BAB II
DASAR TEORI

2.1 Komunikasi Data Paralel

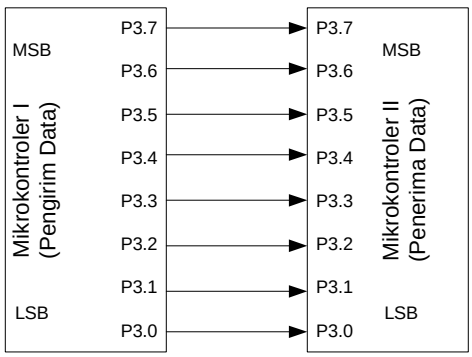
Prinsip dasar dari sistem komunikasi data *paralel* adalah suatu cara untuk pengiriman atau pertukaran data dari kedua pihak dengan menggunakan sirkuit yang terhubung lebih dari satu serta waktu pengiriman yang cepat dibandingkan dengan komunikasi data secara *serial*. Pada Gambar 2.1 dijelaskan sebuah contoh sistem komunikasi data satu arah sederhana terkait pengiriman data mikrokontroler I ke mikrokontroler II pada perancangan alat ”Sistem Pengontrolan Kunci Pintu pada Ruangan Berbasis Mikrokontroler AT89C51”.



Gambar 2.1 Blok Pengiriman Data Mikrokontroler I ke Mikrokontroler II

Untuk mengirimkan data dari satu lokasi ke lokasi yang lain, tiga elemen sistem harus tersedia, yaitu pengirim data, media pengirim data, dan penerima data. Jika salah satu elemen tidak ada, maka pengiriman tidak akan dapat dilakukan. Gambar 2.1 dapat memperjelas mengenai ketiga elemen tersebut. Berikut ini penjelasan mengenai ketiga elemen komunikasi data tersebut:

1. Mikrokontroler I berfungsi sebagai pengirim data
2. Media pengiman data berfungsi sebagai jalur yang menghubungkan antara mikrokontroler I dengan mikrokontroler II
3. Mikrokontroler II berfungsi sebagai penerima data



Gambar 2.2 Transmisi Paralel dari Mikrokontroler I ke Mikrokontroler II

Cara kerja dari pengiriman data ini sangat sederhana, dimana mikrokontroler I yang berfungsi sebagai pengirim, di program untuk mengeluarkan data yang diinginkan. Data dibuat berupa angka atau kode sehingga mengeluarkan bit-bit tertentu pada *port* yang ditunjuk sebagai *port output* untuk mengrimkan data. Angka atau kode yang dibuat dapat berupa bilangan desimal, heksadesimal maupun bilangan *biner* yang nantinya akan dikirimkan ke mikrokontroler II dengan terus-menerus pada saat-saat tertentu. Misalnya pada saat waktu menunjukan pukul 07.00, data yang dikirimkan ialah 1CH (00011100b) atau dalam bentuk bilangan angka desimal ialah 28. Data berupa angka tersebut dikirimkan melalui kabel 8 bit yang dihubungkan ke *port* yang ditunjuk sebagai penerima atau masukan data pada mikrokontroler II.

Mikrokontroler II yang berfungsi sebagai penerima data, di program untuk membandingkan atau membaca data yang dikirimkan dari *port output* mikrokontroler I, jika data yang dibaca oleh mikrokontroler II tidak sama, maka mikrokontroler II tidak mengasumsikan sebagai pukul 07.00 tetapi jika data yang dibaca atau diterima sama maka mikrokontroler II mengasumsikan bahwa waktu tersebut menunjukan pukul 07.00.

Berikut tabel pengiriman data yang dikirimkan dari mikrokontroler I ke Mikrokontroler II:

Tabel 2.1 Tabel Pengiriman Data

Jam	**Pengirim Data (Mikro I)**			**Penerima Data (Mikro II)**		
	Data yang dikirim			**Data yang diterima**		
	Desimal	**Biner**	**Hex**	**Desimal**	**Biner**	**Hex**
00.00.00	0	00000000	00	0	00000000	00
07.00.00	28	00011100	1C	28	00011100	1C
08.00.00	32	00100000	20	32	00100000	20
11.45.00	47	00101111	2F	47	00101111	2F
13.00.00	52	00110100	34	52	00110100	34
16.00.00	64	01000000	40	64	01000000	40
17.00.00	68	01000100	44	68	01000100	44

2.2 Mikrokontroler AT89C51

Mikrokontroler sebagai salah satu terobosan teknologi *mikroprosesor* dan *mikrokontroler* hadir untuk kebutuhan pasar dan teknologi baru. Sebagai salah satu teknologi baru, yaitu teknologi semikonduktor dengan kandungan transistor yang lebih banyak namun hanya membutuhkan ruang yang kecil.

Tidak seperti sistem komputer yang mampu menangani berbagai macam aplikasi, mikrokontroler hanya dapat digunakan untuk aplikasi tertentu saja (hanya satu program yang dapat disimpan dalam mikrokontroler ini). Mikrokontroler telah dikombinasikan dengan *input*/ *output* (I/O) dan memori sebagai komponen utamanya dalam bentuk *single chip*. Mikroprosesor telah dikombinasikan dengan *input*/ *output* (I/O) dan memori baik *RAM* maupun *ROM* dengan *kompleksitas* tinggi dan mempunyai pemrosesan pusat (*Central Processing Unit)* sehingga disebut sebagai mikrokomputer.

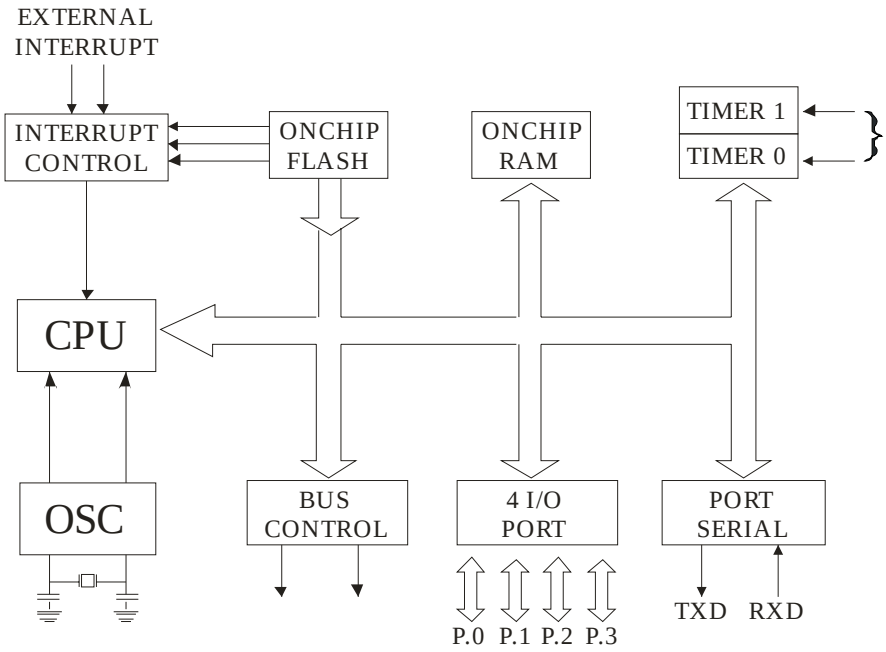
Secara umum sistem mikrokomputer terdiri dari:

