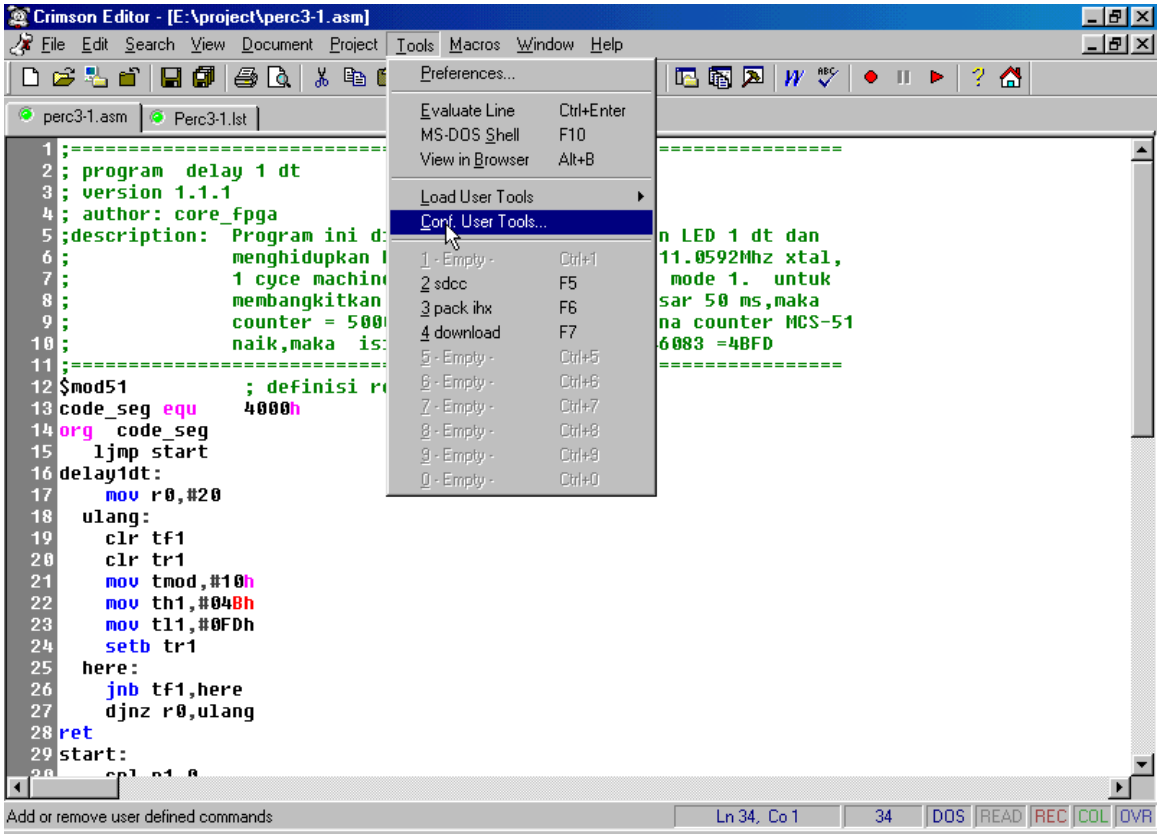
```
;====================================================================
; program  delay 1 dt
; version 1.1.1
; author: core_fpga
;description:  Program ini digunakan untuk menyalakan LED 1 dt dan
;              menghidupkan LED selama 1 dt. Dengan 11.0592Mhz xtal,
;            1 cyce machine =12/Xtal=1.085us timer mode 1.  untuk
;              membangkitkan delay pada timer 1 sebesar 50 ms,maka
;              counter = 50000us/1.085us=46083. karena counter MCS-51
;              naik,maka  isi counter dalam hexa= -46083 =4BFD
;====================================================================
$mod51          ; definisi register pada metalink 51
code_seg equ    4000h
org  code_seg
   ljmp start
delay1dt:
    mov r0,#20
  ulang:
    clr tf1
    clr tr1
    mov tmod,#10h
    mov th1,#04Bh
    mov tl1,#0FDh
    setb tr1
  here:
    jnb tf1,here
    djnz r0,ulang
ret
start:
    cpl p1.0
    lcall delay1dt
    ljmp start
end
```

- Simpan dengan nama perc3-1.asm

STOP	penyimpanan nama lain harus mengikuti kaidah penulisan standard dos baik direktory maupun nama file, karena kompiler metalink hanya mengenal foldeer dan nama file yang berformat dos • Pastikan posisi jumper berada pada 1 dan 2

Pembuatan program sudah selesai. Langkah selanjutnya adalah perakitan program. Untuk perakitan kita membutuhkan compiler metalink. Penggunaan dan pemilihan compiler ini mempertimbangkan kemudahan dan sifat licensinya yang freeware. Langkah compile program sebagai berikut

- Letakkan software metalink di direktori c:\asm51. Apabila belum ada instal software di direktori tersebut
- Kembali ke editor Crimson pada menu tool pilih conf user tools. Lihat gambar 3-8

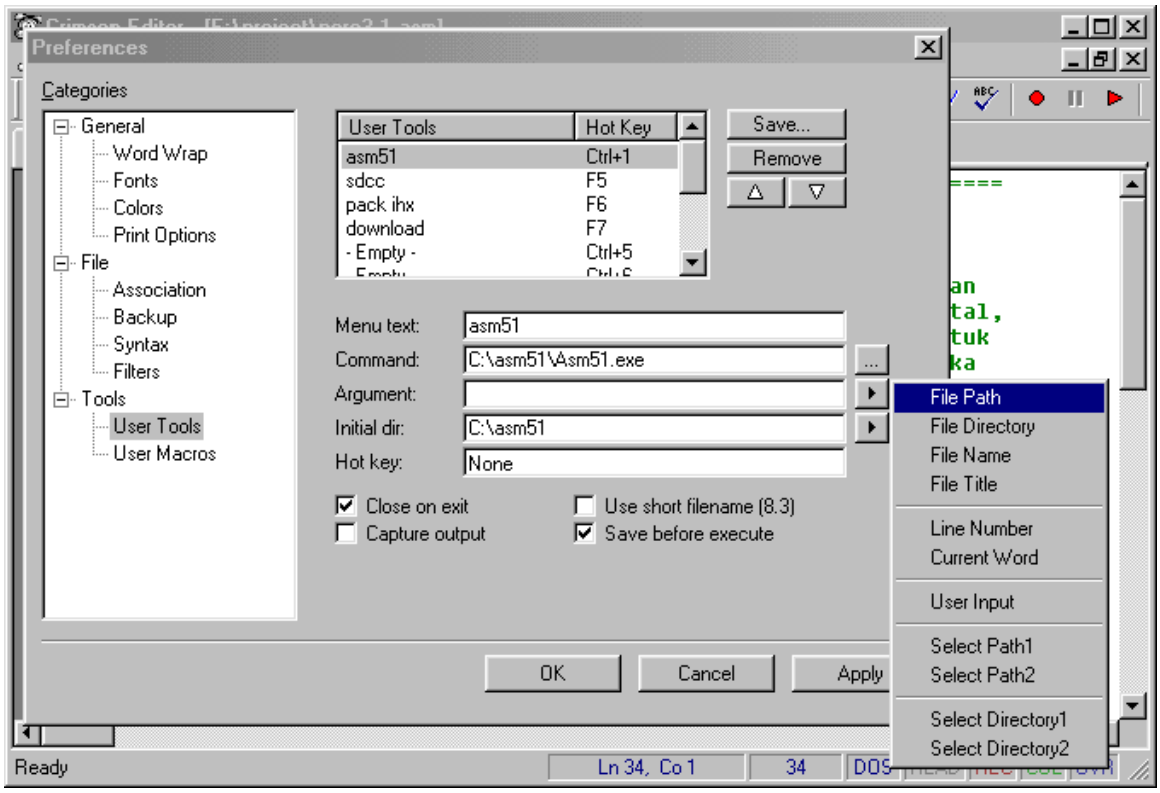


gambar 3-8

- Pada menu text ketik asm51
- pada menu command klik icon disebelah kanan menu itu ..., kemudian arahkan ke

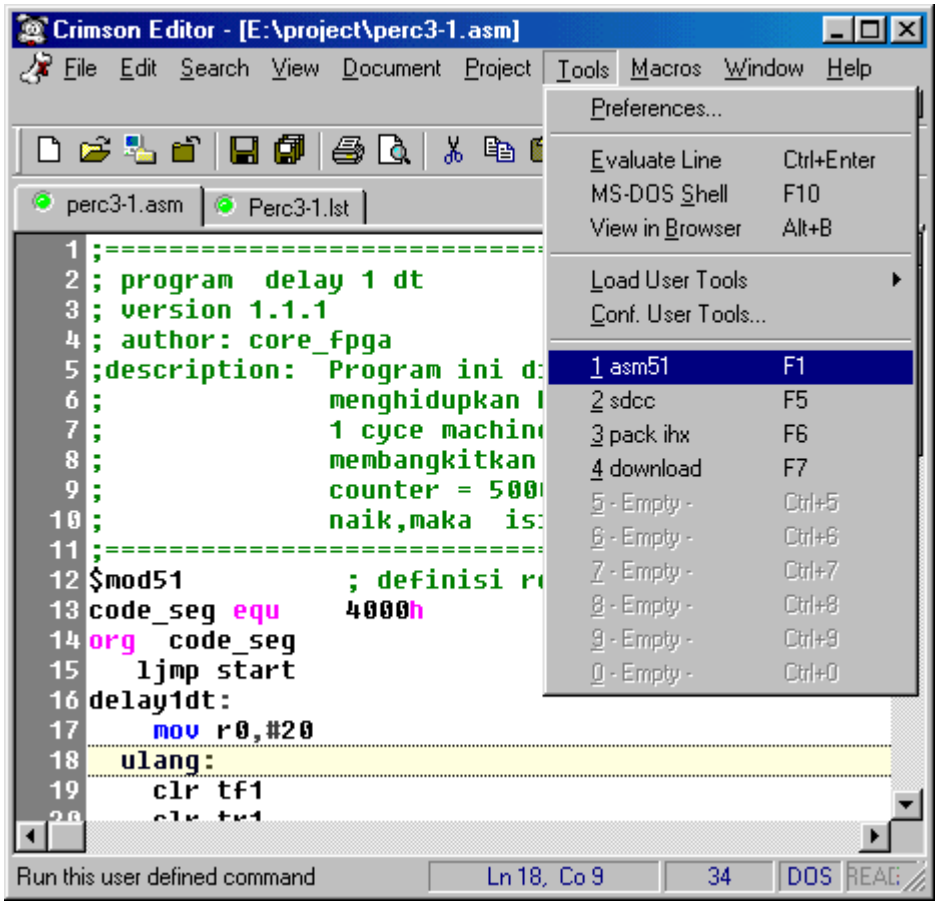
direktori c:\asm\asm51.exe atau langsung diketik.

- Pada baris argument klik icon disebelahnya kemudian pilih file path (gambar 3-9).



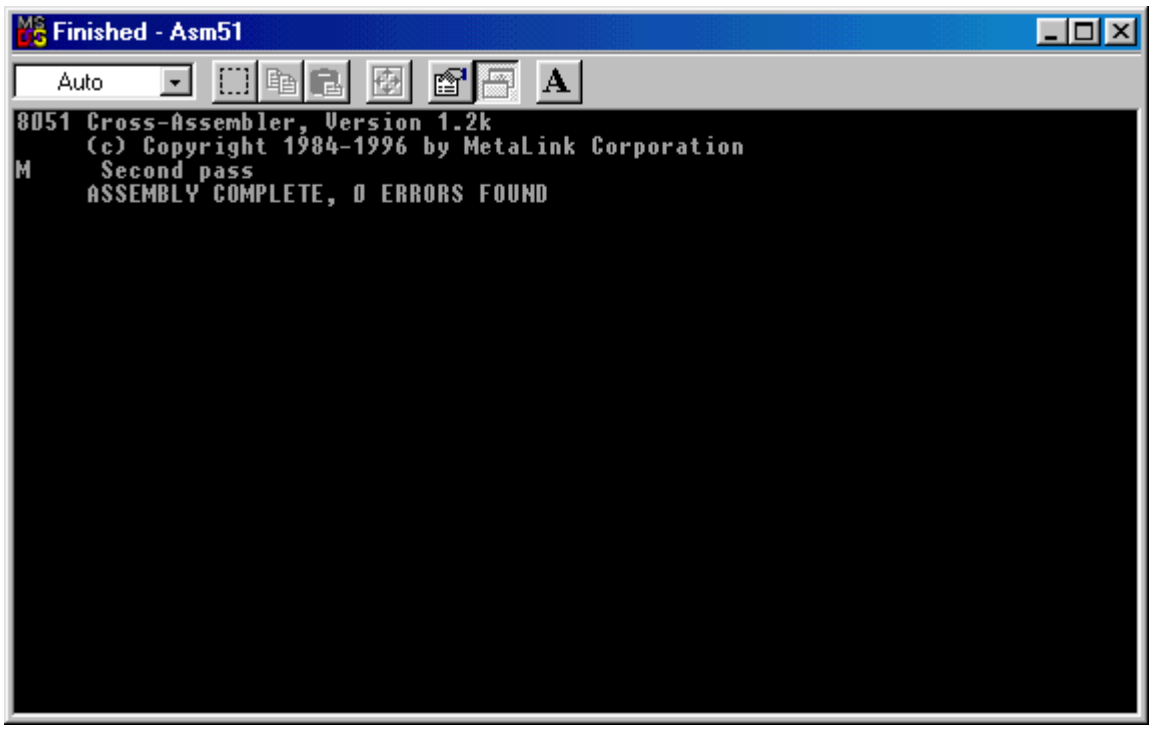
gambar 3-9

- Pada baris initial dir pilih klik icon disebelah kanannya ▶, kemudian pilih browse, pilih direktori c:\asm51. Dapat juga diketikkan langsung
- Hotkey adalah pilihan untuk shotcut pilih F1. setelah selesai klik ok
- Kemudian compile program dengan cara pilih tool, klik asm51. Untuk kemudahan bisa langsung ditekan F1.



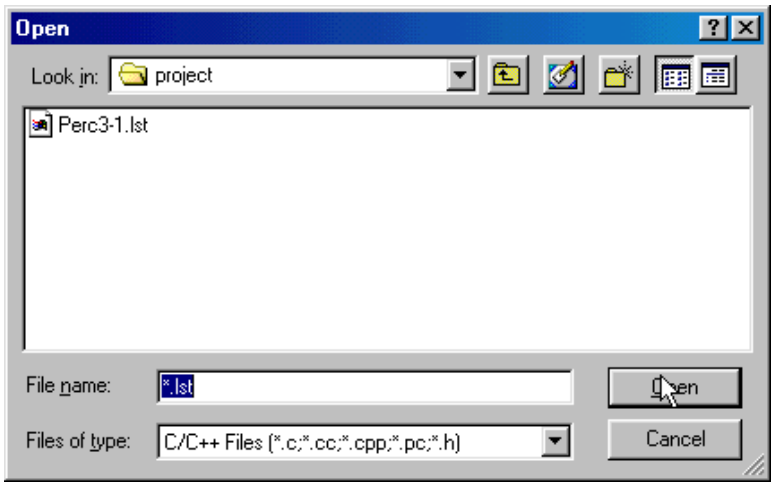
gambar 3-10

- Setelah itu compiler akan dijalankan secaraa otomatis oleh crimson editor hasilnya ditunjukkan pada gambar 3-11. pada gambar terlihat report dari compiler menyatakan tidak ada error.



