BAB II
TEORI DASAR

2.1 *Radio Frequency Identification* (RFID)

RFID merupakan suatu teknologi yang memanfaatkan frekuensi radio sebagai identifikasi otomatis terhadap suatu objek. RFID dapat dipandang sebagai salah satu cara dalam pelabelan suatu objek secara eksplisit dengan menggunakan peralatan-peralatan komputer. Dengan kata lain, RFID adalah teknologi penangkapan data yang dapat digunakan secara elektronik untuk mengidentifikasi, melacak dan menyimpan informasi yang tersimpan dalam *tag* RFID.

Para pengamat RFID menganggap bahwa RFID merupakan suatu kemajuan dari *barcode* optik yang banyak terdapat pada barang-barang dagangan. Keunggulan pembeda dari RFID dengan *barcode* yaitu.

a. **Identifikasi yang unik,** artinya sebuah *barcode* mengindikasikan tipe obyek tempat ia dicetak, misalnya “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 “sebatang coklat merek ABC dengan kadar 70% dan berat 100 gram, dengan nomor seri 897348738”. *Identifier* yang unik dalam RFID dapat berperan sebagai pointer terhadap entri basis data yang menyimpan banyak histori transaksi untuk item-item individu.

b. **Otomasi,** artinya sebuah *barcode* memerlukan kontak *line-of-sight* dengan *reader* yaitu dengan cara *scan* optik, dan tentu saja peletakan fisik harus tepat dari obyek yang di-*scan*. Kecuali pada lingkungan yang benar-benar terkontrol, selain itu *scanning* terhadap *barcode* memerlukan campur tangan manusia. Sebaliknya, *tag* RFID dapat dibaca tanpa kontak *line-of-sight* dan tanpa penempatan yang presisi. *Reader* RFID dapat melakukan *scan* terhadap *tag* RFID sebanyak ratusan perdetik.

Perbedaan mendasar antara RFID dengan *barcode* dapat dilihat seperti pada Tabel 2.1 berikut ini.

Tabel 2.1 *Perbedaan RFID dan Barcode*

RFID	***Barcode***
Dapat ditempel dan tersembunyi, tidak memerlukan pandangan langsung	Harus dengan pandangan langsung
Dapat dibaca meskipun terhalang benda kecuali benda logam	Tidak dapat dibaca jika terhalang
Dapat diprogram/ entri ulang dalam keadaan bergerak	Tidak dapat
Dapat diterapkan dalam lingkungan yang keras, seperti di luar rumah, sekitar bahan kimia dan kelembaban	Harus ditempatkan ditempat yang terlingdungi agar tidak merusak kode
Tag RFID berisikan 1MB memori (1miliar karakter) bahkan sampai fraksi terkecil dari 64 bits	Jumlah informasi terbatas sekitar 20 karakter

Secara garis besar sebuah sistem RFID terdiri atas tiga komponen utama, yaitu *tag*, *reader* dan basis data (seperti terlihat pada 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 terdapat dalam *tag* tersebut.



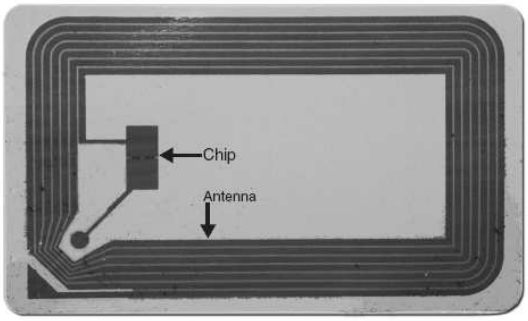
Gambar 2.1 *Komponen Utama Suatu Sistem RFID*

2.1.1 *Tag* RFID

Sistem RFID merupakan suatu sistem identifikasi otomatis yang bertujuan untuk memungkinkan data ditransmisikan oleh peralatan *portable* yang disebut *tag*. Kemudian *tag* tersebut 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 *microchip* dan sebuah antena, (seperti terlihat pada Gambar 2.2). *Microchip* itu sendiri dapat berukuran sekecil butiran pasir, sekitar 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 *mikrochip* mengirimkan informasi ke *reader* RFID. 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* RFID bergerak maupun stasioner.



Gambar 2.2 *Tag RFID atau Transponder*

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 pasif adalah *tag* paling sederhana, 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 oleg *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. Contoh aplikasi *tag* pasif adalah pada pas transit, pas masuk gedung, barang-barang konsumsi. 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*. Sebagian *tag* semipasif tetap diam hingga menerima sinyal dari *reader*. *Tag* semi pasif dapat dihubungkan dengan sensor untuk menyimpan informasi untuk peralatan keamanan kontainer. Rentang baca yang dijangkau *tag* semipasif dapat mencapai 100 kaki.

Tag aktif adalah *tag* yang selain memiliki antena dan chip juga memiliki catu daya sendiri 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.

2.1.2 *Reader* RFID

Untuk berfungsinya sistem RFID, maka diperlukan sebuah *reader* atau alat *scanning* yang dapat membaca *tag* dengan benar dan mengkomunikasikan hasilnya ke suatu *database* yang ada. 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 genggam, atau stasioner seperti peralatan *point-of-sale* di supermarket. *Reader* dibedakan berdasarkan kapasitas penyimpanannya, kemampuan pemrosesannya, serta frekuensi yang dapat dibacanya.



Gambar 2.3 *Beberapa Jenis IC yang Digunakan untuk Reader RFID*

2.1.3 *Databases*

Database 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. Sebagai contoh, data yang disimpan pada sistem pembayaran tol akan berbeda dengan yang disimpan pada rantai *supply*. 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 *database* lokal.

