BAB II

LANDASAN TEORI

2.1 *Radio Frequency Identification* (RFID)

RFID adalah sebuah teknologi penangkapan data yang memanfaatkan frekuensi radio dalam sistem kerjanya yang dapat digunakan secara elektronik untuk mengidentifikasi, melacak dan menyimpan informasi yang tersimpan dalam *tag* RFID. Perhatian terhadap RFID dalam lingkungan media massa maupun akademis yang populer telah meningkat dalam beberapa tahun ini. Salah satu buktinya adalah usaha dari organisasi-organisasi yang besar seperti Wal-Mart, Procter and Gamble, dan Departemen Pertahanan Amerika Serikat untuk menggunakan RFID sebagai suatu alat untuk mengontrol secara otomatis terhadap rantai supply mereka. Harga *tag* yang menurun dan standarisasi yang dinamis telah menyebabkan kita berada pada ambang ledakan penggunaan RFID.

Para pengamat RFID menganggap RFID sebagai suksesor dari *barcode* optik yang banyak dicetak pada barang-barang dagangan dengan dua keunggulan pembeda.

a) Identifikasi yang unik

   Sebuah *barcode* mengindikasikan tipe obyek tempat ia dicetak, misalnya “Ini adalah sebatang coklat merek ABC dengan kadar 70% dan berat 100 gram”. Sebuah *tag* RFID selangkah lebih maju dengan mengemisikan sebuah nomor seri unik di antara jutaan obyek yang identik, sehingga ia dapat mengindikasikan “Ini adalah sebatang coklat merek ABC dengan kadar 70%

dan berat 100 gram, nomor seri 897348738". Identifikasi yang unik dalam RFID ini dapat berperan sebagai pointer terhadap masukan basis data yang menyimpan banyak *history* transaksi untuk item-item individu.

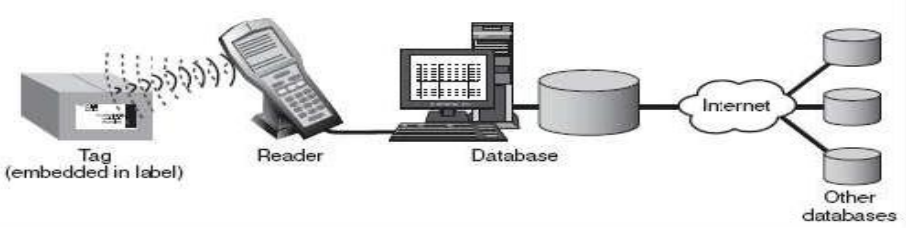
b) Otomasi

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.

Sebagai suksesor dari *barcode*, RFID dapat melakukan kontrol otomatis untuk banyak hal. Sistem-sistem RFID menawarkan peningkatan efisiensi dalam pengendalian inventaris (*inventory control*), logistik dan manajemen rantai *supply* (*supply chain management*).

2.1.1 Komponen-komponen Utama Sistem RFID

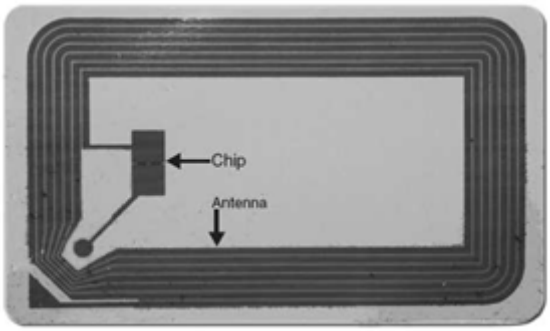
Secara garis besar sebuah sistem RFID terdiri atas tiga komponen utama, yaitu *tag*, *reader* dan basis data (Gambar 2.1). Secara ringkas, mekanisme kerja yang terjadi dalam sebuah sistem RFID adalah bahwa sebuah *reader* frekuensi radio melakukan *scanning* terhadap data yang tersimpan dalam *tag*,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



Gambar 2.1 *Komponen Utama Sistem RFID*

2.1.1.1 *Tag* RFID

Sebuah *tag* RFID atau *transponder*, terdiri atas sebuah mikro (*microchip*) dan sebuah antena seperti yang terlihat pada Gambar 2.2. Chip mikro itu sendiri dapat berukuran sekecil butiran pasir atau 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 maupun stasioner menggunakan gelombang radio.



Gambar 2.2 *Tag RFID*

Tag RFID sangat bervariasi dalam hal bentuk dan ukuran. *Tag* versi paling sederhana adalah *tag* pasif, yaitu *tag* yang tidak memiliki catu daya sendiri serta tidak dapat menginisiasi 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.

Harga *tag* pasif lebih murah dibandingkan harga versi lainnya. Perkembangan *tag* murah ini telah menciptakan revolusi dalam adopsi RFID dan memungkinkan penggunaannya dalam skala yang luas baik oleh organisasi-organisasi pemerintah maupun industri.

Tag semipasif adalah versi *tag* yang memiliki catu daya sendiri (baterai) tetapi tidak dapat menginisiasi 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 memiliku catu daya dan pemancar serta mengirimkan sinyak 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.

Dalam keadaan yang sempurna, sebuah *tag* dapat dibaca dari jarak sekitar 10 hingga 20 kaki. *Tag* pasif dapat beroperasi pada frekuensi rendah (*low frequency*, LF), frekuensi tinggi (*high frequency*, HF), frekuensi ultra tinggi (*ultrahigh frequency*, UHF), atau gelombang mikro (*microwave*). Untuk lebih jelasnya dapat dilihat pada Tabel 2.1 berikut.

Tabel 2.1 *Karakteristik Umum Tag RFID*

Jenis *tag*	*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*
Usia *tag*	Dapat mencapai 20 tahun	2 sampai 7 tahun	5 samapai 10 tahun

2.1.1.2 *Reader* RFID

Untuk berfungsinya sistem RFID diperlukan sebuah *reader* atau alat *scanning device* yang dapat membaca *tag* dengan benar dan mengkomunikasikan hasilnya ke suatu basis data. Gambar 2.3 memperlihatkan beberapa contoh IC *reader* RFID yang terdapat di Indonesia.



Gambar 2.3 *Beberapa Contoh IC Reader RFID*

Sebuah *reader* menggunakan antenanya sendiri untuk berkomunikasi dengan *tag*. Ketika *reader* memancarkan gelombang radio, seluruh *tag* yang dirancang pada 6 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 genggam, atau stasioner seperti peralatan *point of sale* di supermarket. *Reader* dibedakan berdasarkan kapasitas penyimpanannya, kemampuan pemrosesannya, serta frekuensi yang dapat dibacanya.

2.1.2 Frekuensi Radio sebagai Karakteristik Operasi Sistem RFID

Pemilihan frekuensi radio merupakan kunci karakteristik operasi sistem RFID. 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. Sebagai contoh, gelombang LF memiliki kemampuan penetrasi terhadap dinding tembok yang lebih baik dibandingkan dengan gelombang dengan frekuensi yang lebih tinggi, tetapi frekuensi yang lebih tinggi memiliki laju data (*data rate*) yang lebih cepat.

Sistem RFID menggunakan rentang frekuensi yang tak berlisensi dan diklasifikasikan sebagai peralatan *industrialscientific-medical* atau peralatan

berjarak pendek (*short-range device*) yang diizinkan oleh *Federal ommunications Commission* (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.