Gambar 3-11

- Bila masih terdapat error, anda perlu mengetahui detail error agar dapat diperbaiki. Detail error dapat dilihat pada file dengan ekstensi.lst buka file tersebut dan lihat errornya. Lihat gambar 3-12



Gambar 3-12

- Setelah selesai download program yang berekstensi .hex di direktori yang sama dengan nama file assembler yang anda buat.

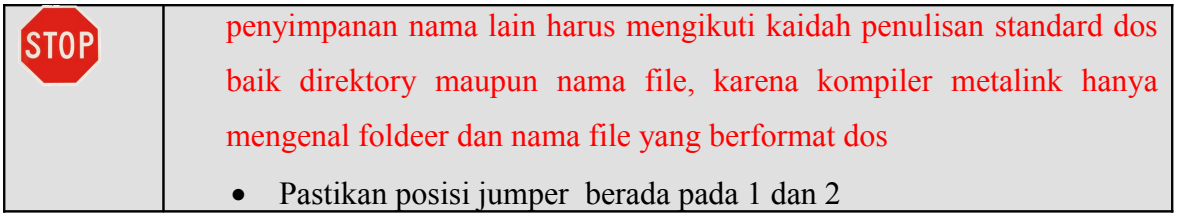
Percobaan 2- Square wave generator

Pada percobaan ini anda mencoba untuk membuat pembangkit pembangkit gelombang kotak sebesar 10 kHz dengan duty cycle sebesar 50%.

- Buka file baru
- Pastikan alamat awal program anda 4000h
- Tulis program dibawah ini

```
;=====================================================================
; program  square wave
; version 1.1.1
; author: core_fpga
;description:  this program is used for generating 10 khz square wave.
;              With 11.0592Mhz xtal, 1 cyce machine is 12/Xtal=1.085us
;              timer mode 2 autoreload. To generate 10 khz
;              (100us periode) and duty cycle 50%, then counter for
;              pulse high is 50us and counter for pulse low is 50us.
;              Th0 is 50us/1.085us=46. because counter of MCS-51 is
;               increasethen actual contain of th0 is -46 or 0D2h
;=====================================================================
$mod51
code_seg equ    4000h
org  code_seg
start:
    mov tmod,#02h
    mov th0,#0D2h
    setb tr0
here: jnb tf0,here
    clr tf0
    cpl p1.0
    ljmp here
end
```

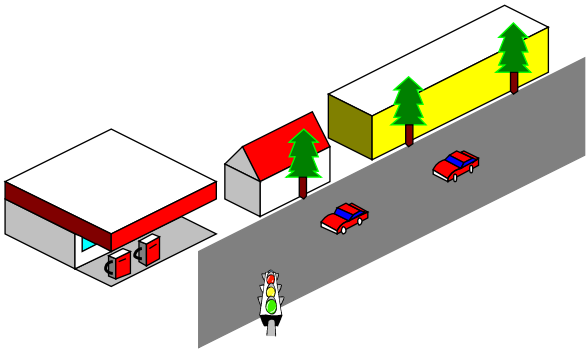
- Simpan program dengan nama perc3-2.asm
- Compile program.
- Apabila masih terdapat error buka dan lihat perc3-2.lst untuk mengetahui error secara detail
- Setelah tidak ada error download program dengan menggunakan program dt51win



Percobaan 3 Stupid Traffic light controller

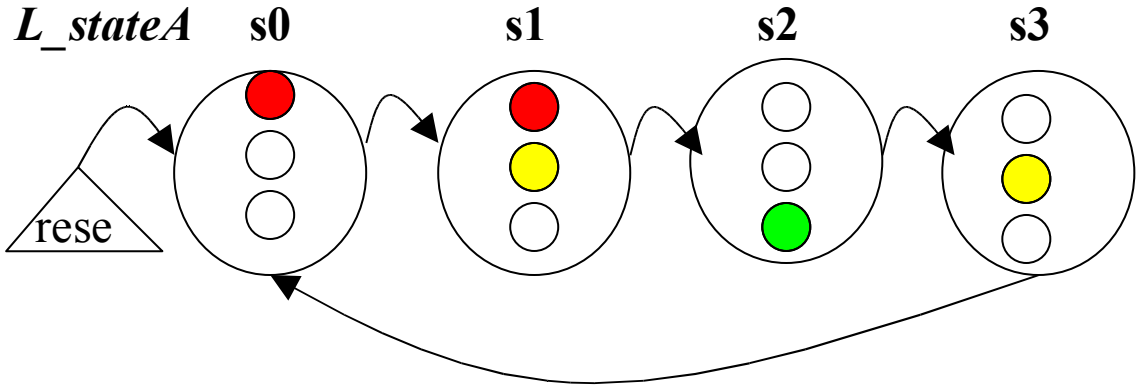
Pada percobaan ke tiga anda mendesain traffic light controller yang mempuyai pattern sebagai berikut:

I. Merah (2dt)
II. merah kuning(1dt)
III. hijau (3dt)
IV. kuning(1dt)



Gambar 3-13

Untuk melihat aliran program secara detail dapat digunakan state diagram pada gambar 3-14. Pada saat reset state awal lampu yang nyala adalah merah kemudian diikuti lampu merah+kuning seperti yang ditunjukkan pada gambar 3-14



Gambar 3-14

- Buka file baru
- Pastikan alamat awal program berada di 4000h
- Ketik program dibawah ini

```
$mod51
;  traffic2 - Traffic light controller pattern version v1.1.1
; case studies 1
;
;
;  akhmad hendriawan
;  sabtu 27 Juni 2004
;
;  Hardware Notes:
; output active rendah

Hijau  equ  P1.2
Kuning equ P1.1
Merah  equ P1.0
;merah(s0) -> merah-kuning(s1) -> hijau(s2) -> kuning(s3) ->
;merah(s0): 4 states
;
code_seg equ 4000h
state    equ 20h
vdly     equ 21h
dthigh   equ 30h
dtlow    equ 31h
org code_seg
    ljmp start
org code_seg+100h
;prosedure delay 1 detik
delay:
      push vdly
lfirst:
      mov dthigh,#10
loopdtlow:
      mov dtlow,#100
loopms:
      mov tl0,#67
      mov th0,#0fch
      setb tr0      ;timer start
```

```
here: jbc tf0,loop1
      sjmp here
loop1:
      djnz dtlow,loopms
      djnz dthigh,loopdtlow
      clr tr0
      djnz vdly,lfirst
      pop vdly
      ret

start:
; inisisalisasi
     mov tmod,#01
     mov tcon,00h

s0:
     setb merah
     clr  kuning
     clr hijau
     mov vdly,#2
     lcall delay
     sjmp s1
s1:
     setb merah
     setb kuning
     clr  hijau
     mov vdly,#1
     lcall delay
     sjmp s2
s2:
     clr   merah
     clr  kuning
     setb  hijau
     mov vdly,#3
     lcall delay
     sjmp s3
s3:
     clr   merah
     setb  kuning
     clr  hijau
```

```
      mov  vdly,#1
      lcall delay
      sjmp s0

end;
```

- Simpan program dengan nama perc3-3.asm
- Compile program.
- Apabila masih terdapat error buka dan lihat perc3-3.lst untuk mengetahui error secara detail
- Setelah tidak ada error download program dengan menggunakan program dt51win

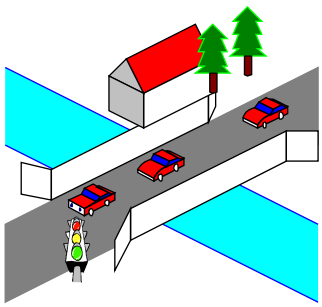
penyimpanan nama lain harus mengikuti kaidah penulisan standard dos baik direktory maupun nama file, karena kompiler metalink hanya mengenal folder dan nama file yang berformat dos

- Pastikan posisi jumper berada pada 1 dan 2

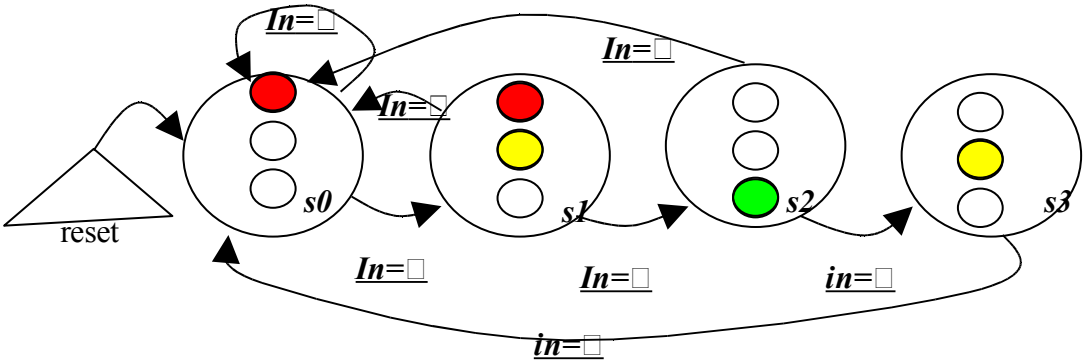
Percobaan 4 simple Traffic light controller

Disain Stupid Traffic light controller yang mempunyai pattern sebagai berikut

- Merah (2dt)
- merah kuning(1dt)
- hijau (3dt)
- kuning(1dt)



Jika anda menekan tombol pada traffic light, lampu akan merah pada state berikutnya



Device	Port
Lampu merah	P1.0
Lampu kuning	P1.1
Lampu Hijau	P1.2
Tombol	Port C.0

```
;definisi port ppi
porta equ 2000h
portb equ 2001h
portc equ 2002h
cw equ 2003h
datacw equ %10001001

;definisi lampu traffic light
```

```
Hijau equ P1.2
Kuning equ P1.1
Merah equ P1.0 ;merah(s0) -> merah-kuning(s1) -> hijau(s2) -> kuning(s3) ->
;merah(s0): 4 states
 code_seg equ 4000h
state equ 20h
vdly equ 21h
dthigh equ 30h
dtlow equ 31h
org code_seg
    ljmp start
org code_seg+100h ;prosedure delay 1 detik
delay:
      push vdly
lfirst:
       mov dthigh,#10
loopdtlow:
          mov dtlow,#100
loopms:
       mov tl0,#67
       mov th0,#0fch
       setb tr0 ;timer start
here: jbc tf0,loop1
      sjmp here
loop1:
      djnz dtlow,loopms
      djnz dthigh,loopdtlow
      clr tr0
      djnz vdly,lfirst
      pop vdly
      ret

reset:
     mov dptr,#portC
     movx a,@dptr
     anl  a,#01h
     pop dph
     pop dpl
     cjne a,#01h,salah
     jmp  s0
salah:
     clr A
     jmp @a+dptr
     ret

start: ; inisisalisasi
       mov tmod,#01
       mov tcon,00h

       mov dptr,#cw     ;inisialisasi ppi
       mov a,#datacw
       movx @dptr,a

s0:
    setb merah
    clr kuning
    clr hijau
    mov vdly,#2
    lcall delay
    lcall reset
    sjmp s1
```

```
s1:
    setb merah
    setb kuning
    clr hijau
    mov vdly,#1
    lcall delay
    lcall reset
    sjmp s2
s2:
    clr merah
    clr kuning
    setb hijau
    mov vdly,#3
    lcall delay
    lcall reset
    sjmp s3
s3:
    clr merah
    setb kuning
    clr hijau
    mov vdly,#1
    lcall delay
    lcall reset
    sjmp s0
end;
```

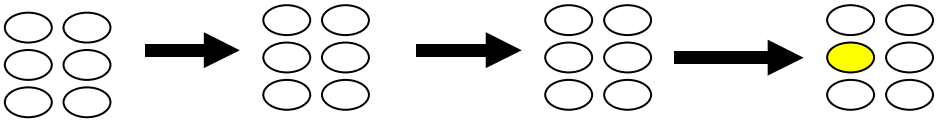
VII. Laporan sementara

Tugas berikut dikerjakan sebagai laporan sementara

1. Traffic light simpang 4

Desain traffic light controller pada persimpangan 4 dimana pattern utara = selatan, pattern barat=timur

State	North-South	East-West	Waktu
S0	Merah	Hijau	3
S1	Merah-kuning	Hijau	1
S2	Hijau	Merah	3
S3	Kuning	Merah-Kuning	1



Device	North-South	East-West
Lampu merah	P1.0	P1.3
Lampu kuning	P1.1	P1.4
Lampu Hijau	P1.2	P1.5

VIII. Laporan resmi

Analisa tugas yang diberikan pada laporan sementarat

IX. Tambahan

Berikan saran atau komentar guna pengembangan lebih lanjut praktikum ini.

Bab 4

Pemanfaatan komunikasi serial

I. Tujuan

Setelah Melakukan praktikum ini yang anda peroleh adalah:

- Dapat menjelaskan register yang digunakan untuk keperluan komunikasi serial
- Dapat mengimplementasikan fungsi komunikasi serial RS-232 pada mikrokontroller dan mengirim datanya ke PC.
- Dapat mengimplementasikan program pooling dan program interrupsi pada komunikas serial RS-232

II. Pendahuluan

Pada praktikum ini, anda akan mencoba memanfaatkan komunikasi serial pada mikrokontroller atmel 89C51 yang merupakan variant dari mikrokontroller MCS-51(8051). Pada praktikum sebelumnya anda mengimplementasikan counter/timer sebagai delay, kemudian mencoba mengimplementasikannya sebagai lite function generator diakhiri dengan implementasi traffic light controller menggunakan delay hardware. Sekarang anda akan mencoba memanfatkan delay hardware dalam pembuatan generator baud rate untuk keperluan serial

III. Gambaran desain

Komunikasi antara komputer dengan 8051 dapat terjadi dengan dua metode, yaitu metode sikron dan metode asinkron. Pada metode sinkron terdapat satu pin yang digunakan untuk mensinkronkan kecepatan pengiriman data antara dua device. Umumnya digunakan untuk

mensikronkan data. Pada metode asinkron tidak ada clock untuk mensinkronkan data. Salah satu cara untuk mensinkronkan data pada komunikasi serial adalah mengirimkan data pada kecepatan yang sama. Hal ini dimungkinkan dengan cara masing –masing device mengirimkan data dengan baudrate yang sama.

Format data pada komunikasi serial sangat penting. Selain menjamin data pada device pengirim(transmitter) dimengerti oleh device penerima (reciever), format data juga menjamin bahwa error akibat perjalanan data bisa diketahui oleh penerima.