2.1.4 Frekuensi 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. 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. Kombinasi dari level-level frekuensi dan daya yang dibolehkan menentukan rentang fungsional dari suatu aplikasi tertentu seperti keluaran daya dari *reader*.

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. *Tag* dengan 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 1.000 *tag* perdetik. *Tag* ini umumnya digunakan pada pelacakan kontainer, truk, trailer, terminal peti kemas, serta telah diadopsi oleh peritel besar dan Departemen Pertahanan Amerika Serikat. Sebagai tambahan, di Amerika Serikat, band MHz digunakan untuk mengidentifikasi isi kontainer dalam area komersial dan industri untuk meningkatkan ketepatan waktu dan

akurasi transmisi data. Menurut FCC penggunaan semacam itu menguntungkan perusahaan pengapalan komersial dan memberikan manfaat keamanan yang signifikan dengan dimungkinkannya seluruh isi kontnainer teridentifikasi dengan mudah dan cepat serta dengan dapat diidentifikasinya kerusakan selama pengapalan.

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

2.1.5 Kategori Sistem RFID

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

a. Sistem *Electronic Article Surveillance* (EAS) : 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.

b. Sistem *Portable Data Capture* : dicirikan oleh penggunaan *reader* RFID yang portabel yang memungkinkan sistem ini digunakan dalam seting yang bervariasi.

c. 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.

d. Sistem *Positioning* : Digunakan untuk identifikasi lokasi item-item atau kendaraan.

2.2 Mikrokontroler

Pada tahun 1970 mikroposesor atau disebut juga *computer on a chip* intel 4004 pertama kali diproduksi secara masal, pertumbuhannya sangat cepat. Pengembangan mikoprosesor dengan menggunakan teknik pabrikasi yang sama dan konsep pemrograman yang sama melahirkan mikrokontroler. Mikrokontroler tidak dikenal secara luas karena hanya ditujukan untuk komunitas teknik.

Yang dimaksud dengan mikrokontroler adalah sebuah sistem mikroprosesor lengkap yang terkandung didalam sebuah chip. Mikrokontroler berbeda dari mikroprosesor serba guna yang digunakan didalam sebuah PC, karena dalam sebuah mikrokontroler umumnya telah berisi komponen komponen pendukung, seperti prosesor, memori, dan I/O. Sedangkan mikroprosesor adalah CPU komputer. Oleh karena itu, untuk dapat digunakan secara umum,masih diperlukan:

a. *Memory* (RAM, ROM) dan juga *decoder memory*

b. I/O dan *decoder* I/O

c. Piranti khusus (*Interupsi, timer, counter,* dll)

d. Clock Generator

Mikrokontroler didesain sebagai *True computer on a chip*, jadi mikrokontroler sudah mempunyai piranti-piranti tambahan untuk membangun suatu sistem komputer digital. Mikrokontroler dirancang dengan intruksi yang bisa mengakses piranti terprogram dan dioptimalkan untuk instruksi bit dan byte, sementara komputer PC didesain dengan instruksi untuk mengases data-data *multybyte*.

2.2.1 Prinsip Kerja

Prinsip kerja mikrokontroler adalah sebagai berikut.

a. Berdasarkan nilai yang berada pada register *Program Counter*, mikrokontroler mengambil data pada ROM dengan alamat sebagaimana yang tertera pada register *Program Counter*. Selanjutnya isi dari register *Program Counter* ditambah dengan satu (*Increment*) secara otomatis. Data yang diambil pada ROM merupakan urutan instruksi program yang telah dibuat dan diisikan sebelumnya oleh pengguna.
b. Instruksi yang diambil tersebut diolah dan dijalankan oleh mikrokontroler. Proses pengerjaan bergantung pada jenis instruksi, bisa membaca, mengubah nilai-nilai pada register, RAM, isi Port, atau melakukan pembacaan dan dilanjutkan dengan pengubahan data.
c. *Program Counter* telah berubah nilainya baik karena penambahan secara otomatris atau karena pengubahan data. Selanjutnya yang dilakukan mikrokontroler adalah mengulang kembali siklus ini pada langkah pertama. Demikian seterusnya hingga catu daya dimatikan

Dapat disimpulkan bahwa pada dasarnya unjuk kerja mikrokontroler sangatlah tergantung pada urutan instruksi yang dijalankannya, yaitu program yang ditulis dalam ROM. Dan jika dikaitkan dengan *embedded system*, mikrokontroler bertugas untuk membagi kerja dari sistem yang ditambahkan berdasarkan cara kerja sistem tersebut. Sehingga walau telah ditambahkan sistem atau proses yang lain, sistem yang ada sebelumnya tetap bisa melakukan proses sebagaimana mestinya yang diatur dengan mikrokontroler.

Secara teknis hanya ada 2 jenis mikrokontroler yaitu RISC dan CISC dan masing-masing mempunyai keturunan/keluarga sendiri-sendiri.

a. *Reduced Instruction Set Computer* (RISC) : instruksi terbatas tapi memiliki fasilitas yang lebih banyak

b. *Complex Instruction Set Computer* (CISC*)* : instruksi bisa dikatakan lebih lengkap tapi dengan fasilitas secukupnya.

2.2.2 Mikrokontroler AT89S8252

Mikrokontroller tipe Atmel AT89S8252 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), dan 2K *byte* EEPROM. Perangkat ini dibuat menggunakan teknologi memori *non volatile* artinya tidak kehilangan data apabila kehilangan daya listrik. Set instruksi dan kaki keluaran AT89S8252 sesuai dengan standar industri 80C51.