- Unit mikroprosesor *(microprocessing unit)*
- RAM *(Random Acces Memory)*
- ROM *(Read Only Memory)*
- I/O *(Input/Output unit)*

Mikrokontroler AT89C51 adalah mikrokontroler buatan ATMEL *Corporation.* Mikrokontroler ini termasuk kedalam bagian keluarga mikrokontroler MCS51, varian mikrokontroler 8051 yang memiliki ciri-ciri sebagai berikut:

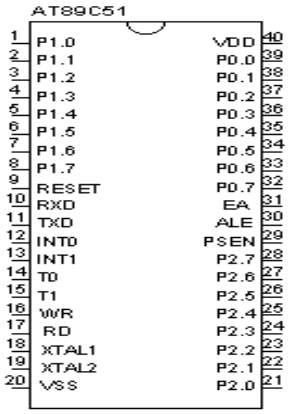
- CPU *(Central Processing Unit)* 8 bit untuk aplikasi pengontrolan
- 4 kilo*byte internal* ROM
- 128 *byte internal* RAM
- 4 buah *port* I/O 8 bit
- 2 buah *timer* 16 bit
- 5 jalur *interupsi*, terdiri dari 3 *interupsi internal* dan 2 *interupsi external*
- Saluran komunikasi *serial*
- 32 baris *Programmable* I/O
- Mampu beroperasi statis secara penuh pada 9 Hz to 24 Mhz

Di bawah ini dapat dilihat gambar masing-masing perangkat keras (*hardware*) yang ada pada mikrokontroler MCS51. Didalamnya terdapat *CPU, Memory*, dan perangkat keras lain yang akan menjalankan instruksi sesuai keinginan *programmer*.



Gambar 2.3 Blok Diagram Dasar AT89C51

2.2.1 Fungsi Pin Mikrokontroler AT89C51



Gambar 2.4 Konfigurasi Pin Mikrokontroler AT89C51

1. *Port* 0

   Merupakan jalur 8 bit yang dapat dijadikan *input/ output* data. *Port* 0 berada pada pin 32 hingga pin 39. bila digunakan untuk mengakses eksternal memori, *port* 0 akan *memultipleks* alamat memori dan data.

2. *Port* 1

   Merupakan jalur 8 bit yang mempunyai fungsi sebagai saluran *input* maupun saluran *output*. *Port* 1 berada pada pin 1 hingga pin 8

3. *Port* 2

Port 2 yang berada pada pin 21 hingga pin 28 mempunyai fungsi sebagai saluran *input* maupun *output.* Bila digunakan untuk mengakses eksternal memori, *port* 2 akan mengirimkan *byte* atau ke eksternal memori

4. *Port* 3

   Port 3 yang berada pada pin 10 hingga pin 17 merupakan saluran 8 bit yang dapat dijadikan saluran *input* maupun *output.* Selain itu *port* ini berfungsi sebagai saluran komunikasi *serial* dan sebagai jalur interupsi.

   Di bawah ini adalah fungsi khusus dari *port* 3:

   a. Pin P3.0 / RXD (*Port Serial Input*) : Merupakan jalur *input* untuk menerima data *serial*

   b. Pin P3.1 / TXD *(Port Serial Output)* : Merupakan jalur *output* untuk mengirim data *serial*

   c. Pin P3.2 / INTO *(Input Interupsi External* 0*)* : Jalur untuk melakukan *external interrupt*

   d. Pin P3.3 / INTI *(Input Interupsi External 1)* : Jalur untuk melakukan *external interrupt*

   e. Pin P3.4 / TO *(Input Interupt Timer 0)* : jalur untuk menerima data *clock* dengam mode *timer* 0

   f. Pin P3.5 / T1 *(Input Interupt Timer 1)* : jalur untuk menerima data *clock* dengam mode *timer* 1

   g. Pin P3.6 / WR : sebagai jalur untuk pengontrol proses penulisan data eksternal RAM

   h. Pin P3.7 / RD : sebagai jalur untuk pengontrol proses pembacaan data eksternal RAM

5. RST (*Reset*)

   Berfungsi untuk me-*reset* sistem mikrokontroler

6. ALE / PROG

   ALE (*Address Latch Enable*) digunakan untuk pengatur *address bus* dan *data bus* pada pengoperasian dengan memori eksternal. Sinyal Ale akan menahan alamat memori eksternal selama pelaksanaan instruksi. Pin ini juga akan berfungsi sebagai pulsa program (PROG) pada proses pemrograman *flash.*

7. PSEN

   Program Store Enable merupakan pin untuk membaca eksternal program memori pada mikrokontroler yang dioperasikan dengan program memori eksternal.

8. *EA/ VPP*

   External Acces Enable merupakan pin yang dihubungkan dengan *Vcc* bila menggunakan memori program *internal* yang beralamat 0000H hingga 0FFFH. Apabila menggunakan memori eksternal maka alamat dari memori eksternal ialah mulai 1000H hingga FFFFH, EA harus diubungkan dengan *ground* agar dapat mengakses memori program eksternal yang beralamat mulai 0000H hingga FFFFH
9. VCC

   Merupakan *input* catu daya untuk mikrokontroler dan diberi tegangan sebesar 5 Volt
10. VSS

    Pin *ground* tegangan catu daya untuk mikrokontroler dan diberi *input* tegangan 0 Volt.
11. XTAL1 dan XTAL2

    XTAL1 merupakan *input* osillator dan XTAL2 merupakan *output* osillator eksternal.

2.2.2 Port Input/ Output AT89C51

Mikrokontroler AT89C51 mempunyai 4 buah *port*, yaitu *port* 0, *port* 1, *port* 2 dan *port* 3 yang terletak pada alamat 80H, 90H, A0H dan B0H. *port* 0, *port* 2 dan *port* 3 tidak dapat digunakan sebagai fungsi-fungsi khusus seperti interupsi eksternal, *serial* ataupun *timer* eksternal. Untuk itu digunakan *port* 1 yang dapat di pakai untuk perintah dengan fungsi umum.