IV. Teori Dasar

Mikrokontroller 8051 mempunyai fasilitas serial komunikasi diantaranya:

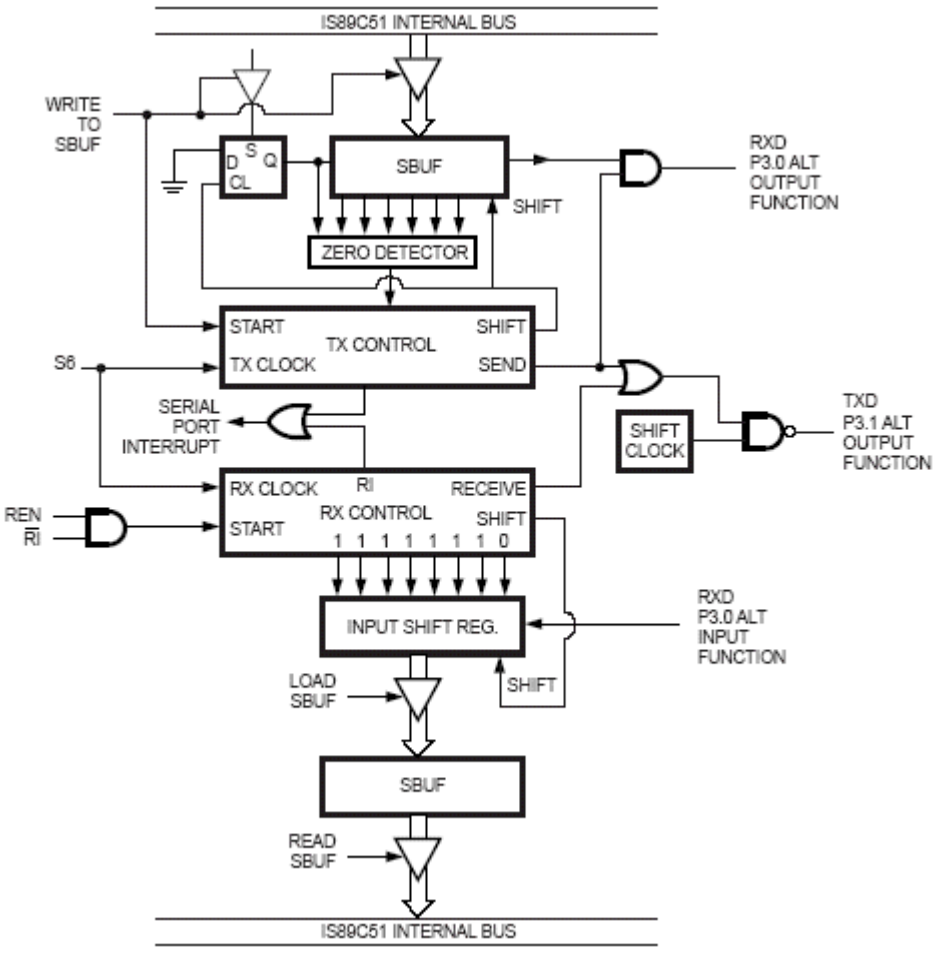
- Full duplex UART.
- Empat mode dari operation:
    - Synchronous serial I/O expansion.
    - Asynchronous serial I/O dengan variable baud rate.
    - Mode dengan data 9 bit dengan variable baud rate.
    - Mode dengan data 9 bit dengan fixed baud rate.
- 10 atau11 bit frames.
- Interrupt enable atau metode pooling.

Komunikasi data serial pada 8051 dapat dilakukan dengan cara menghubungkan pin TXD dan pin RXD dari 8051 ke device serial lainnya. TXD dan RXD adalah output serial dan input pin (Port 3, bits 1 and 0).

Komunikasi serial pada 8051 ditangani oleh beberapa register. Register SCON (Serial control register) digunakan untuk menentukan mode komunikasi serial, register SBUF digunakan untuk menerima dan mengirim data serial, register PCON digunakan untuk mengubah setting baudrate standar dengan cara melipatkan gandakan dua kali, register TMOD dan register TCON digunakan sebagai pembangkit baudrate (baudrate generator).

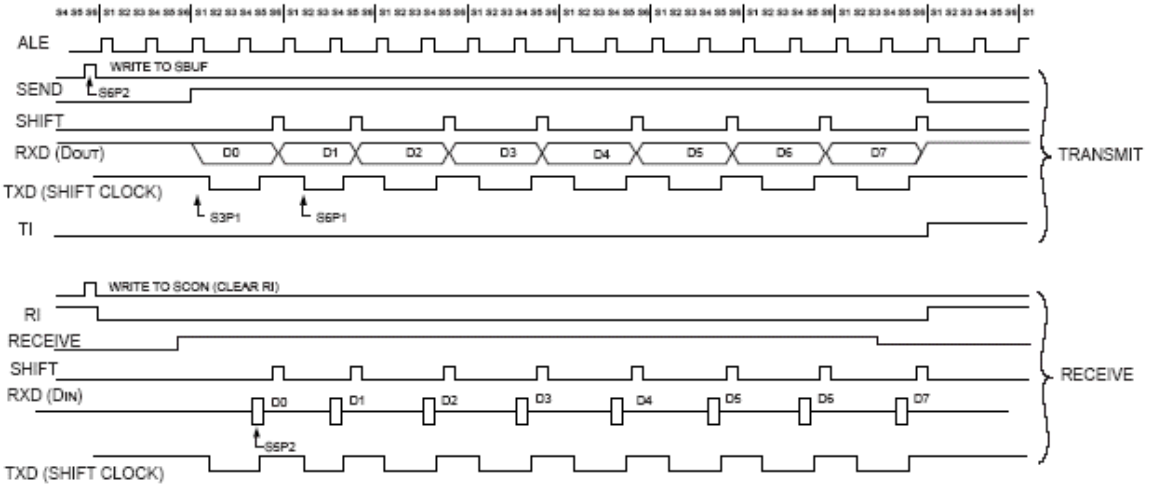
Komunikasi serial pada 8051 dapat dilakukan dengan 4 mode yaitu mode 0, mode 1, mode 2 dan mode 3.

Pada mode 0 data serial dikirim dan diterima oleh pin RXD. TXD output shift clock. Baudrate adalah 1/12 dari frekuensi clock. Gambar 4-1 akan memperjelas pengertian ini.



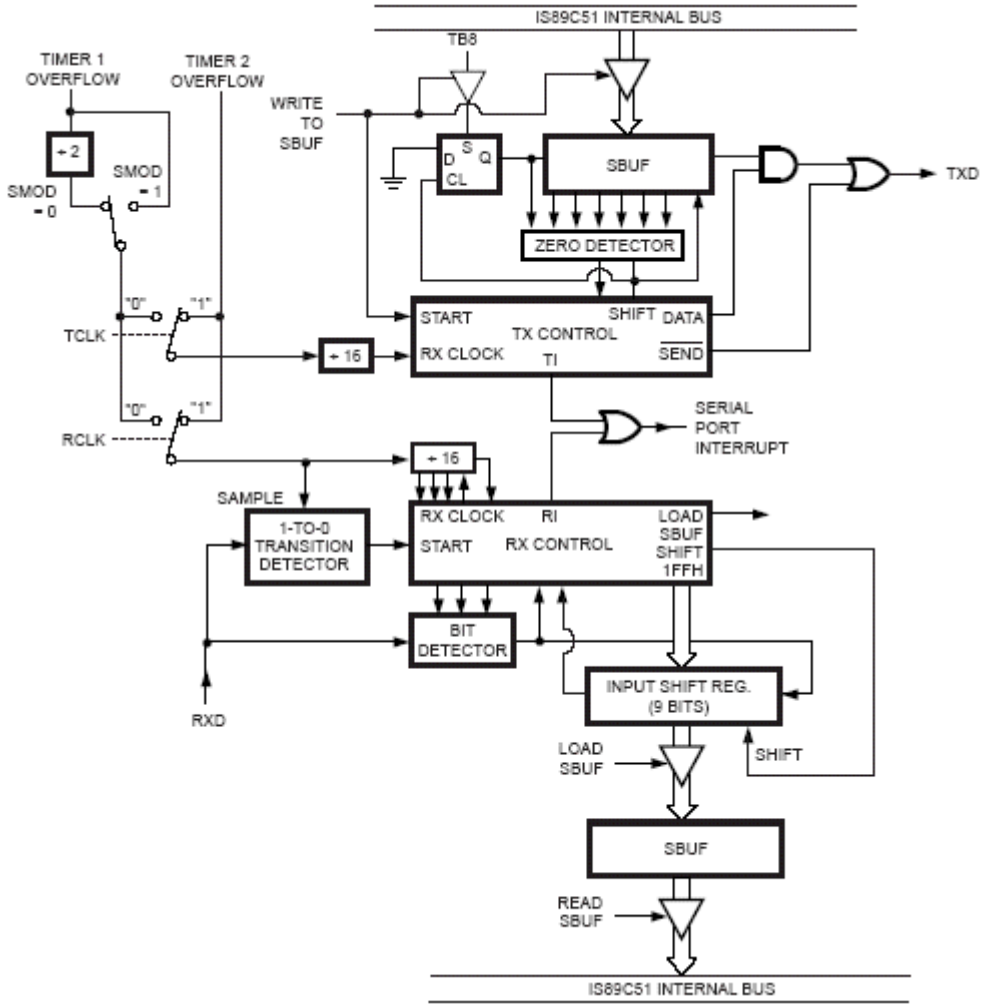
Gambar 3-1 blok diagram untuk mode 0

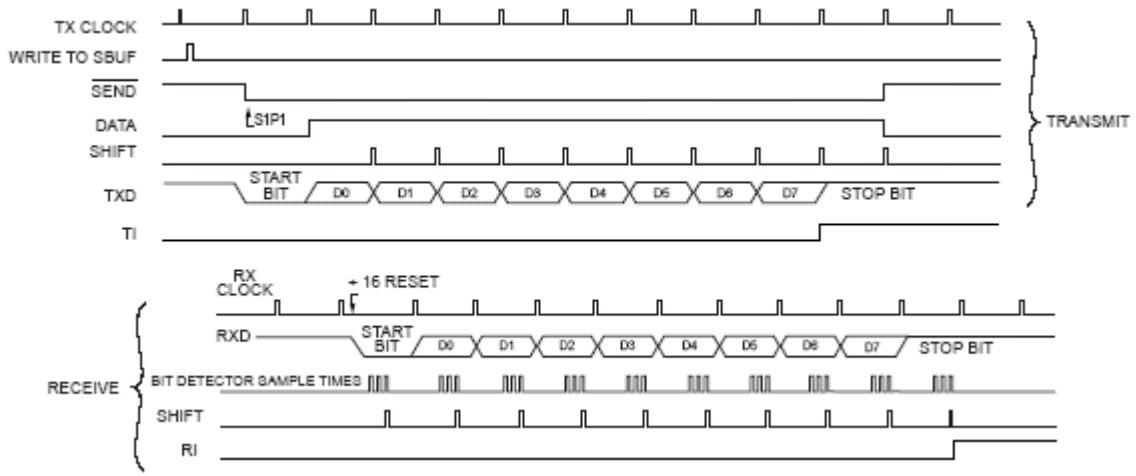
Pada gambar 3-1 pin RXD mempunyai dua fungsi, yaitu sebagai input dan output. Sementara pin TXD pada mode ini tidak digunakan untuk data output melainkan digunakan untuk clock sinkronisasi. Data dapat diterima apabila pin Ri clearkan terlebih dahulu. Pin ini di-*clear* oleh user tetapi diset oleh hardware ketika ada data masuk. Timing diagram komunikasi serial mikrokontroller 8051 pada mode 0 ditunjukkan pada gambar 3-2.



gambar 3-2 timing diagram mode 0

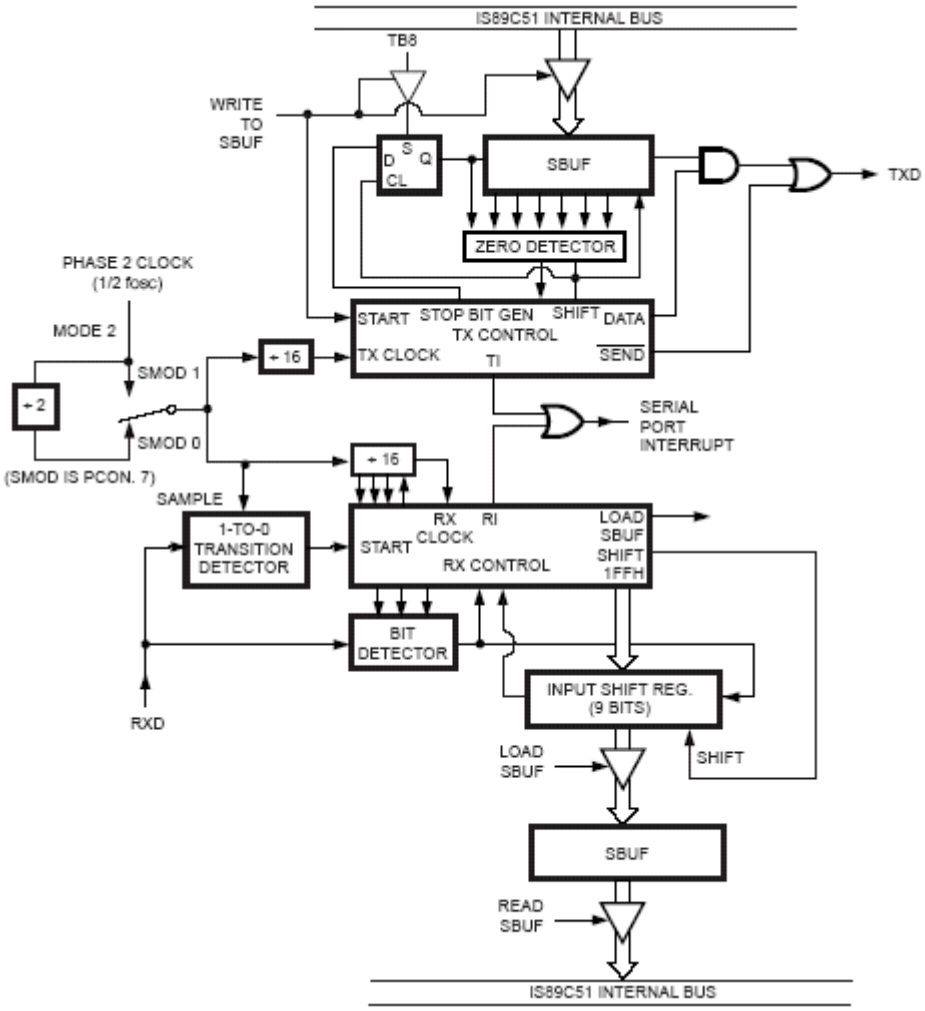
Pada mode 1 data dikirim atau diterima sebesar 10 bit yang terdiri dari start (bit 0), 8 bit data yang dimulai dari LSB dan 1 stop bit. Baud Rate Clock menggunakan variable Timer 1 overflow atau hitungan input luar (Gambar 3-3)

