BAB II
DASAR TEORI

Bab ini akan membahas mengenai teori dan komponen penunjang yang akan digunakan dalam pembuatan tugas akhir ini. Pembahasannya berisi tentang Mikrokontroler AT89S51, LCD LMB162A, IC EEPROM AT24C62, RS232, RS485, Bahasa Assembly dan Visual Basic 6.0 sebagai komponen instrumentasi.

2.1 Mikrokontroler AT89S51

Mikrokontroler AT89S51 termasuk dalam *MCS-51™* dari Intel. Sebuah mikrokontroler tidak dapat bekerja bila tidak diberi program kepadanya. Program tersebut memberitahukan mikrokontroler apa yang harus dilakukan. Salah satu keunggulan dari AT89S51 adalah dapat diisi ulang dengan program lain sebanyak 1000 kali pengisian. Instruktur-instruktur perangkat lunak berbeda untuk masing-masing jenis mikrokomtroler. Instruksi-instruksi hanya dapat dipahami oleh jenis mikrokontroler yang bersangkutan. Instruksi-instruksi dikenal sebagai bahasa pemrograman sistem mikrokontroler.

Sebuah mikrokontroler tidak dapat memahami instruksi-instruksi yang berlaku pada mikrokontroler lain. Sebagai contoh, mikrokontroler buatan Intel dengan mikrokontroler buatan Motorolla memiliki perangkat instruksi yang berbeda.

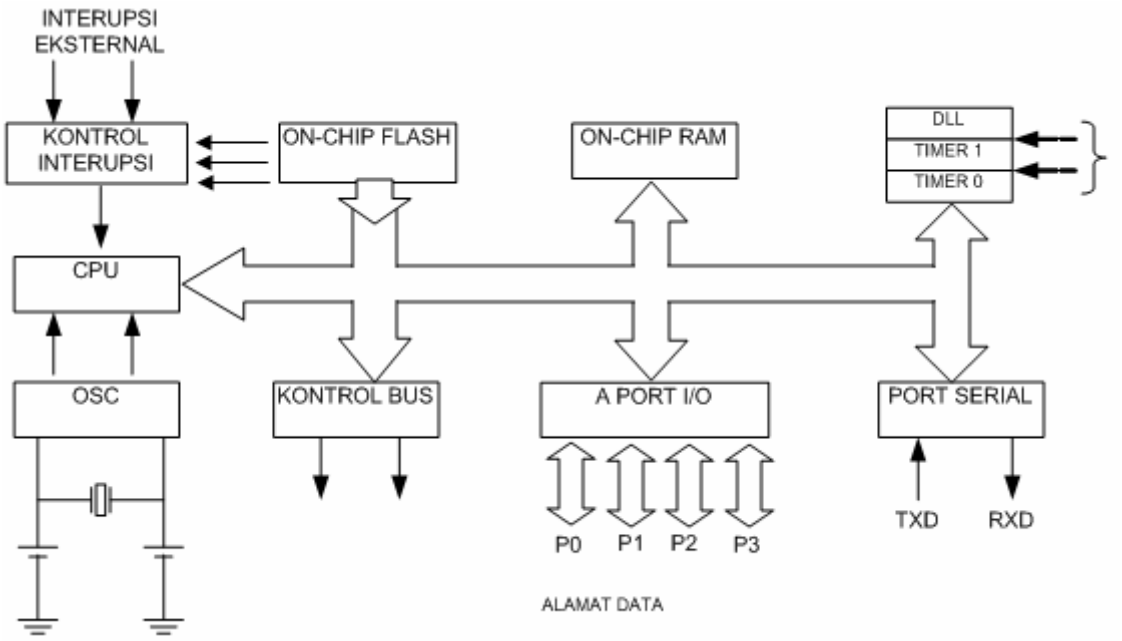
2.1.1 Karakteristik Mikrokontroler AT89S51

Mikrokontroler AT89S51 memiliki beberapa fasilitas, diantaranya sebagai berikut:

1. Sebuah CPU (*Central Prossesing Unit*) 8 bit yang termasuk keluarga Osilator *internal* dan rangkaian *timer*.
2. Flash memori 4 Kbyte.
3. RAM internal 128 byte.
4. Empat buah *programmable* port I/O, masing-masing terdiri atas 8 buah jalur I/O.
5. Lima buah jalur interupsi (2 buah interupsi eksternal dan 3 buah internal).
6. Sebuah port serial dengan kontrol *serial full duplex* UART.
7. Kemampuan melaksanakan operasi perkalian, pembagian dan operasi boolean (bit).

8. Kecepatan pelaksanaan interuksi per siklus 1 mikrodetik pada frekuensi clock 1 MHz.

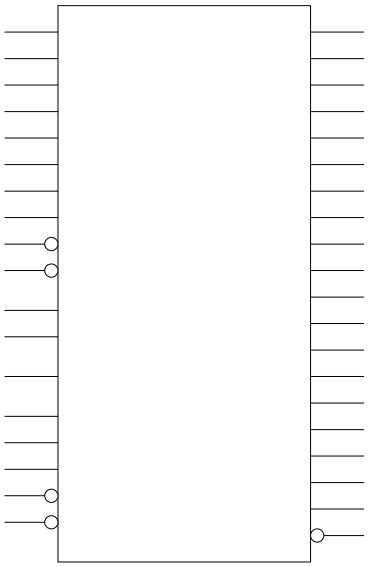
Dengan fasilitas seperti diatas, pembuatan alat menggunakan AT89S51 menjadi lebih sederhana dan tidak memerlukan IC pendukung yang banyak. Agar lebih jelasnya dapat dilihat pada gambar 2.1 dibawah ini.



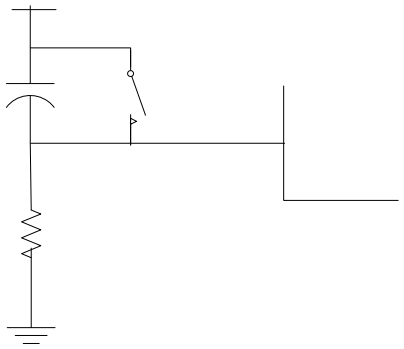
Gambar 2.1 Diagram Blok Mikrokontroler AT89S51

2.1.2 Deskripsi Pin AT89S51

Susunan pin-pin mikrokontroler AT89S51 memperlihatkan pada gambar 2.2 dibawah ini. Penjelasan dari masing-masing pin adalah sebagai berikut :



Gambar 2.2 Diagram Pin Mikrokontroler AT89S51



Gambar 2.3 Rangkaian Reset

1. Pin 1 sampai 8 (Port 1) merupakan port pararel 8 bit dua arah (*output-input*) yang dapat digunakan untuk berbagai keperluan (*general purpose*).
2. Pin 9 (Reset) adalah input reset (aktif tinggi). Pulsa transisi dari rendah ke tinggi akan me-reset AT89S51. pin ini dihubungkan dengan rangkaian power on reset seperti ditunjukan pada gambar 2.3 diatas.
3. Pin 10 sampai 17 (Port 3) adalah port pararel 8 bit dua arah (output-input) yang memiliki fungsi pengganti. Fungsi pengganti meliputi TxD (*Transmision Data*), RxD (*Received Data*), Int 0 (*Interupsi 0*), Int 1 (*Interupsi 1*), T0 (*Timer 0*), T1 (*Timer 1*), WR (*Write*) dan RD (*Read*). Bila fungsi pengganti tidak dipakai, pin-pin ini dapat digunakan sebagai *port* pararel 8 bit serba guna.
4. Pin 18 dan 19 (XTAL 1 dan XTAL 2) adalah pin *input* kristal, yang merupakan *input clock* bagi rangkaian osilator *internal*.
5. Pin 20 (*Ground*) dihubungkan ke Vss atau Ground.
6. Pin 21 sampai 28 (Port 2) adalah *port* pararel 2 selebar 8 bit dua arah. Port 2 ini mengirimkan byte alamat bila dilakukan pengaksesan memori eksternal.
7. Pin 29 adalah pin PSEN (*Program Strobe Enable*) yang merupakan sinyal pengontrol yang membolehkan program memori eksternal masuk ke dalam bus selama proses pemberian/pengambilan instruksi.
8. Pin 30 adalah pin output ALE (*Address Latch Enable*) yang digunakan untuk menahan alamat memori eksternal selama pelaksanaan instruksi.
9. Pin 31 (EA). Bila pin ini diberi logika tinggi, maka mikrokontroler akan melaksanakan instruksi dari ROM ketika isi program counter kurang dari 4096. Bila diberi logika rendah, maka mikrokontroler akan melaksanakan seluruh instruksi dari memori program diluar.

10. Pin 32 sampai 39 (*Port 0*) merupakan *port* pararel 8 bit open drain dua arah. Bila diberi logika rendah, mikrokontroler akan melaksanakan seluruh instruksi dari memori program luar.
11. Pin 40 (Vcc) dihubungkan ke Vcc (+5 Volt).

2.1.3 Organisasi Memori

Semua mikrokontroler dalam keluarga MCS-51 memiliki pembagian ruang alamat (*address space*) untuk program dan data. Pemisahan memori program dan memori data membolehkan memori data untuk diakses oleh alamat 8 bit. Meskipun demikain, alamat data memori 16 bit dapat dihasilkan melalui register DPTR (Data Pointer Register).

Memori program hanya dapat dibaca tidak bisa ditulis, karena disimpan dalam *Flash Memori*. Memori program sebesar 64 Kbyte dapat dimasukkan dalam EPROM eksternal. Seperti tampak pada gambar 2.4

Sinyal yang membolehkan pembacaan dari memori program eksternal adalah pin PSEN. Pada gambar 2.5 memperlihatkan memori data yang terletak pada ruang alamat terpisah dari memori program. RAM ekternal 64 Kbyte dapat dialamati dalam ruang memori data eksternal. CPU menghasilkan sinyal *read* dan *write* selama menghubungi memori data eksternal.

Mikrokontroler AT89S51 memiliki 5 buah ruang alamat, yaitu :

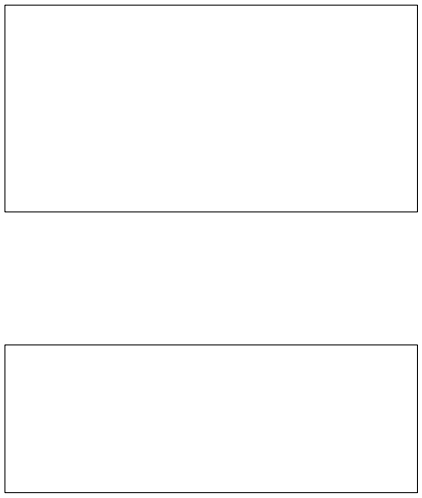
(a) Ruang alamat kode (*Code Address Space*) sebanyak 64 Kbyte, yang seluruhnya merupakan ruang alamat kode eksternal.

(b) Ruang alamat memori data *internal* yang dapat dialamati secara langsung, yang terdiri atas :
   1. RAM sebanyak 128 byte
   2. Hardware register sebanyak 128 byte

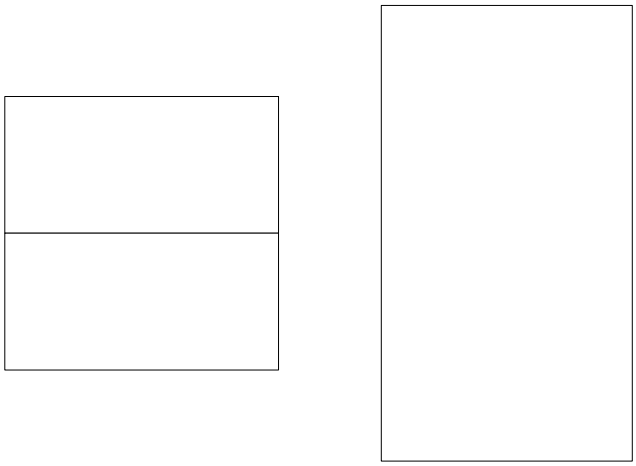
(c) Ruang alamat memori data *internal* yang dialamati secara tidak langsung sebanyak 128 byte, seluruhnya diakses dengan pengalamatan tidak langsung.

(d) Ruang alamat memori data eksternal sebanyak 64 Kbyte yang dapat ditambahkan oleh pemakai.

(e) Ruang alamat bit. Dapat diakses dengan pengalamatan langsung.



Gambar 2.4 Struktur Program Memori AT89S51



Gambar 2.5 Struktur Data Memori AT89S51

2.1.4 Special Function Register (SFR)

SFR berisi register-register dengan fungsi tertentu yang disediakan oleh mikrokontroler seperti timer dan lain-lainnya. AT89S51 memiliki 21 SFR yang terletak pada memori 80H-FFH. Masing-masing ditunjukkan pada tabel 2.1 yang meliputi simbol, nama dan alamatnya.