Port 0, *port* 2 dan *port* 3 memiliki fungsi alternatif. Masing-masing dari pin ini dapat digunakan sebagai jalur *input/ output* digital secara umum atau alternatifnya dapat digunakan untuk fungsi keduanya. Fungsi lain dari *port* 0 dan *port* 2 adalah untuk menghubungkan memori eksternal. Ketika program eksternal atau memori data sedang di akses. *Port* 2 mengluarkan *byte* tinggi dari alamat 16-bit. *Port* 0 awalnya mengeluarkan *byte* rendah dari alamat 16-bit, kemudian mengirimkan atau menerima *byte* data. Semua *port* ini dapat di akses dengan pengalamatan secara bit hingga dapat dilakukan perubahan *output* pada tiap-tiap pin dari *port* ini tanpa mempengaruhi pin-pin yang lainnya.

Ketika sebuah *port* digunakan sebagai *port output*, data diletakan di SFR (*Special Fungtion Register*) yang sesuai. Nilai yang tertulis di SFR akan diteruskan sampai dengan operasi penulisan selesai. Nilai dari *port output* akan berubah bila nilai baru dimasukan. Ketika sebuah *port* digunakan sebagai *port input*, nilai FFH harus

dituliskan pertama kali ke *port*, kemudian setiap *input* yang menggunakan tegangan rendah akan dianggap sebagai nilai 0, dan *port* tersebut dibaca SFR yang sesuai.

Meskipun SFR yang sama sedang digunakan, terdapat perbedaan dua operasi yang keluar, dimana membedakan kapan sebuah *port* yang sedang ditulis dan kapan *port* yang sedang dibaca. Operasi dilaksanakan seluruhnya di perangkat keras dan pengguna dapat menjaga arah jalur dimana data tersebut dipindahkan. Sedangkan fungsi lain dari *port* 3 termasuk *interrupt* dan *input timer*, *serial port input* dan *output*, dan sinyal kontrol untuk menghubungkan dengan memori eksternal.

2.2.3 Struktur Memori

AT89C51 mempunyai struktur memori sebagai berikut:

1. RAM Internal.

   Ram Internal memiliki memori sebesar 128 *byte* yang biasanya digunakan untuk menyimpan *variabel* atau data yang bersifat sementara. *Ram Internal* dialamati oleh RAM *Address Register* (Register Alamat RAM). *Ram internal* ini terdiri atas:

   - Register Banks

     AT89C51 mempunyai 8 buah register yang terdiri dari R0 sampai dengan R7. Register-register tersebut selalu terletak pada alamat 00H hingga 07H pada setiap kali di-*reset*.

   - Bit Addressable RAM

     RAM pada alamat 20H hingga 2FH dapat di akses secara pengalamatan bit sehingga hanya sebuah instruksi setiap bit dapat di-*set*, *clear*, AND dan OR

   - RAM dengan fungsi umum

     RAM dengan fungsi umum ini di mulai pada alamat 30H hingga 7FH dan dapat di akses dengan pengalamatan langsung maupun tak langsung.

2. Special Function Register (Register dengan fungsi khusus)

   Memori yang berisi register-register yang mempunyai fungsi-fungsi khusus yang disediakan oleh mikrokontroler seperti, *timer serial* dan lain-lain.

3. Flash FEROM

   Memori yang digunakan untuk menyimpan instruksi-instruksi MCS-51 di alamati oleh Register Alamat Program. AT89C51 memiliki 4 Kb *Flash PEROM* yang menggunakan *Atmel High-Density Non Volatile Technology*

   Program yang ada pada *flash PEROM* akan dijalankan jika pada saat sistem di-*reset*, pin EA/VP berlogika 1 maka mikrokontroler aktif berdasarkan program yang ada

pada *Flash PEROM*-nya. Namun jika pin EA/VP berlogika 0, mikrokontroler aktif berdasarkan program yang ada pada memori eksternal.

2.2.4 Timer AT89C51

Mikrokontroler AT89C51 mempunyai 2 buah *timer*, yaitu *timer* 0 dan *timer* 1, setiap *timer* terdiri dari 16 bit *timer* yang tersimpan dalam 2 buah register yaitu THx untuk *timer High Byte* dan TLx untuk *timer Low Byte* yang keduanya dapat berfungsi sebagai *counter* maupun sebagai *timer*.

Timer mempunyai sumber *clock* dengan frekuensi tertentu yang sudah pasti sedangkan *counter* mendapat sumber *clock* dari pulsa yang hendak di hitung jumlahnya. Aplikasi dari *timer* atau pewaktu dapat digunakan untuk aplikasi mengitung lamanya suatu kejadian yang terjadi sedangkan *counter* atau penghitung biasa digunakan untuk aplikasi menghitung jumlah kejadian yang terjadi dalam periode tertentu.

Register *THx* dan *TLx* diatur oleh register *TMOD* dan register *TCON*. *Timer* dapat diaktifkan melalui perangkat keras maupun perangkat lunak.

Periode waktu *timer/counter* dapat dirumuskan sebagai berikut:

1. Sebagai *timer/ counter* 8 bit

   T = (255 – TLx) * 1 µs ..... (2.1)

   TLx merupakan isi register TL0 atau TL1

2. Sebagai *timer/ counter* 16 bit

   T = (65535 – THx TLx) * 1 µs ..... (2.2)

   THx merupakan isi register TH0 atau TH1 dan TLx merupakan isi register TL0 atau TL1

	Register Timer				Register Interupsi			
88H	TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0
	TCON 7	TCON 6	TCON5	TCON 4	TCON 3	TCON 2	TCON 1	TCON 0

(a) TCON ; dapat diakses secara bit

	Timer I				TIMER 0			
89H	GATE	C/T	M1	M0	GATE	C/T	M1	M0

(b) TMOD ; tidak dapat diakses secara bit

Gambar 2.5 Register TCON dan TMOD

Pengontrolan kerja *timer/ counter* di atur oleh register TCON. Register ini bersifar *bit addressable* sehingga bit TF1 dapat disebut TCON.7 dan seterusnya hingga bit IT0 sebagai TCON 0. Register ini hanya mempunyai 4 bit saja yang berhubungan dengan *timer* seperti diperlihatkan pada gambar 2.5 serta akan dijelaskan pada tabel 2.2.

Tabel 2.2 Fungsi Bit Register TCON yang Berhubungan dengan Timer

Nama	Bit Fungsi
TF1	*Timer* 1 *Overflow Flag* yang akan diset jika *timer overflow*
TR1	Membuat *timer* 1 aktif (*set*) dan non aktif (*clear*)
TF0	*Timer* 0 *Overflow Flag* yang akan di-*set* jika *timer overflow*
TR0	Membuat *timer* 0 aktif *(set*)dan nonaktif (*clear*)

Register TMOD berfungsi untuk pemilihan mode operasi *timer/ counter* dengan fungsi tiap bitnya adalah sebagai berikut:

- Gate berfungsi pada saat TRx =1, *timer* akan berjalan tanpa memperhatikan nilai pada *Gate* (*timer* di kontrol *software*)
- C/T berfungsi sebagai pemilihan fungsi *timer* (0) atau *counter* (1).
- M1 & M0 berfungsi untuk memilih mode *timer* dengan variasi seperti pada tabel 2.3.