Berikut ini adalah empat frekuensi utama yang digunakan oleh sistem RFID.

a) Band LF berkisar dari 125 KHz hingga 134 KHz. Band ini paling sesuai untuk penggunaan jarak pendek (*short-range*) seperti sistem antipencurian, identifikasi hewan dan sistem kunci mobil.
b) Band HF beroperasi pada 13.56 MHz. Frekuensi ini memungkinkan akurasi yang lebih baik dalam jarak tiga kaki dan karena itu dapat mereduksi risiko kesalahan pembacaan *tag*. Sebagai konsekuensinya band ini lebih cocok untuk pembacaan pada tingkat item (*item-level reading*). *Tag* pasif dengan frekuensi 13.56 MHz dapat dibaca dengan laju 10 to 100 *tag* perdetik pada jarak tiga kaki atau kurang. *Tag* RFID HF digunakan untuk pelacakan barang-barang di perpustakaan, toko buku, kontrol akses gedung, pelacakan bagasi pesawat terbang, pelacakan item pakaian.
c) Band UHF beroperasi di sekitar 900 MHz dan dapat dibaca dari jarak yang lebih jauh dari *tag* HF, berkisar dari 3 hingga 15 kaki. *Tag* ini lebih sensitif terhadap faktor-faktor lingkungan daripada *tag-tag* yang beroperasi pada frekuensi lainnya. Band 900 MHz muncul sebagai band yang lebih disukai untuk aplikasi rantai *supply* disebabkan laju dan rentang bacanya. *Tag* UHF pasif dapat dibaca dengan laju sekitar 100 hingga 1000 *tag* perdetik. *Tag* ini

umumnya digunakan pada pelacakan kontainer, truk, trailer, terminal peti kemas.

d) *Tag* yang beroperasi pada frekuensi gelombang mikro, biasanya 2.45 GHz dan 5.8 GHz, mengalami lebih banyak pantulan gelombang radio dari obyek-obyek didekatnya yang dapat mengganggu kemampuan *reader* untuk berkomunikasi dengan *tag*. *Tag* RFID gelombang mikro biasanya digunakan untuk manajemen rantai *supply*.

Untuk lebih jelasnya dapat dilihat pada Tabel 2.2 berikut.

Tabel 2.2 *Frekuensi RFID Yang Umum Beroperasi Pada Tag Pasif*

Gelombang	Frekuensi	Rentang dan laju baca
LF	125 Khz	▪ Dapat mencapai jarak ± 30 cm ▪ Kecepatan baca rendah
HF	13,56 Mhz	▪ Dapat mencapai jarak ± 90 cm ▪ Kecepatan baca sedang
UHF	860-930 Mhz	▪ Dapat mencapai jarak ± 4 meter ▪ Kecepatan baca tinggi
Gelombang mikro	2,45/5,8 Ghz	▪ Dapat mencapai jarak diatas 5 meter ▪ Kecepatan baca tinggi

2.2 Mikrokontroler AT89S52

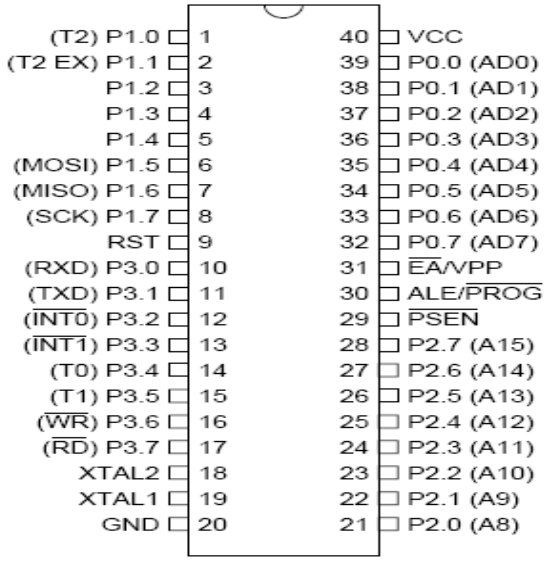
Mikrokontroller tipe Atmel AT89S52 termasuk dalam keluarga MCS51 merupakan suatu mikrokomputer CMOS 8-bit dengan daya rendah, kemampuan tinggi, memiliki 8K byte *Flash Programable and Erasable Read Only Memory* (PEROM). Perangkat ini dibuat menggunakan teknologi memori *non volatile* artinya tidak kehilangan data apabila kehilangan daya listrik. Set instruksi dan kaki keluaran AT89S52 sesuai dengan standar industri 80C51 dan 80C52. Atmel

AT89S52 adalah mikrokomputer yang sangat bagus dan fleksibel dengan harga yang rendah untuk banyak aplikasi sistem kendali. Spesifikasi penting AT89S52 :

- kompatibel dengan keluarga mikrokontroler MCS51 sebelumnya
- 8 K Bytes *In system Programmable* (ISP) *flash memori* dengan kemampuan 1000 kali baca/tulis
- tegangan kerja 4-5.0V
- bekerja dengan rentang 0 – 33MHz
- 256x8 bit RAM internal
- 32 jalur I/0 dapat diprogram
- 3 buah 16 bit *Timer/Counter*
- 8 sumber interrupt
- mode pemrograman ISP yang fleksibel (Byte dan *Page Mode*)

2.2.1 Konfigurasi Mikrokontroller AT89S52

Mikrokontroller keluarga MCS51 memiliki port-port yang lebih banyak (40 port I/O) dengan fungsi yang bisa saling menggantikan sehingga mikrokontroller jenis ini menjadi sangat digemari karena hanya dalam sebuah chip sudah bisa memenuhi banyak kebutuhan. Gambar 2.4 adalah gambar konfigurasi kaki-kaki mikrokontroler.



Gambar 2.4 *Konfigurasi Pin Mikrokontroler AT89S52*

2.2.1.1 Port 0

Port 0 adalah port dua arah masukan/keluaran 8-bit saluran terbuka. Sebagai port keluaran, tiap kaki dapat menerima masukan TTL. Ketika logika 1 dimasukkan ke kaki-kaki port 0, kaki-kaki dapat digunakan sebagai masukan impedansi tinggi. Port 0 juga dapat diatur sebagai bus alamat/data saat mengakses program dan data dari memori luar. Pada mode ini port 0 memiliki *pull-up internal*. Port 0 juga menerima byte-byte kode saat pemprograman Flash dan mengeluarkan byte kode saat verifikasi. *Pull-up eksternal* diperlukan saat memverifikasi program.

2.2.1.2 Port 1

Port 1 adalah port dua arah masukan/keluaran 8-bit dengan *pull-up internal*. Sebagai tambahan, P1.0 dan P1.1 dapat diatur sebagai pewaktu/ pencacah-2 eksternal masukan pencacah (P1.0/T2) dan pewaktu/pencacah-2 masukan pemicu

(P1.1/T2EX). Port 1 juga menerima byte-byte alamat saat pemrograman dan verifikasi flash.

2.2.1.3 Port 2

Port 2 adalah port masukan/keluaran dua arah 8-bit dengan internal *pull-up*. Port 2 juga menerima bit-bit alamat dan beberapa sinyal kendali saat pemrograman dan verifikasi flash.

2.2.1.4 Port 3

Port 3 adalah port masukan/keluaran dua arah 8-bit dengan internal *pull-up*. Port 3 juga menyediakan fasilitas berbagai fungsi khusus dari AT89S52. Port 3 juga menerima beberapa sinyal kendali saat pemrograman dan verifikasi flash.

2.2.1.5 RST (Reset)

Masukan tinggi pada kaki ini selama dua siklus instruksi mesin akan me-reset perangkat.

2.2.1.6 ALE/ PROG (*Address Latch Enable*)

ALE adalah pulsa keluaran untuk mengunci bit rendah dari alamat saat mengakses memori eksternal. Kaki ini juga digunakan sebagai masukan pulsa (PROG) saat pemprograman Flash. Pada operasi biasa, ALE mengeluarkan rata-rata 1/6 kali frekuensi osilator dan mungkin digunakan sebagai pewaktu atau denyut.

2.2.1.7 PSEN (*Program Store Enable*)

PSEN adalah *strobe* pembacaan program pada memori eksternal. Ketika AT89S52 melakukan eksekusi program dari memori eksternal, PSEN diaktifkan

dua kali setiap siklus instruksi mesin, kecuali bahwa dua aktifasi PSEN diabaikan setiap mengakses data memori eksternal.

2.2.1.8 EA / Vpp (*External Access Enable*)

EA harus dihubungkan ke GND supaya memfungsikan perangkat untuk mengambil kode program dari lokasi memori eksternal dimulai dari 0000H hingga FFFFH.

