BAB II
TINJAUAN PUSTAKA

Bab ini akan membahas mengenai teori dan komponen penunjang yang akan digunakan dalam pembuatan tugas akhir ini. Pembahasannya berisi tentang *RFID (Radio-Frequency Identification)*, Mikrokontroler, RS232, RS 485, Bahasa Assembly dan Borland Delphi sebagai komponen instrumentasi.

2.1 RFID (*Radio-Frequency Identification*)

RFID adalah sebuah teknologi yang memanfaatkan frekuensi radio untuk identifikasi otomatis terhadap obyek. Pengidentifikasian objek dapat dikenali dengan menambahkan sebuah *tag* RFID pada objek tersebut *(Himanshu Bhatt,Bill Glover 2006)*.

RFID dianggap sebagai suksesor dari *barcode* optik yang banyak dicetak pada barang-barang dagangan dengan dua keunggulan pembeda :

1. **Identifikasi yang unik :** Sebuah *tag* RFID selangkah lebih maju dengan mengemisikan sebuah nomor seri unik di antara jutaan obyek yang identik. *Identifier* yang unik dalam RFID dapat berperan sebagai pointer terhadap entri basis data yang menyimpan banyak histori transaksi untuk item-item individu.
2. **Otomasi** : Barkode di-*scan* secara optik, memerlukan kontak *line-of-sight* dengan *reader*, dan tentu saja peletakan fisik yang tepat dari obyek yang di*scan*. Kecuali pada lingkungan yang benar-benar terkontrol, *scanning* terhadap *barcode* memerlukan campur tangan manusia, sebaliknya *tag-tag* RFID dapat dibaca tanpa kontak *line-of-sight* dan tanpa penempatan yang presisi. *Reader* RFID dapat melakukan *scan* terhadap *tag-tag* sebanyak ratusan perdetik.

2.1.1 Komponen-komponen Utama Sistem RFID

Secara garis besar sebuah sistem RFID terdiri atas tiga komponen utama, yaitu *tag*, *reader* dan basis data. Secara singkat, mekanisme kerja yang terjadi dalam sebuah sistem RFID adalah bahwa sebuah *reader* frekuensi radio melakukan *scanning* terhadap data yang tersimpan dalam *tag*, kemudian mengirimkan informasi tersebut ke sebuah basis data yang menyimpan data yang terkandung dalam *tag* tersebut, yang terlihat Pada gambar 2.1.

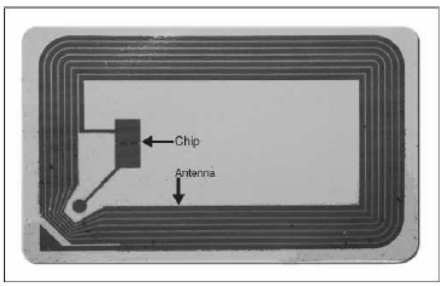


Gambar 2.1 Komponen utama sistem RFID

2.1.2 *Tag* RFID

Sistem RFID merupakan suatu tipe sistem identifikasi otomatis yang bertujuan untuk memungkinkan data ditransmisikan oleh peralatan *portable* yang disebut *tag*, yang dibaca oleh suatu *reader* RFID dan diproses menurut kebutuhan dari aplikasi tertentu. Data yang ditrasmisikan oleh *tag* dapat menyediakan informasi identifikasi atau lokasi, atau hal-hal khusus tentang produk-produk ber*tag*, seperti harga, warna, tanggal pembelian dan lain-lain.

Sebuah *tag* RFID atau *transponder*, terdiri atas sebuah mikro (*microchip*) dan sebuah antena (Gambar 2.2). *Chip* mikro itu sendiri dapat berukuran sekecil butiran pasir, seukuran 0.4 mm. Chip tersebut menyimpan nomor seri yang unik atau informasi lainnya tergantung kepada tipe memorinya. Tipe memori itu sendiri dapat *read-only*, *read-write*, atau *write-onceread-many*. Antena yang terpasang pada chip mikro mengirimkan informasi dari chip ke *reader*. Biasanya rentang pembacaan diindikasikan dengan besarnya antena. Antena yang lebih besar mengindikasikan rentang pembacaan yang lebih jauh. *Tag* tersebut terpasang atau tertanam dalam obyek yang akan diidentifikasi. *Tag* dapat di-*scan* dengan *reader* bergerak menggunakan gelombang radio.



Gambar 2.2 Tag RFID

Tag RFID sangat bervariasi dalam hal bentuk dan ukuran. Sebagian *tag* mudah ditandai, misalnya *tag* anti-pencurian yang terbuat dari plastik keras yang dipasang pada barang-barang di toko. *Tag* untuk tracking hewan yang ditanam di bawah kulit berukuran tidak lebih besar dari bagian lancip dari ujung pensil. Bahkan ada *tag* yang lebih kecil lagi yang telah dikembangkan untuk ditanam di dalam serat kertas uang.

Tag versi paling sederhana adalah *tag* pasif, yaitu *tag* yang tidak memiliki catu daya sendiri serta tidak dapat menginisialisasi komunikasi dengan *reader*. Sebagai gantinya, *tag* merespon emisi frekuensi radio dan menurunkan dayanya dari gelombang-gelombang energi yang dipancarkan oleh *reader*. Sebuah *tag* pasif minimum mengandung sebuah indentifier unik dari sebuah item yang dipasangi *tag* tersebut. Data tambahan dimungkinkan untuk ditambahkan pada *tag*, tergantung kepada kapasitas penyimpanannya.