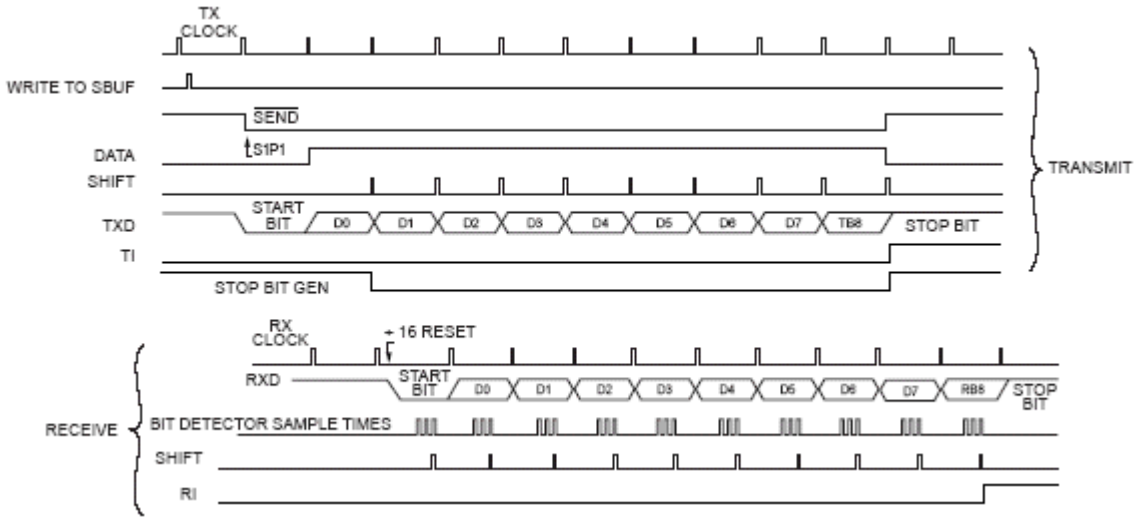




gambar 3-3

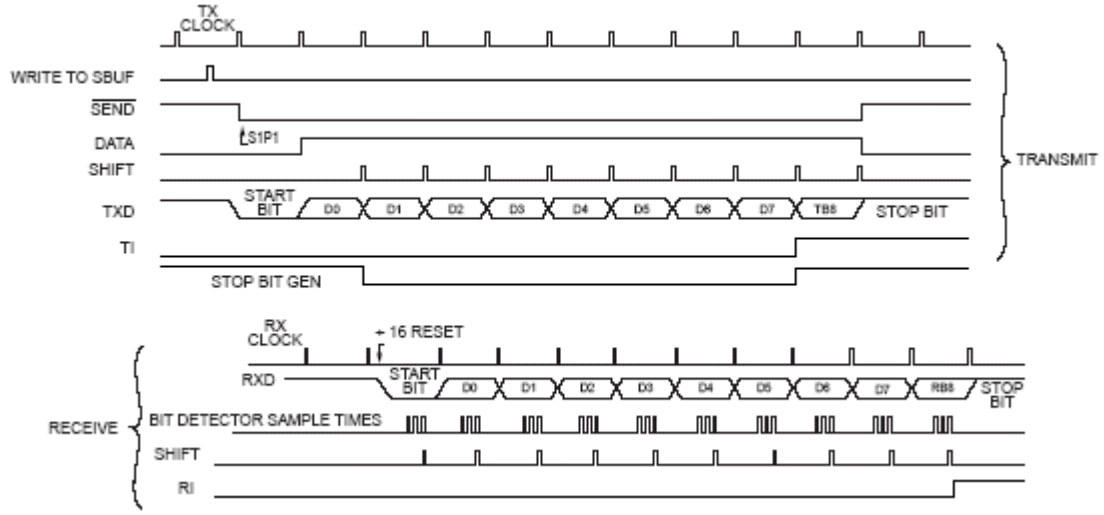
Pada mode 2 data dikirim atau diterima sebesar 11 bit dengan start bit sebesar 1 bit (bit ke 0), data sebesar 8 bit yang dimulai LSB, bit ke 9 bisa diprogram serta 1 stop bit. Baudrate dapat diprogram 1/32 atau 1/64 osc frekuensi (gambar 3-4 dan 3-5)



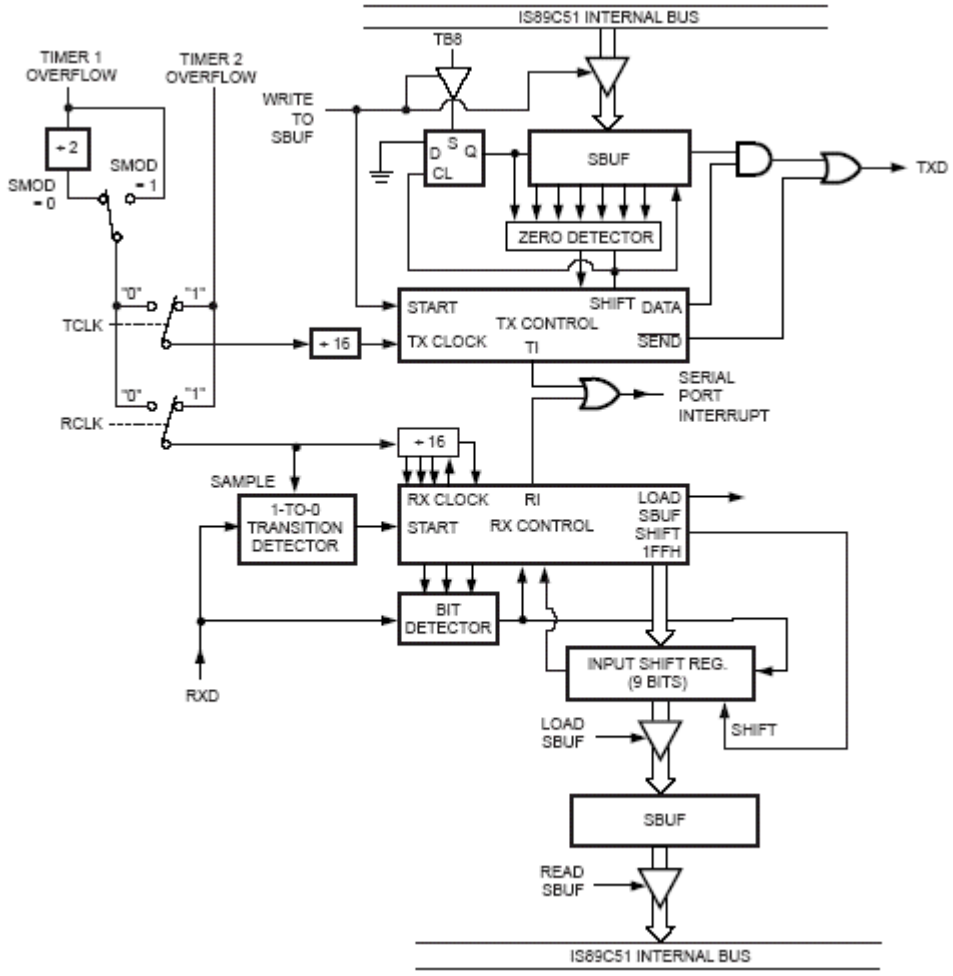


gambar 3-4

Pada mode 3 data sebesar sebesar 11 bit untuk transmit atau receiver yang terdiri dari start bit (bit ke 0), data sebesar 8 bit(mulai dari LSB), 1 bit yang dapat diprogram dan 1 stop bit. Mode 3 sama dengan mode 2 kecuali baudrate (gambar 3-5 dan gambar 3-6).



Gambar 3-5 timing diagram mode 3



gambar 3-6 blok diagram komunikasi serial mode 3

$$BaudRateMode0 = \frac{FrekeunsiOsilator}{12}$$

$$BaudrateMode2 = \frac{2^{S\,\mathrm{mod}}}{64} xFrekuensiOsilator$$

$$BaudrateMode1,3 = \frac{2^{S\,\mathrm{mod}}}{32} xTimer1OverflowRate$$

$$BaudrateMode1,3 = \frac{2^{S\,\mathrm{mod}}}{32} x \frac{FrekuensiOsilator}{12x[256 - TH1]}$$

V. Peralatan yang dibutuhkan:

1. Modul board DT 51

2. Simulator MCS-51 (UMPS)
3. Cross Assembler MCS-51 (UMPS)
4. IBM PC kompatibel
5. LED modul.

VII. Prosedur percobaan

Percobaan 1- Simple Serial Communication

1. ketik program dibawah ini
2. Buka file baru
3. Pastikan alamat awal program berada di 4000h
4. Ketik program dibawah ini

```
;*****************************************************
;MCS-51 program serial                                **
;author    : fpga_core                                **
;            serial mode 1 (standar uart)             **
;mode    : serial                                     **
;baudrate  : 9600 N 8 1                               **
;*****************************************************
$mod51
baudrate  equ     0fDh
code_seg equ      4000h

 org code_seg
       ljmp  start

;********************
;procedure init
;********************
init:
      mov scon,#50h
      mov th1,#baudrate
      anl pcon,#7fh            ;set 1x baudrate  9600
      mov tmod,#21h             ;timer1 set mode2, timer0 set mode1
      mov tcon,#40h             ;enable timer1 disable timer0
      ret
;---------------------------------------------------------------------
;main program
;---------------------------------------------------------------------
```

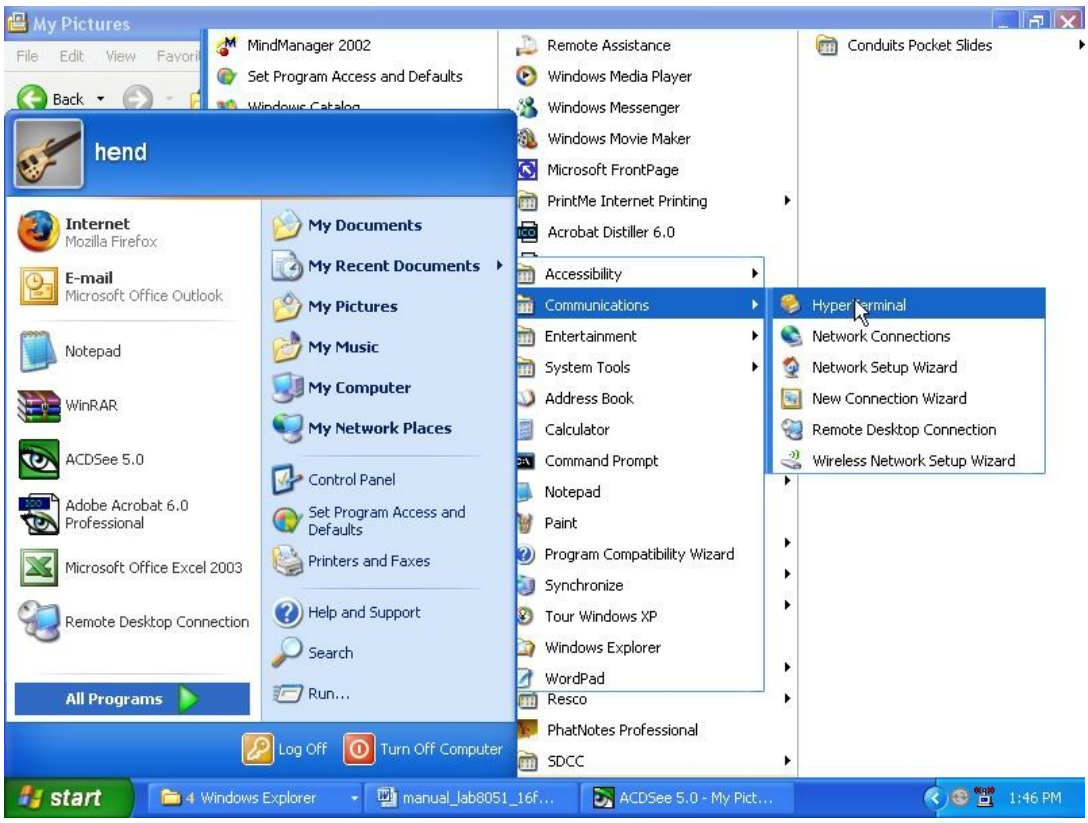
```
start:      mov sp,#10h              ;sp=10h
      lcall init               ;inisialisasi
loop: mov a,#'a'
      mov sbuf,a
wait: jnb scon.1,wait
      clr scon.1
      sjmp  loop               ;jika ya loop
      end
```

5. simpan dengan nama p4-1.asm
6. download data ke modul board 8951
7. Compile program.
8. Apabila masih terdapat error buka dan lihat perc3-2.lst untuk mengetahui error secara detail
9. Setelah tidak ada error download program dengan menggunakan program dt51win

penyimpanan nama lain harus mengikuti kaidah penulisan standard dos baik direktory maupun nama file, karena kompiler metalink hanya mengenal folder dan nama file yang berformat dos

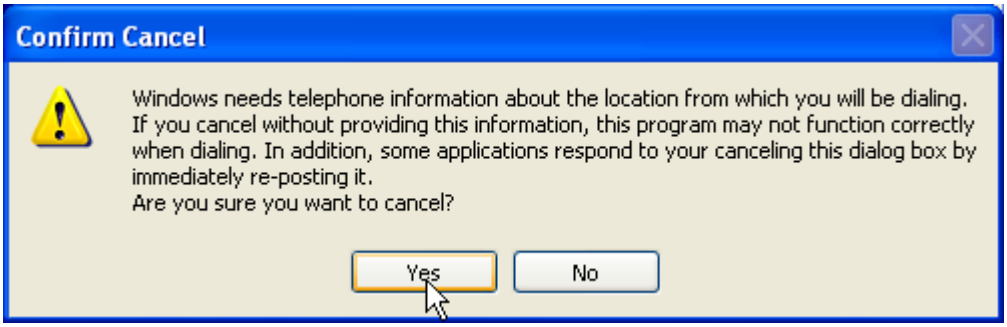
- Pastikan posisi jumper berada pada 1 dan 2

10. Jalankan hyperterminal dengan cara pada startmenu klik pada bagian accessories, klik communication selanjutnya klik hyperterminal. Ini ditunjukkan pada gambar 3-7



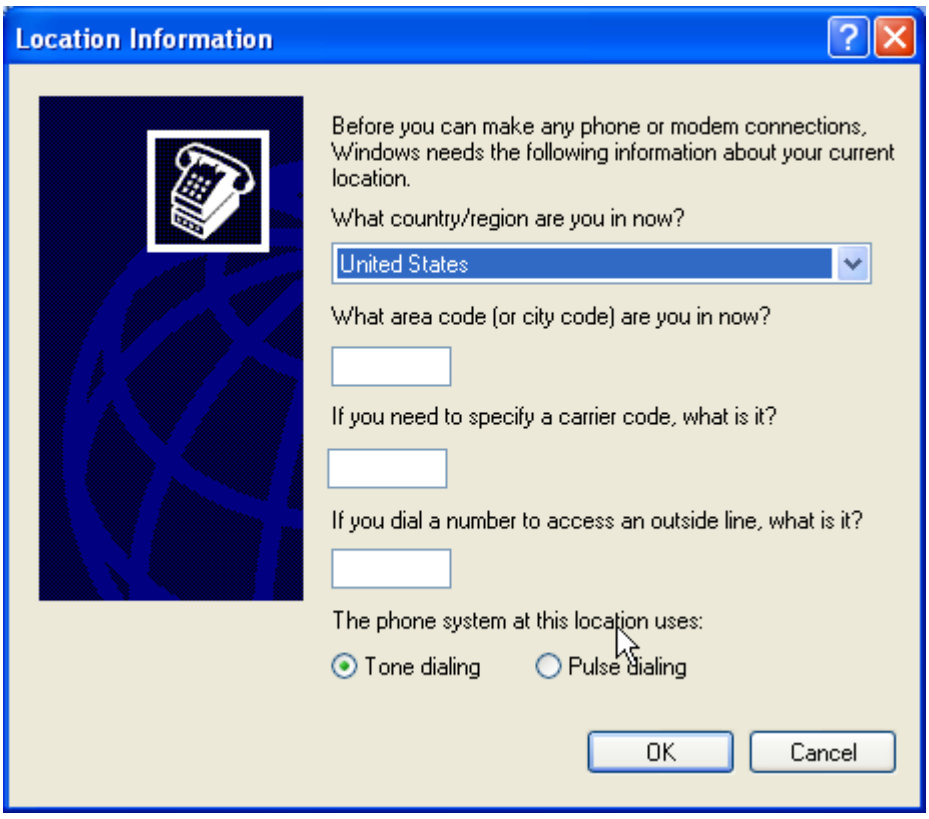
Gambar 3-7

11. Apabila terdapat pertanyaan seperti pada gambar 3-8 ini maka pilih no



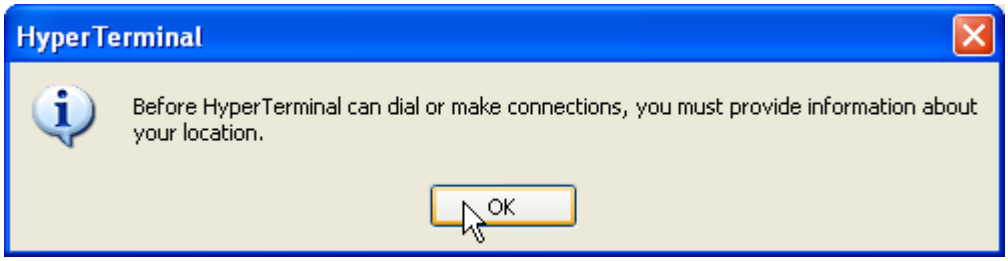
Gambar 3-8

12. Selanjutnya pilihan untuk modem, karena kita tidak menggunakan modem, maka pilih cancel (gambar 3-9).


Location Information
Before you can make any phone or modem connections, Windows needs the following information about your current location.
What country/region are you in now?
United States
What area code (or city code) are you in now?
If you need to specify a carrier code, what is it?
If you dial a number to access an outside line, what is it?
The phone system at this location uses:
Tone dialing
Pulse dialing
OK
Cancel

gambar 3-9

13. pilih ok untuk konfirmasi.