Mikrokontroler *S-series* (AT89S5x) maupun *C-series* (AT89C5x) memiliki konfigurasi pin yang tidak jauh berbeda. Yang membedakan adalah cara pengisian program (*flash programming*). Pada mikrokontroler AT89S5x terdapat fasilitas ISP (*In System Programming*), artinya mikrokontroler ini mampu diprogram meskipun dalam kondiasi bekerja. Letak perbedaan pada hardware adalah adanya MOSI, MOSI, dan SCK. Pin ini berguna saat *flash programming*. Adapun fungsi dari pin-pin yang lain, fungsinya sama seperti pada seri sebelumnya. Konfigurasi secara keseluruhan dapat dilihat pada Tabel 2.3 di bawah ini.

Tabel 2.3 *Nama dan Fungsi Pin Mikrokontroler*

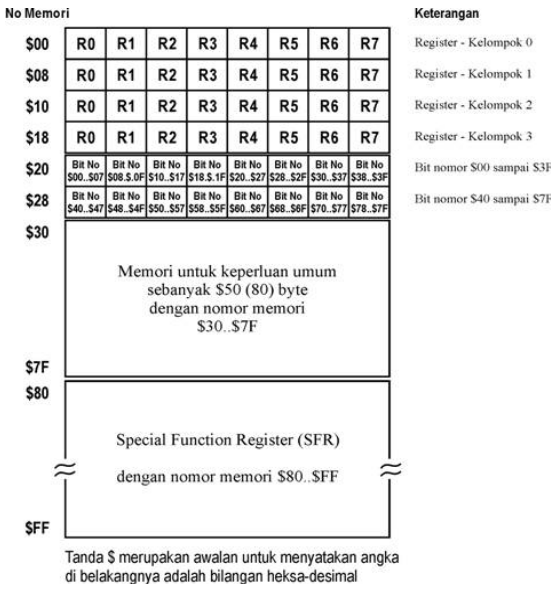
Pin	Nama	Alternatif	Keterangan
1…8	Port 1 (P1.0…P1.7)		Sebagai port I/0 biasa, mempunyai internal *pull up* dan berfungsi sebagai input dengan memberikan logika 1. Terdapat pin MISO, MOSI, SCK
9	RST		Reset aktif dengan logika 1 minimal 2 siklus
10…17	Port 3 (P3.0…P3.7)		Sebagai I/O biasa, namun juga mempunyai fungsi khusus
10	P3.0	RXD	Port serial input
11	P3.1	TXD	Port serial output
12	P3.2	INT0	External Interupt 0
13	P3.3	INT1	External Interupt 1
14	P3.4	T0	External timer 0 input
15	P3.5	T1	External timer 1 input

16	P3.6	WR	External data memory *write strobe*
17	P3.7	RD	External data memory *read strobe*
18	XTAL2		*Output Oscillator*
19	XTAL1		*Input Oscillator*
20	GND		Sebagai Kaki Suplay GND
21…28	Port 2 (P2.0…P2.7)		Port 0 sebagai I/0 biasa, atau sebagai *high order address*, pada saat mengakakses memori eksternal.
29	PSEN		Pin ini berfungsi pada saat mengeksekusi program yang terletak pada memori eksternal. PSEN akan aktif dua kali setiap siklus.
30	ALE	PROG	Pin ini dapat berfungsi sebagai *Address Latch Enable* (ALE) yang me-*latch low byte address* pada saat mengakses memori eksternal. Sedangkan pada saat *Flash Programming* (PROG) berfungsi sebagai pulsa masukan. Pada operasi normal ALE akan mengeluarkan sinyal clock sebesar 1/16 frekwensi oscillator kecuali pada saat mengakses memori eksternal. Sinyal clock pada pin ini dapat pula didisable dengan men-set bit 0 dari *Special Function Register* di alamat 8EH. ALE hanya akan aktif pada saat mengakses memori eksternal (MOVX & MOVC)
31	EA	VPP	Pada kondisi *low* maka pin ini akan berfungsi sebagai EA yaitu mikrokontroler akan menjalankan program yang ada pada memori eksternal setelah sistem direset. Apabila berkondisi *high* maka pin ini akan berfungsi untuk menjalankan program yang ada pada memori internal.
32…39	Port 0 (P0.0…P0.7)		Port 0 dapat berfungsi sebagai I/0 biasa, juga bisa sebagai alamat rendah dan bus data untuk memori eksternal
40	VCC		Sebagai Kaki Suplay VCC

2.2.2 Data Memori

AT89S51/52 mempunyai memori yang disebut sebagai memori data internal. Memori data internal terdiri dari RAM internal sebesar 128 byte dengan alamat 00H-7FH yang dapat diakses menggunakan RAM *address register*. RAM internal ini terdiri dari *register banks* dengan 8 buah register (R0-R7). Memori lain yaitu 21 buah *special function register* dimulai dari alamat 80H-FFH. Jika diperlukan, memori data eksternal untuk menyimpan variabel yang ditentukan oleh user dapat ditambah berupa IC RAM atau ROM maksimal sebesar 64KB.

AT89S52 menggunakan 256 bytes RAM dimana 128 bytes bagian atas menempati alamat parallel ke *special function register* (SFR). Artinya 128 bagian atas mempunyai alamat yang sama dengan SFR namun secara fisik terpisah dari SFR. Ketika instruksi mengakses lokasi internal diatas 7FH, mode alamat yang digunakan pada instruksi menentukan apakah CPU mengakses 128 btyes atas atau SFR. Instruksi yang menggunakan pengalamatan langsung akan mengakses ruang SFR.



Gambar 2.5 *Denah Memori Data*

Seperti terlihat dalam denah memori-data Gambar 2.5, memori-data dibagi menjadi dua bagian, memori nomor $00 sampai $7F merupakan memori seperti RAM selayaknya meskipun beberapa bagian mempunyai kegunaan khusus, sedangkan memori nomor $80 sampai $FF dipakai sangat khusus yang dinamakan sebagai *Special Function Register* (SFR). Memori-data nomor $00 sampai $7F bisa dipakai sebagai memori penyimpan data biasa, dibagi menjadi 3 bagian:

a. memori nomor $00 sampai $18 selain sebagai memori-data biasa, bisa pula dipakai sebagai Register Serba Guna (*General Purpose Register*)
b. memori nomor $20 sampai $2F selain sebagai memori-data biasa, bisa dipakai untuk menyimpan informasi dalam level bit
c. memori nomor $30 sampai $7F (sebanyak 80 byte) merupakan memori-data biasa, bisa dipakai untuk menyimpan data maupun dipakai sebagai *stack.*

Sebagai perbandingan kapasitas memori**,** Tabel 2.4 menampilkan kapasitas memori dari mikrokontroler seri AT89X.

Tabel 2.4 *Kapasitas Memory Mikrokontroller Seri AT89X*

Tipe	RAM	Flash Memori	EEPROM
AT89C51/ AT89S51	8 X 128 byte	4 Kbyte	Tidak
AT89C52/ AT89S52	8 X 256 byte	8 Kbyte	Tidak
AT89C55	8 X 256 byte	20 Kbyte	Tidak
AT89S53	8 X 256 byte	12 Kbyte	Tidak
AT89S8252	8 X 256 byte	8 Kbyte	2 Kbyte

2.2.3 *Special Function Register* (SFR)

Mikrokontroler mempunyai sebuah peta memori yang disebut sebagai *Special Function Register* (SFR) . Register khusus SFR adalah satu daerah RAM dalam IC

keluarga MCS51 yang dipakai untuk mengatur perilaku MCS51 dalam hal-hal khusus, misalnya tempat untuk berhubungan dengan port paralel P1 atau P3, dan sarana input/output lainnya, tapi tidak umum dipakai untuk menyimpan data seperti layaknya memori data. Register khusus SFR dalam port 0 berada di alamat 80h, port 1 di alamat 90h, port 2 di alamat A0h dan P3 di alamat B0h.