Tabel 2.3 Mode Timer

M1	M0	Mode	Operasi
0	0	0	*Timer* 13 bit
1	1	1	*Timer* / *counter* 16 bit
1	0	2	Register THx berisi nilai isi ulang yang akan dikirim ke TLx setiap overflow
1	1	3	Pada mode ini, AT89C51 seperti memiliki 3 buah *timer*. *Timer* 0 terpisah menjadi 2 buah *timer* 8 bit (TL0 – TF0 dan TH0 –TF1) dan *timer* 1 tetap 16 bit

2.3 Indikator Seven Segment

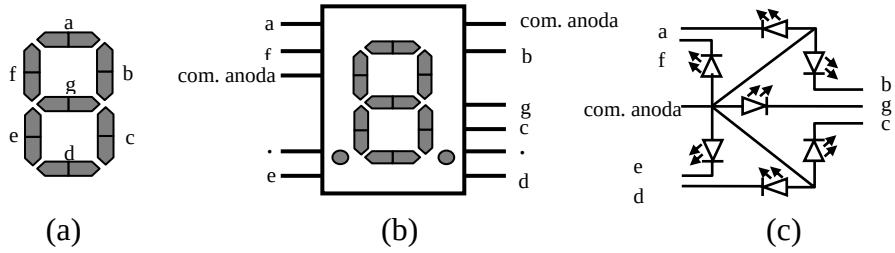
Seven Segment merupakan suatu komponen yang sering digunakan untuk keperluan *display* peraga bilangan *biner* ke angka desimal atau heksadesimal. Pada peraga *seven segment* di rancang dari lampu-lampu led yang di susun membentuk

segment sehingga dapat di susun penyalaannya untuk membentuk suatu angka. Salah satu cara untuk menampilkan angka atau huruf adalah dengan metode *matrik*.

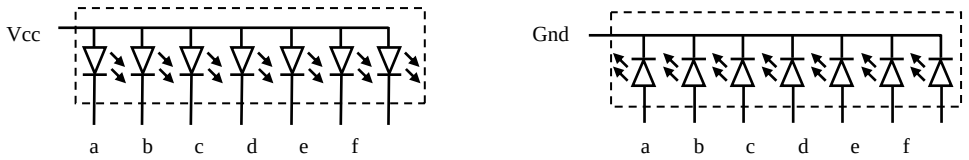
Dengan metoda *matrik*, beberapa garis atau *segment* yang membentuk angka 0 – 9. Pada metoda ini digunakan tujuh buah ruas atau *segment* yang tersusun sehingga dapat menyalakan garis-garis tertentu atau membentuk angka desimal umumnya disebut tampilan *seven segment*.

Indikator *Seven Segment* ada dua tipe, yaitu:

- Tipe *Common Anoda* : semua kaki *anoda* terhubung ke *Vcc*.
- Tipe *Common Katoda* : semua kaki *katoda* terhubung ke *ground*.



Gambar 2.6 Bentuk Seven Segment
(a) Tampilan seven segment dengan urutan segmentnya.
(b) Salah satu bentuk indikator seven segment dengan urutan kaki-kakinya.
(c) Susunan LED pada seven segment dengan common anoda.



Gambar 2.7 Susunan LED pada Seven Segment
(a) Common Anoda, (b) Common Katoda

2.4 Penyearah (*Rectifier)*

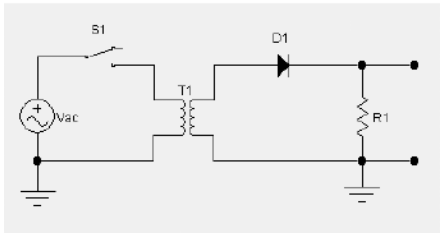
Penyearahan adalah proses meyearahkan arus bolak-balik menjadi arus searah. Arus bolak-balik ini berasal dari tegangan jala-jala. Komponen utama yang diperlukan dalam penyearahan adalah transformator, dioda dan kapasitor elektrolit.

Secara umum penyearah terdiri dari:

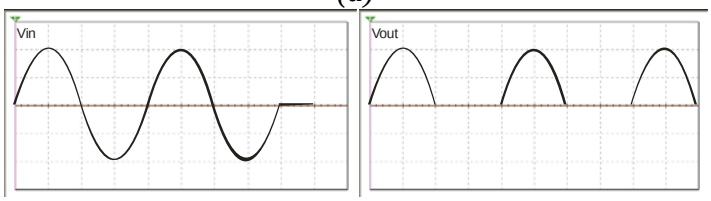
- Penyearah setengah gelombang
- Penyearah gelombang penuh sistem CT
- Penyearah gelombang penuh sistem jembatan
- Penyearah tergulasi

2.4.1 Penyearah Setengah Gelombang

Penyearah ini bekerja dengan menggunakan satu dioda. Sehingga hanya pulsa positif yang dapat terambil.



(a)



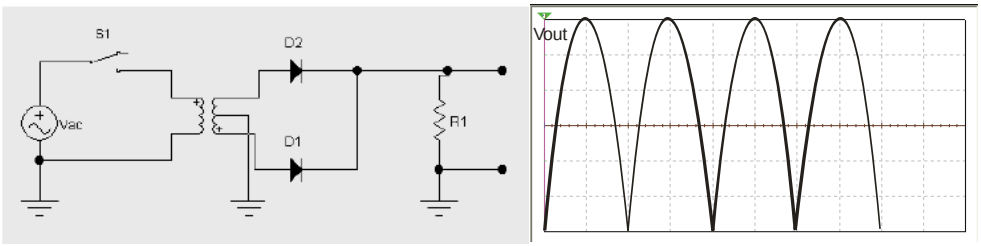
(b) *(c)*

Gambar 2.8 Penyearah Setengah Gelombang

(a) Rangkaian Penyearah Setengah Gelombang (b) Sinyal Input (c) Sinyal Output

2.4.2 Penyearah Gelombang Penuh Sistem CT

Penyearah Gelombang Penuh Sistem CT menggunakan transformator jenis CT dengan dua buah dioda sebagai penyearah. Dioda secara bergantian untuk mengambil pulsa positif dan negatif. Sehingga keluaran berupa gelombang penuh

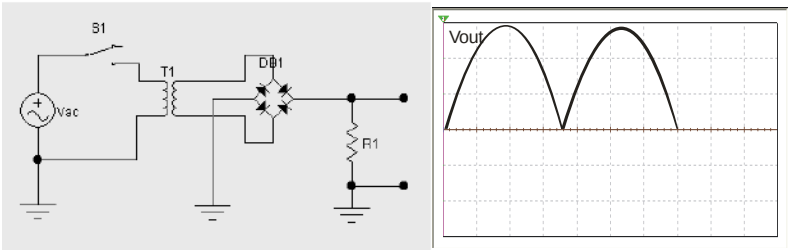


(a) *(b)*

Gambar 2.9 Penyearah Gelombang Penuh Sistem CT
(a) Rangkaian Penyearah Gelombang Penuh Sistem CT (b) Sinyal Output

2.4.3 Penyearah Gelombang Penuh Sistem Jembatan

Penyearah ini menggunakan 4 buah dioda sebagai penyearah. Pada siklus pertama dua buah dioda bekerja untuk mengambil pulsa positif. Dan dua dioda selanjutnya mengambil pulsa negatif. Keuntungan menggunakan penyearah ini ialah keluaran yang dihasilkan berupa gelombang penuh dan jika salah satu dioda rusak, maka dioda yang satunya lagi akan tetap bekerja.