Atmel AT89S8252 adalah mikrokomputer yang sangat bagus dan fleksibel untuk banyak aplikasi sistem kendali.

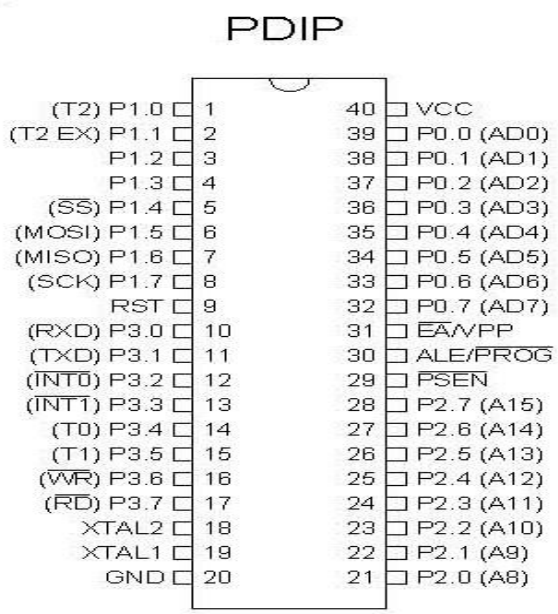
Spesifikasi penting AT89S8252 :

- kompatibel dengan keluarga mikrokontroler MCS51 sebelumnya
- 8K *bytes In system Programmable* (ISP) *flash memori* dengan kemampuan 1000 kali baca/tulis dan antarmuka serial SPI untuk mengunduh program
- 2K *bytes* EEPROM dengan kemampuan 100000 kali baca/tulis
- tegangan kerja 4.0 – 6V
- bekerja dengan rentang 0 – 24MHz
- 256x8 bit RAM internal
- 32 jalur I/0 dapat diprogram
- 3 buah 16 bit *Timer/Counter*
- saluran serial UART
- antarmuka serial SPI
- *watchdog timer*
- *dual data pointer*

2.2.3 Konfigurasi Mikrokontroller AT89S8252

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

mengkafer untuk banyak kebutuhan. Gambar 2.4 adalah gambar konfigurasi kaki-kaki mikrokontroler AT89S8252.



Gambar 2.4 *Konfigurasi Pin Mikrokontroler AT89S8252*

Konfigurasi dan deskripsi kaki-kaki mikrokontroler AT89S8252 sebagai berikut.

2.2.3.1 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.

2.2.3.2 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.

Selanjutnya , P1.4, P1.5, P1.6 dan P1.7 dapat dikonfigurasikan sebagai port SPI *slave* yang dipilih, pin data masukan/keluaran dan clok masukan/keluaran dapat ditujukkan seperti Tabel 2.2 berikut.

Tabel 2.2 *Fungsi Alternatif Port 1*

Port Pin	**Nama Alternatif**	**Fungsi**
P1.0	T2	Masukan pencacah eksternal ke pewaktu/pencacah-2 (*clok-out*)
P1.1	T2EX	Pewaktu/pencacah-2 pemicu
P1.4	SS	Masukan port *slave* yang terpilih
P1.5	MOSI	Pin data keluaran *master*, data masukan *slave* untuk saluran SPI
P1.6	MISO	Pin data masukan *master*, data keluaran *slave* untuk saluran SPI
P1.7	SCK	Pin clok keluaran *master*, clok masukan *slave* untuk saluran SPI

2.2.3.3 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.

2.2.3.4 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 AT89S8252. Port 3 juga menerima beberapa sinyal kendali saat pemrograman dan verifikasi flash. Selain itu port 3 juga memiliki fungsi alternatif seperti pada Tabel 2.3 berikut.

Tabel 2.3 *Fungsi Alternatif Port 3*

Port Pin	**Nama Alternatif**	**Fungsi**
P3.0	RXD	Port masukan serial
P3.1	TXD	Port keluaran serial
P3.2	INT0	Interupt 0 eksternal
P3.3	INT1	Interupt 1 eksternal
P3.4	T0	Masukan timer 0 eksternal
P3.5	T1	Masukan timer 1 eksternal
P3.6	WR	Memoridata *write strobe* eksternal
P3.7	RD	Memori data *read strobe* eksternal

2.2.3.5 *Reset* (RST)

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

2.2.3.6 *Address Latch Enable* (ALE)/ PROG

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.

Catatan, satu pulsa ALE diabaikan saat setiap pengaksesan data memori eksternal. Jika diinginkan, operasi ALE dapat di-*disable* dengan menseting bit 0 dari SFR pada lokasi 8EH. Dengan bit yang diset, ALE aktif hanya saat menjalankan perintah MOVX dan MOVC. Selain itu, kaki ini dapat juga di-pull tinggi. *Setting bit ALE-disable* tidak berpengaruh jika mikrokomputer pada mode eksekusi eksternal.

2.2.3.7 *Program Store Enable* (PSEN)

PSEN adalah *strobe* pembacaan program pada memori eksternal. Ketika AT89S8252 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.

2.2.3.8 *External Access Enable* (EA)/ Vpp

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

Catatan, jika *lock-bit* diprogram, EA akan dikunci secara internal pada saat reset. EA harus dihubungkan dengan Vcc untuk eksekusi program internal. Kaki ini juga menerima tegangan yang memungkinkan pemrograman 12 Volt saat memprogram flash bila pemrograman 12 Volt dipilih.