2.2.4 Register Serba Guna

Register Serba Guna (*General Purpose Register*) menempati memori-data nomor $00 sampai $18, memori sebanyak 32 byte ini dikelompokkan menjadi empat kelompok register (*Register Bank*), 8 byte memori dari masing-masing kelompok itu dikenali sebagai register 0, register 1 sampai register 7. Dalam penulisan program, memori-memori ini bisa langsung disebut sebagai R0, R1, R2, R3, R4, R5, R6 dan R7, tidak lagi dengan nomor memori. Dengan cara ini instruksi yang terbentuk bisa lebih sederhana dan bekerja lebih cepat. Pengertian ini bisa diperjelas dengan contoh 2 instruksi berikut :

MOV A, #04H

MOV A, R4

Instruksi pertama mempunyai makna isi memori data nomor 4 di-*copy*-kan ke akumulator A, sedangkan instruksi kedua artinya isi R4 di-*copy*-kan ke akumulator A. Khusus untuk register 0 dan register 1 (R0 dan R1) masih punya mempunyai kemampuan lain, kedua register ini bisa dipakai sebagai register penampung alamat yang dipakai dalam penyebutan memori secara tidak langsung (*indirect memori addressing).*

Empat kelompok register serba guna itu tidak dapat dipakai secara bersamaan, saat setelah reset yang aktif dipakai adalah kelompok register 0 (*Register Bank* 0). Kalau yang diaktifkan adalah kelompok register 1, maka yang dianggap sebagai R0 bukan lagi memori data nomor 0 melainkan memori data nomor 8, demikian pula kalau yang diaktifkan kelompok register 3 maka memori data nomor 18h yang menjadi R0. Kelompok register yang aktif dipilih dengan cara mengatur bit RS0 dan RS1 yang ada di dalam register PSW (*Program Status Word*).

2.2.5 Register Dasar MCS51

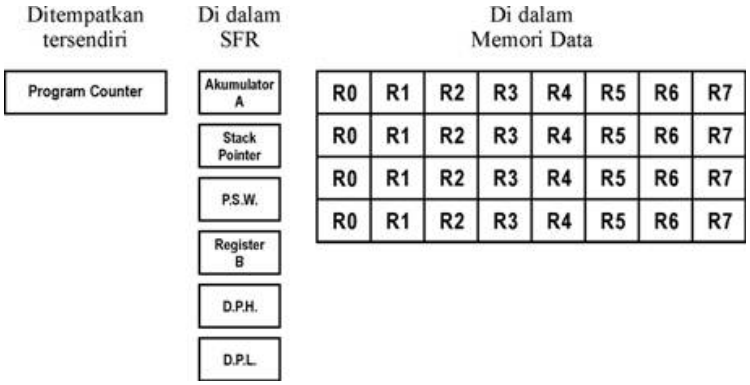
Untuk keperluan penulisan program, setiap mikroprosesor/mikrokontroler selalu dilengkapi dengan register dasar. Ada beberapa macam register yaitu register baku yang bisa dijumpai disemua jenis mikroprosesor/ mikrokontroler dan register yang spesifik pada masing-masing prosesor.

Yang termasuk register baku antara lain program *counter*, akumulator, *stack pointer* register, dan program status register. MCS51 mempunyai semua register baku ini. Kemudian sebagai register yang khas dari MCS51 antara lain adalah register B, Data Pointer High Byte dan Data Pointer Low Byte. Di samping itu, MCS51 masih mempunyai Register Serba Guna R0..R7 yang sudah disebut dibagian atas.

Dalam mikroprosesor/mikrokontroler yang lain, register-register dasar biasanya ditempatkan ditempat tersendiri dalam inti prosesor, tapi dalam MCS51 register-register itu ditempatkan secara terpisah.

- Program *counter* ditempatkan ditempat tersendiri di dalam inti prosesor

- Register serba guna R0..R7 ditempatkan di salah satu bagian dari memori-data
- Register lainnya ditempatkan dalam *Special Function Register* (SFR).



Gambar 2.6 *Susunan Register Dasar MCS51*

2.2.5.1 *Program Counter*

Program Counter (PC) dalam AT89S5x merupakan register dengan kapasitas 16 bit. Di dalam PC dicatat nomor memori-program yang menyimpan instruksi berikutnya yang akan diambil (*fetch*) sebagai instruksi untuk dikerjakan (*execute*). Saat setelah reset PC bernilai 0000h, berarti MCS51 akan segera mengambil isi memori-program nomor 0 sebagai instruksi. Nilai PC otomatis bertambah 1 setelah prosesor mengambil instruksi 1 byte. Ada instruksi yang hanya 1 byte, ada instruksi yang sampai 4 byte, dengan demikian pertambahan nilai PC setelah menjalankan instruksi, tergantung pada jumlah byte instruksi bersangkutan.

2.2.5.2 Akumulator

Sesuai dengan namanya, akumulator adalah sebuah register yang berfungsi untuk menampung (*accumulate*) hasil-hasil pengolahan data dari banyak instruksi MCS51. Akumulator bisa menampung data 8 bit (1 byte) dan merupakan register

yang paling banyak kegunaannya, lebih dari setengah instruksi-instruksi MCS51 melibatkan Akumulator.

2.2.5.3 *Stack Pointer Register*

Salah satu bagian dari memori data dipakai sebagai *stack*, yaitu tempat yang dipakai untuk menyimpan sementara nilai PC sebelum prosesor menjalankan *sub-rutin*, nilai tersebut akan diambil kembali dari *stack* dan dikembalikan ke PC saat prosesor selesai menjalankan *sub-rutin*. *Stack Pointer Register* adalah register yang berfungsi untuk mengatur kerja *stack*, dalam *Stack Pointer Register* disimpan nomor memori data yang dipakai untuk operasi *stack* berikutnya.

2.2.5.4 *Program Status Word*

Program Status Word (PSW) berfungsi mencatat kondisi prosesor setelah melaksanakan instruksi.

2.2.5.5 Register B

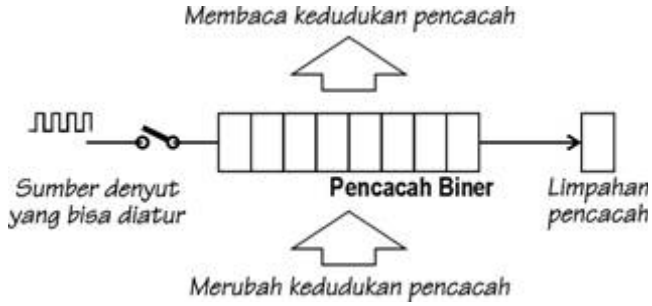
Register B merupakan register dengan kapasitas 8 bit, merupakan register pembantu akumulator saat menjalankan instruk perkalian dan pembagian.

2.2.5.6 DPH dan DPL

Data Pointer High Byte (DPH) dan *Data Pointer Low Byte* (DPL) masing-masing merupakan register dengan kapasitas 8 bit, tapi dalam pemakaiannya kedua register ini digabungkan menjadi satu register 16 bit yang dinamakan sebagai *Data Pointer Register* (DPTR). Sesuai dengan namanya, register ini dipakai untuk mengalamati data dalam jangkauan yang luas.

2.2.6 Timer dan Counter

Pada dasarnya sarana input yang satu ini merupakan seperangkat pencacah biner (*binary counter*) yang terhubung langsung ke saluran data mikrokontroler, sehingga mikrokontroler bisa membaca kedudukan pancacah. Seperti layaknya pencacah biner, apabila sinyal denyut (*clock*) yang diumpankan sudah melebihi kapasitas pencacah, maka pada bagian akhir untaian pencacah akan timbul sinyal limpahan. Sinyal ini merupakan suatu hal yang penting sekali dalam pemakaian pencacah. Terjadinya limpahan pencacah ini dicatat dalam sebuah flip-flop tersendiri. Di samping itu, sinyal denyut yang diumpankan ke pencacah harus dapat dikendalikan dengan mudah.