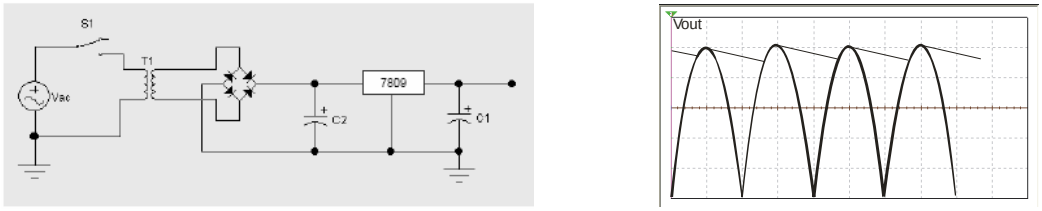


(a) *(b)*

Gambar 2.10 Penyearah Gelombang Penuh Sistem Jembatan
(a) Rangkaian Penyearah Gelombang Penuh Sistem Jembatan
(b) Sinyal Output

2.4.4 Penyearah dengan Regulator

Pada penyearah ini keluaran belum tentu stabil untuk memperoleh suatu catu daya dengan nilai keluaran yang tetap, maka dapat digunakan sebuah IC *regulator* 78xx untuk catu daya positif. (xx adalah nilai tegangan yang dikeluarkan dari regulator tersebut.), misalnya dengan memakai IC 7805 maka keluarannya akan bernilai 5 Volt, jika dengan menggunakanan IC 7812 maka keluarannya akan 12 Volt, sedangkan jika menggunakan IC 7809 maka keluarannya akan 9 Volt. Seperti contoh pada gambar di bawah ini.

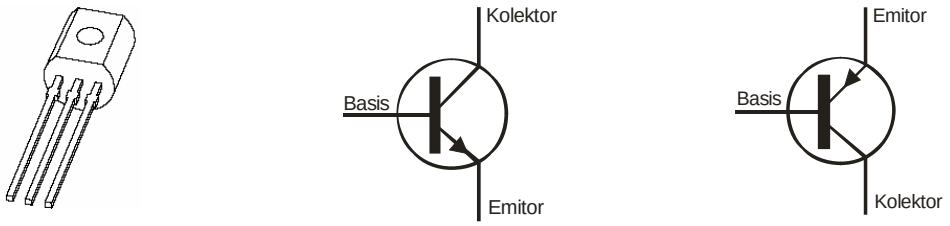


(a) *(b)*

Gambar 2.11 Penyearah dengan Regulator
(a) Rangkaian Penyearah dengan Regulator (b) Sinyal Output

2.5 Transistor

Transistor merupakan komponen aktif yaitu transistor mampu mengadakan penguatan atau penguatan sinyal. Simbol skematik untuk transistor *dwikutub* menggunakan arus *konvensional* pada transistor NPN seperti ditunjukan pada gambar di bawah ini panah emitor menunjuk keluar dari basis



(a) Bentuk Transistor *(b) Simbol Transistor NPN* *(b) Simbol Transistor PNP*

Gambar 2.12 Bentuk dan Simbol Transistor

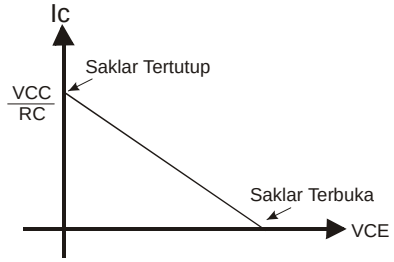
Penguatan dari transistor *dwikutub* terletak pada kenyataan bahwa arus kolektor tergatung pada arus basis ke emitor. Hal ini disebut penyearah (*biasing*).

Jika dipanjar dengan arah ini akan lebih banyak arus emitor (Ie) yang mengalir melalui transistor dari pada arus basis (Ib) atau arus kolektor (Ic). Kenyataanya arus emitor pada dasarnya adalah sama dengan jumlah dari kedua arus lainnya.

Ie = Ib + Ic ..... (2.3)

Transistor Sebagai Saklar

Transistor akan bekerja sebagai saklar dengan mengoperasikan transistor pada salah satu dari *saturasi* atau *titik sumbat*. Jika transistor berada dalam keadaan *saturasi*, transistor tersebut seperti saklar tertutup dari kolektor ke emitor. Jika transistor tersumbat (*cutt off*) maka transistor seperti sebuah saklar terbuka.



Gambar 2.13 Garis Beban DC

Pada titik sumbat arus basis nol dan arus kolektor sangat kecil sehingga dapat diabaikan maka digunakan suatu pendekatan bahwa tegangan kolektor, emitor sama dengan Vcc.

V_{CE} = Vcc ..... (2.4)

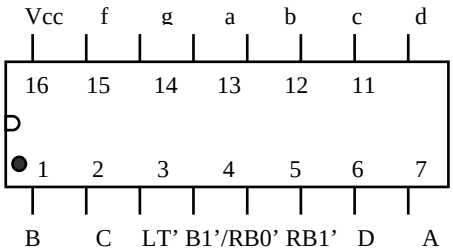
Pada titik *saturasi* arus basis sama dengan Ic (*sat*) dan arus kolektor adalah maksimum, digunakan suatu pendekatan arus kolektor pada *saturasi* adalah :

$$Ic\ (sat) = \frac{V_{CC}}{Rc} \quad \text{..... (2.5)}$$

2.6 IC TTL 74LS47 (Decoder)

IC TTL 74LS47 ialah komponen yang berfungsi untuk mengkodekan data biner sehingga dapat dibaca atau ditampilkan oleh *seven segment*. IC ini mempunyai 4 masukan *BCD* (8421) dan 7 keluaran. Cara kerja dari pengkodean IC 74LS47 dilakukan dengan 7 gerbang *AND* dan *OR* yang mempunyai keluaran kolektor terbuka dimasukan

ke kendali indikator *segment*. Setiap keluaran dari *decorder* akan menyalakan *seven segment* dengan kondisi rendah (0).