2.2.4 Memori Data

AT89S8252 memiliki 2K byte EEPROM internal untuk penyimpanan data dan 256 *byte* RAM. Lebih dari 128 *byte* RAM menempati ruang paralel untuk register khusus (SFR). Dengan kata lain lebih dari 128 *byte* memiliki alamat yang sama dengan SFR tetapi secara fisik terpisah dari SFR.

Memori data EEPROM internal dipilih dengan meng-set bit EEMEN di dalam register WMCON pada lokasi alamat SFR 96H. Jarak alamat EEPROM dari 000H sampai 7FFH. Instruksi MOVX digunakan untuk mengakses EEPROM. Untuk mengakses EEPROM eksternal dengan menggunakan instruksi MOVX, maka EEMEN harus di-set “0”.

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

Tabel 2.4 *Kapasitas Memori Mikrokontroller Seri AT89X*

Type	RAM	Flash Memory	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

2.2.5 *Special Function Register* (SFR)

Mikrokontroler mempunyai sebuah peta memori yang disebut sebagai *Special Function Register* (SFR) . Port 0 berada di alamat 80h, port 1 90h, port 2 A0h dan

P3 di alamat B0h. Sedangkan SBUF untuk komunikasi serial berada pada alamat 99h.

Register Khusus SFR (*Special Function Register*) 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 masukan/keluaran lainnya, tapi tidak umum dipakai untuk menyimpan data seperti layaknya memori data.

Meskipun demikian, dalam hal penulisan program SFR diperlakukan persis sama dengan memori data. Untuk mengisi memori data nomor 60H dengan bilangan 0FH, instruksi yang dipergunakan adalah :

MOV 60H, #0FH

Sedangkan untuk memenyimpan 0FH ke Port 1 yang di SFR menempati memori data nomor 90H, instruksi yang dipergunakan adalah :

MOV 90H, #0FH

Membandingkan kedua instruksi di atas bisa dimengerti dalam segi penulisan program SFR diperlakukan persis sama dengan memori data.

Meskipun demikian, dalam menyebut memori data bisa dipakai dua cara, yakni penyebutan nomor memori secara langsung (*direct memory addressing*) dan penyebutan nomor memori secara tidak langsung (*indirect memory addressing*) lewat bantuan R0 dan R1. Tapi untuk SFR hanya bisa dipakai penyebutan nomor memori secara langsung (*direct memory addressing*) saja.

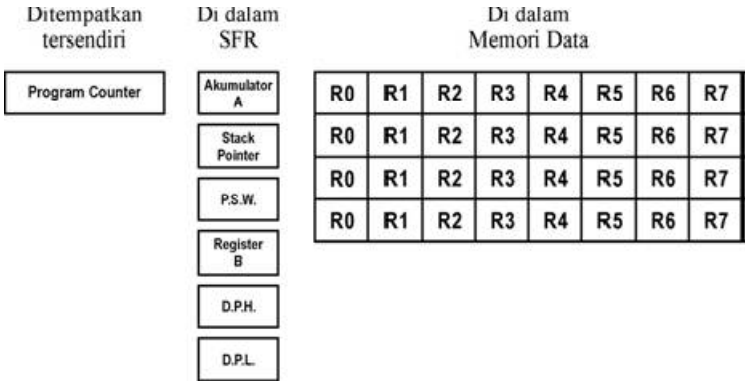
2.2.6 Register Dasar MCS51

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, 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 2.5 *Susunan Register Dasar MCS51*

Kegunaan dan pemakaian register-register dasar tersebut antara lain sebagai berikut.

2.2.6.1 *Program Counter*

Program Counter (PC) dalam MCS51 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.

2.2.6.2 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.

2.2.6.3 *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.

2.2.6.4 *Program Status Word*

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

2.2.6.5 Register B

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

2.2.6.6 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.

2.3 Bahasa Pemrograman C

Sebuah mikrokontroller memerlukan program atau perintah untuk dapat melakukan sesuatu. Pembuatan program tersebut dapat menggunakan beberapa bahasa pemrograman, salah satunya adalah bahasa pemrograman C. Bentuknya sendiri mirip dengan bahasa pemrograman tingkat tinggi lainnya, yaitu : judul program, daftar *header* file, deklarasi, deskripsi.

a. Judul Program

Judul program sifatnya sebagai dokumentasi saja, tidak signifikan terhadap proses program. Ditulis dalam bentuk baris komentar.

Contoh :

```
/* Program Menghitung Rata-Rata */
```

b. Header File

C menyediakan sejumlah file judul (*header file*) yaitu file yang umumnya berisi prototipe fungsi, definisi makro, variabel dan definisi tipe. File ini mempunyai ciri yaitu namanya diakhiri dengan extension .h.

Contoh :

```
#include <stdio.h>
```

Keterangan : menyatakan bahwa agar membaca file bernama stdio.h saat pelaksanaan kompilasi.

c. Deklarasi

Deklarasi adalah bagian untuk mendefinisikan semua nama yang dipakai dalam program. Nama tersebut dapat berupa nama tetapan (konstanta), nama variabel, nama tipe, nama prosedur, nama fungsi.

d. Deskripsi

Bagian inti dari suatu program yang berisi uraian langkah-langkah penyelesaian masalah. Program C pada hakekatnya tersusun atas sejumlah blok fungsi. Sebuah program minimal mengandung sebuah fungsi. Setiap fungsi terdiri dari satu atau beberapa pernyataan, yang secara keseluruhan dimaksudkan untuk melaksanakan tugas khusus. Bagian pernyataan fungsi (disebut tubuh fungsi) diawali dengan tanda “{“ dan diakhiri dengan tanda “}”.