Dalam keadaan yang sempurna, sebuah *tag* dapat dibaca dari jarak sekitar 10 hingga 20 kaki (1 kaki = 3,2 meter). *Tag* pasif dapat beroperasi pada frekuensi rendah (*low frequency*, LF), frekuensi tinggi (*high frequency*, HF), frekuensi ultra tinggi (*ultra-high frequency*, UHF), atau gelombang mikro (*microwave*). Contoh aplikasi *tag* pasif adalah pada waktu masuk gedung, barang-barang konsumsi dan lain-lain.

Tag semipasif adalah versi *tag* yang memiliki catu daya sendiri (baterai) tetapi tidak dapat menginisialisasi komunikasi dengan *reader*. Dalam hal ini baterai digunakan oleh *tag* sebagai catu daya untuk melakukan fungsi yang lain seperti pemantauan keadaan lingkungan dan mencatu bagian elektronik internal *tag*, serta untuk memfasilitasi penyimpanan informasi. *Tag* versi ini tidak secara aktif memancarkan sinyal ke *reader*. *Tag* semi pasif dapat dihubungkan dengan sensor untuk menyimpan informasi untuk peralatan keamanan kontainer.

Tag aktif adalah *tag* yang selain memiliki antena dan chip juga memiliki catu daya dan pemancar serta mengirimkan sinyal kontinyu. *Tag* versi ini biasanya memiliki kemampuan baca tulis, dalam hal ini data *tag* dapat ditulis ulang dan/atau dimodifikasi. *Tag* aktif dapat menginisiasi komunikasi dan dapat berkomunikasi pada jarak yang lebih jauh, hingga 750 kaki, tergantung kepada daya baterainya. Harga *tag* ini merupakan yang paling mahal dibandingkan dengan versi lainnya.

Tabel 2.1 Karakteristik Umum Tag RFID

	***Tag* pasif**	***Tag* semipasif**	***Tag* aktif**
Catu daya	Eksternal (dari *reader*)	Baterai internal	baterai internal
Rentang baca	dapat mencapai 20 kaki	dapat mencapai 100 Kaki	dapat mencapai 750 Kaki
Tipe memori	umumnya read-only	Read-write	read-write
Harga	$.20 hingga beberapa dolar	$2 hingga $10	$20 atau lebih
Usia *tag*	dapat mencapai 20 tahun	2 sampai 7 tahun	5 sampai 10 tahun

Seperti telah dilihat pada tabel 2.1 bahwa *tag* memiliki tipe memori yang bervariasi yang meliputi *read-only*, *read/write*, dan *write-once read-many*. *Tag read-only* memiliki kapasitas memori minimal (biasanya kurang dari 64 bit) dan mengandung data yang terprogram permanen sehingga tidak dapat diubah. Informasi utama yang terkandung di dalam *tag* seperti ini adalah informasi identifikasi item. *Tag* dengan tipe memori seperti ini telah banyak digunakan di perpustakaan dan toko persewaan video.

Pada *tag* dengan tipe memori *read/write*, data dapat dimutakhirkan jika diperlukan. Sebagai konsekuensinya kapasitas memorinya lebih besar dan harganya lebih mahal dibandingkan *tag read-only*. *Tag* seperti ini biasanya digunakan ketika data yang tersimpan didalamnya perlu pemutakhiran seiring dengan daur hidup produk, misalnya di pabrik.

Tag dengan tipe memori *write-once read-many* memungkinkan informasi disimpan sekali, tetapi tidak membolehkan perubahan berikutnya terhadap data. *Tag* tipe ini memiliki fitur keamanan *read-only* dengan menambahkan fungsionalitas tambahan dari *tag read/write*.

Keunggulan dari tag pasif yang dipakai dalam sistem ini, yaitu :

1. Terdapat data yang telah diprogram secara permanen sehingga tidak dapat diubah.
2. Tidak memerlukan catu daya (baterai). Sumber tegangan berasal dari *reader* yang digunakan.

3. Usia pemakaian lebih lama dibandingkan dengan *tag* yang lainnya.
4. Harganya lebih murah dibandingkan dengan *tag* yang lainnya.
5. *Tag* pasif dapat beroperasi pada frekuensi rendah (*low frequency*, LF), frekuensi tinggi (*high frequency*, HF), frekuensi ultra tinggi (*ultra-high frequency*, UHF), atau gelombang mikro (*microwave*)

2.1.3 RFID *Reader*

Untuk berfungsinya sistem RFID diperlukan sebuah *reader* atau alat *scanning* device yang dapat membaca *tag* dengan benar dan mengkomunikasikan hasilnya ke suatu basis data yang dapat dilihat pada gambar 2.3.



Gambar 2.3 RFID Reader

Sebuah *reader* menggunakan antenanya sendiri untuk berkomunikasi dengan *tag*. Ketika *reader* memancarkan gelombang radio, seluruh *tag* yang dirancang pada frekuensi tersebut serta berada pada rentang bacanya akan memberikan respon. Sebuah *reader* juga dapat berkomunikasi dengan *tag* tanpa *line of sight* langsung, tergantung kepada frekuensi radio dan tipe *tag* (aktif, pasif atau semipasif) yang digunakan. *Reader* dapat memproses banyak item sekaligus. Menurut bentuknya, *reader* dapat berupa *reader* bergerak seperti peralatan tangan atau stasioner seperti peralatan *point-of-sale* di supermarket. *Reader* dibedakan berdasarkan kapasitas penyimpanannya, kemampuan pemrosesannya, serta frekuensi yang dapat dibacanya.