gambar 3-10

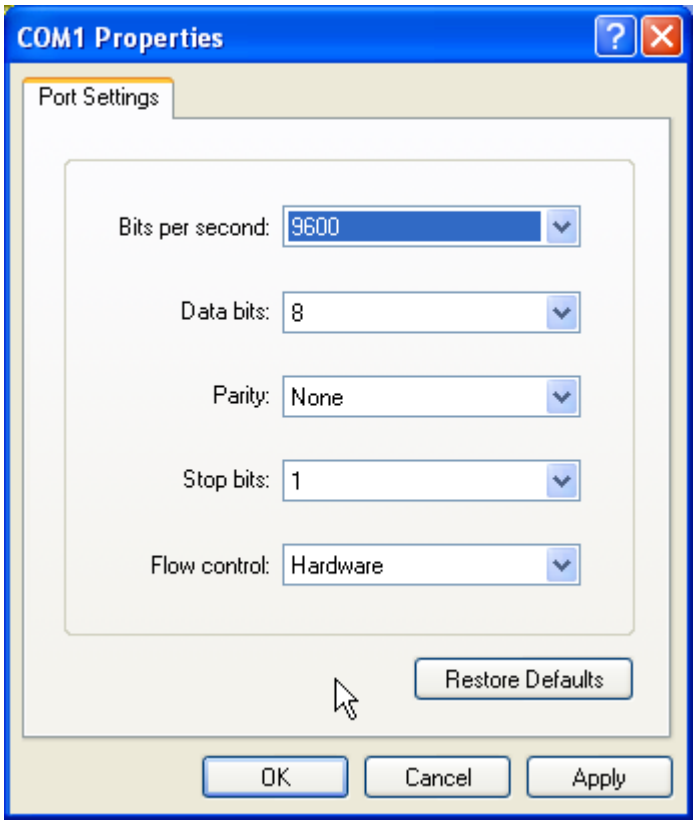
14. Ketik nama koneksi yang diinginkan.(gambar 3-11).

gambar 3-11

15. Langkah selanjutnya adalah pemilihan device serial untuk. Device serial untuk komputer biasanya terdiri dari dua. Device ini dinamakan COM1 dan COM1. Apabila device anda terhubung ke COM1 pilih COM1 (gambar 3-12).

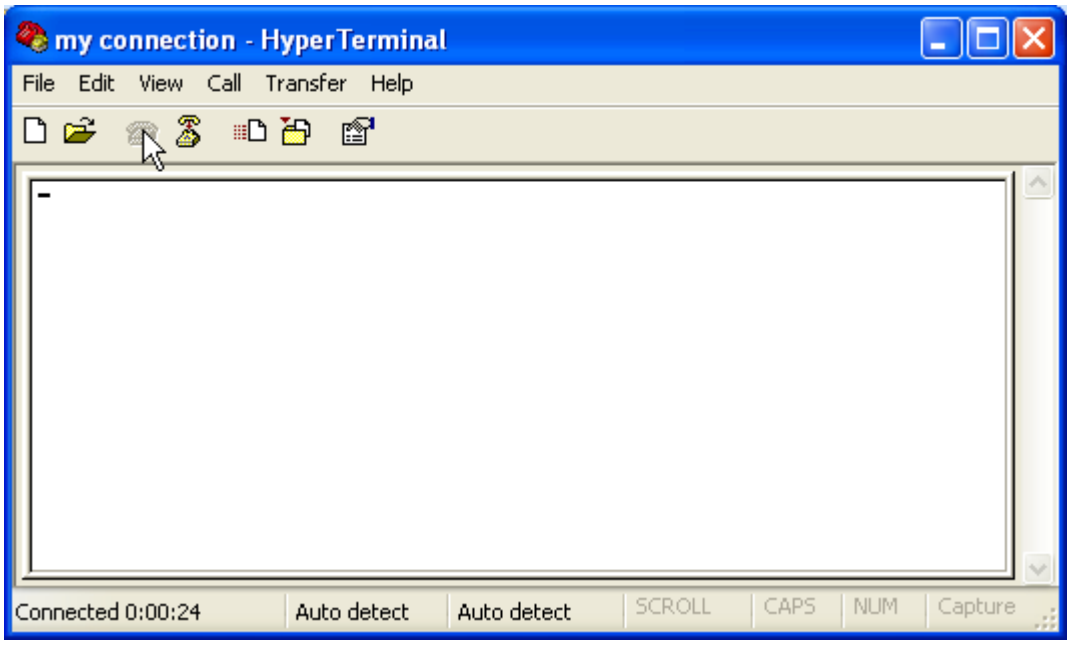
Gambar 3-12

16. Baudrate digunakan untuk sinkronisasi kecepatan. Kecepatan pengiriman dan penerimaan harus sama. Pada percobaan pertama(lihat listring program) adalah 9600 dengan data sebesar 8 bit dan 1 stop bit tanpa parity. Ubah port setting sesuai dengan program anda (gambar 3-13). Setelah itu klik OK.



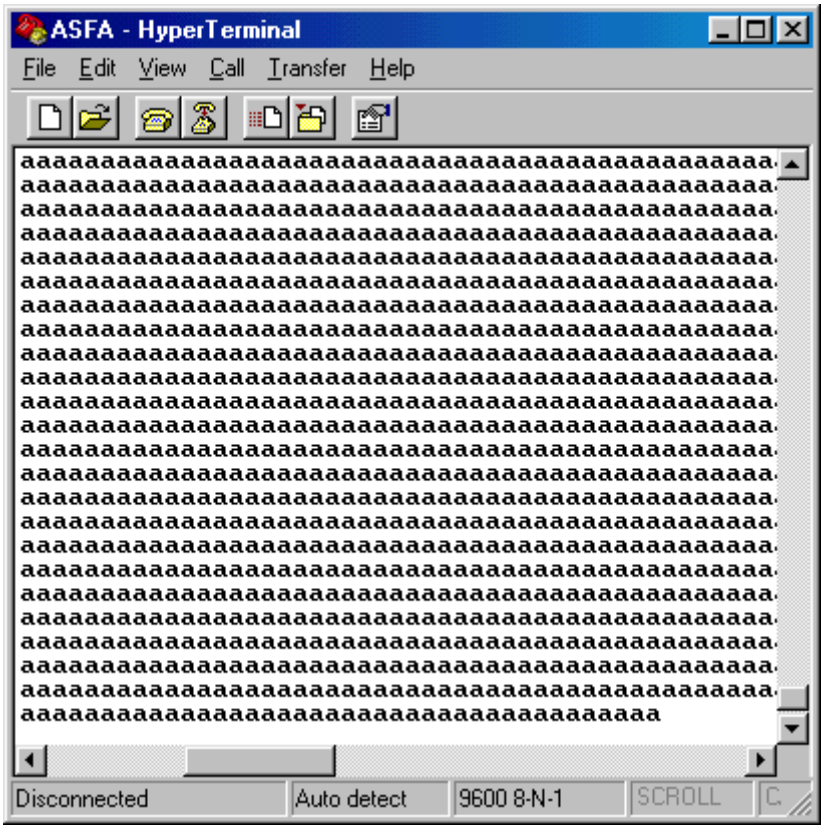
Gambar 3-13

17. Apabila sudah selesai maka akan terlihat seperti gambar 3-14. icon dibawah digunakan untuk melakukan koneksi ke device, sementara icon disebelah kanan digunakan untuk melakukan menonaktifkan komunikasi serial dengan device.



Gambar 3-14

18. Ubah posisi jumper dari 1&2 ke 2&3.
19. restart modul 8951. lakukan secara hardreset
20. display terminal akan bernilai ascii a sesuai dengan data yang kita kirim. Tampilan seperti gambar 3-15



gambar 3-15

Percobaan 2- Advance Serial Communication

1. ketik program dibawah ini
2. Buka file baru
3. Pastikan alamat awal program berada di 4000h
4. Ketik program dibawah ini
5. Ketikkan program dibawah ini

```
;*****************************************************************
;MCS-51 program serial
;ABSTRACT :
; program ini digunakan untuk komunikasi antara mikrokontroller
; dengan komputer IBMPC kompatibel menggunakan metode full duplex
; bank 0 digunakan sebagai buffer penerimaan data sedang
; bank1 digunakan sebagai buffer pengiriman data
; timer 1 mode 2 (auto reload) digunakan untuk pembangkitan
; baudrate untuk serial komunikasi
;
;program name: easytest
;Version : 1.2
```

```
;date created: 27 SEPT 2004
;AUTHOR : HENDRI
;Xcompiler : Metalink51
;systemboard : dt51 ver3
;mode serial : mode 1
;baudrate : 9600 nonparity 8 bit non parity 1 stopbit
;*****************************************************************
;----------------------------------------
; mapping memori
;R1 \\\\\\\\\\\\\\\\\\6F fullbuff
;    \ Xmit -> bank 1 \
;R0 \\\\\\\\\\\\\\\\\\60
;
;
;R1 \\\\\\\\\\\\\\\\\\5F  fullbuf
;     \ RXD -> bank 0   \
;R0 \\\\\\\\\\\\\\\\\\30
;--------------------------------------
;*****************************************************************
; perhitungan baudrate
;Baud_Rate = Fosc/12 * (2^smod/32) / (256-TH1)
;where 2^smod = 2 for smod=1, and 1 for smod=0
;Fosc is oscillator frequency; and TH1 is timer 1 reload value.
;TH1 = 256 - (Fosc/12 * (2^smod/32) / Baud_Rate)
;
;Fosc = 12MHz 16MHz 20MHz 11.0592 14.7456 18.4320 smod
;Rate
;150  030H  --    --    040H    000H         --      0
;300  098H  075H  052H  0A0H    080H         060H    0
;600  0CCH  0BBH  0A9H  0D0H    0C0H         0B0H    0
;1200 0E6H  0DEH  0D5H  0E8H    0E0H         0D8H    0
;2400 0F3H  0EFH  0EAH  0F4H    0F0H         0ECH    0
;4800 *       *   0F5H  0FAH    0F8H         0F6H    0
;9600 --    --    *     0FDH    0FCH         0FBH    0
;19200      --      --  --      0FDH        0FCH         0FBH    1
;38400      --      --  --      --    0FEH         --    1
;76800      --      --  --      --    0FFH         --    1
;*****************************************************************
$mod51
TXFLAG       EQU TI
RXFLAG       EQU RI
```

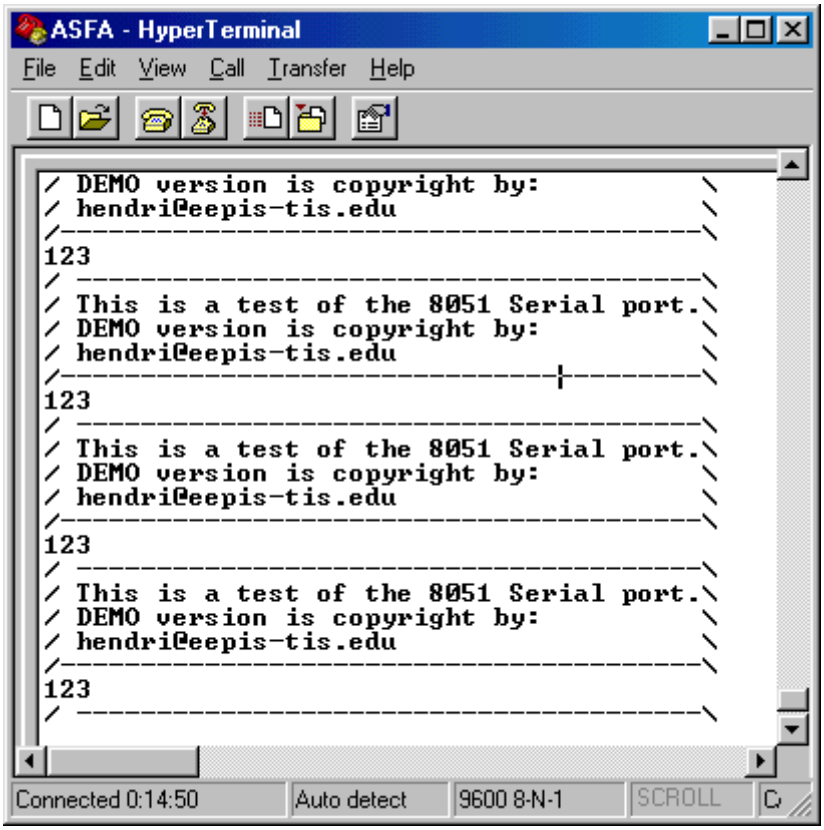
```
BAUDRATE EQU 0FDH ;9600
DT_XMIT  EQU 60H
TX_FULL  EQU 6FH
DT_READ  EQU 30H
RXDFULL  EQU 5fH
CODE_SEG EQU 4000H
INT_SEG  EQU 4023H
org CODE_SEG
      LJMP START
ORG INT_SEG
      LJMP INT_SRV

ORG CODE_SEG+100H
;-------------------------------------
;rutin interrupt serial
;-------------------------------------
INT_SRV:
  PUSH PSW               ;simpan status register
  PUSH ACC               ;dan accumulator
  JBC RXFLAG,RCVE        ;cek apakah int receive terjadi
  JBC TXFLAG,XMIT        ;cek apakah int transmit terjadi
  SJMP GO
RCVE:
  CLR PSW.3                   ;pilih bank 0
  MOV A,SBUF            ;ambil karakter
  MOV @R1,A             ;simpan ke buffer
  CJNE R1,#RXDFULL,ROK  ;deteksi apakah buffer full
  DEC R1                ;koreksi array
  ROK: INC R1           ;incr top address
  JBC TXFLAG,XMIT       ;cek apakah ada pengiriman
  SJMP GO               ;tidak, reti
XMIT:
  SETB PSW.3            ;ya, pilih bank 1
  MOV A,R0              ;reg a=pointer1
  CJNE A,09,MOR         ;uji apakah buffer kosong
  MOV R0,#DT_XMIT       ;ya, pointer 1 = bottom buffer
  MOV R1,#DT_XMIT       ;pointer1 = pointer2
  SJMP GO               ;reti
MOR:
  MOV A,@R0             ;ambil data yang dikirim
  MOV SBUF,A            ;mulai transmisi
```

```
  INC R0                ;point ke karakter selanjutnya
GO:
  POP ACC
  POP PSW
  RETI
;-------------------------------------------------------------
; optional procedure for
;=======
;
;STROUT
; Copy in-line character string to console output device.
; uses: DPTR,ACC
;
STROUT:
  POP DPH ;get in-line string address from stack
  POP DPL
STRO_1:
  CLR A
  MOVC A,@A+DPTR         ;baca byte berikutnya
  CJNE A,#'$',LOOPST
  JMP ENDSTR
LOOPST:
  MOV @R1,A
  INC R1
  INC DPTR              ;Bump pointer.
  SETB TI
HERE:
  CJNE R1,#DT_XMIT,HERE
  SJMP STRO_1
endstr:
  INC DPTR
  CLR A
  JMP @A+DPTR           ;Return to program.
;---------------------------------------------------------------------
;main program
;---------------------------------------------------------------------
START:
;-----------inisialisasi serial komuikasi
  MOV SP,#10H           ; top stack 1
  MOV R0,#DT_READ       ; inisialisasi bank 0
  MOV R1,#DT_READ       ; untuk menerima data
  SETB PSW.3            ; pilih bank1
```

6. simpan program dengan nama int_ser.asm

STOP	penyimpanan nama lain harus mengikuti kaidah penulisan standard dos baik direktory maupun nama file, karena kompiler metalink hanya mengenal folder dan nama file yang berformat dos

7. compile program
    - bila sudah tidak terdapat kesalahan download program ke board 8951. sebelumnya terlebih dahulu posisi jumper harus berada pada 1 dan 2 .
8. ubah kembali posisi jumper ke posisi 2 dan 3.
9. jalankan program hyperterminal
10. hardreset mikro (matikan kemudian hidupkan)
11. tampilan awal akan terlihat seperti gambar 3-16



gambar 3-16.