2.3.1 Baris Komentar

Dalam program C ada yang disebut dengan baris komentar yaitu baris-baris yang menjelaskan maksud dari perubah yang digunakan atau maksud dari program itu sendiri. Hal ini dimaksudkan untuk memudahkan pelacakan atas perubah yang digunakan apabila program yang digunakan cukup besar atau memudahkan orang lain memahami program yang kita buat. Dalam program, baris komentar diletakkan diantara tanda "/*" dan "*/" dan baris ini tidak dikerjakan oleh komputer, hanya dianggap sebagai baris kosong.

2.3.2 Variabel

Variabel dalam program digunakan untuk menyimpan suatu nilai tertentu dimana nilai tersebut dapat berubah-ubah. Setiap variabel mempunyai tipe dan hanya data yang bertipe sama dengan tipe variabel yang dapat disimpan di dalam variabel tersebut. Setiap variabel mempunyai nama. Pemisahan antar variabel dilakukan dengan memberikan tanda koma.

Contoh :

```
int jumlah;
float harga_per_unit, total_biaya;
```

Dari contoh di atas, variabel `jumlah` hanya boleh menerima data yang bertipe *integer* (bulat), tidak boleh menerima data bertipe lainnya. Variabel `harga_per_unit` dan `total_biaya` hanya bisa diisi dengan bilangan float (pecahan). Macam-macam variabel antara lain variabel global, variabel lokal, variabel statik.

2.3.2.1 Variabel Global

Variabel yang dideklarasikan di luar blok fungsi dan bersifat dikenali oleh semua bagian program. Data-data yang tersimpan dalam sebuah variabel dapat diakses di setiap blok fungsi. Disarankan untuk tidak digunakan, karena variabel ini dapat men-*sharing*-kan data dan dapat diubah secara tidak sengaja oleh suatu blok fungsi, sehingga nilainya bisa berubah.

2.3.2.2 Variabel Lokal

Variabel yang dideklarasikan dalam suatu blok fungsi tertentu dan hanya dikenal oleh blok fungsi tersebut. Variabel lokal akan dihapus dari memori jika proses sudah meninggalkan blok letak variabel lokalnya.

2.3.2.3 Variabel Statik

Variabel statik sering dipakai sebagai variabel lokal. Perbedaan variabel lokal dan variabel statik adalah variabel lokal akan dihapus dari memori jika proses sudah meninggalkan blok letak variabel lokalnya, sedangkan variabel statik akan dihapus dari memori jika program dimatikan.

Contoh :

```
int i,j;        /* variabel global */
main ()
{
int k,l;        /* variabel lokal */
}
fungsi()
{
static int m,n; /* variabel statik */
}
```

2.3.3 Konstanta

Berbeda dengan variabel yang isinya bisa berubah selama eksekusi program berlangsung, nilai suatu konstanta tidak bisa berubah.

Contoh :

```
const int m = 8;
#define pajak 0.05
```

2.3.4 Fungsi main()

Fungsi main() harus ada pada program, karena fungsi inilah yang menjadi titik awal dan titik akhir eksekusi program. Tanda “{“ di awal fungsi menyatakan awal tubuh fungsi sekaligus awal eksekusi program, sedangkan tanda “}” di akhir fungsi merupakan akhir tubuh fungsi dan sekaligus akhir eksekusi program.

2.3.5 Pemilihan Kondisi (*Selection*)

2.3.5.1 Pilihan Tunggal

Bentuk paling sederhana pilihan tunggal adalah jika hanya ada satu pilihan kondisi yang disediakan.

```
if kondisi
{
true statement
}
```

2.3.5.2 Pilihan Ganda

Digunakan untuk menentukan tindakan yang akan digunakan bila kondisi bernilai benar dan salah.

```
if kondisi
{
true statement
}
else
{
false statement
}
```

2.3.5.3 Pilihan Majemuk

Untuk menentukan tindakan yang akan digunakan disediakan lebih dari 2 alternatif. Merupakan bentuk statement if dengan statement if lain di dalam if sebelumnya.

```
if kondisiA
{
     if kondisiB
     {
          true statementB
     }
     else
     {
          false statementB
     }
}

else
{
false statementA
}
```

2.3.5.4 Struktur Case (*Statement Switch*)

Untuk masalah dengan dua pilihan atau lebih, struktur CASE dapat menyederhanakan penulisan IF yang bertingkat-tingkat.

```
Switch(kondisi)
{
case konstanta1 : {Statement-statement ; break}
case konstanta1 : {Statement-statement ; break}
case konstanta1 : {Statement-statement ; break}
case konstanta1 : {Statement-statement ; break}
.......
}
```

2.3.6 Pengulangan (*Looping*)

2.3.6.1 Struktur For

Struktur ini digunakan bila kita mengetahui secara pasti banyaknya pengulangan yang akan dilakukan. Pernyataan FOR mempunyai 3 parameter yaitu :

a. nilai awal (*initial value*)

b. test kondisi yang menentukan akhir loop (*condition expression*)

c. penentu perubahan nilai (*incremental expression*)

Bentuk for :

```
for (initial value; condition expression; incremental expression)
```

Keterangan :

- *initial value* : memberikan nilai awal pada variabel kontrol
- *condition expression* : ekspresi yang menyatakan berhentinya pengulangan. Jika tes kondisi bernilai salah maka loop akan berhenti.