Gambar 2.14 Skema Pin IC TTL 7447

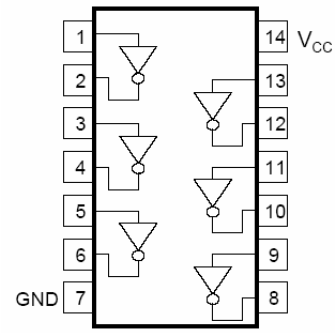
Berikut ini adalah tabel kebenaran dari *IC decoder* 7447.

Tabel 2.4 Tabel Kebenaran IC Decoder 7447

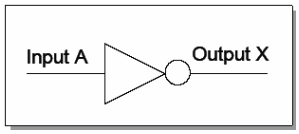
Desimal	Hexa Desimal	D	C	B	A	a	b	c	d	e	f	g	7-LED
0	0	0	0	0	0	0	0	0	0	0	0	1	
1	1	0	0	0	1	1	0	0	1	1	1	1	
2	2	0	0	1	0	0	0	1	0	0	1	0	
3	3	0	0	1	1	0	0	0	0	1	1	0	
4	4	0	1	0	0	1	0	0	1	1	0	0	
5	5	0	1	0	1	0	1	0	0	1	0	0	
6	6	0	1	1	0	1	1	0	0	0	0	0	
7	7	0	1	1	1	0	0	0	1	1	1	1	
8	8	1	0	0	0	0	0	0	0	0	0	0	
9	9	1	0	0	1	0	0	0	1	1	0	0	
10	A	1	0	1	0	1	1	1	0	0	1	0	
11	B	1	0	1	1	1	1	0	0	1	1	0	
12	C	1	1	0	0	1	0	1	1	1	0	0	
13	D	1	1	0	1	0	1	1	0	1	0	0	
14	E	1	1	1	0	1	1	1	0	0	0	0	
15	F	1	1	1	1	1	1	1	1	1	1	1	

2.7 Inverter

IC ini adalah IC TTL yang terdiri dari 6 *input* dan 6 *output* yang membentuk 6 buah gerbang *NOT*. IC ini digunakan untuk membalikan kondisi sinyal masukan tinggi (1) maka keluarannya akan berkondisi rendah (0) atau sebaliknya.



(a) Konfigurasi IC TTL 7404



(b) Simbol Gerbang Logika NOT

Gambar 2.15 Simbol IC 7404 dan Gerbang Logika NOT

Tabel 2.5. Tabel Kebenaran Gerbang NOT/ INVERTER

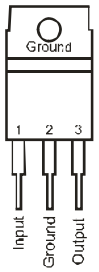
Input A	Output X
0	1
1	0

Keterangan operasi gerbang *NOT*:

- Jika *input* A *high*, maka kondisi X akan *low*
- Jika *input* A *low*, maka kondisi X akan *high*

2.8 IC Regulator

IC *regulator* mempunyai 3 buah kaki yang digunakan sebagai komponen pendukung dari Vcc untuk menghasilkan tegangan yang diinginkan. IC ini dinamai dengan 78xx untuk catu daya positif dan IC regulator 79xx untuk catu daya negatif. (xx adalah nilai tegangan yang dikeluarkan dari regulator tersebut.) IC *regulator* ini dapat berfungsi sebagai penstabil tegangan dan dapat bekerja dengan baik jika tegangan *input* (*Vin*) lebih besar dari pada tegangan *output* (*Vout*).



Gambar 2.16 Simbol IC Regulator

2.9 Resistor

Resistor merupakan komponen dasar elektronika yang berfungsi untuk membatasi arus pada rangkaian, bila tahanannya semakin besar maka semakin kecil pula arus dari keluaran tersebut, sebaliknya jika nilai hambatannya kecil maka akan besar pula arusnya. Resistor bersifat resistif dan umumnya terbuat dari bahan karbon. Dari hukum *Ohm* diketahui, resistansi berbanding terbalik dengan jumlah arus yang mengalir melaluinya. Satuan resistansi dari suatu resistor disebut *Ohm* (Ω).

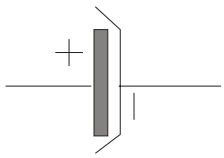
Tipe resistor yang sekarang banyak digunakan yaitu resistor dari karbon yang terdapat cicin warna yang dicatkan pada bahan resistor untuk menyatakan nilai hambatan.



Gambar 2.17 Simbol Resistor

2.10 Kapasitor

Kapasitor merupakan komponen yang dibuat untuk mendapatkan kapasitansi tertentu yaitu menyimpan muatan/ tegangan listrik atau menahan arus searah. Kapasitor ELCO terbuat dari keping alumunium dan elektrolit yang dikandung dalam lembaran kertas berpori. Plat alumunium bersifat sebagai *isolator* dan elektrolit berfungsi sebagai *konduktor*. Kapasitor ELCO mempunyai kutub atau polaritas yaitu tanda positif dan tanda negatif. Jika dalam pemasangan kutub-kutub ELCO terbalik maka kapasitor akan rusak. Untuk satuan dari ELCO ini adalah Mikro, kapasitor keramik adalah piko dan kapasitor milar adalah Nano.



Gambar 2.18 Simbol ELCO

2.11 Dioda

Dioda adalah suatu komponen elektronik yang dapat melewatkan arus pada satu arah saja. Ada berbagai macam dioda diantaranya dioda tabung, dioda sambungan P-N dan dioda kontak titik dan sebagainya.

Fungsi dioda sangat berperan penting pada rangkaian elektronika, diantaranya mempunyai fungsi untuk menghasilkan tegangan searah dari tegangan bolak-balik.

Bentuk dioda yang sering digunakan pada saat sekarang ini terdiri dari semikonduktor jenis P yang dibuat bersambung dengan semikonduktor jenis N.



Gambar 2.19 Simbol Dioda

2.12 Bahasa Assembly

Program Bahasa *Assembly* adalah sebuah program yang terdiri atas label-label, *mnemonic* dan lain sebagainya. Masing-masing pernyataan berhubungan dengan suatu instruksi mesin. Bahasa *Assembly* , sering juga disebut kode sumber (*source code*) atau kode simbolik (*Symbolic code*) tidak dapat dijalankan oleh prosesor.

Bahasa *Assembly* menggantikan kode-kode biner dari bahasa mesin dengan "*mnemonic"* yang mudah diingat. Misalnya sebuah instruksi penambahan dalam bahasa mesin disajikan dengan kode : "10110011" yang dalam bahasa *assembly* dapat disajikan dalam *mnemonic* ADD, sehingga mudah diingat di banding sederetan angka 0 dan 1.