VII. Laporan sementara

Tugas berikut dikerjakan sebagai laporan sementara

Modifikasi program yang sudah anda coba, untuk mengirimkan data ke komputer dengan ketentuan:
Data yang dikirim ke komputer adalah nama kelompok anda dan anggota kelompok, data yang diterima komputer adalah berupa penekanan tombol keyboard untuk tombol numerik [0-9]. Bila di keyboard komputer ditekan 0 maka led di board mikro mati semua yang menyatakan kode binernya.

VIII.Laporan resmi

Sebagai analisa dari laporan resmi, gambarkan dan jelaskan diagram alir dari tugas yang anda kerjakan. Berilah keterangan pada program yang telah dibuat

IX. Tambahan

Berikan saran atau komentar guna pengembangan lebih lanjut praktikum ini

Bab 5

Aplikasi PWM

I. Tujuan

Setelah Melakukan praktikum ini yang anda peroleh adalah:

- Dapat memanfaatkan fasilitas Interupsi dalam MCS-51
- Dapat menggunakan pwm untuk Mengatur, nyala lampu led dan mengatur kecepatan motor dc

II. Pendahuluan

Pada praktikum ini anda akan mempelajari cara mengembangkan suatu embbedded controller digunakan sebagai kontroller motor. Penggunaan teknik programing suatu hal yang menarik untuk diperlajari pada praktikum ini. pada Pada praktikum sebelumnya anda telah mempelajari penggunaan timer dan counter untuk delay, pembuatan traffic light controller. Pada praktikum ini anda akan mempelajari cara pemrograman menggunakan teknik pooling dan penggunaan teknik interupsi untuk pwm controller berbasis delay hardware.

III. Gambaran Desain

Anda membuat project baru dengan dengan menggunakan crimson editor, memulai program pada alamat 4000h dan memulai melakukan pemrograman dengan menggunakan dua pendekatan. Pada percobaan pertama anda akan mempelajari implementasi pwm controller dengan menggunakan metode pooling. Satu delay digunakan untuk mengatur tlow delay yang lain digunakan untuk mengatur thigh. Apabila delay thigh dinaikkan maka delay dari tlow harus

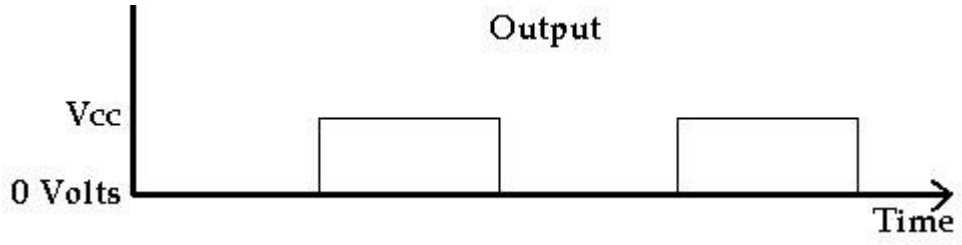
diturunkan demikian juga sebaliknya. Pengaturan duty cyle pwm digunakan dua register, yaitu register R3 dan register R4. R4 naik maka register R3 harus diturun dan sebaliknya R4 turun maka R3 harus naik.

Pada percobaan ke dua pengaturan duty cyle digunakan satu buah register (R7). PWM pada percobaan ke dua ini bersifat timer independen, artinya timer hardware digunakan untuk mengatur delay sementara prossesor tetap dapat digunakan untuk fungsi yang lain, timer hanya akan menginterupsi prosessor apabila timer selesai menghitung.

IV. Dasar Teori

Sangatlah mudah untuk mengontrol nyala led, tetapi kita hanya bisa mematikan dan menghidupkannya. Tetapi bagaimana apabila kita ingin mengatur brightnes dari LED ? masalah sama akan terjadi jika kita ingin dalam bidang robotic jika kita ingin mengatur kecepatan motor dengan mikrokontroller. Tidaklah cukup hanya dengan menghidupkan dan menyalakan motor. Untuk mengontrol nyala led dan mengatur kecepatan motor dibutuhkan cara untuk mengontrol arus yang melalui perangkat tersebut. Tetapi bagaimana ?

Salah satu solusi adalah membuat LED atau motor hidup dan mati dengan sangat cepat arus hanya akan mengalir apabila output dari mikrokontroller berlogic low (bila mikrokontroller aktif low). Output dari mikrokontroller akan seperti gambar 5-1

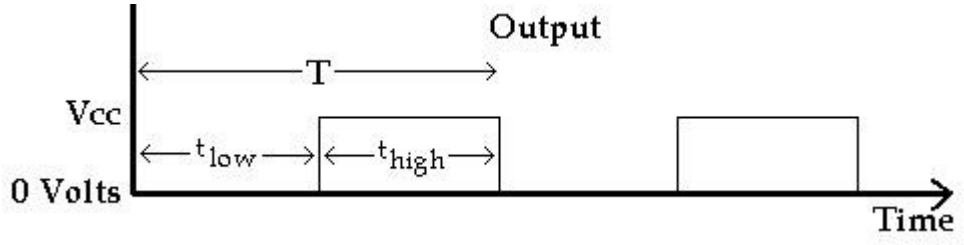


gambar 5-1

Jika kita hidupkan dan mematikan LED atau motor cukup cepat, akan terlihat LED tetap nyala atau motor tetap berputar. Jika kita mencoba mengatur mati dan hidup selama 20 kali perdetik akan terlihat LED berkedip atau pada motor kerjanya tidak halus. Solusi ini tidak bagus sebab LED masih terlihat berkedip bila kita mecoba untuk mengatur brighness lebih rendah.

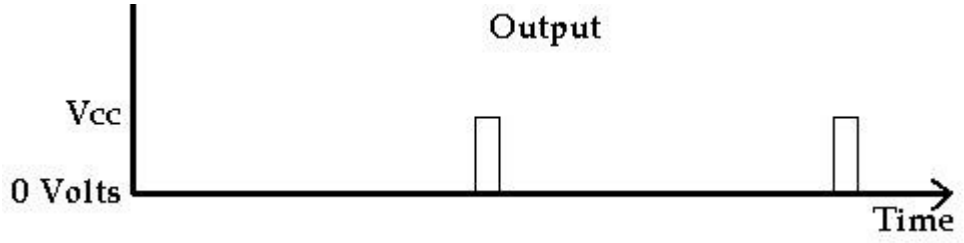
Solusi yang lain adalah mengatu berapa lama waktu on dan off. Kalau kita lihat output mikrokontroller ada dua periode waktu yang penting yaitu periode high (thigh) dan periode low

(tlow). Thigh +tlow =T. T adalah periode keseluruhan dari output. Thigh biasanya disebut pulsa output atau pulsa saja



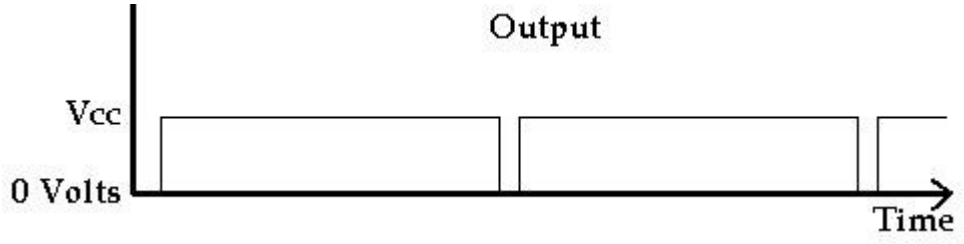
gambar 5-2

Kita harus menjaga agar periode mempunyai lebar yang sama dalam satu siklus. Jika kita melebarkan thigh maka tlow harus dikecilkan. Jika kita mengecilkan thigh maka tlow harus dibesarkan. Apabila thigh kecil output seperti pada gambar 5-3



gambar 5-3

Apabila thigh besar maka output seperti pada gambar 5-4



Gambar 5-4

Output pada gambar 5-4 adalah jika kita mematikan LED (asumsikan LED aktif LOW). Arus hanya mengalir melalui LED pada waktu yang sinkat selama kita mematikan LED. Tetapi kita mematikan dan menghidupkan LED dengan sangat cepat (sekitar 100 kali perdetik.) sehingga kita tidak dapat melihat LED berkedip. Pada motor dapat digunakan untuk mengatur kecepatan

putar motor dengan halus. Cara kerja ini dinamakan PWM (Pulse wave modulation)

Bagaimana bekerjanya ?
Contoh pertama adala pwmled.asm. pada contoh ini digunakan menggunakan dua subrutin delay. Satu delay digunakan untuk mengatur tlow delay yang lain digunakan untuk mengatur thigh. Apabila delay thigh dinaikkan maka delay dari tlow harus diturunkan demikian juga sebaliknya. Pengaturan ini dilakukan dengan cara mengatur register R4 dan R3. R4 naik maka register R3 harus diturun dan sebaliknya R4 turun maka R3 harus naik.

Contoh kedua adalah pwmled2.asm. pada contoh ini digunakan timer yang ada pada mikrokontroller. Timer hardware digunakan untuk mengatur delay sementara prossesor dapat digunakan untuk fungsi yang lain. Timer ini bersifat independen dan bekerja tanpa menggunakan prosessor. Jika timer selesai menghitung (ada overflow) maka timer membangkitkan interrupt dan prosessor mengerjakan subrutin interrupt. Register R7 digunakan untuk mengatur lebar pulsa(brightness LED atau kecepatan motor). Kita menset timer 0 sebagai 8 bit counter sehingga nilainya mulai dari 0 sampai 255. untuk membuat T mempunyai waktu konstan maka thigh akan menghitung dari R7 sampai 255 apabila hitungan telah selesai maka tlow aka mulai menghitung sisanya (255-R7) sampai 255. ini dapat dilakukan dengan menggunakan perintah subb

V. Peralatan yang dibutuhkan:

1. Modul board DT 51+ kabel serial
2. Cross compiler Metalink MCS-51
3. Crimson Editor
4. IBM PC kompatibel
5. LED modul dan switch modul

VI. Prosedur percobaan

Percobaan 1 - PWM LED 1

1. Buat file baru dengan menggunakan editor crimson
2. pastikan alamat awal 4000h
3. ketik program berikut ini

```
$mod51
code_seg     equ 4000h
;*****************************************************************
;* LED PWM
;* Modified bye: hendri@eepis-its.edu
;*****************************************************************
; RESET                        ;reset routine
  ORG  code_seg+0H      ;locate routine at 00H
AJMP   START                   ;jump to START
;*****************************************************************
; INTERRUPTS  (not used)       ;place     interrupt     routines     at
appropriate
                               ;memory locations
   ORG       code_seg+03H      ;external interrupt 0
      RETI
   ORG       code_seg+0BH      ;timer 0 interrupt
      RETI
   ORG       code_seg+13H      ;external interrupt 1
      RETI
   ORG       code_seg+1BH      ;timer 1 interrupt
      RETI
   ORG       code_seg+23H      ;serial port interrupt
      RETI
   ORG       code_seg+25H      ;locate beginning of rest of program
;*****************************************************************
INITIALIZE:                    ;set up control registers
      MOV   TCON,#00H
      MOV   TMOD,#00H
      MOV   PSW,#00H
      MOV   IE,#00H            ;disable interrupts
      RET
;*****************************************************************
;         Real  code  starts  below.  The  first  two  routines  are  for
delays ;        so we can slow down the blinking so we can see it.
(Without a ;       delay, it would blink so fast it would look like
it was
;      always on.
;*****************************************************************
DELAYONE:
LOOPONEA:
   INC       R4                ;increase R4 by one
```

```
   MOV       A,R4              ;move value in R4 to Accumlator
   CJNE   A,#0FFH,LOOPONEA     ;compare A to FF hex (256).
                              ;If not equal go to LOOPA
   RET                        ;return to the point that this routine
                             ;was called from
;*****************************************************************
DELAYTWO:
LOOPTWOA:
   INC       R3               ;increase R3 by one
   MOV       A,R3             ;move value in R3 to Accumlator
   CJNE  A,#0FFH,LOOPTWOA     ;compare  A  to  FF  hex  (256).  If  not
equal
                              ;go to LOOPA
   RET                        ;return to the point that this routine
                              ;was called from
;*****************************************************************
START:                        ;main program
   ACALL INITIALIZE           ;set up control registers
LOOP:
   MOV R4, #0FEH              ;used to control tlow
   MOV R3, #0FFH              ;used to control thigh
   CLR P1.0                   ;turn on
   ACALL DELAYONE
   SETB P1.0                 ;then turn right back off
   ACALL DELAYTWO
   AJMP LOOP                 ;go to LOOP
END                           ;end program
```

4. simpan dengan nama pwmled.asm
5. compile program
6. pastikan tidak ada error
7. jalankan program downloader dt51lwin, buka file pwmled.hex
8. setelah itu download ke board 8951 dengan terlebih dahulu posisi jumper di ubah ke 1&2

Percobaan 2- PWM LED 2

1. Buka file baru dengan menggunakan crimson editor
2. pastikan alamat awal program 4000h
3. ketik program berikut ini

```
;*********************************************************************
;* LED PWM 2
;* dimodifikasi oleh: hendri@eepis-its.edu
;*********************************************************************
$mod51
Code_seg  equ 4000h
;*********************************************************************
; RESET                       ;reset routine
   ORG       Code_seg         ;locate routine at 00H
   AJMP   START               ;jump to START
;*********************************************************************
; INTERRUPTS  (not used)
;place interrupt routines
;memory locations
   ORG       Code_seg+03H     ;external interrupt 0
      RETI
   ORG       Code_seg+0BH     ;timer 0 interrupt
      AJMP TIMER_0_INTERRUPT  ;go to interrupt routine
   ORG       Code_seg+13H     ;external interrupt 1
      RETI
   ORG       Code_seg+1BH     ;timer 1 interrupt
      RETI
   ORG       Code_seg+23H     ;serial port interrupt
      RETI
   ORG       Code_seg+25H     ;locate beginning
;*********************************************************************
INITIALIZE:
; set up control registers
   MOV TMOD,#00H        ; set timer 0 to Mode 0
   SETB TR0             ; turn on timer 0
   MOV PSW,#00H
   SETB EA              ; Enable Interrupts
   SETB ET0             ; Enable Timer 0 Interrupt
   RET
```