Tabel 2.1 *Spesial Function Register*

Simbol	Nama	Alamat
ACC	Akumulator	E0H
B	*B register*	F0H
PSW	*Program Status Word*	D0H
SP	*Stack Pointer*	81H
DPTR	*Data Pointer* 16 bit *DPL byte* rendah *DPH byte* tinggi	 82H 83H
P0	*Port 0*	80H
P1	*Port 1*	90H
P2	*Port 2*	A0H
P3	*Port 3*	B0H
IP	*Interupt Priority Control*	B8H
IE	*Interupt Enable Control*	A8H
TMOD	*Timer/Counter Mode Control*	89H
TCON	*Timer/Counter Control*	88H
TH 0	*Timer/Counter High Low byte*	8CH
TL 0	*Timer/Counter Low byte*	8AH
TH 1	*Timer/Counter High byte*	8DH
TH 1	*Timer/Counter Low byte*	8BH
SCON	*Serial Control*	98H
SBUF	*Serial Data Buffer*	99H
PCON	*Power Control*	87H

2.1.5 Timer AT89S51

Mikrokontroler AT89S51 mempunyai dua buah timer, yaitu Timer 0 dan Timer 1, setiap timer terdiri dari 16 bit timer yang tersimpan dalam dua buah register yaitu THx untuk *Timer High Byte* dan TLx untuk *Timer Low Byte* yang keduanya dapat berfungsi sebagai *counter* maupun sebagai timer. Secara fisik timer juga merupakan rangkaian T flip-flop yang dapat diaktifkan dan dinonaktifkan setiap saat. Perbedaan keduanya terletak pada sumber *clock* dan aplikasinya.

Timer mempunyai sumber *clock* dengan frekuensi tertentu yang sudah pasti sedangkan *counter* mendapat sumber *clock* dari pulsa yang hendak dihitung jumlahnya.

Aplikasi dari timer atau pewaktu biasa digunakan untuk aplikasi menghitung lamanya suatu kejadian yang terjadi sedangkan *counter* atau penghitung biasa digunakan untuk aplikasi menghitung jumlah kejadian yang terjadi dalam periode tertentu. Perilaku dari register THx dan TLx diatur oleh register TMOD dan TCON. Timer dapat diaktifkan melalui perangkat keras maupun perangkat lunak.

Periode waktu *timer/counter* dapat dihitung menggunakan rumus 2.1 dan 2.2 sebagai berikut.

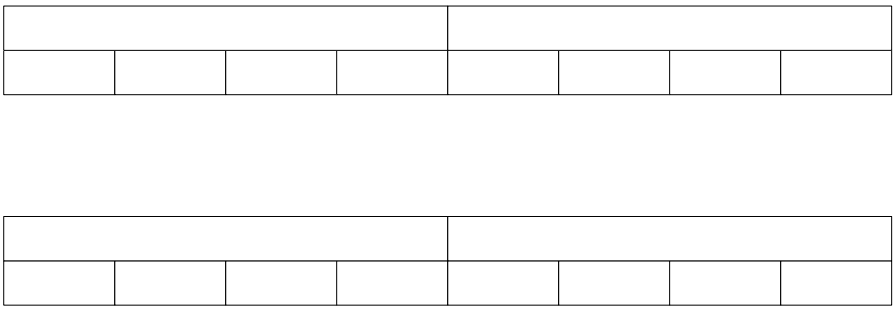
Sebagai *timer/counter* 8 bit

$$T = (255 - TLx) * \frac{12}{frekuensiXTAL}\,\mu s \quad \textbf{.....(2.1)}$$

Sebagai *timer/counter* 16 bit

$$T = (65535 - THxTLx) * \frac{12}{frekuensiXTAL}\,\mu s \quad \textbf{.....(2.2)}$$

Di mana : THx = isi register TH0 atau TH1 dan TLx = isi register TL0 atau TL1.



Gambar 2.6 Register TCON dan TMOD

Pengontrolan kerja *timer/counter* diatur oleh register TCON. Register ini bersifat *bit addresable* 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 gambar 2.6 dan 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 nonaktif (*clear*)
TF0	*Timer* 0 *overflow flag* yang akan diset jika *timer overflow*
TR0	Membuat timer 0 aktif (*set)* dan nonaktif (*clear*)

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

Gate : Pada saat TRx = 1, *timer* akan berjalan tanpa memperlihatkan nilai pada Gate (timer dikontrol software).

C/T : Pemilihan fungsi timer (0) atau counter (1).

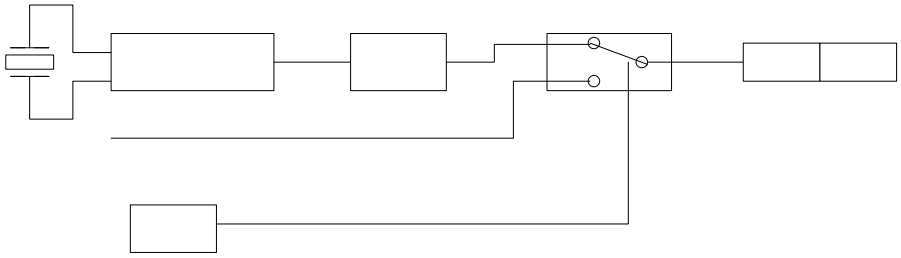
M1 & M0 : Untuk memilih mode timer dengan variasi seperti pada tabel 2.3.

Tabel 2.3 *Mode Timer*

M1	M0	Mode	Operasi
0	0	0	Timer 13 bit
0	1	1	Timer/Counter 16 bit
1	0	2	Timer 8 bit di mana nilai timer tersimpan pada TLx. Register THx berisi nilai isi ulang yang akan dikirim ke TLx setiap *overflow*.
1	1	3	Pada mode ini, AT89S51 bagaikan memiliki 3 buah timer. Timer 0 terpisah menjadi 2 buah timer 8 bit (TL0-TF0 dan TH0-TF1) dan timer tetap 16 bit.

2.1.5.1 Prinsip Kerja Timer

Pada gambar 2.7 *Timer* mempunyai dua buah sumber *clock* untuk beroperasi, yaitu sumber *clock internal* dan sumber *clock* eksternal. Jika *timer* menggunakan sumber *clock* eksternal, maka bit C/T harus di-set atau berkondisi *high*, saklar akan menghubungkan sumber *clock timer* ke pin Tx (T0 untuk timer 0, T1 untuk timer 1). Apabila sumber *clock* internal digunakan, *input clock* berasal dari osilator yang telah dibagi 12. Maka bit C/T harus di-clear atau berkondisi *low* sehingga saklar akan menghubungkan sumber *clock timer* ke osilator yang telah dibagi 12.



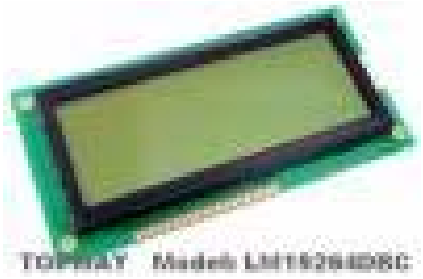
Gambar 2.7 Operasi *Timer*

2.2 LCD (Liquid Crystal Display)

LMB162A adalah modul LCD matrix dengan konfigurasi 16 karakter dan 2 baris dengan setiap karakternya dibentuk oleh 8 baris pixel dan 5 kolom pixel (1 baris terakhir adalah kursor).

Memori LCD terdiri dari 9.920 bir CGROM, 64 byte CGRAM dan 80x8 bit DDRAM yang diatur pengalamatannya oleh *Address Counter* dan akses datanya (pembacaan maupun penulisan datanya) dilakukan melalui Register Data.

Pada LMB162A terdapat Register Data dan Register Perintah. Proses akses data ke atau dari Register Data akan mengakses ke CGRAM, DDRAM atau CGROM bergantung pada kondisi Address Counter, sedangkan proses akses data ke atau dari Register Perintah akan mengakses *Instruction Decoder* (dekoder instruksi) yang akan menentukan perintah–perintah yang akan dilakukan oleh LCD. Bentuk LCD LMB162A diperlihatkan pada gambar 2.8.



Gambar 2.8 LCD LMB162A

2.2.1 Deskripsi Pin LCD

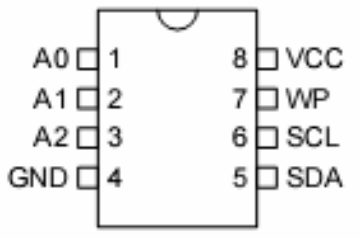
Untuk keperluan antar muka suatu komponen elektronika dengan mikrokontroler, perlu diketahui fungsi dari setiap kaki yang ada pada komponen tersebut.

a. Kaki 1 (GND) : Kaki ini berhubungan dengan tegangan +5 Volt yang merupakan tegangan untuk sumber daya.

b. Kaki 2 (VCC) : Kaki ini berhubungan dengan tegangan 0 volt (*Ground*).
c. Kaki 3 (VEE/VLCD) : Tegangan pengatur kontras LCD, kaki ini terhubung pada cermet. Kontras mencapai nilai maksimum pada saat kondisi kaki ini pada tegangan 0 volt.
d. Kaki 4 (RS) : *Register Select*, kaki pemilih register yang akan diakses. Untuk akses ke Register Data, logika dari kaki ini adalah 1 dan untuk akses ke Register Perintah, logika dari kaki ini adalah 0.
e. Kaki 5 (R/W) : Logika 1 pada kaki ini menunjukan bahwa modul LCD sedang pada mode pembacaan dan logika 0 menunjukan bahwa modul LCD sedang pada mode penulisan. Untuk aplikasi yang tidak memerlukan pembacaan data pada modul LCD, kaki ini dapat dihubungkan langsung ke *Ground*.
f. Kaki 6 (E) : *Enable Clock LCD*, kaki mengaktifkan *clock* LCD. Logika 1 pada kaki ini diberikan pada saat penulisan atau pembacaan data.
g. Kaki 7 – 14 (D0 – D7) : Data bus, kedelapan kak LCD ini adalah bagian di mana aliran data sebanyak 4 bit ataupun 8 bit mengalir saat proses penulisan maupun pembacaan data.
h. Kaki 15 (Anoda) :Berfungsi untuk tegangan positif dari *backlight* LCD sekitar 4,5 volt (hanya terdapat untuk LCD yang memiliki *backlight*)
i. Kaki 16 (Katoda) : Tegangan negatif backlight LCD sebesar 0 volt (hanya terdapat pada LCD yang memiliki *backlight*).

2.3 IC 24C64 (EEPROM)

24CXX *programmer* adalah program yang digunakan untuk membaca dan menulis data pada IC I2C serial EEPROM 24CXX. XX merupakan angka yang mengindikasikan kapasitas serial EEPROM itu dalam satuan KiloBit. Contoh, 24C64 merupakan IC I2C serial EEPROM berkapasitas 64 KiloBit. Gambar 2.9 adalah IC I2C serial EEPROM 24CXX dari ATMEL. Huruf AT merupakan kode pabrik dari ATMEL.



Gambar 2.9 IC I2C Serial EEPROM 24CXX dari ATMEL

Tabel 2.4 Keluarga IC I2C serial EEPROM 24CXX dari ATMEL

Type IC	Kapasitas (Byte)	Penomoran Chip	Maksimal Chip Per Bus
AT24C01	128	Tidak Ada	1
AT24C01A	128	A0,A1,A2	8
AT24C02	256	A0,A1,A2	8
AT24C04	512	A1,A2	4
AT24C08	1024	A2	2
AT24C16	2048	Tidak Ada	1
AT24C164	2048	A0,A1,A2	8
AT24C32	4096	A0,A1,A2	8
AT24C64	8192	A0,A1,A2	8

Keluarga AT24CXX mempunyai kaki IC yang sama seperti terlihat pada gambar 2.9. Fungsi – fungsi kaki pada IC AT24CXX ini adalah sebagai berikut:

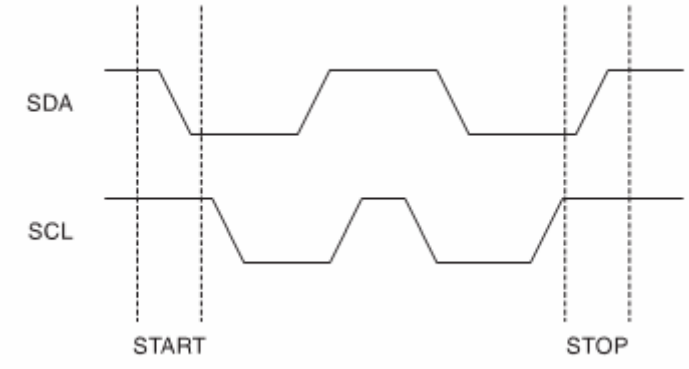
1. SDA (*Serial Data / Address*) adalah saluran dua arah yang digunakan untuk melakukan transfer data ke/dari IC AT24CXX.
2. SCL (*Serial Clock*) merupakan kaki input yang digunakan untuk sinyal *clock* IC AT24CXX.
3. WP (*Write Protect*), jika kaki ini dihubungkan ke VCC, maka IC AT24CXX hanya bisa dibaca. Isinya tidak dapat diganti. Jika kaki ini dihubungkan ke GND, maka operasi baca/tulis pada IC ini dapat dilakukan.
4. A1,A2,A3 adalah kaki–kaki untuk pengalamatan *chip*, hal ini digunakan jika dalam satu rangkaian digunakan lebih dari satu IC EEPROM sejenis.