2.12.1 Konstruksi Program Assembly

Program *assembly* terdiri dari kumpulan baris-baris perintah dan biasanya di simpan dengan *extension .ASM* dengan satu baris untuk satu perintah. Biasanya program bahasa *assembly* ini berisikan:

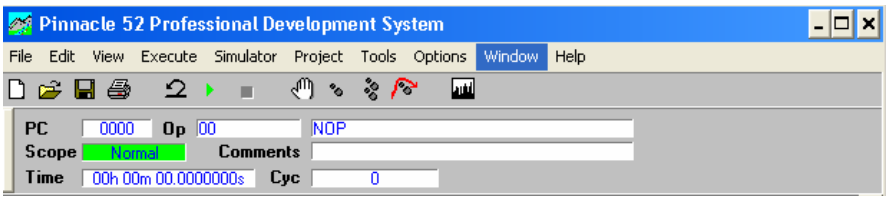
- Instruksi-instruksi mesin : (misalnya *MOV*)
- Pengarah-pengarah assembler yang merupakan instruksi ke program assembler yang mendefinisikan struktur program, simbol-simbol, data, konstanta dan lain-lainnya (misalnya : *ORG*)
- Kontrol-kontrol assembler yang mengatur (menentukan) mode-mode assembler dan aliran assembey langsung (misalnya *$TITLE*)
- Komentar-komentar. Komentar perlu dituliskan agar mudah di baca, tidak harus perinstruksi dapat juga sekumpulan instruksi yang mengerjakan suatu operasi.

Baris–baris program yang mengandung instruksi mesin atau pengarah assembler harus mengikuti aturan program assembler. Masing-masing baris atas beberapa *field* yang dipisahkan dengan spasi atau tabulasi. Format umumnya adalah:
[label:] mnemonic [operan] [,operan] […] [;komentar]

2.12.2 Program Aplikasi Pinnacle 52

Ada beberapa *software* yang dapat digunakan sebagai program Bantu untuk membuatdan mensimulasi program untuk AT89C51, diantaranya *Pinnacle 52*. program ini merupakan suatu program yang dapat mensimulasikan proses yang terjadi dalam prosesor AT89C51 dan menampilkan kondisi-kondisi pada memorinya serta dapat mengkompilasi program yang dibuat ke dalam bentuk OBJ, LST dan HEX

Berikut akan dijelaskan beberapa menu yang sering digunakan pada pembuatan dan mensimulasi program. Pada bagian atas program aplikasi ini terdapat beberapa menu utama, diantaranya; Menu *File, Edit, View, Execute, Simulator, Project, Tools, Options, Windows* dan *Help*



Gambar 2.20 Tampilan Menu Utama Program Pinnacle 52

2.12.2.1 Membuka dan Menyimpan Program

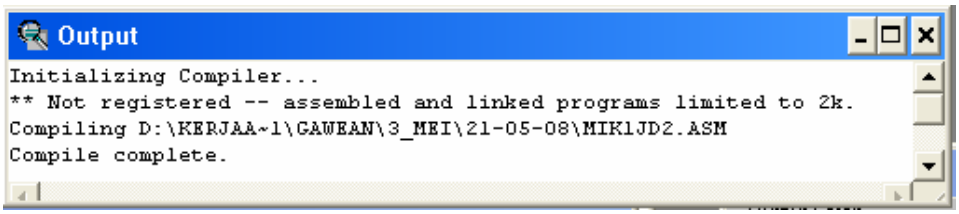
Pada menu file terdapat submenu sebagi berikut:

- New : membuat editor baru
- Open : membuka file HEX ataupun ASCII yang ada
- Close : menutup editor yang sedang aktif
- Save : menyimpan editor yang sedang aktif
- Save As : menyimpan editor yang sedang aktif kedalam nama file yang baru
- Save All : menyimpan seluruh editor yang sedang dibuka
- Print : mencetak file yang sedang aktif
- Exit : mengakhiri program *pinnacle*

Pembuatan program dilakukan pada editor. Setelah program selesai dibuat, sebaiknya program di simpan dan selanjutnya dilakukan kompilasi agar diperoleh file OBJ. atau langsung dilakukan kompilasi dan link agar diperoleh file *OBJ* dan *HEX* (file *HEX* digunakan untuk dimasukan ke dalam IC AT89C51).

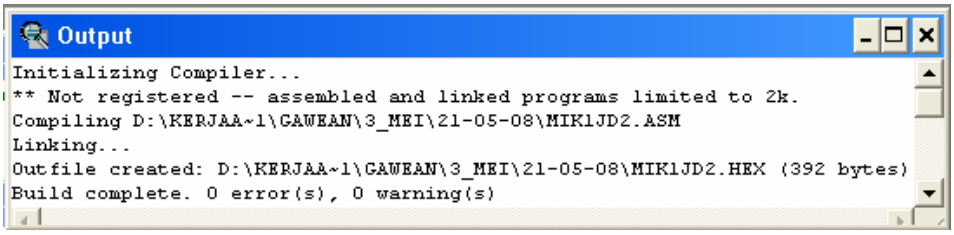
2.12.2.2 Kompilasi Program

Proses kompilasi dapat dilakukan melalui menu project kemudian pilih *Compile* <namafile.asm> atau langsung menekan tombol F2. Berikut contoh tampilan hasil kompilasi.



Gambar 2.21 Contoh Tampilan Kompile

Selain dengan cara di atas dapat dilakukan dengan cara menekan CTRL+F2. Cara ini untuk mengkompile dan sekaligus menghasilkan file berekstensi *HEX* yang akan di *download* oleh IC AT89C51. Berikut contoh kompile untuk hasil file *HEX*.

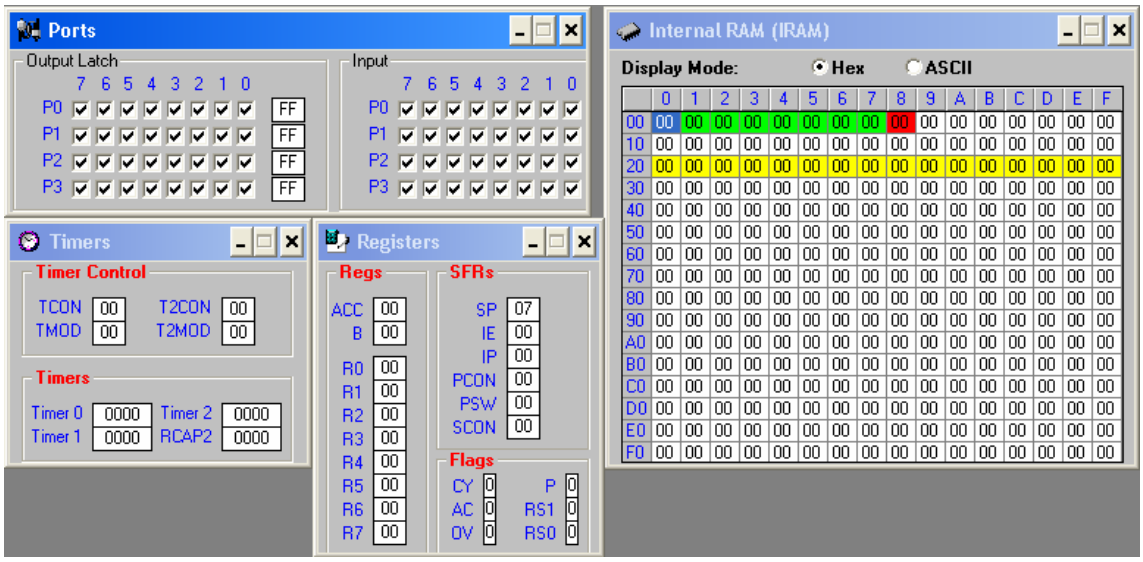


Gambar 2.22 Contoh Tampilan Kompile untuk File .HEX.