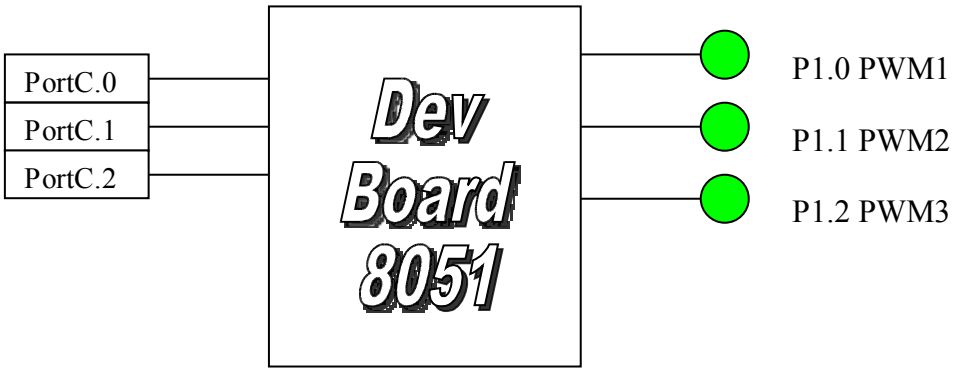
```
;*****************************************************************
;        Real code starts below.
;*****************************************************************
; The LED is off during the high section and on during the low section
; The Flag F0 is used to remember whether we are timing tlow or thigh.

TIMER_0_INTERRUPT:
   JB F0, HIGH_DONE     ; If F0 is set then we just finished the
                        ; high section of the
LOW_DONE:               ;cycle so Jump to HIGH_DONE
   SETB F0              ; Make F0=1 to indicate start of high section
   SETB P1.0            ; Turn off LED
   MOV TH0, R7          ; Load high byte of timer with R7
   CLR TF0              ; Clear the Timer 0 interrupt flag
   RETI                 ; Return from Interrupt
HIGH_DONE:
   CLR F0               ; Make F0=0 to indicate start of low section
   CLR P1.0             ; Turn on LED
   MOV A, #0FFH         ; Move FFH (255) to A
   CLR C                ; Clear C (the carry bit) so it does not
                        ; affect the subtraction
   SUBB A, R7           ; Subtract R7 from A. A = 255 - R7.
   MOV TH0, A           ; so the value loaded into TH0 + R7 = 255
   CLR TF0              ; Clear the Timer 0 interrupt flag
   RETI                 ; Return from Interrupt to where the program
                        ; came from
;*****************************************************************
START:                  ;main program
   ACALL INITIALIZE     ;set up control registers
   MOV R7, #0127H       ; set pulse width control to dim
LOOP:
   AJMP LOOP            ;go to LOOP
END                     ;end program
```

4. simpan dengan nama pwmled2.asm
5. compile program
6. pastikan tidak ada error, setelah itu download ke board 8951 dengan terlebih dahulu posisi jumper di ubah ke 1&2

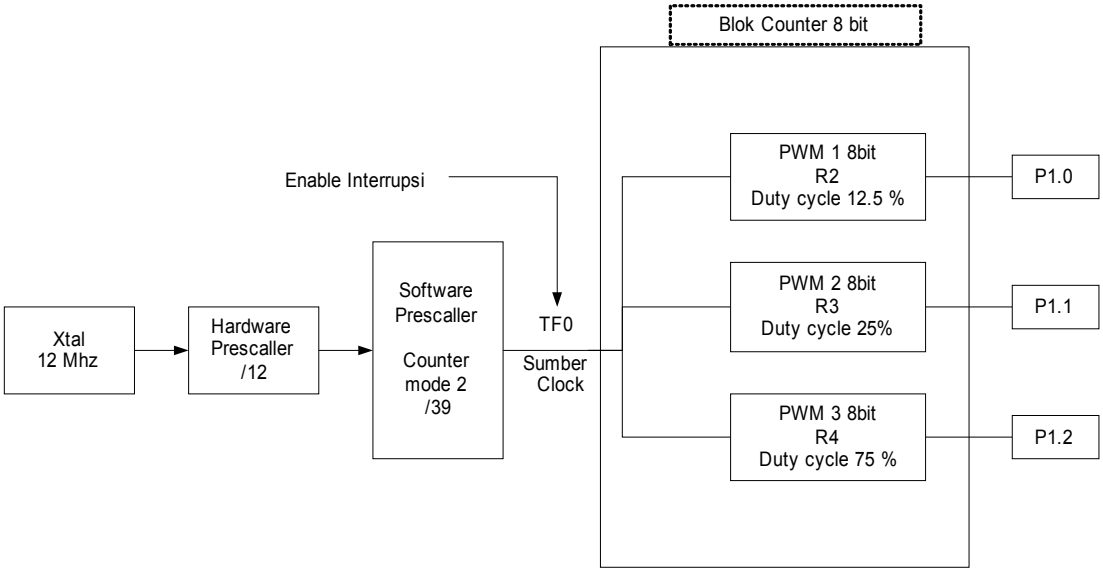
VII. Laporan sementara

Tugas diberikut dikerjakan sebagai laporan sementara.

Desain pwm controller 3 channel yang mempunyai duty cyle sebesar 12.5 % (P1.0), 25% (P1.1), dan 75% (P1.2). PWM 1 akan aktif bila PortC.0 berlogik1, PWM 2 akan aktif bila portC.1, PWM 3 akan aktif bila portC.2 Berlogik 1.(gambar 5-4 dan 5-5)



Gambar 5-4



Gambar 5-5

VIII. Laporan resmi

Sebagai analisa pada laporan resmi, gambarkan dan jelaskan diagram alir dari tugas yang anda kerjakan. Berikan keterangan pada program yang telah anda buat

IX. Tambahan

Berikan saran atau komentar guna pengembangan lebih lanjut praktikum ini

Bab 6

State mesin pada 8051

I. Tujuan

Setelah Melakukan praktikum ini yang anda peroleh adalah:

- Dapat Menggunakan Algoritma State machine pada MCS-51
- Dapat Menerapkan Algoritma State machine untuk membuat Vending Machine

II. Pendahuluan

Pada praktikum ini anda akan mempelajari cara mengembangkan suatu embbedded controller dengan menggunakan state mesin. Penerapan non linear programming dengan menggunakan state machine akan mempermudah pemahaman anda pada proess sekuensial. Pada project ini non linier programming diterapkan pada pembuatan vending machine.

III. Gambaran Desain

Anda membuat project baru dengan dengan menggunakan crimson editor, memulai program pada alamat 4000h dan memulai melakukan pemrograman dengan teknik non linear programming. Pada percobaan pertama anda akan mempelajari desain vending machine sederhana dengan menggunakan state mesin, pada percobaan kedua anda belajar mengimplementasikan vending machine yang lebih komplek menggunakan algoritma state mesin.

IV. Dasar Teori

Untuk beberapa aplikasi dibutuhkan pendekatan non linear programming, Contoh dari pendekatan non linier programming adalah state machine.

Pada state machine, variabel digunakan untuk menyimpan *current state* dari program. Pada praktikum percobaaan pertama variabel state digunakan diwakili oleh listing program pada baris mulai 38-42

```
;define state
st0       equ 0
st500     equ 1
St1000    equ 2
stdrop    equ 3
```

Pada listing program terlihat setiap selesai menjalankan prosedur state maka program akan lompat ke loop_state:

```
loop_state:
   lcall state_awal
   ljmp loop_state
```

Setiap kali nilai state diisi dengan 0,1,2,3 yang diwakili oleh st0,st500, st1000, stdrop maka program akan lompat ke next state yang diwakili oleh s0,s500, s1000 dan sdrop. Supaya state lompat ke label setelah jmp@+dptr maka harus dikalikan dengan 3(jmp @a+dptr memerlukan 3 byte code) sehingga tepat ke label state.

Ini ditunjukkan pada perintah dibawah ini

```
   mov b,#3
   mul ab          ; fix jmp start
   jmp @a+dptr
next_state:
   ljmp s0
   ljmp s500
   ljmp s1000
   ljmp sdrop
```

V. Peralatan yang dibutuhkan:

1. Modul board DT 51

2. Cross compiler Metalink MCS-51
3. Crimson Editor
4. IBM PC kompatibel
5. LED modul dan switch modul

VI. Prosedur percobaan

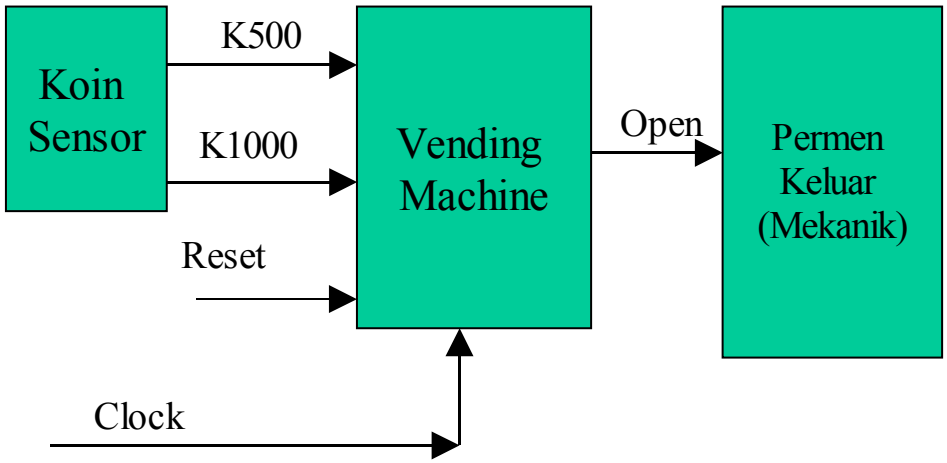
Percobaan 1 - Simple Vending Machine

Pada percobaan pertama anda diminnta membuat disain mesin penjual permen (candy machine) yang mempunyai harga permen Rp 1500,00. Mesin menerima masukan berupa uang logam **Rp 500** dan uang logam **Rp 1000**.

Spesifikasi:

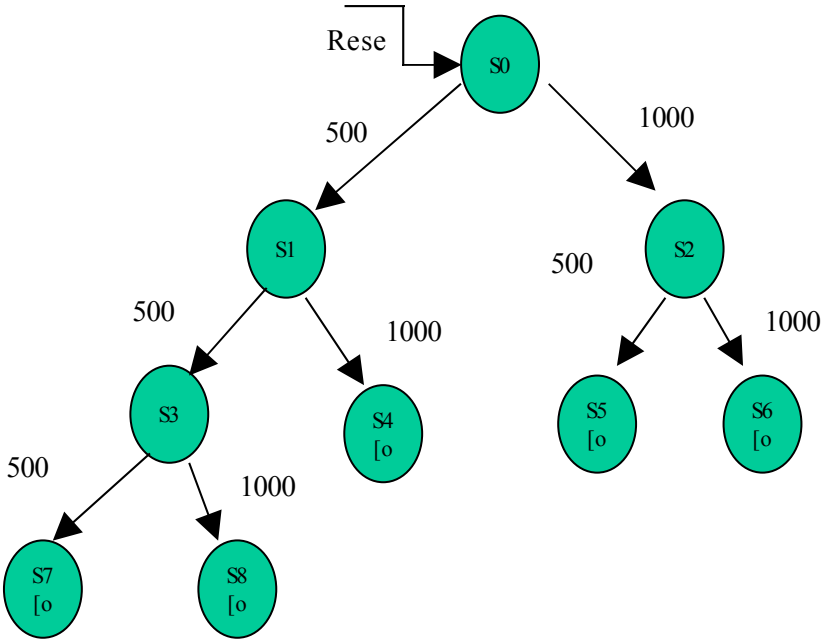
- Mesin tidak mengembalikan kelebihan uang
- Asumsikan bahwa mesin mempunyai clock yang cepat sehingga hanya satu input yang terjadi pada satu saat.
- Apabila permen sudah dikeluarkan mesin kembali memulai state awal (menunggu transaksi selanjutnya)

1. Desain Sistem

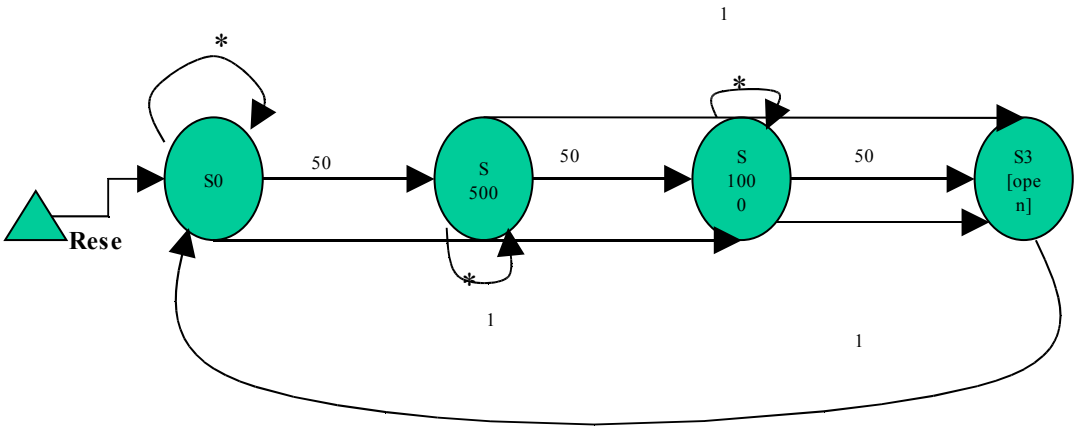


2. Buat State machine

3.



Sederhanakan State



4. Desain Program dengan menggunakan Crimson Editor. Isi dari Program seperti pada Listing dibawah ini

```
$mod51
;  vending machine
;  case studies 1
;  aplikasi ini menerangkan bagaimana MCS-51 menggunakan
;  state diagram dengan memanfaatkan Table jump
;  program ini menggunakan simulator MCS-51
;  akhmad hendriawan
;  selasa 15 juni 2004
;  Hardware Notes:

;Input Definition
 i500    equ   acc.0
 i1000   equ   acc.1

;Output Definition
porta   equ    2000h   ;
portb   equ    2001h
portc   equ    2002h   ; input
cw      equ    2003h
O500    equ p1.0       ;Kembalian 500
O1000   equ p1.1       ;Kembalian 1000
Odrink  equ p1.2

;define segment
code_seg equ 4000h

;define variabel
state    equ 20h
vdly     equ 21h
dthigh   equ 30h
dtlow    equ 31h
keyb     equ 32h

;define state
st0        equ 0
st500     equ 1
St1000    equ 2
```

```
stdrop     equ 3

org code_seg
    ljmp start
org code_seg+100h

;------------------------
; prosedur delay 1/10 detik
;------------------------
delay:
      push vdly
lfirst:
      mov dtlow,#100
loopms:
      mov tl0,#67
      mov th0,#0fch
      setb tr0      ;timer start
here: jbc tf0,loop1
      sjmp here
loop1:
      djnz dtlow,loopms
      djnz vdly,lfirst
      pop vdly
      ret
;------------------------
; prosedur waitkey for 20ms
;------------------------
waitkey:
      push 0
      mov R0,#20
loop20ms:
      mov tl0,#67
      mov th0,#0fch
      setb tr0      ;timer start
l1:   jbc tf0,l2
      sjmp l1
l2:   djnz R0,loop20ms
      Clr  tr0
      pop  0
      ret
;------------------------
```

```
; prosedur baca input   -
;------------------------
ReadInput:
     push dph
     push dpl
;pastikan tidak ada penekanan sebelumnya
ChkFreeBtn:
     mov  dptr,#portc
     movx a,@dptr
     cjne a,#0ffh,ChkFreeBtn
;read input
read_key:
     mov  dptr,#portc
     movx a,@dptr
     cjne a,#0ffh,ChkGlitch
     sjmp read_key
ChkGlitch:
     Call waitkey
;chkfor glitch
     mov  dptr,#portc
     movx a,@dptr
     cjne a,#0FFh,btnok
     sjmp ChkFreeBtn
btnok:
     cpl   a
     pop  dpl
     pop  dph
     ret

;=================
; main program   =
;=================
start:
; inisialisasi ppi
     mov a,#89h
     mov dptr,#cw
     movx @dptr,a
     mov  p1,#0ffh           ; matikan semua led

; inisisalisasi
     mov tmod,#01
```

```
     clr tr0
     mov state,#st0
     mov dptr,#next_state

loop_state:
     lcall state_awal
     ljmp loop_state
state_awal:
     mov a,state     ; make current state <= next state
     mov b,#3
     mul ab          ; fix jmp start
     jmp @a+dptr
next_state:
     ljmp s0
     ljmp s500
     ljmp s1000
     ljmp sdrop
s0:
   push  dph
   push  dpl
   call  ReadInput
   jb   I500,S0_1
   jb   I1000,S0_2
   mov  state,#st0
   jmp  end_s0
S0_1:
   mov  state,#st500
   jmp  end_s0
S0_2:
   mov  state,#St1000
end_s0:
   pop dpl
   pop dph
   ret

s500:
   push  dph
   push  dpl
   call  ReadInput
   jb   I500,S500_1
   jb   I1000,S500_2
```

```
   mov  state,#st500
   jmp  end_s500
S500_1:
   mov  state,#st1000
   jmp  end_s500
S500_2:
   mov  state,#Stdrop
end_s500:
   pop dpl
   pop dph
   ret

s1000:
   push  dph
   push  dpl
   call  ReadInput
   jb   I500,S1000_1
   jb   I1000,S1000_2
   mov  state,#st1000
   jmp  end_s1000
S1000_1:
   mov  state,#stdrop
   jmp  end_s1000
S1000_2:
   mov  state,#Stdrop
end_s1000:
   pop dpl
   pop dph
   ret

sdrop:
   mov  state,#st0
   clr Odrink
   mov  vdly,#10
   call delay
   setb  Odrink
end_sdrop:ret
end;
```