Gambar 2.7 *Konsep Dasar Timer/Counter Sebagai Sarana Input*

Sinyal denyut yang diumpankan ke pencacah bisa dibedakan menjadi 2 macam, yang pertama adalah sinyal denyut dengan frekuensi tetap yang sudah diketahui besarnya dan yang kedua adalah sinyal denyut dengan frekuensi tidak tetap. Jika sebuah pencacah bekerja dengan frekuensi tetap yang sudah diketahui besarnya, maka pencacah tersebut bekerja sebagai timer, karena kedudukan pencacah tersebut setara dengan waktu yang bisa ditentukan dengan pasti. Sedangkan jika sebuah pencacah bekerja dengan frekuensi yang tidak tetap, maka pencacah

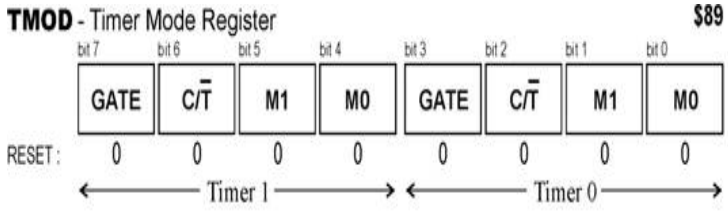
tersebut bekerja sebagai *counter*, karena kedudukan pencacah tersebut hanyalah menyatakan banyaknya pulsa yang sudah diterima pencacah.

2.2.6.1 Sarana *Timer* dalam MCS51

Keluarga mikrokontroler MCS51, misalnya AT89S51 dan AT89x051, dilengkapi dengan dua perangkat *timer*, masing-masing dinamakan sebagai *timer* 0 dan *timer* 1. Sedangkan untuk jenis yang lebih besar, misalnya AT89S52, mempunyai tambahan satu perangkat *timer/counter* lagi yang dinamakan sebagai *timer* 2. Perangkat *timer/counter* tersebut merupakan perangkat keras yang menjadi satu dalam chip mikrokontroler MCS51. Bagi pemakai mikrokontroler MCS51 perangkat tersebut dikenal sebagai SFR (*Special Function Register*) yang berkedudukan sebagai memori data internal.

2.2.6.2 Register Pengatur *Timer*

Register TMOD dan register TCON merupakan register pembantu untuk mengatur kerja *timer* 0 dan *timer* 1. Kedua register ini dipakai bersama oleh *timer* 0 dan *timer* 1.



Gambar 2.8 *Denah Susunan Bit Pada Register TMOD*

Register TMOD dibagi menjadi dua bagian secara simetris, bit 0 sampai 3 register TMOD (TMOD bit 0 .. TMOD bit 3) dipakai untuk mengatur *timer* 0, bit 4

sampai 7 register TMODE (TMOD bit 4 .. TMOD bit 7) dipakai untuk mengatur *timer* 1, pemakaiannya sebagai berikut :

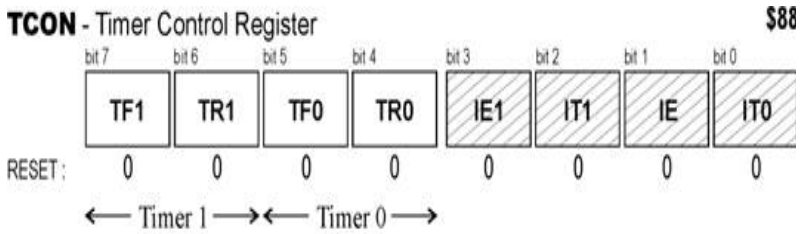
- bit M0/M1 dipakai untuk menentukan mode *timer* seperti yang terlihat dalam Tabel 2.5 di bawah ini.

Tabel 2.5 *Penentuan Mode Timer*

M1	**M0**	**Mode**
0	0	0
0	1	1
1	0	2
1	1	3

Jika bit pada M1 = M0 = 0 maka, mode timer yang digunakan adalah mode 0. Sedangkan jika M1 dan M0 = 1, maka yang digunakan adalah mode 3. Begitu selanjutnya seperti yang terlihat pada Tabel 2.3 di atas

- bit C/T* dipakai untuk mengatur sumber sinyal denyut yang diumpankan ke pencacah biner. Jika C/T*=0 sinyal denyut diperoleh dari osilator kristal yang frekuensinya sudah dibagi 12, sedangkan jika C/T*=1 maka sinyal denyut diperoleh dari kaki T0 (untuk *Timer* 0) atau kaki T1 (untuk *Timer* 1)
- bit GATE merupakan bit pengatur saluran sinyal denyut. Bila bit GATE=0 saluran sinyal denyut hanya diatur oleh bit TRx (maksudnya adalah TR0 atau TR1 pada register TCON). Bila bit GATE=1 kaki INT0 untuk *Timer* 0 atau kaki INT1 untuk *Timer* 1 dipakai juga untuk mengatur saluran sinyal denyut, seperti terlihat pada gambar berikut.



Gambar 2.9 *Denah Susunan Bit Pada Register TCON*

Register TCON dibagi menjadi dua bagian, 4 bit pertama (bit 0 .. bit 3, bagian yang diarsir dalam Gambar 2.9) dipakai untuk keperluan mengatur kaki INT0 dan INT1. Sisa 4 bit dari register TCON (bit 4...bit 7) dibagi menjadi dua bagian secara simetris yang dipakai untuk mengatur *timer* 0 atau *timer* 1, yaitu sebagai berikut :

- bit TFx (maksudnya adalah TF0 atau TF1) merupakan bit penampung limpahan. TFx akan menjadi ‘1’ setiap kali pencacah biner yang terhubung padanya melimpah (kedudukan pencacah berubah dari $FFFF kembali menjadi $0000). Bit TFx di-nol-kan dengan istruksi CLR TF0 atau CLR TF1. Jika sarana interupsi dari *Timer* 0/*Timer* 1 dipakai, TRx di-nol-kan saat MCS51 menjalankan rutin layanan interupsi ISR (*Interupt Service Routine*)
- bit TRx (maksudnya adalah TR0 atau TR1) merupakan bit pengatur saluran sinyal denyut, bila bit ini sama dengan ‘0’, sinyal denyut tidak disalurkan ke pencacah biner sehingga pencacah berhenti mencacah. Bila bit GATE pada register TMOD = 1, maka saluran sinyal denyut ini diatur bersama oleh TRx dan sinyal pada kaki INT0/INT1

2.3 Bahasa Pemrograman C

Bahasa C merupakan salah satu bahasa pemrograman yang sangat populer di mata para pakar dunia komputer, terutama di kalangan pendidikan, karena bahasa C dianggap sebagai bahasa yang memiliki banyak keunnggulan dibanding bahasa yang lain. Dilihat dari sisi sintaksnya Bahasa C sangat mudah dipelajari baik bagi kalangan pelajar/mahasiswa maupun kalangan masyarakat umum, dan dari segi kecepatan Bahasa C sangat tangguh karena bisa berjalan secepat bahasa tingkat rendah (rakitan). Karena keunggulannya tersebut, bahasa C dipakai sebagai mata kuliah wajib hampir di seluruh perguruan tinggi di dunia.

Bahasa pemrograman tingkat tinggi lebih berorientasi kepada manusia yaitu bagaimana agar pernyataan-pernyataan yang ada dalam program mudah ditulis dan dimengerti oleh manusia. Sedangkan bahasa tingkat rendah lebih berorientasi ke mesin, yaitu bagaimana agar mesin dapat langsung mengintepretasikan pernyataan-pernyataan program.

2.3.1 Struktur Dasar Bahasa C

2.3.1.1 Tipe Data

Tipe data merupakan bagian program yang paling penting karena tipe data mempengaruhi setiap instruksi yang akan dilaksanakan oleh computer. Misalnya saja 5 dibagi 2 bisa saja menghasilkan hasil yang berbeda tergantung tipe datanya. Jika 5 dan 2 bertipe integer maka akan menghasilkan nilai 2, namun jika keduanya bertipe float maka akan menghasilkan nilai 2.5000000. Pemilihan tipe data yang tepat akan membuat proses operasi data menjadi lebih efisien dan efektif.