2.3.1 Sinyal – Sinyal Dasar dan Protokol I2C Bus

Inter–Integrated Circuit atau sering disebut I2C adalah standar komunikasi serial dua arah menggunakan 2 saluran yang didesain khusus untuk pengontrolan IC (*Integrated Circuit*). Sistem I2C terdiri dari saluran SCL (*Serial Clock*) dan saluran SDA (*Serial Data*) yang membawa informasi data antara IC dengan pengontrolnya. Peranti yang dihubungkan dengan sistem I2C bus dapat dioperasikan sebagai *Master* atau *Slave*. *Master* adalah peranti yang memulai transfer data pada I2C bus dengan

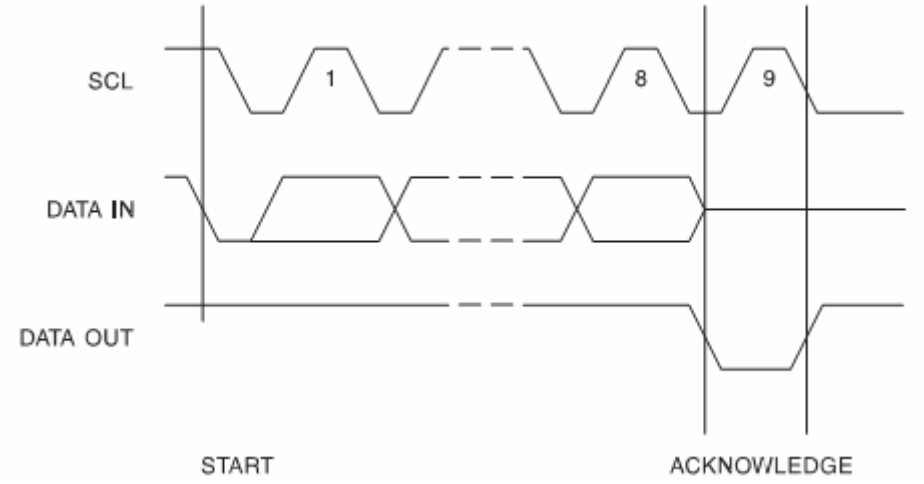
membentuk sinyal *start*, mengakhiri transfer data dengan membentuk sinyal *stop*, dan membangkitkan sinyal *clock*. *Slave* adalah peranti yang dialamati *master*.

Sinyal *start* merupakan sinyal untuk memulai semua perintah, didefinisikan sebagai perubahan tegangan SDA dari ‘1’ menjadi ‘0’ pada saat SCL=’1’. Sinyal *stop* merupakan sinyal untuk mengakhiri semua perintah, didefinisikan sebagai perubahan tegangan SDA dari ‘0’ menjadi ‘1’ pada saat SCL=’1’. Kondisi sinyal *start* dan sinyal *stop* seperti tampak pada gambar 2.10.



Gambar 2.10 Kondisi sinyal *Start* dan sinyal *Stop*

Sinyal dasar yang lain dalam I2C bus adalah sinyal *Acknowledge* yang disimbolkan dengan ACK. Setelah transfer data oleh *master* berhasil diterima oleh *slave*, *slave* akan menjawab dengan mengirim sinyal *Acknowledge*, yaitu dengan membuat SDA menjadi ‘0’ selama siklus *clock* ke-9. Ini menunjukkan bahwa *slave* telah menerima 8 bit data dari *master*. Kondisi sinyal *Acknowledge* seperti tampak pada gambar 2.11.

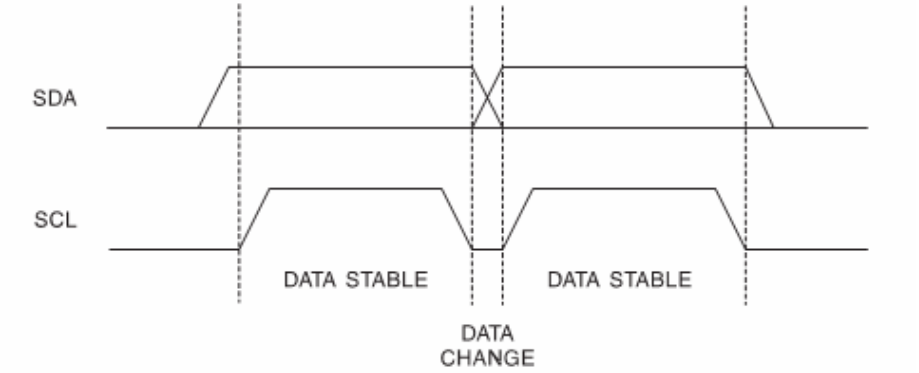


Gambar 2.11 Kondisi Sinyal *Acknowledge*

Dalam melakukan transfer data pada I2C bus, kita harus mengikuti tata cara yang telah ditetapkan yaitu:

1. Transfer data hanya bisa dilakukan ketika bus tidak dalam kondisi sibuk.
2. Selama proses transfer data, keadaan data pada SDA harus stabil selama SCL dalam kondisi tinggi. Keadaan perubahan ‘1’ atau ‘0’ pada SDA hanya dapat dilakukan selama SCL dalam kondisi rendah. Jika terjadi perubahan keadaan SDA pada saat SCL dalam kondisi tinggi, maka perubahan itu akan dianggap sebagai sinyal *start* atau sinyal *stop.*

Kondisi keadaan saluran untuk transfer data seperti tampak pada gambar 2.12.



Gambar 2.12 Kondisi transfer data pada I2C Bus

2.3.2 Pengalamatan IC I2C Serial EEPROM 24CXX

Setelah *master* mengirimkan sinyal *start* sebagai tanda awal transmisi, *master* kemudian mengirimkan ALAMAT SLAVE ke peranti *slave* yang ingin dituju, dalam hal ini IC I2C serial EEPROM. Empat bit pertama adalah Nomor Group yang untuk IC serial EEPROM adalah 1010. Pengalamatan ini telah ditetapkan oleh perusahaan Philips sebagai pencipta I2C. Untuk kelompok IC serial EEPROM Nomor Group adalah 1010. Tiga bit selanjutnya (A2,A1,A0) adalah Nomor *Chip* yang digunakan untuk menentukan IC mana yang akan diakses oleh *master*. Hal ini terutama digunakan jika dalam satu rangkaian terdapat lebih dari satu IC EEPROM yang sejenis. Bit terakhir digunakan untuk menentukan operasi baca atau operasi tulis yang akan dibentuk. Bit ini diisi 1 untuk membentuk operasi tulis. Setelah *master* mengirimkan sinyal *start* dan ALAMAT SLAVE ke peranti *slave* yang dituju, *slave* akan merespon dengan mengirimkan sinyal *Acknowledge* ke *master*.

Setelah *slave* mengirimkan sinyal *Acknowledge*, *master* kemudian mengirimkan BYTE ALAMAT yang digunakan untuk menentukan alamat memori fisik pada *slave*

yang ingin diakses oleh *master*. Jika kapasitas data dalam IC 24CXX tidak cukup dengan pengalamatan 8 *bit*, maka dipakai metode pengalamatan 11 bit (8 *bit* N0…N7dan 3 *bit* A0,A1,A2). Ini bisa dipakai untuk mengalamati sampai kapasitas 2048 *byte*, yakni dipakai pada AT24C16. Untuk kapasitas yang lebih besar, dipakai metode pengalamatan 16 *bit* yang dikirimkan pada *byte* kedua dan *byte* ketiga setelah sinyal *start*, yang memakai cara ini adalah AT24C164, AT24C32, dan AT24C64.

2.3.3 Proses Pengisian IC I2C Serial EEPROM 24CXX

Setelah *master* mengirimkan sinyal *start* dan ALAMAT SLAVE (dengan *bit* R/W diisi 0) kepada IC serial EEPROM, IC 24CXX kemudian akan mengirimkan sinyal *Acknowledge*. *Master* kemudian mengirimkan BYTE ALAMAT yang digunakan untuk menetukan alamat memori fisik yang akan diakses. Setelah menerima sinyal *Acknowledge* lagi, *master* kemudian mengirimkan BYTE DATA yang akan ditulis pada lokasi memori seperti pada BYTE ALAMAT. Sekali lagi IC 24CXX akan mengirimkan sinyal *Acknowledge* dan *master* akan membangkitkan sinyal *stop* untuk mengakhiri komunikasi. Proses pengisian *internal* akan dilakukan oleh IC 24CXX. Selama proses pengisian *internal* ini, IC 24CXX tidak akan melayani jika ada permintaan dari *master* sampai proses pengisian *internal* selesai.

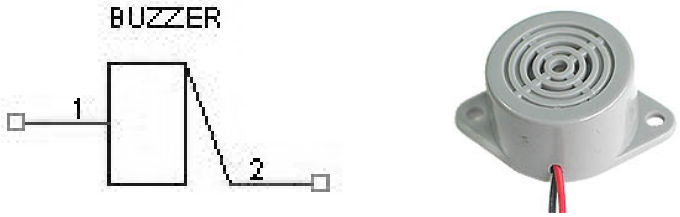
2.3.4 Proses Pembacaan IC I2C Serial EEPROM 24CXX

Proses pembacaan data dari 24CXX dilakukan mula–mula dengan mengirimkan operasi ‘dummy’ ke serial EEPROM yang akan dibaca isinya. Proses ini mirip dengan bagian awal pengisian serial EEPROM yang dibahas di atas. Setelah itu *master* mengirimkan sinyal *start* sekali lagi, disusul dengan perintah untuk membaca isi serial EEPROM yaitu dengan mengisi *bit* R/W dengan 1. IC serial EEPROM selanjutnya akan merespon dengan mengirimkan sinyal *Acknowledge* dan BYTE DATA yang diinginkan *master*. Selesai membaca isi serial EEPROM, *master* menutup komunikasi dengan mengirimkan sinyal *stop*.

2.4 Buzzer

Buzzer adalah sebuah komponen elektronika yang berfungsi untuk mengubah getaran listrik menjadi getaran suara. Pada dasarnya prinsip kerja *buzzer* hampir sama dengan *loud speaker*, jadi *buzzer* juga terdiri dari kumparan yang terpasang pada *diafragma* dan kemudian kumparan tersebut dialiri arus sehingga menjadi

elektromagnet, kumparan tadi akan tertarik ke dalam atau keluar, tergantung dari arah arus dan polaritas magnetnya, karena kumparan dipasang pada *diafragma* maka setiap gerakan kumparan akan menggerakkan *diafragma* secara bolak-balik sehingga membuat udara bergetar yang akan menghasilkan suara. *Buzzer* biasa digunakan sebagai indikator bahwa proses telah selesai atau terjadi suatu kesalahan pada sebuah alat (*alarm*). Pada gambar 2.13.a tampak simbol dari *buzzer* sedangkan bentuk dari *buzzer* tampak pada gambar 2.13.b

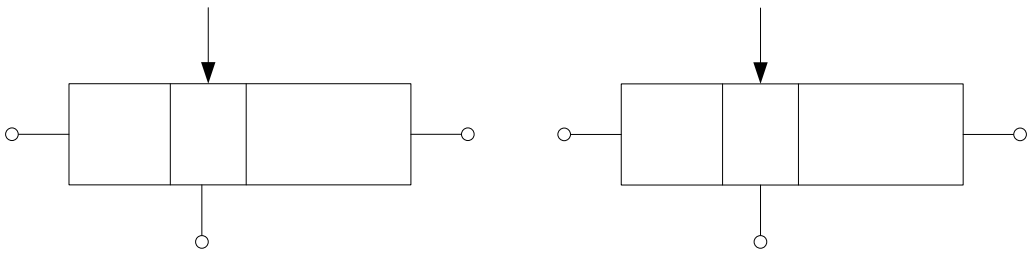


(a) *(b)*

Gambar 2.13 *a. Simbol buzzer, b. Bentuk Buzzer*

2.5 Transistor Sebagai Saklar

Transistor adalah komponen elektronik yang mempunyai fungsi sebagai penguat sinyal kecil. Pada umumnya transistor ini disebut juga sebagai transistor *bipolar* karena bekerja dengan 2 (B_1) muatan yang berbeda yaitu elektron sebagai pembawa muatan *negatif* dan hole sebagai pembawa muatan *positif.* Jenis-jenis transistor *bipolar* dapat dilihat pada gambar 2.14