2.1.4 Basis Data

Basis data merupakan sebuah sistem informasi logistik pada posisi *back-end* yang bekerja melacak dan menyimpan informasi tentang item ber-*tag*. Informasi yang tersimpan dalam basis data dapat terdiri dari *identifier* item, deskripsi, pembuat, pergerakan dan lokasinya. Tipe informasi yang disimpan dalam basis data dapat bervariasi tergantung kepada aplikasinya. Basis data juga dapat dihubungkan dengan jaringan lainnya seperti *local area network* (LAN) yang dapat menghubungkan basis

data ke Internet. Konektivitas seperti ini memungkinkan *sharing* data tidak hanya pada lingkup basis data lokal.

2.1.5 Frekuensi Radio sebagai Karakteristik Operasi Sistem RFID

Pemilihan frekuensi radio merupakan kunci kerakteristik operasi sistem RFID. Frekuensi sebagian besar ditentukan oleh kecepatan komunikasi dan jarak baca terhadap *tag*. Secara umum tingginya frekuensi mengindikasikan jauhnya jarak baca. Frekuensi yang lebih tinggi mengindikasikan jarak baca yang lebih jauh.

Pemilihan tipe frekuensi juga dapat ditentukan oleh tipe aplikasinya. Aplikasi tertentu lebih cocok untuk salah satu tipe frekuensi dibandingkan dengan tipe lainnya karena gelombang radio memiliki perilaku yang berbeda-beda menurut frekuensinya.

Sistem RFID menggunakan rentang frekuensi yang tak berlisensi dan diklasifikasikan sebagai peralatan *industrialscientific-medical* atau peralatan berjarak pendek (*short-range device*) yang diizinkan oleh FCC. Peralatan yang beroperasi pada bandwidth ini tidak menyebabkan interferensi yang membahayakan dan harus menerima interferensi yang diterima. FCC juga mengatur batas daya spesifik yang berasosiasi dengan masing-masing frekuensi.

Kombinasi dari level-level frekuensi dan daya yang dibolehkan menentukan rentang fungsional dari suatu aplikasi tertentu seperti keluaran daya dari *reader*. Berikut ini adalah penjelasan tentang frekuensi utama yang digunakan oleh sistem RFID, yang terdapat pada tabel 2.2.

Tabel 2.2 Frekuensi RFID yang umum beroperasi pada tag pasif.

Gelombang	Frekuensi	Rentang dan laju baca	Contoh penggunaan
LF	125 KHz	~1.5 kaki; kecepatan baca rendah	*Access control, animal tracking, point of sale Applications*
HF	13.56 MHz	~3 kaki; kecepatan baca Sedang	*Access control, smart cards, item-level tracking*
UHF	860-930 MHz	up to 15 kaki; kecepatan baca tinggi	*Pallet tracking, supply chain management*
Gelombang mikro	2.45/5.8 GHz	~3 kaki; Kecepatan baca tinggi	*Supply chain management*

2.1.6 Kategori Sistem RFID

Secara kasar sistem-sistem RFID dapat dikelompokkan menjadi empat kategori sebagai berikut :

1. Sistem EAS (*Electronic Article Surveillance*)

   Umumnya digunakan pada toko-toko untuk menyensor ada tidaknya suatu item. Produk-produk diberi *tag* dan *reader* berantena besar ditempatkan di masing-masing pintu keluar toko untuk mendeteksi pengambilan item secara tidak sah.

2. Sistem *Portable Data Capture*

   Dicirikan oleh penggunaan *reader* RFID yang portabel yang memungkinkan sistem ini digunakan dalam seting yang bervariasi

3. Sistem *Networked*

   Dicirikan oleh posisi *reader* yang tetap yang terhubung secara langsung ke suatu sistem manajemen informasi terpusat, sementara transponder berada pada orang atau item-item yang dapat dipindahkan.

4. Sistem *Positioning*

   Digunakan untuk identifikasi lokasi item-item atau kendaraan.

Sistem RFID yang digunakan pada perancangan sistem pembayaran tol ini termasuk ke dalam sistem yang ke-3 yaitu sistem *Networked,* dimana posisi RFID *reader* terhubung secara langsung ke sebuah PC, sementara transponder terdapat pada pelanggan tol.

2.2 Mikrokontroler AT89C51

Mikrokontroler AT89C51 termasuk dalam *MCS-51*^{TM} dari Intel. Sebuah mikrokontroler tidak dapat bekerja bila tidak diberi program kepadanya. Program tersebut memberitahu mikrokontroler apa yang harus dilakukan. Salah satu keunggulan dari AT89C51 adalah dapat diisi ulang dengan program lain sebanyak 1000 kali pengisian. Instruksi-instruksi perangkat lunak berbeda untuk masing-masing jenis mikrokontroler. 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.