Dalam bahasa C terdapat lima tipe data dasar seperti yang terlihat pada Tabel 2.6 berikut ini.

Tabel 2.6 *Tipe Data Dasar Pada Bahasa C*

No.	Tipe data	Ukuran	Range (jangkauan)	Format	Keterangan
1.	Char	1 byte	-128 s/d 127	%c	Karakter
2.	Int	2 byte	- 32768 s/d 32767	%i, %d	Bilangan bulat
3.	Float	4 byte	- 3,4E-38 s/d 3,4E+38	%f	Bilangan pecahan
4.	Double	8 byte	- 1,7E-308 s/d 1,7E+308	%fl	Pecahan presisi berganda
5.	Void	0 byte			Tidak bertipe

2.3.1.2 Konstanta

Konstanta merupakan suatu nilai yang tidak dapat diubah selama proses program berlangsung. Konstanta nilainya selalu tetap. Konstanta harus didefinisikan terlebih dahulu di awal program. Konstanta dapat bernilai integer, pecahan, karakter dan string.

2.3.1.3 Variabel

Variabel adalah suatu pengenal (identifier) yang digunakan untuk mewakili suatu nilai tertentu di dalam proses program. Berbeda dengan konstanta yang nilainya selalu tetap, nilai dari suatu variable bisa diubah-ubah sesuai kebutuhan. Nama dari suatu variable dapat ditentukan sendiri oleh pemrogram dengan aturan sebagai berikut.

- Terdiri dari gabungan huruf dan angka dengan karakter pertama harus berupa huruf. Bahasa C bersifat case-sensitive artinya huruf besar dan kecil dianggap berbeda. Jadi antara nim, NIM dan Nim dianggap berbeda.
- Tidak boleh mengandung spasi.

- Tidak boleh mengandung symbol-simbol khusus, kecuali garis bawah (underscore). Yang termasuk symbol khusus yang tidak diperbolehkan antara lain : $, ?, %, #, !, &, *, (,), -, +, =, dsb
- Panjangnya bebas, tetapi hanya 32 karakter pertama yang terpakai.

2.3.1.4 Deklarasi

Deklarasi diperlukan bila kita akan menggunakan pengenal (identifier) dalam program. Identifier dapat berupa variable, konstanta dan fungsi.

a. Deklarasi Variabel

Bentuk umum pendeklarasian suatu variable adalah :

Nama_tipe nama_variabel;

Contoh :

```
int x;                  // Deklarasi x bertipe integer
char y, huruf, nim[10]; // Deklarasi variable bertipe char
float nilai;            // Deklarasi variable bertipe float
double beta;            // Deklarasi variable bertipe double
int array[5][4];        // Deklarasi array bertipe integer
char *p;                // Deklarasi pointer p bertipe char
```

b. Deklarasi Konstanta

Dalam bahasa C konstanta dideklarasikan menggunakan preprocessor #define. Contohnya :

```
#define PHI 3.14
#define nim “0111500382”
#define nama “Sri Widhiyanti”
```

c. Deklarasi Fungsi

Fungsi merupakan bagian yang terpisah dari program dan dapat diaktifkan atau dipanggil di manapun di dalam program. Fungsi dalam bahasa C ada yang sudah disediakan sebagai fungsi pustaka seperti printf(), scanf(), getch() dan untuk menggunakannya tidak perlu dideklarasikan. Fungsi yang perlu dideklarasikan terlebih dahulu adalah fungsi yang dibuat oleh programmer. Bentuk umum deklarasi sebuah fungsi adalah :

Tipe_fungsi nama_fungsi (parameter_fungsi);

Contohnya :

float luas_lingkaran(int jari);

void tampil();

int tambah(int x, int y);

2.3.1.5 Komentar Program

Komentar program hanya diperlukan untuk memudahkan pembacaan dan pemahaman suatu program (untuk keperluan dokumentasi program). Dengan kata lain, komentar program hanya merupakan keterangan atau penjelasan program. Untuk memberikan komentar atau penjelasan dalam bahasa C digunakan pembatas /* dan */ atau menggunakan tanda // untuk komentar yang hanya terdiri dari satu baris. Komentar program tidak akan ikut diproses dalam program (akan diabaikan).

Contoh penulisan komentar :

```
#include “stdio.h”
#include “conio.h”
void main()
{
```

```
clrscr();              /* Ini untuk membersihkan layar tampilan */
printf("Contoh");      //komentar tidak ikut diproses
getch();               /* Dan ini untuk menahan tampilan di layer */
}
```

2.3.1.6 Operator

a. Operator Penugasan

Operator Penugasan (Assignment operator) dalam bahasa C berupa tanda sama dengan ("="). Contoh :

nilai = 80;

A = x * y;

Artinya : variable "nilai" diisi dengan 80 dan variable "A" diisi dengan hasil perkalian antara x dan y.

b. Operator Aritmatika

Bahasa C menyediakan lima operator aritmatika, yaitu :

- Bintang (*) : untuk perkalian
- Garis Miring (/) : untuk pembagian
- Persen (%) : untuk sisa pembagian (modulus)
- Tambah (+) : untuk pertambahan
- Kurang (-) : untuk pengurangan

Catatan : operator % digunakan untuk mencari sisa pembagian antara dua bilangan.

Misalnya :

9 % 2 = 1 // sembilan dibagi dengan dua memiliki sisa satu

9 % 3 = 0 // sembilan dibagi dengan tiga tidak memiliki sisa

c. Operator Hubungan (Perbandingan)

Operator Hubungan digunakan untuk membandingkan hubungan antara dua buah operand, sebuah nilai atau variable. Operator hubungan dalam bahasa C dapat dilihat pada Tabel 2.7 berikut ini.

Tabel 2.7 *Operator Hubungan Yang Terdapat Pada Bahasa C*

Operator	Arti
<	Kurang dari
<=	Kurang dari sama dengan
>	Lebih dari
>=	Lebih dari sama dengan
==	Sama dengan
!=	Tidak sama dengan

d. Operator Logika

Jika operator hubungan membandingkan hubungan antara dua buah operand, maka operator logika digunakan untuk membandingkan logika hasil dari operator operator hubungan. Operator logika ada tiga macam, yaitu :

- && : Logika AND (dan)
- || : Logika OR (atau)
- ! : Logika NOT (ingkaran)

e. Operator Bitwise

Operator bitwise digunakan untuk memanipulasi bit-bit dari nilai data yang ada di memori. Operator bitwise dalam bahasa C :

- << : Pergeseran bit ke kiri
- >> : Pergeseran bit ke kanan
- & : Bitwise AND
- ^ : Bitwise XOR (exclusive OR)

- | : Bitwise OR
- ~ : Bitwise NOT

2.3.2 Kelebihan dan Kelemahan Pemrograman Dengan Bahasa C

2.3.2.1 Kelebihan Pemrograman Dengan Bahasa C

Dibandingkan dengan bahasa pemrograman lain, pemrograman dengan bahasa C memiliki beberapa kelebihan. Kelebihan tersebut diantaranya adalah.

- Bahasa C tersedia hampir di semua jenis computer.
- Kode bahasa C sifatnya adalah portable dan fleksibel untuk semua jenis computer.
- Bahasa C hanya menyediakan sedikit kata-kata kunci, hanya terdapat 32 kata kunci.
- Proses executable program bahasa C lebih cepat.
- Bahasa C termasuk bahasa tingkat menengah.

2.3.2.2 Kelemahan Pemrograman Dengan Bahasa C

Namun selain memiliki kelebihan, pemrograman dengan bahasa C juga memiliki kekurangan/kelemahan. Kelemahan tersebut diantaranya adalah.