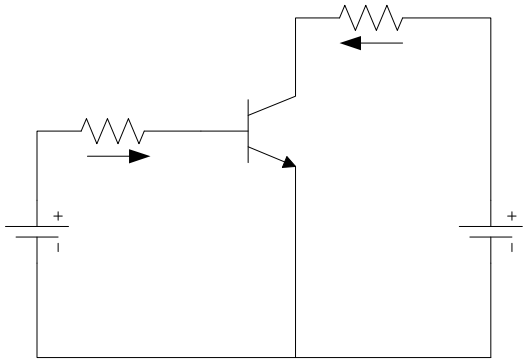


(a) (b)

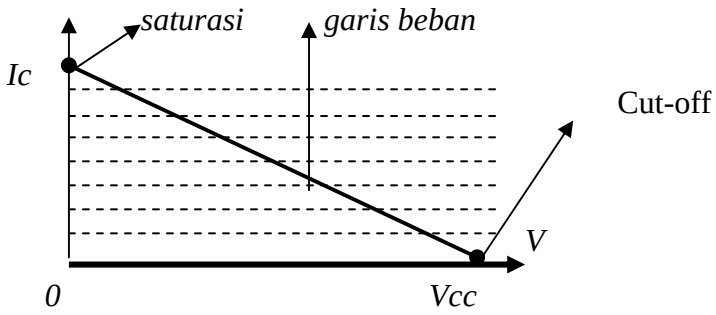
Gambar 2.14 (a) Transistor npn, (b) Transistor pnp

Transistor pada gambar 2.14(a) mempunyai dua sambungan (*junction*), satu di antaranya emiter-basis dan basis-kolektor. Karena inilah, sebuah transistor sama seperti dua buah dioda. Transistor npn disebut juga dioda emiter basis atau singkatnya dioda emiter. Transistor pnp juga disebut dioda kolektor basis atau dioda kolektor.

Gambar 2.14(b) menunjukkan kemungkinan yang lain, yaitu sebuah transistor pnp. Transistor pnp merupakan komplemen dari transistor npn. Pembawa muatan mayoritas pada emiter adalah *hole*, sebagai pengganti dari muatan bebas. Ini berarti, pada transistor pnp dibutuhkan arus dan tegangan yang berlawanan dengan transistor npn.



Gambar 2.15 (a) Rangkaian bias basis



(b) Garis beban dc

Gambar 2.15(a) memperlihatkan rangkaian bias basis. Sebuah sumber tegangan V_{BB} membias maju emiter melalui resistor R_B yang juga berfungsi membatasi arus. Penjumlahan tegangan disekitar *loop input* memberikan :

$$I_B R_B + V_{BE} - V_{BB} = 0 \quad \textbf{.....(2.3)}$$

Sehingga arus bias pada basis adalah :

$$I_B = \frac{V_{BB} - V_{BE}}{R_B} \quad \textbf{.....(2.4)}$$

Dengan V_{BE} = 0,7 V untuk transistor silikon dan 0,3 V untuk germanium. Dalam rangkaian kolektor, sumber tegangan Vcc membias balik dioda kolektor melalui Rc. Persamaan tegangan kolektor emiter dapat diperoleh melalui hukum ohm, yaitu :

$$V_{CE} = V_{CC} - I_C R_C \quad \textbf{.....(2.5)}$$

Dalam rangkaian bias basis yang diperlihatkan gambar 2.15(a), Vcc dam R_C adalah konstan. Pada persamaan 2.5 apabila disederhanakan akan dapat ditentukan besarnya arus Ic, seperti terlihat pada persamaan 2.6

$$I_C = \frac{V_{CC} - C_{CE}}{R_C} \quad \textbf{.....(2.6)}$$

Gambar 2.15(b) menunjukan grafik dari persamaan 2.6 memotong kurva dari kolektor. Perpotongan vertikal adalah Vcc/Rc dan perpotongan horizontal pada Vcc. Garis ini disebut garis *beban dc* karena garis ini menyatakan semua titik operasi yang mungkin. Perpotongan dari garis beban dc dengan arus basis adalah titik operasi kerja dari transistor. Titik di mana garis beban memotong kurva $I_B = 0$ disebut titik sumbat (*cut-off*). Pada titik ini, arus basis nol dan arus kolektor sangat kecil, sehingga dapat diabaikan (hanya ada arus bocor I_{CE0}). Pada titik sumbat, dioda emiter tidak lagi dibias maju dan transistor kehilangan kerja normalnya. Untuk itu digunakan suatu pendekatan, bahwa tegangan kolektor emiter adalah :

$$V_{CE(cut-off)} = V_{CC} \quad \textbf{.....(2.7)}$$

Perpotongan dari garis beban dan kurva $I_B = I_{B(sat)}$ disebut saturasi. Pada titik ini, arus basis sama dengan $I_{B(sat)}$ dan arus kolektor adalah maksimum. Pada saturasi, dioda kolektor tidak lagi dibias balik dan transistor kehilangan kerja normalnya. Untuk itu digunakan suatu pendekatan bahwa arus kolektor pada saturasi adalah seperti diperlihatkan pada persamaan 2.6.

$$I_{C(sat)} = \frac{V_{CC}}{R_C} \quad \textbf{.....(2.8)}$$

dan arus basis yang tepat menimbulkan saturasi adalah seperti diperlihatkan pada persamaan 2.9.

$$I_{B(sat)} = \frac{I_{C(sat)}}{\beta dc} \quad \textbf{.....(2.9)}$$

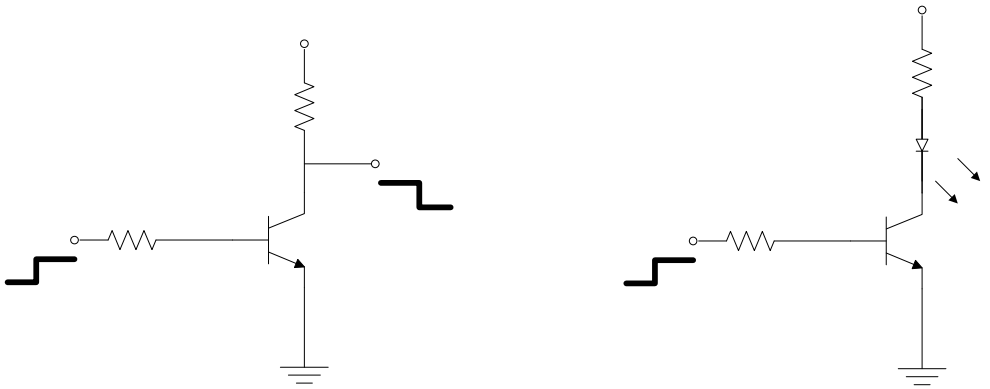
Dengan βdc merupakan penguatan arus. Salah satu penggunaan dari transistor adalah sebagai *switch* atau saklar, artinya bahwa mengoperasikan transistor pada salah satu dari saturasi atau titik sumbat, tetapi tidak di tempat-tempat sepanjang garis beban. Jika sebuah transisitor berada dalam keadaan saturasi, transistor seperti sebuah *switch* yang tertutup dari kolektor ke emiter. Jika transistor tersumbat (*cut-off*), transistor seperti sebuah *switch* yang terbuka. Dalam transistor dikenal istilah aturan disain *soft saturation* dan *hard saturation*. *Soft saturation* berarti kita membuat transistor hampir

saturasi, di mana arus basis hanya cukup untuk mangoperasikan transistor pada ujung atas dari garis beban. *Soft saturation* tidak dapat diandalkan pada produksi massal karena adanya perubahan-perubahan pada βdc dan $I_{B(sat)}$. *Soft saturation* akan mengacu pada rancangan di mana transistor akan jenuh secara terbatas, dalam hal ini penguatan arus hanya sedikit lebih kecil dari penguatan arus aktif.

Pada kondisi *hard saturation*, berarti terdapat arus basis yang cukup kuat untuk membuat transistor saturasi pada semua harga dari βdc . Untuk keadaan yang paling buruk dari temperatur dan arus, hampir semua transistor silikon sinyal kecil mempunyai βdc lebih besar daripada 10. karena itu, semua pedoman desain untuk *hard saturation* adalah mempunyai arus basis kira-kira sepersepuluh dari harga saturasi arus kolektor, ini menjamin *hard saturation* pada semua kondisi kerja. Sebagai contoh, jika ujung atas garis beban mempunyai arus kolektor sebesar 10 mA, maka akan didapatkan arus basis sebesar 1 mA. Hal ini menjamin keadaan saturasi untuk semua transistor, arus, temperatur dan sebagainya. Digunakan aturan 10:1 dalam proses mendesain rangkaian *switching* transistor, kecuali jika ditentukan lain. Ingat, ini hanya sebuah pedoman. Jika nilai tahanan standar menghasilkan perbandingan I_C/I_B sedikit lebih besar daripada 10, hampir setiap transistor sinyal kecil akan menuju keadaan *hard saturation*.

Gambar 2.16(a) menunjukkan sebuah rangkaian *switching* transistor yang digerakkan oleh tegangan *step*. Jika tegangan *input* nol, transistor tersumbat (*cut-off*). Dalam hal ini, transistor kelihatannnya sebuah *switch* yang terbuka. Dengan tidak adanya arus yang mengalir melalui tahanan kolektor, maka tegangan *output* sama dengan $+V_{BB}$.

Gambar 2.16(b) menunjukkan rangkaian *switching* transistor dengan sedikit variasi. Rangkaian ini disebut *LED driver*, karena transistor mengendalikan LED. Jika tegangan *input* rendah (*low*), transistor akan tersumbat (*cut-off*) dan LED padam. Jika tegangan *input* tinggi (*high*), transistor saturasi dan LED menyala.



(a)
Switching transistor *step*

(b)
Switching transistor sebagai LED DRIVER

Gambar 2.16 Contoh Transistor yang Digunakan Sebagai *Switch*

2.6 Penyearah

Penyearah adalah proses menyearahkan 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 dibagi menjadi tiga kategori yaitu :

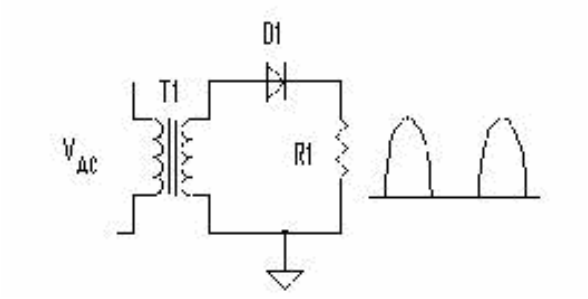
1. Penyearah setengah gelombang
2. Penyearah gelombang penuh sistem CT (*Center Tap*)
3. Penyearah gelombang penuh sistem jembatan

Rb

+Vbb

2.6.1 Penyearah Setengah Gelombang

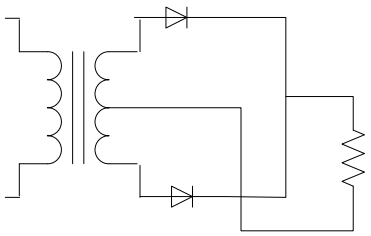
Penyearah ini bekerja dengan menggunakan satu dioda, sehingga hanya pulsa positif yang dapat terambil. Penyearah ini praktis sederhana, tetapi kekurangannya adalah bahwa gelombang keluaran bukan gelombang penuh sehingga rentan sekali akan *ripple*. Ini dapat dilihat pada gambar 2.17.



Gambar 2.17 Penyearah Setengah Gelombang

2.6.2 Penyearah Gelombang Penuh Sistem CT

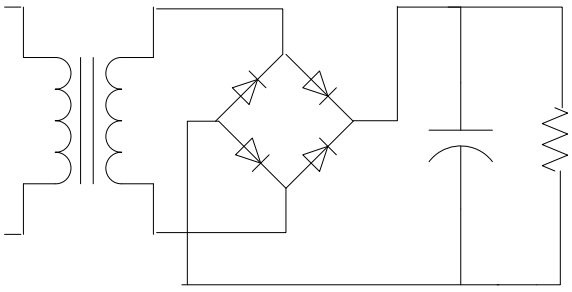
Penyearah ini menggunakan transformator jenis CT dengan dua buah dioda sebagai penyearah. Dioda bekerja secara bergantian untuk mengambil pulsa *positif* dan *negatif*, sehingga keluaran berupa gelombang penuh. Ini dapat dilihat pada gambar 2.18.