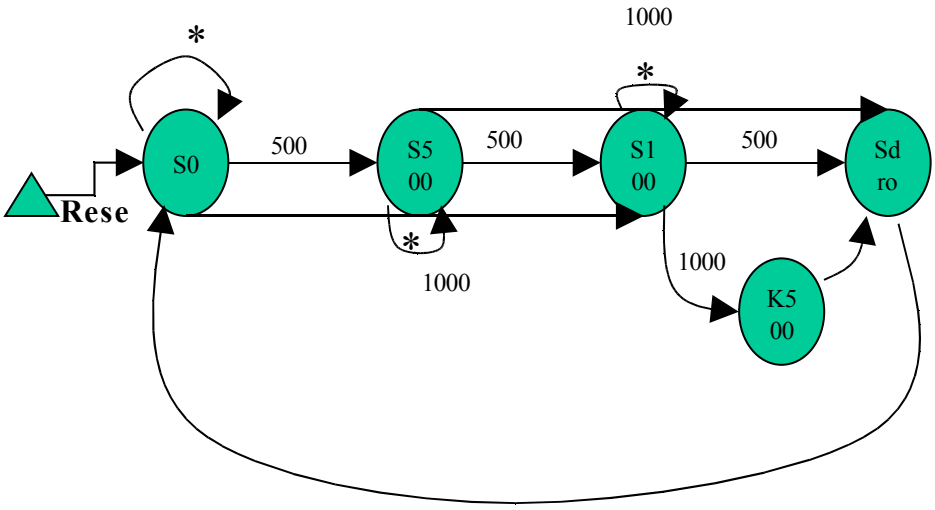
5. compile program tersebut (Ctrl-F9)
6. pastikan tidak ada error, setelah itu download ke board 8951.

7. Jalankan program dengan terlebih dahulu pastikan semua posisi Switch dalam keaadan berlogic 1 semua.
8. Simulasikan rancangan.
    - Untuk koin 500 diwakili oleh switch 0
    - Untuk koin 1000 diwakili oleh switch 1

Percoban 2 - Candy Machine

Pada percobaan kedua anda diminta membuat disain mesin penjual permen (candy machine) yang mempunyai harga permen Rp1500,00. Mesin menerima masukan berupa uang logam Rp500 dan uang logam Rp 1000. Mesin mengembalikan uang 500 jika uang 1000 dimasukkan 2 kali. Asumsikan bahwa mesin mempunyai clock yang cepat sehingga hanya satu input yang terjadi pada satu saat. Apabila permen sudah dikeluarkan mesin kembali memulai state awal (menunggu transaksi selanjutnya) . urutan langkahnya

1. Sederhanakan State



2. Desain Program dengan menggunakan Crimson Editor. Isi dari Program seperti pada Listing dibawah ini

```
$mod51
;  vending machine
;  case studies 1
;  aplikasi ini menerangkan bagaimana MCS-51 menggunakan
;  state diagram dengan memanfaatkan Table jump
;  program ini menggunakan simulator MCS-51
```

```
;
;  akhmad hendriawan
;  selasa 15 juni 2004
;  Hardware Notes:

;Input Definition
 i500    equ   acc.0
 i1000   equ   acc.1


;Output Definition
porta   equ    2000h   ;
portb   equ    2001h
portc   equ    2002h   ; input
cw      equ    2003h


O500    equ p1.0       ;Kembalian 500
O1000   equ p1.1       ;Kembalian 1000
Odrink  equ p1.2

;define segment
code_seg equ 4000h

;define variabel
state    equ 20h
vdly     equ 21h
dthigh   equ 30h
dtlow    equ 31h
keyb     equ 32h

;define state
st0       equ 0
st500     equ 1
St1000    equ 2
StU500    equ 3
stdrop    equ 4

org code_seg
    ljmp start
```

```
org code_seg+100h

;------------------------
; prosedur delay 1/10 detik
;------------------------
delay:
      push vdly
lfirst:
      mov dtlow,#100
loopms:
      mov tl0,#67
      mov th0,#0fch
      setb tr0      ;timer start
here: jbc tf0,loop1
      sjmp here
loop1:
      djnz dtlow,loopms
      djnz vdly,lfirst
      pop vdly
      ret

;------------------------
; prosedur waitkey for 20ms
;------------------------
waitkey:
      push 0
      mov R0,#20
loop20ms:
      mov tl0,#67
      mov th0,#0fch
      setb tr0      ;timer start
l1:   jbc tf0,l2
      sjmp l1
l2:   djnz R0,loop20ms
      Clr  tr0
      pop  0
      ret

;------------------------
; prosedur baca input   -
```

```
;------------------------
ReadInput:
     push dph
     push dpl
;pastikan tidak ada penekanan sebelumnya
ChkFreeBtn:
     mov  dptr,#portc
     movx a,@dptr
     cjne a,#0ffh,ChkFreeBtn
;read input
read_key:
     mov  dptr,#portc
     movx a,@dptr
     cjne a,#0ffh,ChkGlitch
     sjmp read_key
ChkGlitch:
     Call waitkey
;chkfor glitch
     mov  dptr,#portc
     movx a,@dptr
     cjne a,#0FFh,btnok
     sjmp ChkFreeBtn
btnok:
     cpl   a
     pop  dpl
     pop  dph
     ret

;=================
; main program   =
;=================
start:
; inisialisasi ppi
     mov a,#89h
     mov dptr,#cw
     movx @dptr,a
     mov  p1,#0ffh           ; matikan semua led

; inisisalisasi
     mov tmod,#01
```

```
     clr tr0
     mov state,#st0
     mov dptr,#next_state

loop_state:
     lcall state_awal
     ljmp loop_state
state_awal:
     mov a,state     ; make current state <= next state
     mov b,#3
     mul ab          ; fix jmp start
     jmp @a+dptr
next_state:
     ljmp s0
     ljmp s500
     ljmp s1000
     ljmp su500
     ljmp sdrop
s0:
   push  dph
   push  dpl
   call  ReadInput
   jb   I500,S0_1
   jb   I1000,S0_2
   mov  state,#st0
   jmp  end_s0
S0_1:
   mov  state,#st500
   jmp  end_s0
S0_2:
   mov  state,#St1000
end_s0:
   pop dpl
   pop dph
   ret

s500:
   push  dph
   push  dpl
   call  ReadInput
```

```
   jb   I500,S500_1
   jb   I1000,S500_2
   mov  state,#st500
   jmp  end_s500
S500_1:
   mov  state,#st1000
   jmp  end_s500
S500_2:
   mov  state,#Stdrop
end_s500:
   pop dpl
   pop dph
   ret

s1000:
   push  dph
   push  dpl
   call  ReadInput
   jb   I500,S1000_1
   jb   I1000,S1000_2
   mov  state,#st1000
   jmp  end_s1000
S1000_1:
   mov  state,#stdrop
   jmp  end_s1000
S1000_2:
   mov  state,#Stu500
end_s1000:
   pop dpl
   pop dph
   ret

su500:
   mov  state,#stdrop
   clr O500
   mov  vdly,#10
   call delay
   setb  O500
end_su500:ret
```

```
sdrop:
   mov  state,#st0
   clr Odrink
   mov  vdly,#10
   call delay
   setb  Odrink
end_sdrop:ret

end;
```

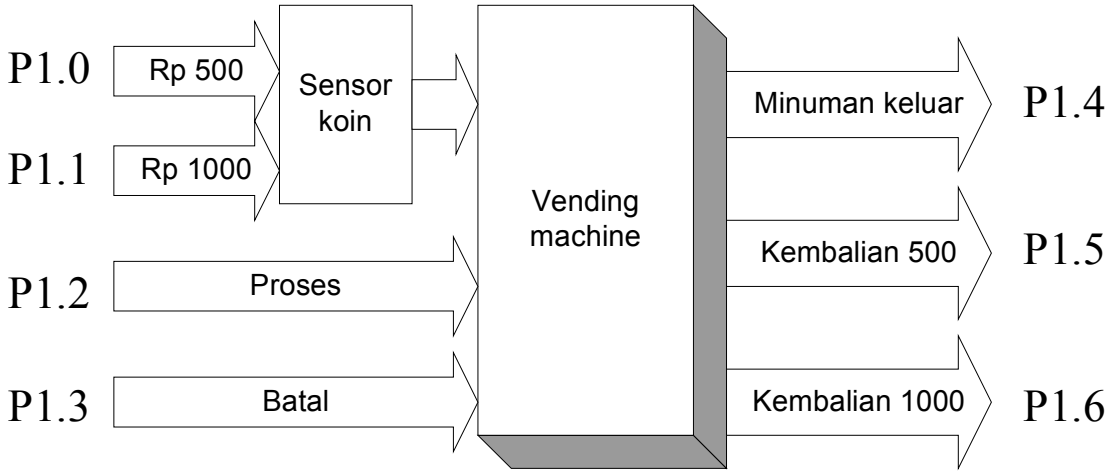
3. pastikan tidak ada error, setelah itu download ke board 8951.
4. Jalankan program dengan terlebih dahulu pastikan semua posisi Switch dalam keaadan berlogic 1 semua.
5. Simulasikan rancangan.
    - Untuk koin 500 diwakili oleh switch 0
    - Untuk koin 1000 diwakili oleh switch 1

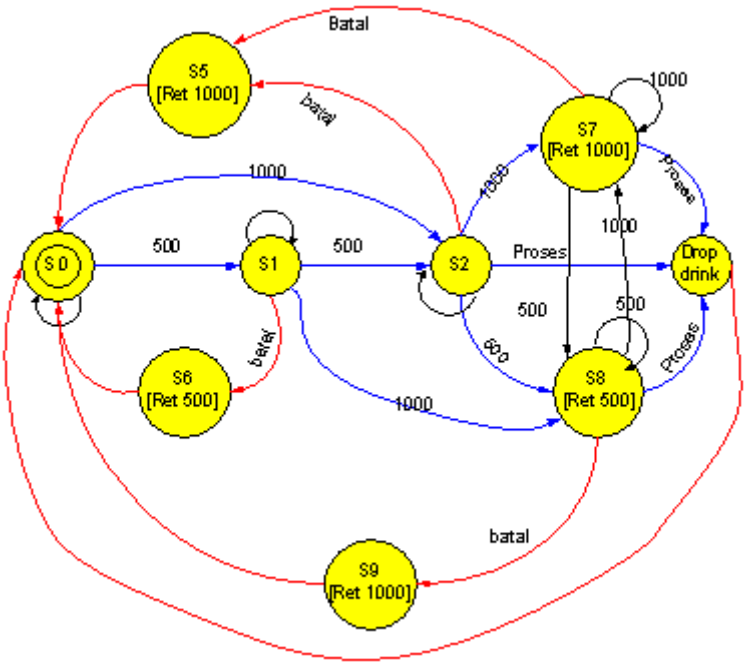
VII. Laporan sementara

Tugas berikut dikerjakan sebagai laporan sementara

Desain vending machine yang mempunyai harga minuman 1000. vending machine hanya menerima uang logam 500 dan uang logam 1000. transaksi di proses menggunakan tombol "proses". Transaksi yang belum diproses masih dapat dibatalkan dengan menekan tombol "batal". Mesin dapat mengeluarkan kembalian jika ada kelebihan uang.

- Gambar kan diagram statenya
- Implementasikan dengan menggunakan program yang sederhana**]**




Batal
S5
[Ret 1000]
batal
1000
S7
[Ret 1000]
1000
Proses
1000
S 0
500
S1
500
S2
Proses
Drop
drink
500
500
500
S6
[Ret 500]
batal
1000
S8
[Ret 500]
Proses
batal
S9
[Ret 1000]

VIII. Laporan resmi

Sebagai analisa pada laporan resmi, gambarkan dan jelaskan diagram alir dari tugas yang anda kerjakan. Berikan keterangan pada program yang telah anda buat

IX. Tambahan

Berikan saran atau komentar guna pengembangan lebih lanjut praktikum ini

Lampiran

REFERENCE

A. Instruksi assembler pada 8051

Mnemonics	Operands	Bytes/Cycle
ADD	A, Rn	1/1
ADDC	A, direct	2/1
SUBB	A, @Ri	1/1
	A, #data	2/1
INC	A	1/1
DEC	Rn	1/1
	direct	2/1
	@Ri	1/1
INC	DPTR	1/2
MUL	AB	1/4
DIV	AB	1/4
DA	A	1/1

Operasi logic

Mnemonic	Operands	Bytes/Cycles

ANL	A, Rn	1/1
ORL	A, direct	2/1
XRL	A, @Ri	1/1
	A, #data	2/1
	direct, A	2/1
	direct, #data	3/2
	C, bit	2/2
	C, /bit	2/2
CLR	A	1/1
CPL	C	1/1
	bit	2/1
RL	A	1/1
RLC	A	1/1
RR	A	1/1
RRC	A	1/1
SWAP	A	1/1
SETB	C	1/1
CLR	bit	2/1
CPL		

Operasi data transfer

Mnemonic	Operands	Bytes/Cycles
MOV	A, Rn	1/1
	A, direct	2/1
	A, @Ri	1/1
	A, #data	2/1
	Rn, A	1/1
	Rn , direct	2/2
	Rn, #data	2/1
	direct, A	2/1
	direct, Rn	2/2
	direct, direct	3/2
	direct, @Ri	2/2
	direct, #data	3/2

MOV		
	@Ri, A	1/1
	@Ri, direct	2/2
	@Ri, #data	2/1
	DPTR, #data16	3/2
		2/1
	C, bit	2/2
	bit, C	
MOVX	A,@DPTR	1/2
	@DPTR,A	1/2
	A,@Ri	1/2
	@Ri,A	1/2
MOVC	A,	1/2
	@A+DPTR	1/2
	A, @A+PC	
PUSH	direct	2/2
POP	direct	2/2