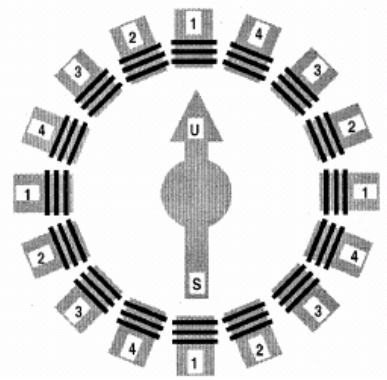
- Banyaknya operator serta fleksibilitas penulisan program kadang-kadang membingungkan pemakai.
- Bagi pemula pada umumnya akan kesulitan menggunakan pointer.

2.4 Motor Stepper

Motor stepper adalah perangkat kendali yang mengkonversi bit-bit inputnya menjadi posisi rotor atau gerakan motor *discret* (berlainan) yang disebut *step*

(langkah). Satu putaran motor memerlukan 360 derajat dengan jumlah langkah yang tertentu perderajatnya. Ukuran kerja dari motor stepper biasanya diberikan dalam jumlah langkah perputaran perdetik. Motor stepper mempunyai kecepatan dan torsi yang rendah namun memiliki kontrol gerakan posisi yang cermat, hal ini dikarenakan motor stepper memiliki beberapa segment kutub kumparan.

Sistem kerja motor stepper adalah motor ini memiliki beberapa pin masukan yang menjadi kutub-kutub magnet didalam motor. Bila salah satu pin lebih tinggi dari pin seberang, pin itu akan mengaktifkan kutub didalam motor sebagai kutub utara, dan kutub seberang sebagai kutub selatan. Dengan adanya dua kutub ini, rotor didalam motor stepper yang memiliki kutub pemanen akan mengarah sesuai masukan. Untuk lebih jelasnya dapat dilihat pada Gambar 2.10 berikut.

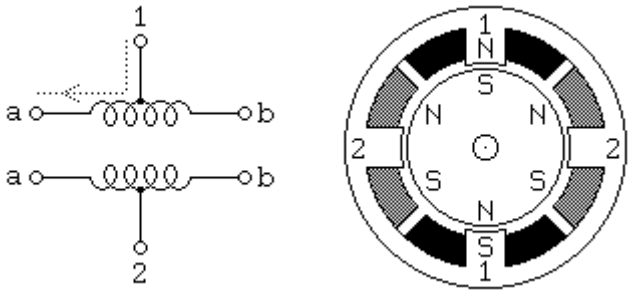


Gambar 2.10 *Penampang Motor Stepper*

Gambar 2.10 diatas menampilkan penampang motor stepper dengan empat koil. Setiap koil memiliki empat kondisi kutub. Bila kondisi satu yang aktif, posisi rotor akan tampak seperti gambar diatas, dan bila kondisi bergeser kedua, rotor akan bergerak kekiri dengan sudut putar sesuai jarak kondisi satu dan dua. Ketelitian sudut putar pada motor stepper sebanding dengan banyaknya koil dan kondisi masukannya.

2.4.1 Motor Stepper Unipolar

Motor stepper dengan tipe unipolar adalah motor stepper yang mempunyai 2 buah lilitan yang masing-masing lilitan ditengah-tengahnya diberikan sebuah tap seperti tampak pada Gambar 2.11 berikut.



Gambar 2.11 *Penampang Motor Stepper Unipolar*

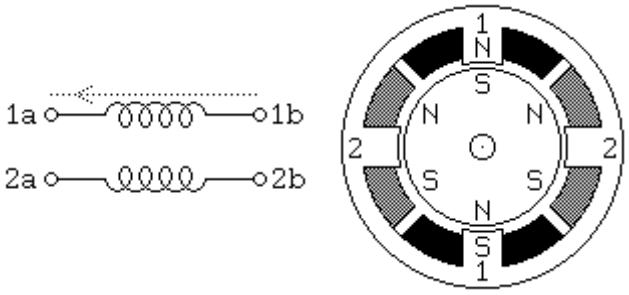
Motor ini mempunyai step tiap 30° dan mempunyai dua buah liliatan yang didistribusikan berseberangan 180° di antara kutub pada stator. Sedangkan pada rotonya menggunakan magnet permanen yang berbentuk silinder dengan mempunyai 6 buah kutub, 3 kutub selatan dan 3 buah kutub utara. Sehingga dengan konstrusi seperti ini maka jika dibutuhkan ke presisian dari motor stepper yang lebih tinggi dibutuhkan pula kutub-kutub pada stator dan rotor yang semakin banyak pula.

Ketelitian dari magnet permanen di rotor dapat sampai 1.8° untuk tiap stepnya. Ketika arus mengalir melalui tap tengah pada lilitan pertama akan menyebabkan kutub pada stator bagian atas menjadi kutub utara sedangkan kutub stator pada bagian bawah menjadi kutub selatan. Kondisi akan menyebabkan rotor mendapat gaya tarik menuju kutub-kutub ini dan ketika arus yang melalui lilitan satu

dihentikan dan lilitan dua diberi arus maka rotor akan mengerak lagi menuju kutub-kutub ini. Sampai di sini rotor sudah berputar sampai 30° atau 1 step.

2.4.2 Motor Stepper Bipolar

Motor dengan tipe bipolar ini mempunyai konstruksi yang hampir sama dengan motor stepper tipe unipolar namun tidak terdapat tap pada lilitannya, seperti tampak pada Gambar 2.12 berikut.

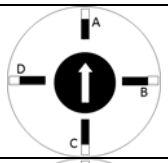
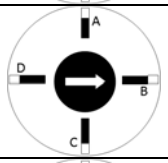
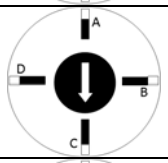
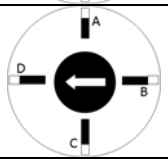


Gambar 2.12 *Penampang Motor Stepper Bipolar*

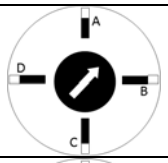
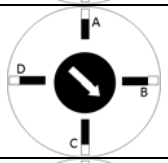
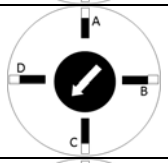
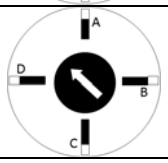
Penggunaan motor dengan tipe bipolar ini membutuhkan rangkaian yang sedikit lebih rumit untuk mengatur agar motor ini dapat berputar dalam dua arah. Biasanya untuk menggerakkan motor stepper jenis ini membutuhkan sebuah driver motor yang sering dikenal sebagai *H Bridge*. Rangkaian ini akan menontrol tiap-tiap lilitan secara independen termasuk dengan polaritasnya untuk tiap-tiap lilitan.

Pengaturan kutub-kutub motor dan proses gerak motor stepper dapat dipahami melalui Tabel 2.8, Tabel 2.9, dan Tabel 2.10 berikut ini.

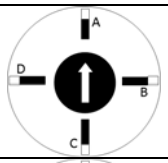
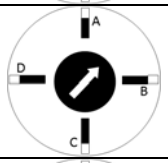
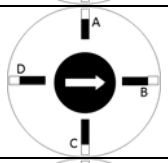
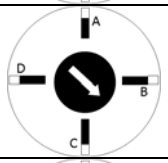
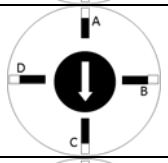
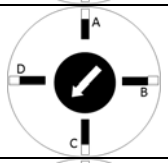
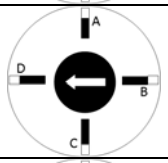
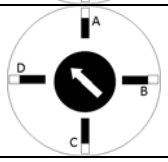
Tabel 2.8 *Mode Full Step Motor Stepper (a)*

Fasa	Kutub				Proses
	A	B	C	D	
1	ON	OFF	OFF	OFF	
2	OFF	ON	OFF	OFF	
3	OFF	OFF	ON	OFF	
4	OFF	OFF	OFF	ON	

Tabel 2.9 *Mode Full Step Motor Stepper (b)*

Fasa	Kutub				Proses
	A	B	C	D	
1	ON	ON	OFF	OFF	
2	OFF	ON	ON	OFF	
3	OFF	OFF	ON	ON	
4	ON	OFF	OFF	ON	