Gambar 2.18 Penyearah Sistem CT

2.6.3 Penyearah Gelombang Penuh Sistem Jembatan

Penyearah ini menggunakan 4 buah dioda sebagai penyearah. Pada siklus pertama dua dioda bekerja untuk menyearahkan atau mengambil pulsa *positif*. Siklus selanjutnya dua dioda berikutnya yang bekerja untuk mengambil pulsa *negatif*. Keuntungan penyearah ini adalah bahwa keluaran berupa gelombang penuh dan jika salah satu dioda rusak, maka dioda yang satunya lagi akan tetap bekerja. Ini dapat dilihat pada gambar 2.19.

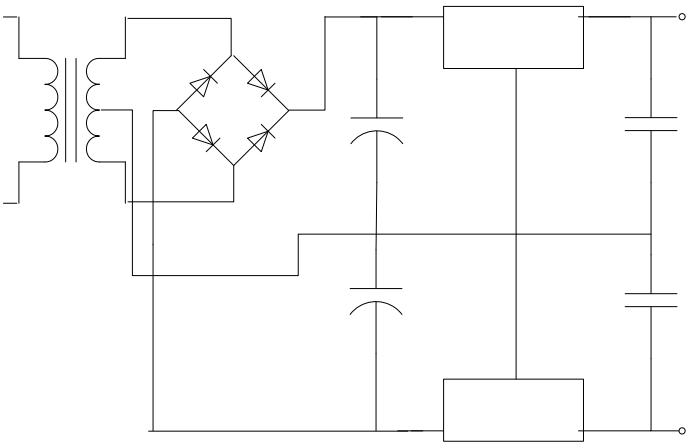


Gambar 2.19 Penyearah dengan sistem jembatan

2.6.4 Penyearah Teregulasi

Tegangan hasil penyearah belum tentu stabil pada suatu titik yang diinginkan, untuk itu harus ada proses untuk menstabilkan tegangan tersebut. Hal ini yang dapat dilakukan dengan menambahkan komponen pada keluaran penyearah, diantaranya menggunakan

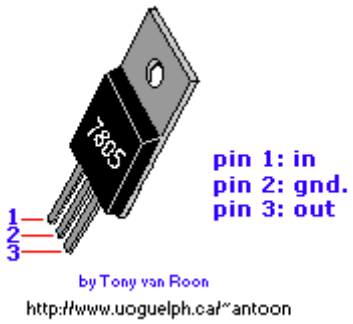
dioda zener, penguat operasional atau dengan IC regulator. Ini dapat dilihat pada gambar 2.20.



Gambar 2.20 Penyearah Teregulasi menggunakan IC

2.7 LM 7805

IC ini mempunyai tiga kaki yang digunakan sebagai komponen pendukung dari Vcc untuk menghasilkan tegangan 5V. IC *regulator* ini berfungsi untuk menstabilkan tegangan 5V dan dapat bekerja dengan baik jika tegangan *input* (Vin) lebih besar dari pada tegangan *output* (Vout). Biasanya perbedaan tegangan *input* dengan *output* yang direkomendasikan tertera pada *datasheet* komponen tersebut. Contoh LM7805 diperlihatkan pada gambar 2.21.



Gambar 2.21 IC LM7805

2.8 Komunikasi Serial

2.8.1 Komunikasi RS-232

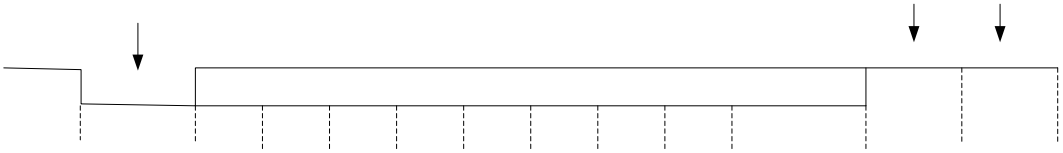
Standar RS232 ditetapkan oleh Industry Association dan Telecomunication Industry Association pada tahun 1962. Nama lengkapnya adalah *EIA/TIA-232 Interface Between*

Data Terminal Equipment and Data Circuit-Terminal Equipment Employing Serial Binary Data Interchange.

Dengan demikian standar ini hanya menyangkut komunikasi data antara komputer (*Data Terminal Equipment* – DTE) dengan alat-alat pelengkap komputer (*Data Circuit-Terminal Equipment* – DCE).

Dalam banyak literatur, DCE sering diartikan sebagai *Data Communication Equipment*, hal ini bisa dibenarkan tapi pengertiannya menjadi lebih sempit karena sebagai *Data Communication Equipment* yang dimaksud dengan DTE hanya sebatas peralatan untuk komunikasi, misalnya Modem. Padahal yang dimaksud dengan *Data Circuit-Terminal Equipment* bisa meliputi macam-macam alat pelengkap komputer yang dihubungkan ke komputer dengan standar RS232, misalnya *printer, optical mark reader, card register, PABX* bahkan jembatan timbang.

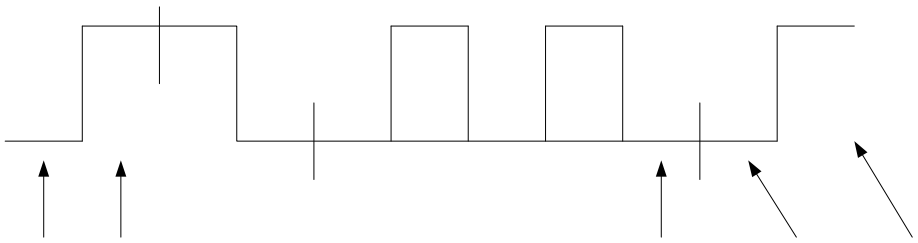
Ada dua macam sistem transmisi dalam komunikasi serial, yaitu asinkron dan sinkron. Transmisi sinkron lebih kompleks dan sangat sulit untuk dibuat percobaan secara sederhana, karena kedua titik komunikasi harus selalu dibuat sinkron. Format pengiriman serial asinkron diperlihatkan pada gambar 2.22 di bawah ini.



Gambar 2.22 Format Pengiriman Data Asinkron

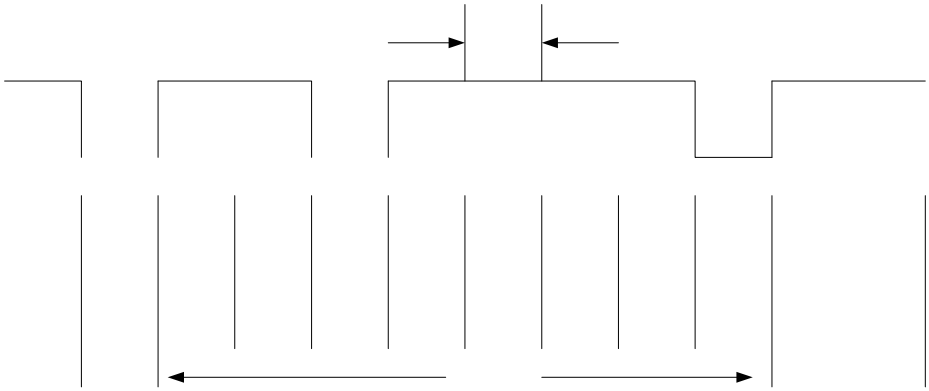
Bit-bit asinkron terdiri atas 1 start bit (setelah low), 6 sampai 8 bit data, 1 bit paritas dan 1 atau 2 stop bit (selalu high). Pada saat tidak ada data (*idle*) yang dikirim, kondisi saluran transmisi selalu *high*.

Kondisi bit paritas ditentukan oleh sistem paritas yang digunakan (ganjil atau genap). Agar tidak terjadi kesalahan interpretasi antara pengirim dan penerima, maka sistem paritas yang hendak dipakai perlu disetujui bersama, paritas genap atau ganjil. Bit paritas berfungsi untuk memeriksa apakah terdapat kesalahan pada data yang dikirim atau tidak. Pada gambar 2.23, misalnya, kita akan mengirim data 01010011, paritas genap dan 1 bit stop. Dikarenakan memakai paritas genap sehingga jumlah format data serial yang dikirim adalah :



Gambar 2.23 Format Pengiriman Data Asinkron (01010011)

Faktor lain yang perlu diperhatikan dalam transfer data serial asinkron adalah kecepatan pengiriman. Besaran kecepatan pengiriman data serial adalah bps (*bit per second*), dan biasa disebut baud rate atau *character per second* (cps). Seperti tampak pada gambar 2.24.



Gambar 2.24 Format Standard Transmisi Data Asinkron

Jika satu bit data membutuhkan waktu 3,33 milidetik, baud rate besarnya adalah 300 bps. Karena 1 byte terdiri atas 11 bit atau 11 x 3,33 = 36,36 milidetik, kecepatan transfer karakter adalah 1/36,36 milidetik atau 27, 3 karakter/detik. Baud rate yang biasa digunakan adalah 110, 300, 1200, 4800, 9600, dan 19200 bps.

Ada 3 pokok yang diatur oleh standar RS232, antara lain :

1. Bentuk sinyal dan level tegangan yang dipakai
2. Penentuan jenis sinyal dan konektor yang dipakai, serta susunan sinyal pada kaki-kaki di konektor.
3. Penentuan tata cara pertukaran informasi antara komputer dan alat-alat pelengkapnya.

2.8.1.1 Karakteristik Sinyal RS232

Standar sinyal komunikasi serial yang banyak digunakan adalah standar RS232 yang dikembangkan oleh *Electrical Industry Association and The Telecommunication Industry Association* (EIA/TIA) yang pertama kali dipublikasikan pada tahun 1962. ini jauh sebelum IC TTL popular sehingga sinyal ini tidak ada hubungan sama sekali dengan level tegangan IC TTL. Standar ini hanya menyangkut komunikasi data antara komputer (*Data Terminal Equipment – DTE)* dengan alat-alat pelengkap komputer (*Data Circuit Terminal Equipment – DCE).* Standar RS232 inilah yang biasa digunakan pada port serial IBM PC kompatibel.

Standar sinyal serial RS232 memiliki ketentuan level tegangan sebagai berikut :

1. Logika “1” disebut “*mark*” terletak antara -3 volt hingga -25 volt.
2. Logika “0” disebut “*space*” terletak antara +3 volt hingga +25 volt.
3. Daerah tegangan antara -3 volt hingga +3 volt adalah *invalid* level, yaitu daerah tegangan yang tidak memiliki level logika sehingga harus dihindari. Demikian juga, level tegangan negatif dari -25 volt atau lebih *positif* dari +25 volt juga harus dihindari karena tegangan tersebut dapat merusak *line driver* pada saluran RS232.

2.8.1.2 Konfigurasi Port Serial

Gambar 2.25 merupakan konektor DB-9 tampak belakang. Pada komputer IBM PC kompatibel biasanya kita dapat menemukan dua konektor port serial DB-9 yang biasa dinamai COM1 dan COM2, terlihat pada gambar 2.25 port serial DB-9 terdapat 9 pin yang mempunyai fungsi yang berbeda, konfigurasi pin DB-9 dapat dilihat pada tabel 2.5.



Gambar 2.25 Port Serial

Pada PC terdapat 2 macam konektor RS232 yaitu jenis 25 pin dan jenis 9 pin. Adapun sinyal dari pin-pin tersebut berisikan data yang dapat diperhatikan pada tabel berikut ini:

Tabel 2.5 Sinyal-sinyal pada konektor RS232

No. PIN	Nama Sinyal	Direction	Keterangan
1	DCD (*Data Carrier Detect/Received Line Signal Detect*)	In	Pada saluran DCD ini, DCE memberitahukan ke DTE bahwa terminal masukkan ada data masuk.
2	RxD (*Receive Data*)	In	Saluran RxD ini digunakan DTE untuk menerima data dari DCE.
3	TxD (*Transmit Data*)	Out	Saluran TxD ini digunakan untuk mengirim data dari DCE.
4	DTR (*Data Terminal Ready*)	Out	Pada saluran DTR, DTE akan memberikan status kesiapan terminalnya.
5	GND (*Ground*)	-	Sebagai saluran ground.
6	DSR (*Data Set Ready*)	In	Sinyal aktif pada saluran DSR ini menunjukkan bahwa DCE sudah siap.
7	RST (*Request To Send*)	Out	Pada saluran ini DCE diminta mengirim data oleh DTE.
8	CST (*Clear To Send*)	In	Pada saluran CST, DCE memberitahukan bahwa DTE boleh mengirimkan data.
9	RI (*Ring Indicator*)	In	Pada saluran RI, DCE memberitahukan DTE bahwa sebuah stasiun menghendaki hubungan dengannya.