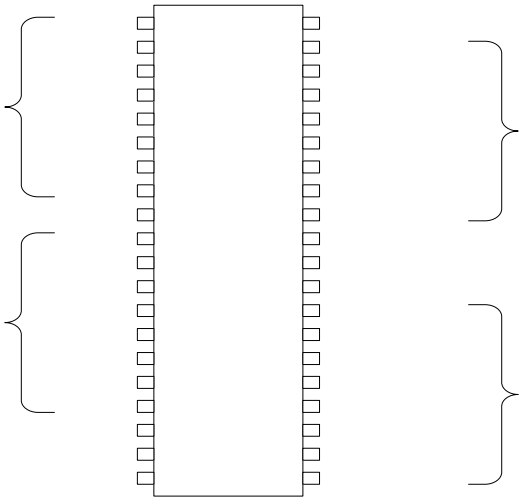
2.2.1 Karakteristik Mikrokontroler AT89C51

Mikrokontroler AT89C51 memiliki beberapa fasilitas, diantaranya sebagai berikut :

a. Sebuah CPU (*Central Prossesing Unit*) 8 bit yang termasuk keluarga Osilator *internal* dan rangkaian *timer*.
b. Flas memori 4 Kbyte.
c. RAM internal 128 byte.
d. Empat buah *programable* port I/O, masing-masing terdiri atas 8 buah jalur I/O.
e. Lima buah jalur interupsi (2 buah interupsi eksternal dan 3 buah internal).
f. Sebuah port serial dengan kontrol *serial full duplex* UART.
g. Kemampuan melaksanakan operasi perkalian, pembagian dan operasi boolean (bit).
h. Kecepatan pelaksanaan instruksi per siklus 1 mikrodetik pada frekuensi clock 12 MHz.

2.2.2 Deskripsi Kaki (Pin)

Susunan pin-pin mikrokontroler AT89C51 memperlihatkan pada gambar 2.4 dibawah ini.



Gambar. 2.4 Diagram Pin Mikrokontroler AT89C51

Penjelasan dari masing-masing pin adalah sebagai berikut :

1. Pin 1 sampai 8 *(Port 1)*

   merupakan port paralel 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 AT89C51. Pin ini dihubungkan dengan rangkaian *power on reset*.

3. Pin 10 sampai 17 *(Port 3)*

   adalah port paralel 8 bit dua arah *(output-input)* yang memiliki fungsi pengganti. Fungsi pengganti meliputi T_xD (*Transmision Data*), R_xD *(Received Data*), Int0 (*Interupsi 0*), Int 1 (*Interupsi 1*), TO (*timer O*), T1 (*timer 1*), WR (*Write*) dan RD (*Read*). Bila fungsi pengganti tidak dipakai, pin-pin ini dapat digunakan sebagai port paralel 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 V_{ss} atau *ground.*

6. Pin 21 sampai 28 *(port 2).*

   adalah port paralel 2 (Port 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)*

   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 O)*

    merupakan port paralel 8 bit open drain dua arah. Bila diberi logika rendah, mikrokontroler akan melaksanakan seluruh instruksi dari memori program luar.

11. Pin 40 (V_{cc}) dihubungkan ke V_{cc} (+5 volt).

2.2.3 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 demikian, alamat data memori 16 bit dapat dihasilkan melalui register DPTR (*Data Pointer Register*).

Memori program hanya dapat dibaca tidak bisa ditulisi, karena disimpan dalam *Flash Memori*. Memori program sebesar 64 Kbyte dapat dimasukan dalam EPROM eksternal.

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

Mikrokontroler AT89C51 memiliki 5 buah ruang alamat, yaitu :

1. Ruang alamat kode (*Code Address Space*) sebanyak 64 Kbyte, yang seluruhnya merupakan ruang alamat kode eksternal.
2. Ruang alamat memori data internal yang dapat dialamati secara langsung, yang terdiri atas RAM sebanyak 128 byte, hardware register sebanyak 128 byte.
3. Ruang alamat memori data internal yang dialamati secara tidak langsung sebanyak 128 byte, seluruhnya diakses dengan pengalamatan tidak langsung.
4. Ruang alamat memori data eksternal sebanyak 64 Kbyte yang dapat ditambahkan oleh pemakai.
5. Ruang alamat bit dapat diakses dengan pengalamatan langsung.

2.2.4 *Special Function Register* (SFR)

SFR berisi register-register dengan fungsi tertentu yang tersediakan oleh mikrokontroler seperti timer, serial, dan lain-lain AT89C51 memiliki 21 SFR yang terletak pada alamat 80H-FFH. Masing-masing ditunjukkan pada tabel 2.3 yang meliputi simbol, nama, dan alamatnya.

Tabel 2.3 Special 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 2*	90H
P2	*Port 3*	A0H
P3	*Port 4*	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 byte*	8CH
TL 0	*Timer/Counter Low byte*	8AH
TH 1	*Timer/Counter High byte*	8DH
TL 1	*Timer/Counter Low byte*	8BH
SCON	*Serial Control*	98H
SBUF	*Serial Data Buffer*	99H
PCON	*Power Control*	87H

2.2.5 Timer AT89C51

Mikrokontroler AT89C51 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 register TCON. Timer dapat diaktifkan melalui perangkat keras maupun perangkat lunak.

Perioda waktu *timer*/*counter* dapat dihitung menggunakan rumus 2.1 dan 2.2 sebagai berikut :

1. Sebagai *timer/counter* 8 bit

$$T = (255 - TLx) * \frac{12}{frekuensi.XTAL}\ \mu s \quad \text{..... (2.1)}$$

2. Sebagai *timer/counter* 16 bit

$$T = (65535 - THx\ TLx) * \frac{12}{frekuensi.XTAL}\ \mu s \quad \text{..... (2.2)}$$

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

Dapat diakses secara bit

	Register Timer				Register Interupsi			
88H	TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0
	TCON.7	TCON.6	TCON.5	TCON.4	TCON.3	TCON.2	TCON.1	TCON.0

Tidak dapat diakses secara bit

	Timer 1				Timer 0			
89H	Gate	C/T	M1	M0	Gate	C/T	M1	M0

Gambar 2.5 Register TCON dan TMOD

Pengontrolan kerja *timer/counter* diatur oleh register TCON. Register ini bersifat *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 gambar 2.5 dan dijelaskan pada tabel 2.4.