Tabel 2.10 *Mode Half Step Motor Stepper*

Fasa	Kutub				Proses
	A	B	C	D	
1	ON	OFF	OFF	OFF	
2	ON	ON	OFF	OFF	
3	OFF	ON	OFF	OFF	
4	OFF	ON	ON	ON	
5	OFF	OFF	ON	OFF	
6	OFF	OFF	ON	ON	
7	OFF	OFF	OFF	ON	
8	ON	OFF	OFF	ON	

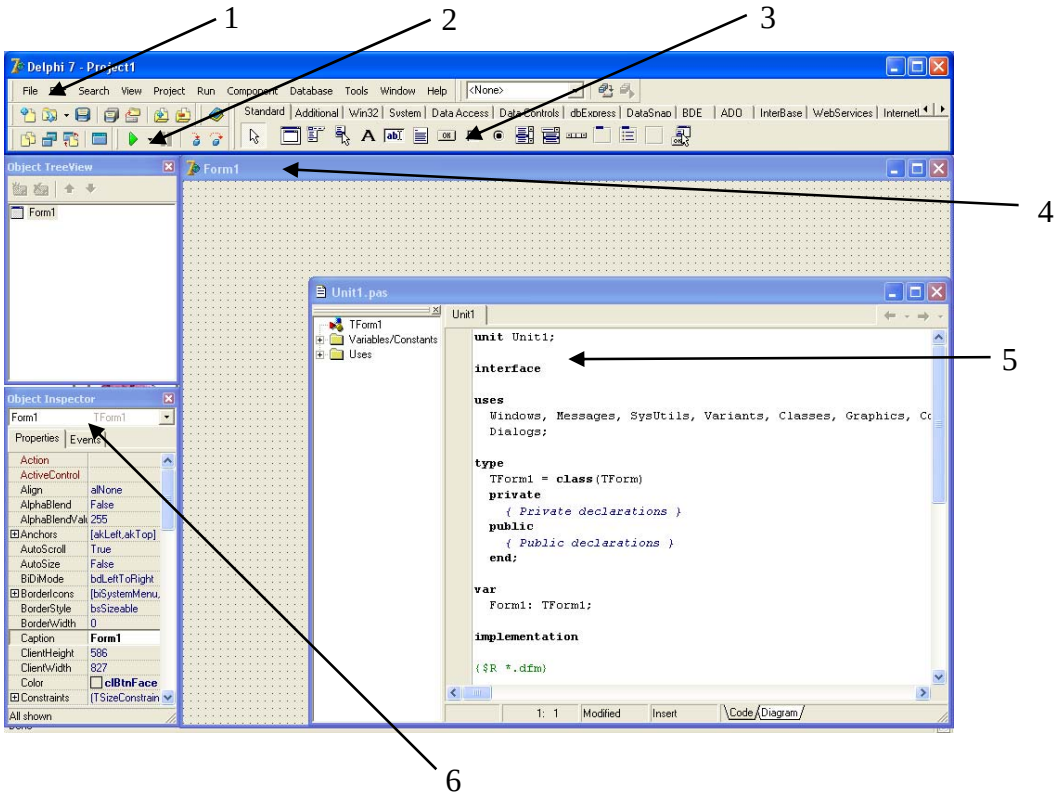
2.5 Borland Delphi 7.0

Delphi adalah sebuah bahasa pemrograman visual di lingkungan windows (*under* windows) yang menggunakan bahasa Pascal sebagai *compiler*. Delphi pertama kali diperkenalkan kepada pengguna komputer pada tahun 1995. Dengan menggunakan Delphi kita dapat mempersingkat waktu pemrograman, karena tidak perlu lagi menuliskan kode program yang rumit dan panjang untuk menggambar, meletakkan dan mengatur komponen. Selain itu kita juga dapat menyusun aplikasi yang lebih interaktif karena Delphi menyediakan cukup banyak pilihan komponen *interface* aplikasi, antara lain berupa tombol menu, drop *down*, menu *pop up*, kotak text, radio button, check box, dan sebagainya.

Delphi 7.0, versi terbaru yang dikeluarkan oleh Borland, memiliki support yang sangat tinggi terhadap *database-database* yang sudah terkenal (seperti MS Accsses, Paradox, Foxpro, Dbase, Oracce, dan lain sebagainya), dan dilengkapi dengan objek-objek yang baru sehingga memudahkan pembuatan *database* maupun program lainnya (Game, Utility dan lainnya).

2.5.1 Tampilan Borland Delphi

Program Delphi dikenal dengan nama IDE (*Integrated Development Environment*), yaitu lingkungan pengembangan aplikasi yang terpadu. Melalui IDE ini dibangun aplikasi-aplikasi dari merancang tampilan untuk pemakai (antarmuka pemakai), menuliskan kode sampai mencari penyebab kesalahan (*debugging*). Tampilan utama dari Delphi dapat dilihat pada gambar 2.13 berikut ini.



Gambar 2.13 *Tampilan Utama Dari Delphi*

Dari gambar diatas dapat dilihat tampilan utama Delphi memiliki beberapa bagian yang dapat digunakan dalam pembuatan aplikasi. Berikut ini adalah penjelasan tentang bagian-bagian tersebut.

a. Menu Bar (pada gambar ditunjukkan oleh nomor 1)

   Berisi tentang perintah-perintah yang dikelompokkan menjadi satu bagian (seperti File, Edit, dsb.). Perintah-perintah seperti *New*, *Open*, *Save*, dan lain sebagainya juga terdapat pada menu bar ini.

b. *Tool* Bar (pada gambar ditunjukkan oleh nomor 2)

   Berisi perintah-perintah eksekusi seperti *compile* dan *run*. Pada tool bar juga terdapat perintah-perintah seperti *New*, *Open*, dan *Save*. Semua perintah-perintah pada tool bar ini juga terdapat pada menu bar. Jadi pada dasarnya tool

bar ini hanya sebuah *shortcut* yang memudahkan kita dalam pembuatan aplikasi.

c. *Component Palette* (pada gambar ditunjukkan oleh nomor 3)

   Component Palette berisi kumpulan objek yang siap untuk digunakan. Kita hanya perlu memilih objek apa yang akan digunakan kemudia meletakkannya pada *Form Designer.*

d. *Form Designer* (pada gambar ditunjukkan oleh nomor 4)

   Form designer merupakan tempat meletakkan objek yang telah dipilih pada *component palette*. Disini kita dapat mendesain tampilan aplikasi yang akan kita buat.

e. *Code Editor* (pada gambar ditunjukkan oleh nomor 5)

   Merupakan tempat untuk menuliskan perintah-perintah program.

f. *Object Inspentor* (pada gambar ditunjukkan oleh nomor 6)

   Berisikan tab properti dan tab even dari objek yang terdapat pada *component palette*. Tab properti berisikan properti dari objek seperti nama objek, ukuran objek, dan lain sebagainya. Sedangkan tab even berisikan perintah yang harus dijalankan oleh objek tersebut.

2.5.2 Kelebihan Dan Kelemahan Delphi

2.5.2.1 Kelebihan Delphi

a. Berbasis objek *orientid programming*, bagian yang ada pada program dipandang sebagai suatu objek yang mempunyai sifat-sifat yang dapat diubah dan diatur, sehingga kita dapat membuat tampilan sebuah program dengan desain kita sendiri tanpa harus membuat *codding* yang panjang.

b. Dapat mengkompilasi menjadi *single executable* (file ber-*extension* *.exe) sehingga program yang dibuat dapat langsung didistribusikan dan dijalankan pada komputer lain tanpa perlu menyertakan file lain.

2.5.2.2 Kelemahan Delphi

a. Borland menetapkan standar bahasa pada produknya sehingga *software-software* yang kompatibilitas harus mengikutinya.
b. Tidak tersedianya *tool* untuk komunikasi serial atau komunikasi paralel sehingga kita harus menambahkannya terlebih dahulu.