Untuk dapat menggunakan port serial kita perlu mengetahui alamatnya. *Base Address* COM1 terdapat pada alamat 1016 (3F8H) dan COM2 terdapat pada alamat 760 (2F8H). Alamat tersebut adalah alamat yang secara umum digunakan, tergantung dari komputer yang digunakan. Tepatnya kita bisa melihat pada peta memori tempat menyimpan alamat tersebut, yaitu memori 0000.0400 untuk *base address* COM1 dan memori 0000.0402 untuk *base address* COM2. Setelah kita mengetahui *base address* nya, maka dapat ditentukan alamat register-register yang akan digunakan untuk

komunikasi port serial ini, register-register yang digunakan tersebut dapat dilihat pada tabel 2.6

Tabel 2.6 Nama Register yang Digunakan Beserta Alamatnya

Nama Register	COM1	COM2
Tx *Buffer*	3F8H	2F8H
Rx *Buffer*	3F8H	2F8H
Baud Rate Divisor Latch LSB	3F8H	2F8H
Baud Rate Divisor Latch MSB	3F9H	2F9H
Interrupt Enable Register	3F9H	2F9H
Interrupt Identification Register	3FAH	2FAH
Line Control Register	3FBH	2FBH
Modem Control Register	3FCH	2FCH
Line Status Register	3FDH	2FDH
Modem Status Register	3FEH	2FEH

Keterangan mengenai fungsi register-register tersebut adalah sebagai berikut :

1. *RX Buffer*, digunakan untuk menampung dan menyimpan data dari DCE.
2. *TX Buffer*, digunakan untuk menampung dan menyimpan data yang akan dikirim ke port serial.
3. *Baud Rate Divisor Latch LSB,* digunakan untuk menampung byte bobot rendah untuk pembagi *clock* pada *IC UART* agar didapat *baud rate* yang tepat.
4. *Baud Rate Divisor Latch MSB*, digunakan untuk menampung byte bobot rendah untuk pembagi *clock* pada *IC UART* sehingga total angka pembagi adalah 4 byte yang dapat dipih dari 0001H hingga FFFFH. Berikut ini adalah tabel pembagi yang sering digunakan.

Tabel 2.7 Angka Pembagi *Clock* pada IC UART

Baud Rate (bit/detik)	Angka Pembagi
300	0180H
600	0C00H
1200	0060H
1800	0040H
2400	0030H
4800	0018H

Tabel 2.7 Lanjutan

9600	000Ch

Sebagai catatan, *Register Baud Rate Divisor Latch* ini bisa diisi jika bit 7 pada *Register Line Control Register* diisi 1.

5. *Interrupt Enable Register*, digunakan untuk men-set interupsi apa saja yang akan dilayani komputer. Berikut ini adalah tabel rincian bit pada *Interrupt Enable Register.*

Tabel 2.8 Rincian Bit pada *Interrupt Enable Register*

Nomor Bit	Keterangan
0	1: Interupsi akan diaktifkan jika menerima data
1	1: Interupsi akan diaktifkan jika register Tx kosong
2	1: Interupsi akan diaktifkan jika ada perubahan keadaan pada *Line Status Register*
3	1: Interupsi akan diaktifkan jika ada perubahan keadaan pada *Status Register*
4,5,6,7	Diisi 0

6. *Interrupt Identification Register*, digunakan untuk menentukan urutan prioritas interupsi. Berikut adalah tabel rincaian bit pada *Interrupt Identification Register.*

Tabel 2.9 Rincian Bit pada *Interrupt Identification Register*

Nomor Bit	Keterangan
0	0: Interupsi menunggu 1: *No Interrupt* pending
1 dan 2	00: Prioritas tertinggi oleh *Line Status Register* 01: Prioritas tertinggi oleh Register Rx jika menerima data 10: Prioritas tertinggi oleh Register Tx jika telah kosong 11: Prioritas tertinggi oleh *Modem Status*
3,4,5,6,7	Diisi 0

7. *Line Control Register*, digunakan untuk menentukan jumlah bit data, jumlah bit parity, jumlah bit stop, serta untuk menentukan apakah *baud rate divisor* dapat

Tabel 2.10 Rincian Bit pada *Line Control Register*

Nomor Bit	Keterangan
0 dan 1	Jumlah bit data 00: Jumlah bit data adalah 5 01: Jumlah bit data adalah 6 10: Jumlah bit data adalah 7 11: Jumlah bit data adalah 8
2	Bit stop 0: Jumlah bit stop adalah 1 1: Jumlah bit stop adalah 1,5 untuk bit data dan 2 untuk 6 hingga 8 bit data
3	Bit Parity 0: tanpa parity 1: dengan parity
4	0: parity ganjil 1: parity genap
5	1: bit parity ikut dikirimkan (*stick parity*)
6	0: *set break control* tidak diaktifkan 1: *set break control* diaktifkan
7	0: *Baud rate divisor* tidak dapat diakses 1: *Baud rate divisor* dapat diakses

8. *Modem Control Register*, digunakan untuk mengatur saluran pengatur modem terutama saluran DTR dan saluran RST. Berikut ini tabel rincian bit pada *Modem Control Register*.

Tabel 2.11 Rincian Bit pada *Modem Control Register*

Nomor Bit	Keterangan
0	Bit DTR 0: Saluran DTR diaktifkan (aktif 0) 1: Saluran DTR dibuat normal (tidak aktif)
1	Bit RST 0: Saluran RST diaktifkan (aktif 0) 1: Saluran RST dibuat normal (tidak aktif)

Tabel 2.11 Lanjutan

2	Bit OUT1, digunakan untuk penghubung ke perangkat lain dapat dibuat logika *high* atau logika *low*. Secara normal tidak digunakan
3	Bit OUT2, digunakan untuk penghubung ke perangkat lain, dapat dibuat logika *high* atau logika *low*
4	0: *Loop back internal* diaktifkan 1: *Loop back internal* tidak diaktifkan
5,6,7	Diisi 0

9. *Line Status Register*, digunakan untuk menampung bit-bit yang menyatakan keadaan penerimaan atau pengiriman data dan status kesalahan operasi. Berikut adalah tabel rincian bit pada *Line Status Register*.

Tabel 2.12 Rincian Bit pada *Line Status Register*

Nomor Bit	Keterangan
0	1: menyatakan adanya data masuk pada *buffer* Rx
1	1: data yang masuk mengalami *overrun*
2	1: terjadi kesalahan pada bit *parity*
3	1: terjadi kesalahan *framing*
4	1: terjadi *break interrupt*
5	1: menyatakan bahwa register Tx telah kosong
6	1: menyatakan bahwa *Transmitter Shift Register*
7	Diisi 0

10. *Modem Status Register*, digunakan untuk menapung bit-bit yang menyatakan status dari saluran hubungan dengan modem. Berikut ini tabel rincian bit pada *Modem Status Register*.

Tabel 2.13 Rincian Bit pada *Modem Status Register*

Nomor Bit	Keterangan
0	1: menyatakan adanya perubahan keadaan di saluran Clear to Send (CST)
1	1: menyatakan adanya perubahan keadaan di saluran Data Set Ready (DSR)

Tabel 2.13 Lanjutan

2	1: menyatakan adanya perubahan keadaan di saluran *Ring Indicator* (RI) dari *low* ke *high*
3	1: menyatakan adanya perubahan di saluran *Receive Line Signal Detect (DCD)*
4	1: menyatakan saluran *Clear to Send* (CTS) sudah dalam keadaan aktif
5	1: menyatakan saluran *Data Set Ready* (DSR) sudah dalam keadaan aktif
6	1: menyatakan bahwa saluran *Ring Indicator* (RI) sudah dalam keadaan aktif
7	1: menyatakan bahwa saluran *Receive Line Signal Detect* (DCD) sudah dalam keadaan aktif

2.8.1.3 Flow Control

Jika kecepatan transfer data dari DTE ke DCE (misalnya komputer ke modem) lebih cepat dari pada transfer data dari DCE ke DCE (misal modem ke modem), cepat atau lambat kehilangan data akan terjadi karena *buffer* pada DCE akan mengalami *overflow*. Untuk itu diperlukan *flow control* untuk mengatasi masalah tersebut.

Dikenal dua macam *flow control*, yaitu secara *software* dan secara *hardware*. *Flow control* secara *software* atau sering disebut Xon atau Xoff *flow control* menggunakan karakter Xon (tipikal karakater *ASCII 17)* dan karakter Xoff (Tipikal karakter *ASCII* 19) untuk melakukan kontrol. DCE akan mengirimkan Xoff ke komputer untuke memberitahukan komputer agar menghentikan pengiriman data jika *buffer* pada DCE telah penuh. Jika *buffer* telah kembali siap menerima data, DCE akan mengirimkan karakter Xon ke komputer dan komputer akan mengirimkan data selanjutnya sampai data terkirim semua atau komputer menerima karakter Xoff lagi. Keuntungan *flow control* secara *software* ini adalah hanya diperlukan kabel sedikit karena karakter kontrol dikirimkan lewat saluran Tx/Rx. Akan tetapi, kecepatan pengiriman data menjadi lambat.

Flow control secara *hardware* atau sering disebut RTS/CTS *flow control* menggunakan dua kabel untuk melakukan pengontrolan. Komputer akan men-set saluran *Request to Send* jika akan mengirimkan data ke DCE. Jika buffer di DCE siap saluran *Clear to Send* dan komputer akan mulai mengirimkan data. Jika *buffer* telah

penuh, maka saluran akan direset dan komputer akan menghentikan pengiriman data sampai saluran ini di-set kembali.

2.8.1.4 Spesifikasi RS232

RS232 yang digunakan adalah MAX232 dari Maxim, yang merupakan salah satu perusahaan besar pembuat IC.

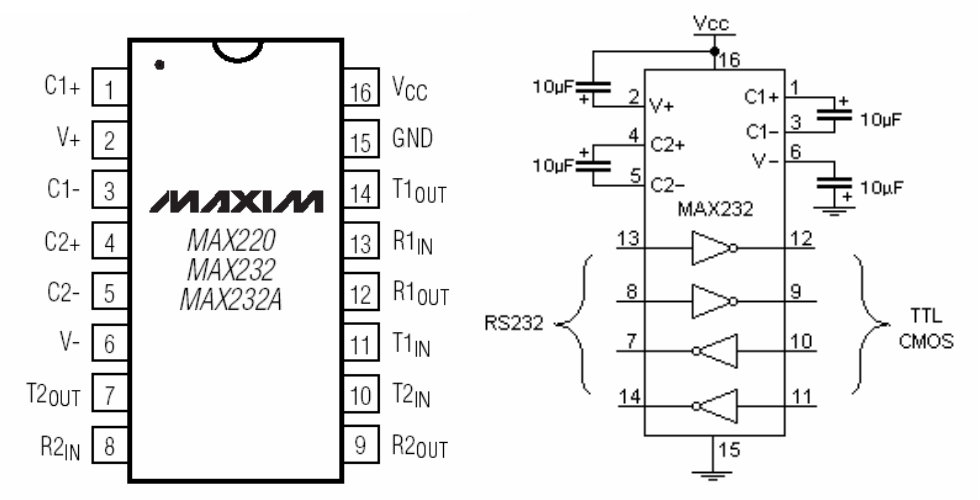
Adapun spesifikasi dari MAX232 ini adalah :

Tabel 2.14 Spesifikasi Max 232

Spesifikasi		**RS232**
Jenis operasi		Single-Ended
Jumlah total *drivers* dan *receivers* dalam satu jalur		1 *Drivers*, 1 *Receivers*
Panjang kabel maksimum		50 Ft
Kecepatan data maksimum		20 kbps
Tegangan keluaran maksimum *drivers*		± 25 V
Signal level keluaran (*Loaded Min*.)	*Loaded*	± 5 V to ± 15 V
Signal level keluaran (*Unloaded Max*.)	*Unloaded*	± 25 V
Drivers Load Impedence (*Ohms*)		3K to 7K
Max. Driver Current dalam *High Z. State*	*Power On*	N/A
Max. Driver Current dalam *High Z. State*	*Power Off*	± 6 mA @ ± 2 V
Kecepatan *slew* (*Max*.)		30 V/ μ S
Jarak tegangan masukan *receiver*		± 15 V
Masukan sensitivitas *receiver*		± 3 V
Masukan resistansi *receiver* (Ohms)		3K to 7K

2.8.1.5 Rangkaian Max 232 Serial Port

Jenis IC MAX 232 ini memiliki 2 input saluran komunikasi dan 2 output saluran yang keduanya dapat digunakan. IC MAX 232 ini mampu mengubah format digital ke dalam sebuah format atau level RS232, yang mana pada level RS232 tegangan *high* memiliki tegangan -3 V sampai -25 V dan tegangan *low* memiliki tegangan +3 V sampai +25 V dalam kondisi maksimum. Pada prakteknya terkadang hanya memiliki tegangan -5 V sampai -10 V untuk tegangan *high* dan +5 V sampai +10 V untuk tegangan *low*. Hal ini dapat diperlihatkan pada gambar 2.26 di bawah ini.