Tabel 2.4 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 memperhatikan 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 2.5

Tabel 2.5 Mode Timer

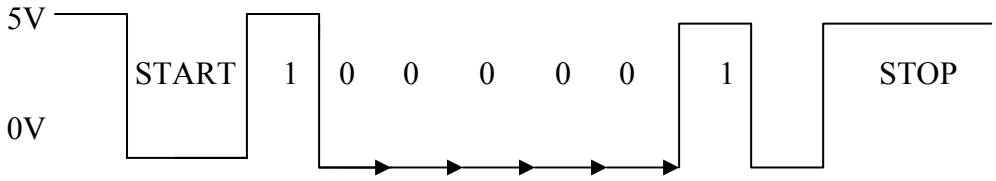
M1	M0	Mode	Operasi
0	0	0	Timer 13 bit
0	1	1	Timer/counter 16 bit
1	0	2	Timer 8 bit di mana nilai timer tersimpan pada TLx. Register THx berisi nilai isi ulang yang akan dikirim ke TLx setiap *overflow*.
1	1	3	Pada mode ini, AT89C51 bagaikan memiliki 3 buah timer. Timer 0 terpisah menjadi 2 buah timer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

2.3 Komunikasi Data Serial

Dikenal dua cara komuniksasi data secara serial, yaitu komunikasi data secara data serial secara *sinkron* dan komunikasi data serial secara *asinkron*. Pada komunikasi data serial *sinkron*, *clock* dikirimkan bersama-sama dengan data serial, sedangkan komunikasi data serial *asinkron*, *clock* tidak dikirimkan bersama data serial, tetapi dibangkitkan secara sendiri-sendiri baik pada sisi pengirim (*Transmitter*) maupun pada sisi penerima (*receiver*). Pada IBM PC kompatibel port serialnya termasuk jenis *asinkron*. Komunikasi data serial ini dikerjakan oleh *UART* (*Universal Asynchronous Reeiver/Transmitter*). *IC* UART dibuat khusus untuk mengubah data parallel menjadi data serial dan menerima data serial yang kemudian diubah menjadi data parallel. *IC UART* 8250 dari *Intel* merupakan salah satunya. Selain berbentuk *IC* mandiri, berbagai macam *mikokontroller* ada yang dilengkapi dengan *UART*, misalnya keluarga *mikrokontroller* MCS51 (termasuk AT89C2051).

Pada *UART,* kecepatan pengiriman data (*baud rate*) dan fase *clock* pada sisi *transmitter* dan pada sisi *receiver* harus sinkron. Untuk itu diperlukan *sinkonisasi* antara *transmitter* dan *receiver*. Hal ini dilakukan oleh bit ‘Start’ dan bit ‘Stop’. Ketika saluran transmisi dalam keadaan *idle*, output *UART* adalah dalam keadaan logika ‘1’. Ketika transmitter ingin mengirimkan data, output *UART* akan diset lebih dulu ke logika ‘0’ untuk waktu satu bit. Sinyal ini pada receiver akan dikenali sebagai sinyal ‘Start’ yang digunakan untuk mensinkronkan fase clocknya sehingga sinkron dengan fase clock transmitter. Selanjutnya, data akan dikirimkan secara serial dari bit paling rendah (bit 0) sampai bit tertinggi. Selanjutnya, akan dikirim sinyal ‘Stop’ sebagai akhir dari pengiriman data serial. Cara pemberian kode data yang disalurkan tidak ditetapkan

secara pasti. Berikut ini adalah contoh pengiriman huruf ‘A’ dalam format ASCII (41 Heksa/10000001) tanpa bit paritas yang terdapat pada gambar 2.6.



Gambar 2.6 Pengiriman huruf ‘A’ tanpa bit paritas

Kecepatan transmisi (baud rate) dapat dipilih bebas dalam rentang tertentu. Baud rate yang umum dipakai adalah 110, 135, 150, 300, 600, 1200, 2400, dan 9600 (bit/detik). Dalam komunikasi data serial, baud rate dari kedua alat yang berhubungan harus diatur pada kecepatan yang sama. Selanjutnya, harus ditentukan panjang data (6,7 atau 8 bit), paritas (genap ganjil atau tanpa pritas), dan jumlah bit ‘STOP’ (1, 1,5, atau 2 bit).

2.3.1 Karakteristik Sinyal RS 232

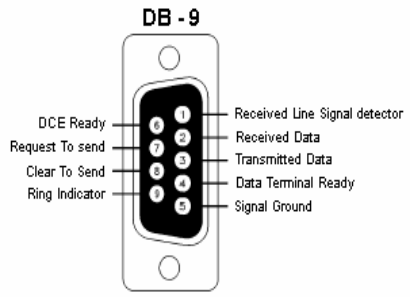
Standar sinyal komunikasi serial yang banyak digunakan adalah standar RS232 yang dikembangkan oleh *Electronic Industry Association and the Telecommunication Industry Association* (EIA/TIA) yang pertama kali dipublikasikan pada tahun 1962. Ini sebelum IC TTL popular sehingga sinyal ini tidak ada hubungan 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-Terminating 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 pasti sehingga harus dihindari. Demikian juga, level tegangan negative dari -25 Volt atau lebih positif dari +25 Volt juga harus dihindari karena tegangan tersebut dapat merusak line driver pada saluran RS232.