2.12.2.3 Simulasi Program

Program dapat disimulasikan melalui menu *execute* kemudian pilih *Run* atau langsung tekan tombol F5. Selain itu simulasi dapat dilakukan secara perinstruksi dengan menekan tombol F8 untuk setiap instruksi.

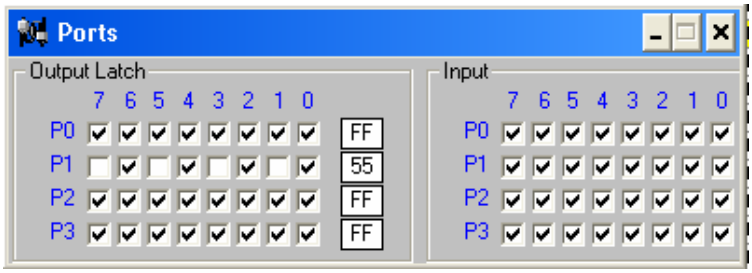
Menu *View* digunakan untuk menampilkan beberapa tampilan seperti *register*, *port*, *timer* dan *internal RAM*. Berikut beberapa tampilan yang dapat di panggil melalui menu View.



Gambar 2.23 Beberapa Tampilan dari Menu View

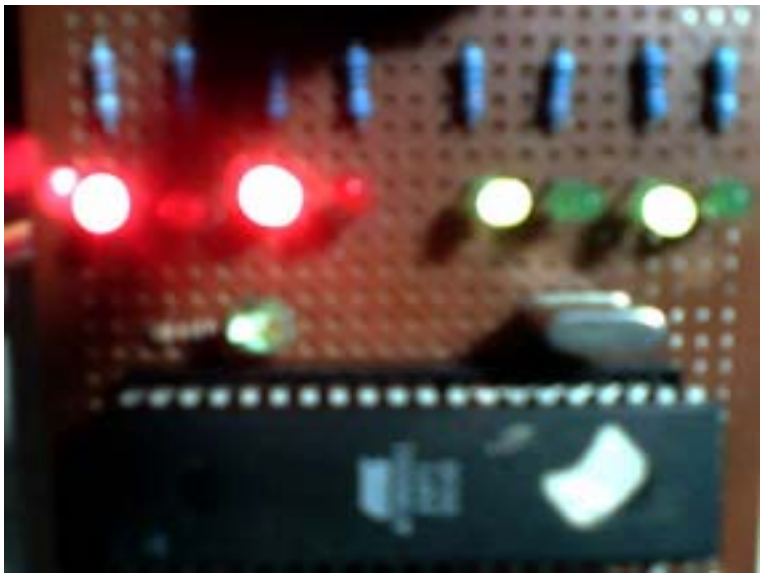
Berikut contoh simulasi program *Pinnacle 52*

```
Org 00h              ; alamat awal berada di 00h
Mulai:               ; label mulai
Mov p1,#01010101B    ; memasukan data 01010101b
Jmp mulai            ; kembali ke label mulai
End
```



Gambar 2.24 Contoh Simulasi Port pada Program Pinnacle 52

Pada potongan program di atas P1.1, P1.3, P1.5, P1.7 diberikan dengan logika 0, dan yang lainnya berlogika 1. Pada simulasi tersebut bit yang berlogika 0 artinya led menyala dan yang berlogika 1 led artinya tidak menyala. Pada gambar 2.25 dapat dilihat tampilan hasil simulasi pada alat.



Gambar 2.25 Tampilan Simulasi Program pada Alat

2.12.3 Instruksi MCS-51 yang Digunakan

Instruksi yang digunakan dalam pembuatan alat ini ialah sebagai berikut:

1. ORG (Origin)

   Digunakan untuk mengarahkan lokasi tempat instruksi yang ada dibawahnya
2. CLR

   Memberikan nilai “0” pada bit tertentu
3. SETB

   Memberikan nilai ”1” pada bit tertentu
4. CALL

   Merupakan instruksi untuk melakukan lompatan dengan area sebesar 2 KB
5. LCALL

   Instruksi ini hampir sama dengan instruksi *CALL*, hanya saja instruksi ini digunakan jika label yang di panggil letaknya lebih jauh dari 2 KB
6. MOV

   Instruksi ini berfungsi melakukan pemindahan data dari *variabel* pada kode operasi kedua dan disimpan di *variabel* pad kode operasi pertama.
7. JMP

   Digunakan untuk melakukan lompatan dan menjalankan program yang berada di alamat yang ditentukan oleh label tertentu.
8. SJMP

   Digunakan untuk melakukan lompatan untuk jarak yang pendek (*Short Jump*)

9. LJMP

   Instruksi ini berfungsi untuk melakukan lompatan untuk jarak yang jauh (*Long Jump*)

10. DJNZ

    Instruksi ini berfungsi melakukan pengurangan pada Rn (R0...R7) dengan 1 dan lompat ke alamat yang ditentukan apabila hasilnya bukan 00. Apabila hasilnya telah mencapai 00, maka program akan terus menjalankan instruksi dibawahnya.

11. CJNE

    Instruksi ini berfungsi melakukan perbandingan antara data tujuan dan data sumber serta melakukan lompatan ke alamat yang ditentukan jika hasil perbandingan tidak sama.

12. RET

    Digunakan untuk melakukan lompatan ke alamat yang di simpan dalam *SP* dan *SP*-1. Instruksi ini biasa digunakan pada saat kembali dari *subrutin* yang di panggil dengan instruksi *ACALL* atau *LCALL*.

13. INC

    Berfungsi menambahkan nilai variabel yang ditunjuk dengan 1 dan hasilnya di simpan di variabel tersebut.

14. END

    Instruksi ini biasanya diletakan di akhir baris dari file program sumber assembler sebagai tanda akhir pernyataan (*statement*) bagi program assembler dalam melakukan proses *assembly*.