Gambar 2.26 Rangkaian Max 232

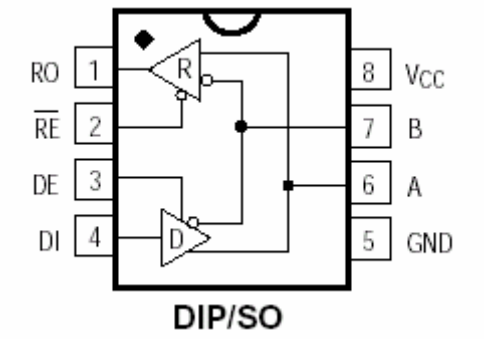
2.8.2 Komunikasi RS485

Rangkaian RS485 adalah suatu rangkaian jembatan (penghubung) antara piranti luar dengan perangkat-perangkat komputer

Standart RS485 ditetapkan oleh *Electronic Industries Association* dan *Telecomunication Industry Association* pada tahun 1983. Nama lengkapnya adalah *EIA/TIA-485*. Digunakan untuk membangun saluran transmisi sampai sejauh 4000 *feet*/ 1,2 Km dengan kecepatan maksimum lebih dari 1 Mb/detik, memiliki tegangan minimum -7V dan tegangan maksimum +12V.

Standard RS485 membicarakan karakteristik sinyal dalam transmisi data secara Balanced *Digital Multipoint System*. RS485 adalah teknik komunikasi data dapat dilakukan pada jarak yang cukup jauh yaitu 1,2 Km.

Konfigurasi pin dari RS485 ini adalah sebagai berikut :



Gambar 2.27 Konfigurasi pin RS485

Berdasarkan pada gambar 2.27 diatas, output dari *Line Generator* dapat diambangkan (*High Impedance*) dengan memberikan nilai ‘0’ pada input (DE), kemampuan ini dimaksudkan untuk menunjang keperluan dalam membentuk rangkaian saluran komunikasi *multidrop* yang menghendaki pada saluran hanya boleh satu *Line Generator* saja yang aktif. Apabila kaki DE berlogika 0 dan RE berlogika 0, maka RS485 berfungsi sebagai penerima data sedangkan bila kaki DE berlogika 1 dan RE berlogika 1 maka RS485 berfungsi sebagai penerima.

2.9 Bahasa Assembly

Assembler adalah program komputer yang men-translitrasi program dari bahasa *assembly* ke bahasa mesin. Sedangkan bahasa *assembly* adalah ekuivalensi bahasa mesin dalam bentuk alpanumerik. *Mnemonics* alpanumerik digunakan sebagai alat bantu bagi *programmer* untuk memprogram mesin komputer daripada menggunakan serangkaian 0 dan 1 (bahasa mesin) yang panjang dan rumit. Program sumber *assembly* terdiri dari kumpulan baris-baris perintah dan biasanya di simpan dengan *extension* .ASM dengan satu baris untuk satu perintah, setiap baris perintah tersebut bisa terdiri atas beberapa bagian, yakni bagian label, bagian *mnemonic*, bagian operand yang bisa lebih dari satu dan terakhir bagian komentar.

Program sumber (*source code*) dibuat dengan program *pinnacle* 52. Hasil kerja program yang telah dikompile dalam bahasa *assembler* ini adalah “*assembly listing*” dan juga “file dengan *extention* .HEX”. File dengan *extention* HEX ini adalah yang akan diisikan ke dalam chip mikrokontroler. Ketentuan penulisan *source code* adalah sebagai berikut :

1 Masing-masing bagian dipisahkan dengan spasi atau TAB, khusus untuk *operand* yang lebih dari satu masing-masing *operand* dipisahkan dengan koma.
2 Bagian-bagian tersebut tidak harus semuanya ada dalam satu baris, jika ada satu bagian yang tidak ada maka spasi atau TAB sebagai pemisah bagian tetap hasus ditulis.
3 Bagian label ditulis mulai huruf pertama dari baris, jika baris bersangkutan tidak mengandung label maka label tersebut digantikan dengan spasi atau TAB, yakni sebagai tanda pemisah antara bagian label dengan bagian *mnemonic*.

Adapun instruksi-instruksi MCS-51 yang digunakan dapat dilihat pada lampiran (program).

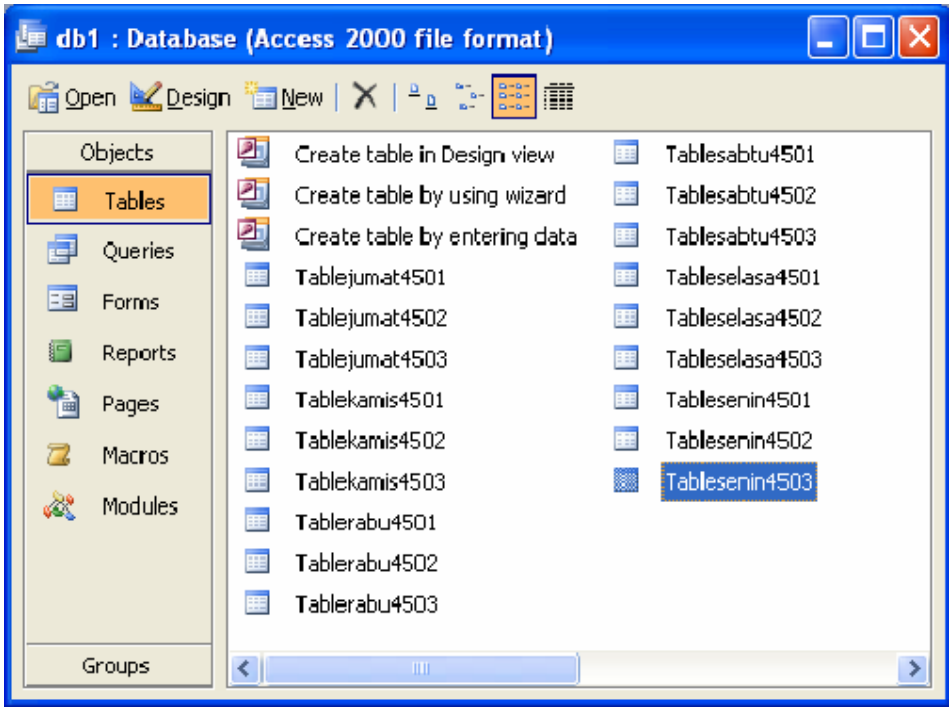
2.10 Database Dengan Microsoft Access

Pada Microsoft Access terdapat fasilitas untuk membuat suatu database. Hal pertama yang harus dilakukan adalah membuat suatu file database. File tersebut menyimpan segala sesuatu yang dibuat untuk database. Tabel adalah suatu pusat database. Tabel sangat mirip dengan *spreadsheet.* Access menyimpan setiap entri database pada barisnya sendiri, itulah yang disebut *record*. *Record* adalah kumpulan informasi tentang suatu hal dalam kasus ini. Setiap tipe rinci dalam bentuk kolom disebut *field*. Pada potongan antara satu *field* dan satu baris adalah data individual untuk *record* khusus tersebut, area ini disebut *cell*. Setiap *file database* dapat mempunyai beberapa tabel. Semua data *field* yang dimasukkan akan berada dalam tabel tersebut.

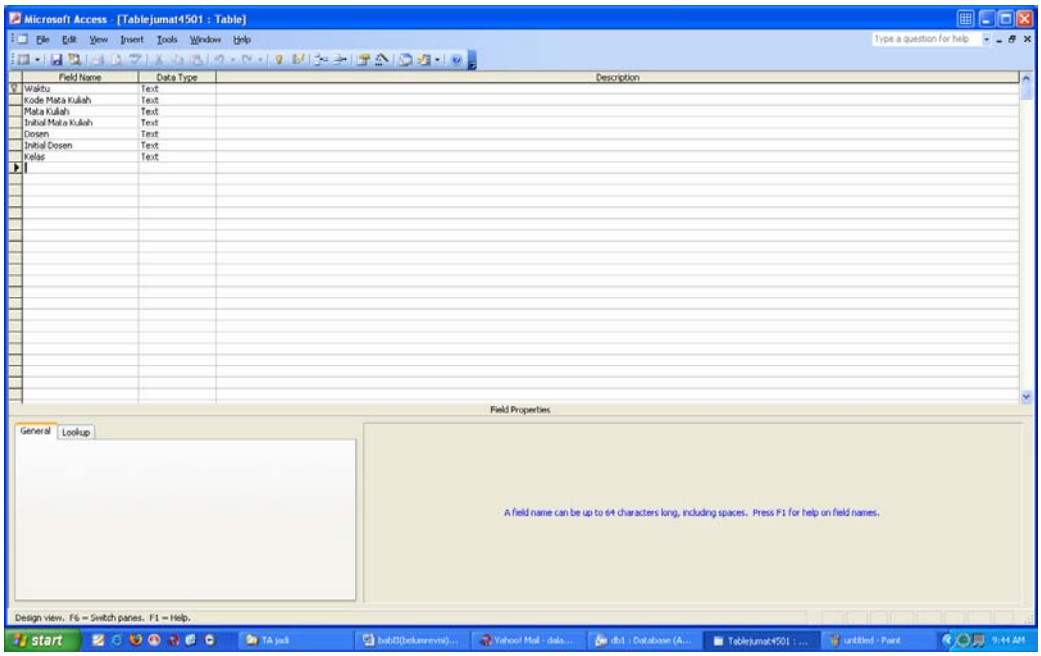
Langkah pertama untuk membuat *database* kosong terlebih dahulu menjalankan program Microsoft Access. Selanjutnya, ikuti langkah berikut ini:

1 Klik **Blank Access Database**, maka jendela **File New Database** terbuka. Ketik suatu *file* baru pada kotak teks **File Name**, kemudian klik **Create** maka Microsoft Access akan membuat *database* baru. Gambar 2.28 memperlihatkan cara membuat sebuah tabel dengan memilih **Create Table in Design View**.

2 Pada gambar 2.29 memperlihatkan cara membuat isi **Field Name**, **Data Type** dan lainnya sesuai dengan kebutuhan. *Close* jendela tabel, beri nama untuk tabel recordnya, pilih **No** jika tidak ingin memakai *primary key*.



Gambar 2.28 Tabel-tabel yang telah dibuat



Gambar 2.29 *Property* masing-masing *field* tergantung pada tipe *field*

Setelah selesai pembuatan file, *record* dan *field*-nya maka database kosong tersebut dapat diisi dan disimpan. Untuk memudahkan pengisian data dapat dibuat model *form* atau lainnya sesuai dengan kebutuhan.

2.11 Bahasa Pemrograman Visual Basic 6.0

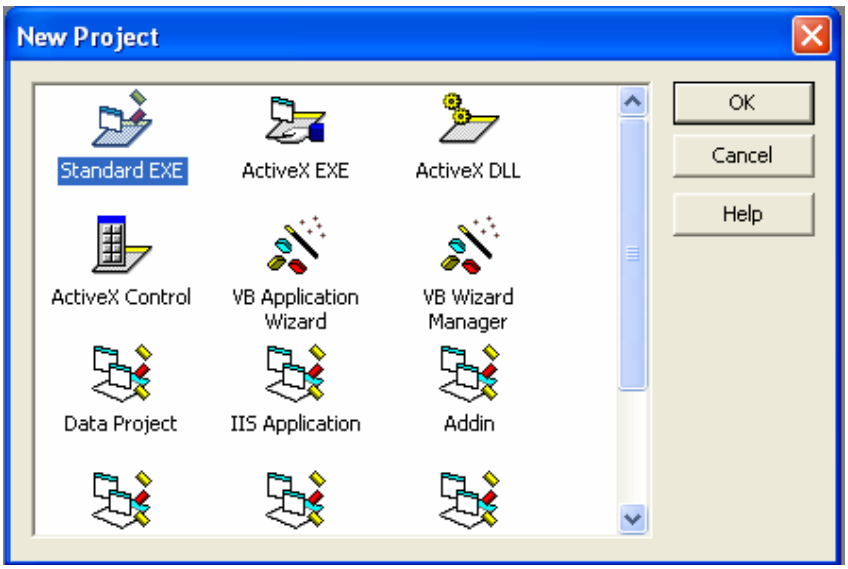
Perancangan software dititik beratkan pada pembangunan sebuah program *interface* yang *user friendly* dan yang terpenting adalah *software* harus mampu berkomunikasi dengan *hardware* sehingga dapat menyampaikan informasi yang sesuai. Pada sistem ini software yang digunakan adalah *Visual Basic* 6.0.