2.3.2 Konfigurasi Port Serial

Gambar 2.7 merupakan konektor port serial 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 2.7 Port serial DB-9 terdapat 9 pin yang mempunyai fungsi yang berbeda, konfigurasi pin DB-9 dapat dilihat pada tabel 2.15



2.7 Konfigurasi 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 2.6 berikut ini :

Tabel 2.6 Sinyal-sinyal pada konektor RS 232

Nomor Pin	Nama Sinyal	Direction	Keterangan
1	DCD (*Data Carrier Detect/Received Line Singnal 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 mengirimkan 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 suda siap.
7	RST (*Request to Send)*	*Out*	Pada saluran ini DCE diminta mengirim data oleh DTE.

Tabel 2.6 lanjutan

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 memberitahu DTE bahwa sebuah stasiun menghendaki hubungan denganya.

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 0000.0400h 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 2.7.

Tabel 2.7 Nama Register yang Digunakan Beserta Alamatnya

Nama Register	COM1	COM2
TX Buffer	3F8h	2F8h
RX Buffer	3F8h	2F8h
Boud Rate Divisor Latch LSB	3F8h	2F8h
Boud 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. *Boud Rate Divisor Latch LSB*, digunakan untuk menampung byte bobot rendah untuk pembagi *clock* pada *IC UART* agar didapat *baud rate* yang tepat.

4. *Boud 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 dipilih dari 0001h sampai FFFFh. Berikut ini adalah tabel angka pembagi yang sering digunakan.

Tabel 2.8 Angka Pembagi Clock pada IC UART

Baud Rate (bit/detik)	Angka Pembagi
300	0180H
600	0C00H
1200	0060H
1800	0040H
2400	0030H
4800	0018H
9600	000CH

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

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

Tabel 2.9 Rincian Bit pada Interrupt Enable Register.

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

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

Tabel 2.10 Rincian Bit pada *Interrupt Identification Register*

Nomor Bit	Keterangan
0	0: Interrupsi 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 Register

Tabel 2.10 Lanjutan

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 diubah atau tidak. Berikut ini adalah tabel rincian bit pada *Line Control Register*.

Tabel 2.11 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 pariti 0: tanpa pariti 1: dengan pariti
4	0: pariti ganjil 1: pariti genap
5	1: bit pariti 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 2.12 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 2.12 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 2.13 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 menampung bit-bit yang menyatakan status dari saluran hubungan dengan modem. Berikut ini tabel rincian bit pada *Modem Status Register*.

Tabel 2.14 Rincian Bit pada Modem Status Register

Nomor bit	Keterangan
0	1: menyatakan adanya perubahan keadaan di saluran *Clear to Send* (CTS)
1	1: menyatakan adanya perubahan keadaan di saluran *Data Set Ready* (DSR)
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* (CST) sudah dalam keadaan aktif
5	1: menyatakan saluran *Data Set Ready* (DSR) sudah dalam keadaan aktif

Tabel 2.14 Lanjutan

6	1: menyatakan bahwa saluran *Ring Indocator* (RI) sudah dalam keadaan aktif
7	1: menyatakan bahwa saluran *Receive Line Signal Detect* (DCD) sudah dalam keadaan aktif

2.3.3 *Flow Control*

Jika kecepatan transfer data dari DTE ke DCE (misal 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.

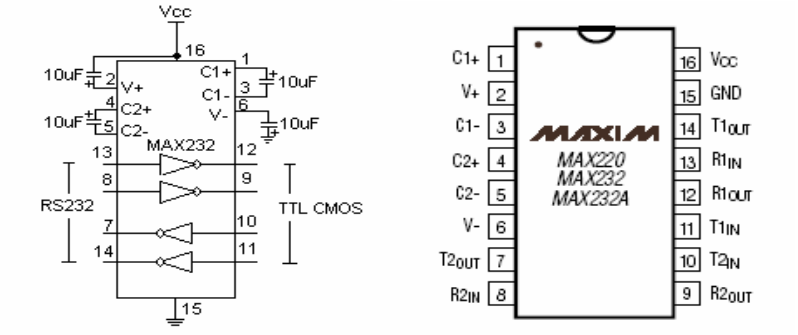
Dikenal dua macam *flow control*, yaitu secara *software* dan secara *hardware*. *Flow control* secara *software* atau sering disebut Xon atau Xoff *flow control* menggunakan karakter Xon (tipikalnya karakter *ASCII* 17) dan karakter Xoff (tipikalnya karakter *ASCII* 19) untuk melakukan kontrol. DCE akan mengirimkan Xoff ke komputer untuk 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 control 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 menset 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 diset kembali.

2.4 RS232

Port serial pada PC dengan standar RS232 merupakan salah satu fasilitas yang tersedia agar sebuah PC dapat dihubungkan dengan perangkat luar. Diperlukan sebuah chip interface untuk mengkonversi level tegangan yang berbeda yaitu chip MAX232. Level tegangan yang berbeda pada tiap saluran port serial adalah ± 12 Volt DC. Pada saat level logika “1”, maka tegangan yang keluar pada port serial adalah sebesar -12

Volt dan sebaliknya pada level logika “0”, maka tegangan yang keluar pada port serial adalah sebesar +12 Volt. Hal ini disebut *negative logic* karena kondisi logika tertentu justru menghasilkan tegangan yang terbalik. MAX232 akan merubah level tegangan RS232 menjadi level tegangan TTL (5 volt) sekaligus mengubah logika negatif menjadi logika positif. Berikut ini adalah gambar konfigurasi dari RS232 yang dapat dilihat pada gambar 2.8.