- *incremental expression* : berfungsi menaikkan/ menurunkan nilai dari variabel kontrol. Dapat berupa nilai positif (penaikan)/ nilai negatif (penurunan).
    - Penaikan : setiap loop operator ++ akan menambah nilai 1 ke variabel kontrol
    - Penurunan : setiap operator -- akan menurunkan nilai 1 pada variabel kontrol

2.3.6.2 Struktur For Bersarang

Dinamakan struktur for bersarang karena di dalam for ada for.

Contoh :

```
for (i=1; i<=3; i++)
{
for (j=1; j<=5; j++)
{
printf("*");
}
printf(" \n");
}
```

2.3.6.3 Struktur While

Struktur ini digunakan bila kita belum mengetahui secara pasti berapakali banyaknya pengulangan yang akan dilakukan. Berakhirnya proses pengulangan ditentukan oleh suatu kondisi. Selama kondisi terpenuhi, maka pengulangan terus dilakukan, dan sebaliknya, bila kondisinya tidak terpenuhi maka pengulangan dihentikan.

Bentuk while :

```
while (condition expression)
{
statement-statement;
}
```

2.3.7 Subprogram/ Fungsi

Program komputer yang dibuat untuk menyelesaikan permasalahan umumnya berukuran besar. Cara terbaik untuk menangani program besar adalah menyusunnya dari potongan-potongan program yang berukuran kecil-kecil (disebut modul) yang merupakan konsep dari pemrograman terstruktur yaitu pemrograman yang menitikberatkan pada pemecahan masalah yang kompleks menjadi masalah yang sederhana.

Program yang terdiri dari modul/ subprogram/ prosedur/ routine lebih mudah ditangani dibanding dengan program yang terdiri dari banyak sekali baris. Modul program dalam C disebut fungsi (function). Fungsi adalah blok dari kode yang dirancang untuk melakukan tugas khusus.

Tujuan pembuatan fungsi :

- program menjadi terstruktur
- menghemat kode program karena dapat mengurangi duplikasi kode
- fungsi dapat dipanggil dari program atau fungsi yang lain
- mempersingkat/memperpendek panjang program
- mempermudah cek kesalahan

2.3.7.1 Pendeklarasian dan Pendefinisian Fungsi

Fungsi harus dideklarasikan di dalam program pemanggil/ program utama, dengan tujuan supaya program pemanggil mengenal nama fungsi tersebut serta cara mengaksesnya. Biasanya deklarasi fungsi diakhiri dengan tanda " ; " sedangkan pendefinisian fungsi tidak diakhiri tanda " ; ".

Aturan pemberian nama fungsi sama dengan aturan penulisan variabel. Blok fungsi diawali dengan "{" dan diakhiri dengan "}" .

Bentuk :

```
tipe_data nama_fungsi (daftar parameter)
```

Keterangan :

daftar parameter : berisi variabel dan tipe variabel yang berfungsi sebagai masukan untuk fungsi tersebut. Masukan tersebut akan diproses untuk menghasilkan nilai tertentu sesuai dengan tipe data fungsi.

contoh : `int tukar (int x, int y)`

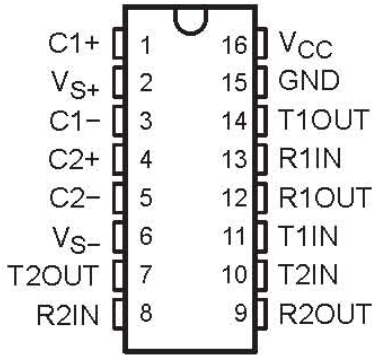
2.3.7.2 Fungsi yang Mengembalikan Nilai

Pada dasarnya semua fungsi mengembalikan nilai, tetapi untuk fungsi yang tidak mengembalikan nilai disebut fungsi bertipe **void**. Untuk mengembalikan nilai sebuah fungsi, digunakan kata **return** yang diikuti dengan nilai yang akan dikembalikan. Dalam sebuah fungsi return bisa mengembalikan beberapa nilai, tetapi setiap kata return hanya bisa mengembalikan sebuah nilai saja.

2.4 Komunikasi Serial RS232

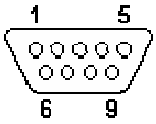
Komunikasi data serial sangat berbeda dengan format pemindahan data paralel. Pengiriman bit-bit pada serial tidak dilakukan sekaligus melalui saluran paralel, tetapi setiap bit dikirimkan satu persatu melalui saluran tunggal. Dalam pengiriman data secara serial harus ada sinkronisasi atau penyesuaian antara pengirim dengan penerima agar data yang dikirimkan dapat diterima dengan tepat dan benar oleh penerima.

Salah satu mode transmisi dalam komunikasi serial adalah mode *asynchronous*. Transmisi serial mode ini digunakan apabila pengiriman data dilakukan satu karakter tiap pengiriman. Antara satu karakter dengan karakter lainnya tidak ada waktu antara yang tetap. Karakter dapat dikirimkan sekaligus ataupun beberapa karakter kemudian berhenti untuk waktu yang tidak tentu, kemudian dikirimkan kembali sisanya. Dengan demikian, bit-bit data ini dikirimkan dengan periode yang acak sehingga pada sisi penerima, data akan diterima kapan saja. Sinkronisasi yang terjadi pada mode transmisi ini adalah dengan memberikan bit-bit penanda awal dan penanda akhir dari data pada sisi pengirim maupun penerima.