Bahasa pemrograman adalah bahasa yang dimengerti oleh *object* untuk melakukan tugas-tugas tertentu, salah satu contoh bahasa *Visual Basic*. Bahasa pemrograman *Visual Basic* yang dikembangkan oleh *Microsoft* sejak tahun 1991 merupakan pengembangan dari pendahulunya yaitu bahasa pemrograman BASIC (*Baginners All-purpose Symbolic Instruction Code*) yang dikembangkan pada era 1950-an. *Visual Basic* merupakan salah satu *Development Tool* yaitu alat bantu untuk membuat berbagai macam program *object*, khususnya yang menggunakan sistem operasi *Windows*, juga salah satu bahasa pemrograman *object* yang mendukung *object* (*Object Oriented Programming* = OOP). Dalam pemrograman berbasis obyek (OOP) kita perlu mengenal istilah *object*, *property*, *method* dan *event*. Berikut adalah keterangan mengenai hal tersebut diatas:

1. *Object* adalah komponen didalam sebuah program
2. *Property* adalah karakteristik yang dimiliki oleh *object*.
3. *Method* adalah aksi yang dapat dilakukan oleh *object*.
4. *Event* adalah kejadian yang dapat dialami oelh *object*.

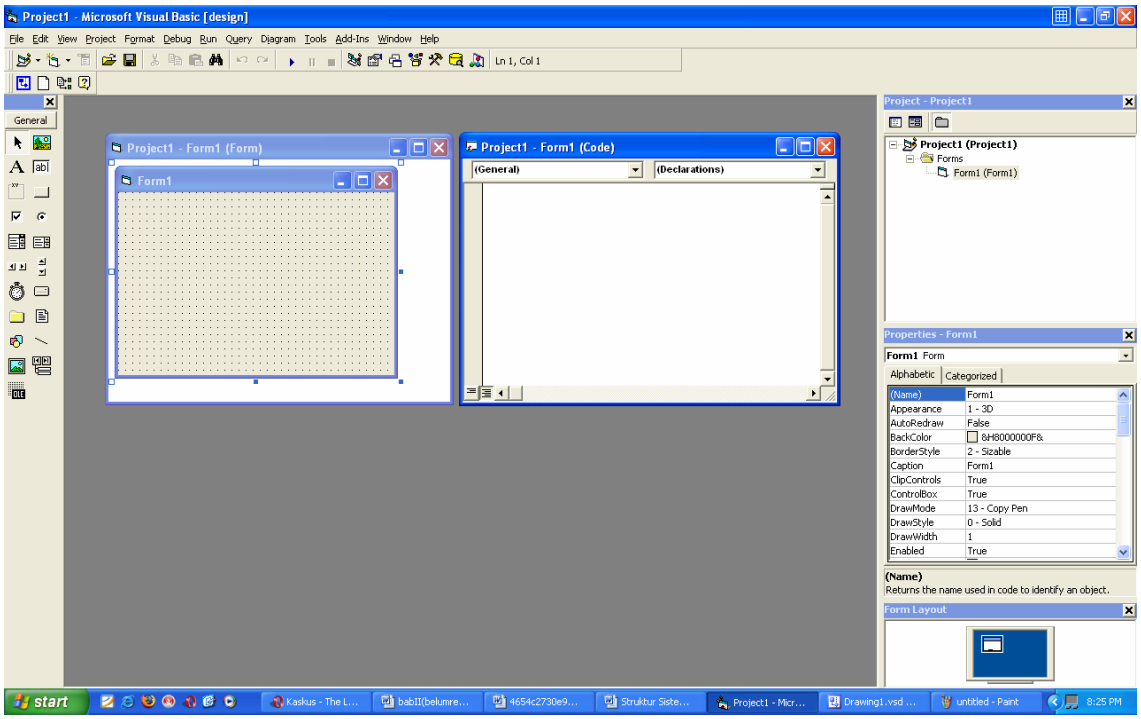
Seperti program berbasis *Windows* lainnya, *Visual Basic* terdiri dari banyak jendela (*windows*) ketika kita akan melalui *Visual Basic* sekumpulan *windows* yang saling berkaitan inilah yang disebut dengan *Integrated Development Environment* (IDE). Program yang berbasis *windows* bersifat *Event-Driven*, artinya program bekerja berdasarkan *event* yang terjadi pada *object* di dalam program tersebut, misalnya jika seorang *user* mengklik sebuah tombol maka program akan memberikan "reaksi" terhadap *event* klik tersebut. Program akan memberikan reaksi sesuia dengan kode-kode program yang dibuat untuk suatu *event* pada *object* tertentu. Pada waktu memulai *Visual Basic* beberapa *windows* kecil berada di dalam sebuah *windows* besar (*windows* induk), bentuk inilah yang dikenal dengan format *Multiple Document Interface* (MDI).

Pada gambar 2.30 memperlihatkan tampilan awal saat akan membuat sebuah **New Project** pada Visual Basic 6.0.



Gambar 2.30 Tampilan awal Visual Basic

Pada gambar 2.31 memperlihatkan contoh tampilan *Integrated Development Environment* (IDE) pada sebuah *project Visual Basic* dengan sebuah *form* dan sebuah *Command Button*.



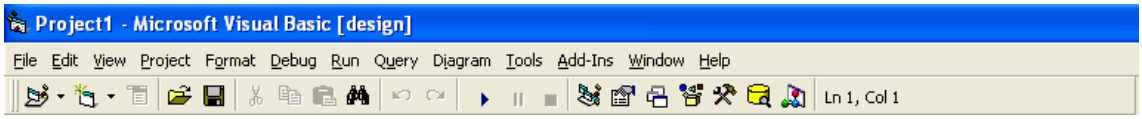
Gambar 2.31 Tampilan IDE Visual Basic

Menu pilihan pada *Visual Basic*:

1. **Menu Bar/Toolbar**

   Menu Bar *Visual Basic* berisi semua perintah *Visual Basic* yang dapat dipilih untuk melakukan tugas tertentu, isi dari menu ini sebagaian hampir sama dengan program-program *windows* pada umumnya.

   Toolbar adalah tombol-tombol (*shortcut*) yang mewakili suatu perintah tertentu pada *Toolbar*. Ini dapat dilihat pada gambar 2.32.



Gambar 2.32 Menu Bar/Toolbar

2. **Toolbox**

   Toolbox adalah sebuah "kotak piranti" yang mengandung semua objek atau "*control*" yang dibutuhkan untuk membentuk suatu program aplikasi. Kontrol adalah suatu objek yang akan menjadi penghubung antara program aplikasi dan *user*nya yang kesemuanya harus diletakkan di dalam jendela *form*. *Toolbox* dapat disembunyikan untuk memberikan ruangan bagi element pada *Intergrated*

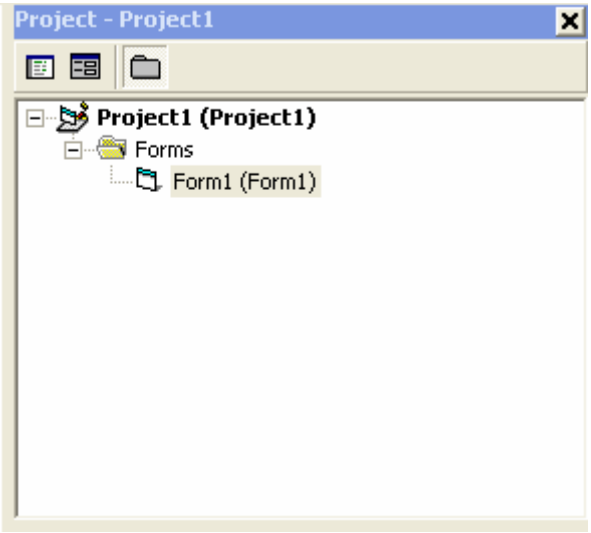
Development Environment (IDE) lainnya. Sehingga lebih mempermudah desain maupun penulisan program. Ini dapat dilihat pada gambar 2.33.



Gambar 2.33 Toolbox

3. Project Window

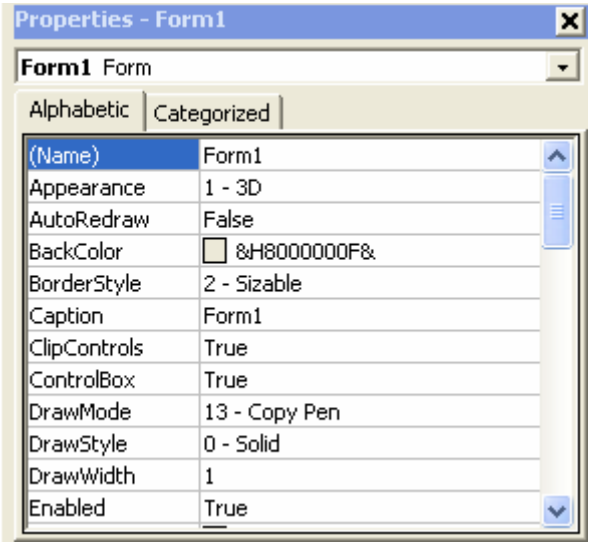
Window ini menampilkan seluruh *form*, *class*, *class module* dan komponen lainnya yang ada pada sebuah *project*. Ini dapat dilihat pada gambar 2.34.



Gambar 2.34 Project Window

4. Property Window

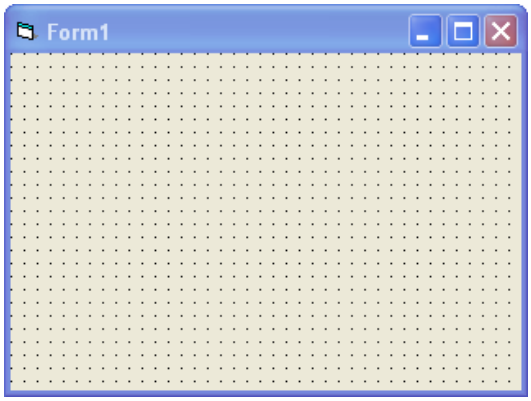
Window ini berisi seluruh *property* dari masing-masing *object* pada sebuah *project* yang meliputi *property form* dan kontrol-kontrol yang ada pada *form* tersebut. Beberapa *property* dapat diisikan pada tahap desain dan adapula *property* yang harus diisikan dengan menuliskan kode selama program dijalankan (*runtime*). Ini dapat dilihat pada gambar 2.35.



Gambar 2.35 Property Window

5. Form

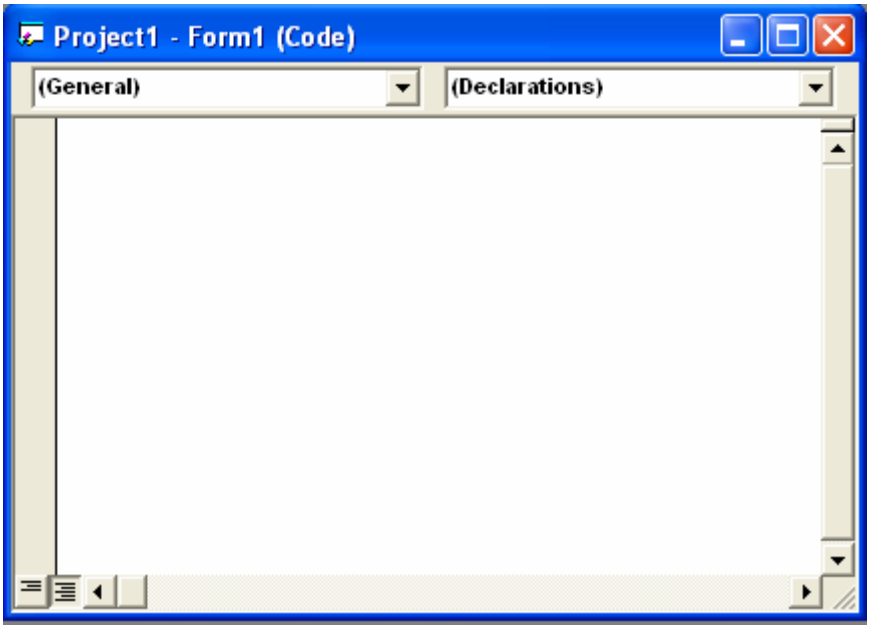
Form adalah sebuah atau beberapa *window* untuk pembuatan program aplikasi. *Form* ini dapat memuat berbagai macam *control* (tombol-tombol maupun teks) yang diperlukan dalam desain program yang sesuai dengan kebutuhan program. Ini dapat dilihat pada gambar 2.36.



Gambar 2.36 Form

6. Code Window

Pada *window* inilah semua kode/perintah tentang program dituliskan dengan memperhatikan *event* apa saja yang diperlukan. Ini dapat dilihat pada gambar 2.37.



Gambar 2.37 Code Window