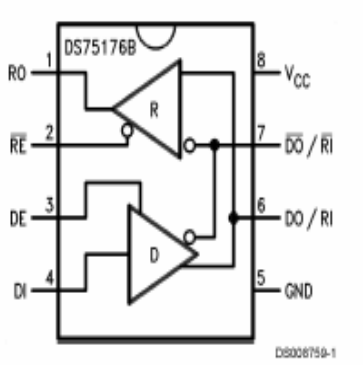


Gambar 2.8 IC dan Konfigurasi Pin MAX232

Dari Gambar 2.8 diatas terlihat adanya 4 buah kapasitor yang berfungsi sebagai “pompa elektronik”, keempat kapasitor inilah yang berfungsi untuk menyesuaikan level tegangan yang berbeda antara level tegangan RS232 dengan TTL. Namun karena keterbatasan dari RS232, seperti panjang komunikasi sepanjang kabel yang digunakan maximal 50 *feet* (15 meter) dan hanya dapat berkomunikasi secara *one to one*, maka dari itu dibutuhkan *converter* dari RS232 ke RS485 agar dapat memanfaatkan keunggulan dari sistem komunikasi RS485.

2.4 RS 485

Untuk melakukan jaringan komunikasi serial menggunakan RS485 dibutuhkan suatu tranceivers DS75176BT seperti yang terlihat pada gambar 2.9. DS75176BT adalah *tri state* yang mempunyai kecepatan yang berbeda. Untuk memenuhi keperluan standar RS485, maka pada bus tranceiver dirancang dengan memperpanjang jarak pada umumnya dari +12 v- 7v. Untuk mentransmisikan data secara *multipoint*. Dalam hal ini dapat diatasi dengan menghubungkan RS 422. Dilihat dari kemampuan *tri state* nya, bahwa yang melebihi *driver* output, semuanya diset dengan cara yang sama dari +12 V – 7 V.



Gambar 2.9 DS75176BT

Bus akan saling berbenturan dikarenakan adanya pemborosan tegangan, dalam hal ini yang mengatur adalah *thermal shutdown circuit*. Ini dapat bekerja pada suhu 0^0 sampai 70 ^{0} dan bekerja pada tegangan 4.75V sampai 5.25V. DS 75176BT dapat dilihat pada gambar 2.9, keistimewaan dari DS75176 adalah

1. Dapat digunakan sebagai jalur komunikasi data secara *multipoint*
2. Transmisi dapat dihubungkan dengan RS422
3. Waktu *delay* nya 22 ns
4. Adanya sistem *Thermal shutdown protection*
5. Bekerja pada tegangan -7V to +12V
6. Adanya perbedaan *ground* pada *bus*
7. Sifatnya *High Impedance* untuk bus dengan *driver* pada *tri state* atau dengan *power off*.

2.6 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* (V_{in}) lebih besar daripada tegangan *output* (V_{out}). Biasanya perbedaan tegangan *input* dengan *output* yang direkomendasikan tertera pada *datasheet* komponen tersebut. Contoh LM7805 yang dapat dilihat pada gambar 2.10.



Gambar 2.10 IC LM7805

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 programer 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 1 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 ekstensi HEX". File dengan ekstensi HEX inilah yang akan diisikan kedalam 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 sebuah baris, jika ada satu bagian yang tidak ada maka spasi atau TAB sebagai pemisah bagian tetap harus 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 dan bagian *mnemonic*.

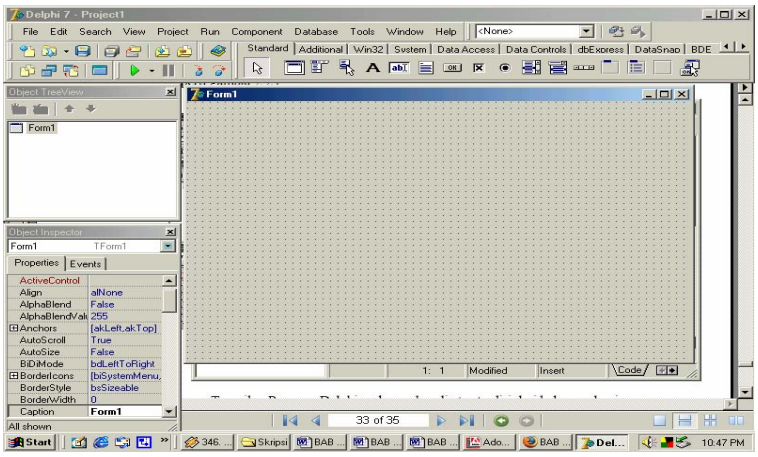
2.10 Bahasa Pemrograman Borland Delphi

Bahasa pemrograman Borland Delphi bekerja dalam sistem operasi Windows. Delphi mempunyai kemampuan yang luas dan canggih. Berbagai aplikasi dapat dilakukan seperti mengolah teks, grafik, angka, *database* dan aplikasi *web*. Keunggulan Delphi terletak pada produktivitas, kualitas, pengembangan perangkat lunak, kecepatan kompilasi dan pola desain yang menarik. Selain itu Delphi juga dapat menangani data dalam berbagai format *database*, misalnya format MS-Acces, Sybase, Oracle, FoxPro dan lain-lain. Untuk memudahkan pemrograman, Delphi menyediakan fasilitas pemrograman dalam dua kelompok yaitu, objek dan bahasa pemrograman. Objek adalah suatu komponen yang mempunyai bentuk fisik dan biasanya dapat dilihat. Objek biasanya dipakai untuk melakukan tugas tertentu dan mempunyai batasan-batasan