Gambar 2.6 *IC Serial MAX232*

Gambar 2.6 di atas merupakan sebuah IC yang digunakakn sebagai konverter RS232 ke TTL dan sebaliknya. RS232 standar yang biasa digunakan adalah konektor DB9. RS232 model ini merupakan salah satu konektor yang biasa digunakan untuk komunikasi serial. RS232 ini mempunyai sembilan pin yang memiliki fungsi masing-masing. Pin yang biasa digunakan adalah pin 2 yang berfungsi sebagai *received* data (RxD), pin 3 sebagai *transmited* data (TxD), dan pin 5 sebagai *ground signal*, seperti terlihat pada Gambar 2.7 berikut.



Gambar 2.7 *Konfigurasi DB9 sebagai Komunikasi Serial (RS232)*

Karakteristik elektrik dari RS232 adalah sebagai berikut :

a. *space (logic 0)* mempunyai level tegangan sebesar +3V sampai +25V.
b. *mark (logic 1)* mempunyai level tegangan sebesar -3V sampai -25V.
c. level tegangan antara -3V sampai +3V tidak terdefinisikan.
d. arus yang melalui rangkaian tidak boleh melebihi dari 500mA, hal ini diperlukan agar sistem yang dibangun bekerja dengan akurat.

Adapun fungsi dari masing-masing pin konektor DB9 diperlihatkan pada Tabel 2.5 berikut ini.

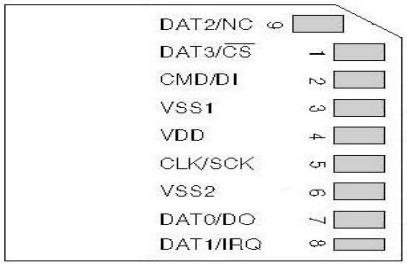
Tabel 2.5 *Fungsi Masing-Masing Pin dari DB9*

Pin	Nama	Fungsi
1	CD	*Received Line Signal Detector (Data Carier Detect)*
2	RxD	*Received Data*
3	TxD	*Transmited Data*
4	DTR	*Data Terminal Ready*
5	GND	*Signal Ground*
6	DSR	*Data Set Ready*
7	RTS	*Request To Send*
8	CTS	*Clear To Send*
9		*Ring Indicator*

2.5 *Multi Media Card* (MMC)

2.5.1 Deskripsi Umum

Multi Media Card atau lebih dikenal dengan MMC merupakan sebuah media penyimpanan *universal* yang umum digunakan untuk berbagai media seperti telepon genggam, PDA, kamera digital, MP3 *player* dan berbagai peralatan elektronik lainnya. MMC dibuat berbasis *removeble flash memory* yang media penyimpanannya memiliki bentuk fisik kecil, berdaya rendah, *non-volatile*, tidak butuh tenaga untuk menjaga data yang telah tersimpan. Sehingga media berbasis flash banyak digunakan sebagai media penyimpanan *portable* yang menggunakan baterai. Konfigurasi pin dari MMC dapat terlihat seperti Gambar 2.8 berikut.



Gambar 2.8 *Konfigurasi Pin MMC/SD Card*

Sistem komunikasi pada MMC didasarkan pada tujuh pin *serial bus* yang dirancang bekerja pada tegangan rendah. Mode komunikasi ini adalah standar komunikasi yang lebih dikenal dengan nama *Multi Media Card Mode*. Untuk menambah kompabilitas antara MMC dengan pengontrol yang ada, maka MMC menawarkan mode komunikasi alternatif yang dibuat berdasarkan standar *Serial Peripheral Interface* (SPI). Fungsi masing-masing pin pada MMC dapat dilihat seperti pada Tabel 2.6 berikut.

Tabel 2.6 *Fungsi Masing-Masing Pin pada MMC*

Pin	**Nama**	**Fungsi SD *Mode***	**Fungsi SPI *Mode***
1	DAT3/ CS	*Data Line 3*	*Chip Select/ Slave Select (SS)*
2	CMD/ DI	*Command Line*	*Master Out Slave In (MOSI)*
3	VSS1	*Ground*	*Ground*
4	VDD	*Supply Voltage*	*Supply Voltage*
5	CLK/ SCK	*Clock*	*Clock (SCK)*
6	VSS2	*Ground*	*Ground*
7	DAT0/ DO	*Data Line 0*	*Master In Slave Out (MISO)*
8	DAT1/ IRQ	*Data Line 1*	
9	DAT2/ NC	*Data Line 2*	

2.5.2 Komunikasi Data SPI

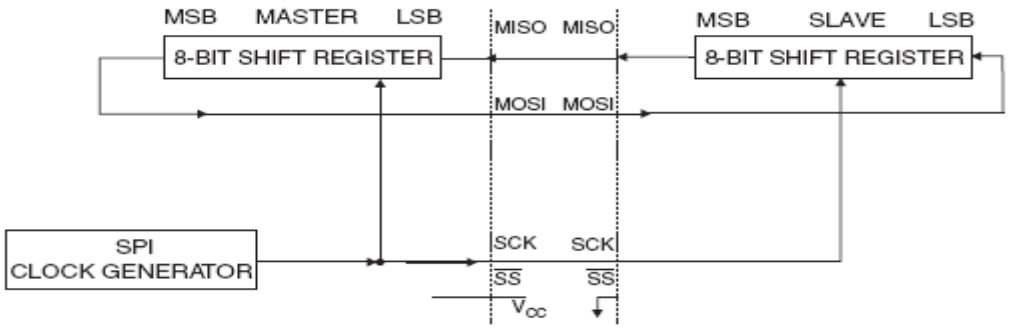
Serial Peripheral Interface (SPI) merupakan salah satu metode pengiriman data dari suatu perangkat ke perangkat lainnya. Metode ini merupakan metode yang bekerja pada metode *full duplex* dan merupakan standar sinkronasi *serial data link* yang dikembangkan oleh Motorola. Pada SPI, perangkat dibagi menjadi dua bagian yaitu *master* dan *slave*, dengan *master* sebagai perangkat yang menginisiasi pengiriman data. Sebuah *master* dalam aplikasinya dapat digunakan untuk mengatur pengiriman data dari atau ke beberapa *slave* sekaligus.