tertentu. Sedangkan bahasa pemrograman adalah kumpulan teks yang mempunyai arti tertentu dan disusun dengan aturan tertentu untuk menjalankan tugas tertentu. Delphi menggunakan struktur bahasa pemrograman Pascal yang sudah sangat dikenal dikalangan pemrogram. Gabungan dari objek dan bahasa pemrograman berorientasi objek disebut dengan *Object Oriented Programming* (OOP).

2.9.1 IDE Delphi

Integrated Development Environment atau IDE Delphi terdiri dari : *Main Window, Toolbar, Component Pallet, Form, Code Editor* dan *Object Inspector*. Untuk memulai Delphi dari sistem Operasi Windows pilih Program, kemudian pilih Borland Delphi lalu klik Delphi maka akan terlihat tampilan utamanya seperti gambar 2.11.



Gambar 2.11 Tampilan Dasar Delphi

Tampilan Program Delphi pada gambar di atas terdiri dari beberapa bagian yang tampilannya dapat diubah-ubah yaitu :

1. Windows Utama Merupakan pusat pengaturan di dalam Delphi yang terletak di bagian atas. Jendela ini berisi menu, *toolbar* dan kumpulan *tab (page)* atau lembaran yang berisi *icon object* sesuai kategori yang disebut sebagai *Component Pallete*.
2. Windows Object Inspector Terletak di bagian kiri bawah. Windows ini memiliki dua buah halaman, yaitu halaman propertis dan halaman *event*. Halaman propertis digunakan untuk mengubah properti komponen. Properti dengan tanda + menunjukkan bahwa properti tersebut mempunyai subproperti. Dan *event* berfungsi untuk menangani kejadian-kejadian berupa prosedur yang dapat direspon oleh sebuah komponen.



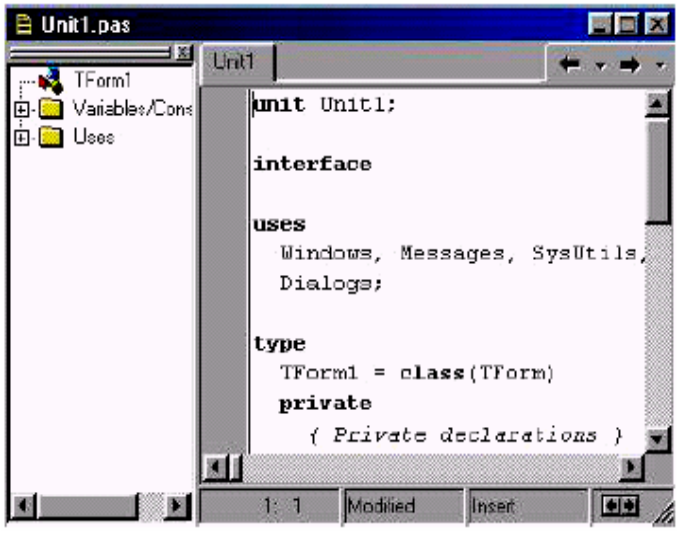
Gambar 2.12 Tampilan Windows Object Inspector

3. Windows Form

   Terletak di bagian kanan *Windows Editor Program*. Untuk proses yang baru Windows ini biasanya ditampilkan dengan judul "*Form1*". *Form* dipakai untuk merancang Windows bagi aplikasi baru yang sedang dibuat. Sebuah aplikasi dapat berisi beberapa *form* dan minimal harus memiliki sebuah *form* yang nantinya dipakai untuk mendesain tampilan program aplikasi

4. Code Editor

   Tempat dimana menuliskan kode program. *Code Editor* dilengkapi dengan fasilitas *highlight* yang memudahkan pemakai menemukan kesalahan pada kode program. *Title bar* yang terletak pada bagian bawah terdapat nomor baris/kolom, *modified* dan *insert/overwrite*.



Gambar 2.13 Lembar Kerja Code Editor

2.9.2 Dasar Pembuatan Program

Langkah-langkah dasar yang harus ditempuh untuk membuat program aplikasi dan yang perlu dipahami oleh pemakai Delphi adalah :

1. Merancang antarmuka visual dari aplikasi dengan memilih komponen-komponen yang diinginkan dari *Component Pallet* dan menempatkannya pada *form*.
2. Dengan memakai *Windows Object Inspector* ubah nilai propertis milik *form* dan objek yang terdapat di dalamnya. Ini bertujuan untuk mendapatkan tampilan yang dikehendaki.
3. Tulis kode pada editor program untuk *event* pada objek yang diinginkan. Perlu diketahui *event* adalah mekanisme penghubung antara suatu kejadian (seperti gerakan *mouse*, penekanan tombol dan lain-lain) pada komponen dengan prosedur yang merespon (menerjemahkan dan merealisasikannya menjadi suatu tindakan kejadian tersebut).