Komunikasi serial data antara *master* dan *slave* pada SPI diatur melalui 4 buah pin yang terdiri dari SCLK, MOSI, MISO, dan SS. Berikut ini adalah penjelasan singkat mengenai ke 4 pin tersebut :

- *serial clock* (SCLK) merupakan data biner yang keluar dari *master* ke *slave* yang berfungsi sebagai *clock* dengan frekuensi tertentu. *Clock* merupakan salah satu komponen prosedur komunikasi data SPI. Dalam beberapa perangkat, istilah yang digunakan untuk pin ini adalah SCK
- *master out slave input* (MOSI) merupakan pin yang berfungsi sebagai jalur data pada saat data keluar dari *master* dan masuk ke dalam *slave*. Istilah lain untuk pin ini antara lain SIMO, SDI, DI, dan SI
- *master input slave output* (MISO) merupakan pin yang berfungsi sebagai jalur data yang keluar dari *slave* dan mesuk ke dalam *master*. Istilah lain untuk pin ini adalah SOMI, SDO, DO, dan SO
- *slave select* (SS) merupakan pin yang berfungsi untuk mengaktifkan *slave* sehingga pengiriman data hanya dapat dilakukan jika *slave* dalam keadaan aktif (*active low*). Istilah lain untuk SS antara lain CS (*chip select*), nCS, nSS, dan STE (*slave transmit enable).*

Pin SCLK, MOSI, dan SS merupakan pin dengan arah pengiriman data dari *master* ke *slave*. Sebaliknya, MISO mempunyai arah komunikasi data dari *slave* ke *master*. Pengaturan hubungan antara pin MISO dan MOSI harus sesuai dengan ketentuan. Pin MISO pada *master* harus dihubungkan dengan pin MOSI pada *slave,* begitu juga sebaliknya. Hal ini penting untuk diperhatikan untuk menghindari terjadinya kesalahan prosedur pada pengiriman data. Istilah pin-pin

SPI untuk berbagai perangkat mungkin saja mempunyai istilah yang berbeda dengan istilah di atas tergantung produsen yang membuatnya.

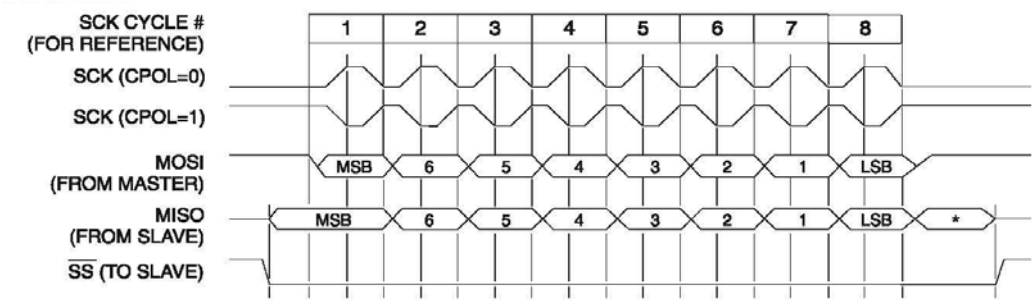


Gambar 2.9 *Prosedur Operasi SPI*

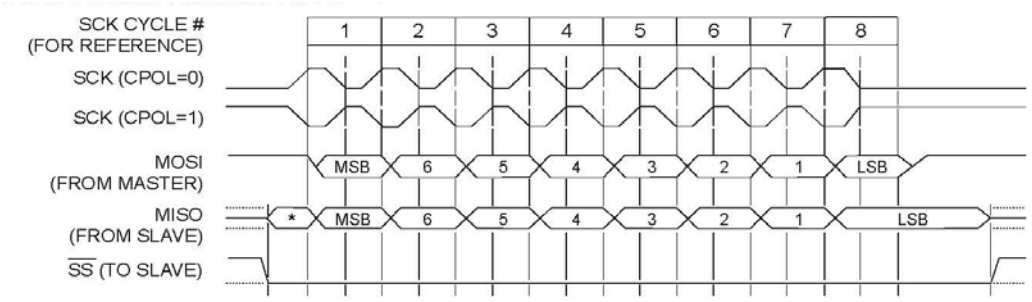
Komunikasi data SPI dimulai pada saat *master* mengirimkan *clock* melalui SCK dengan frekuensi lebih kecil atau sama dengan frekuensi maksimum pada *slave.* Kemudian, *master* memberi logika nol pada SS untuk mengaktifkan *slave* sehingga pengiriman data (berupa siklus *clock*) siap untuk dilakukan. Pada saat siklus *clock* terjadi, transmisi data *full duplex* terjadi dengan dua keadaan sebagai berikut :

- *master* mengirim sebuah bit pada jalur MOSI, *slave* membacanya pada jalur yang sama
- *slave* mengirim sebuah bit pada jalur MISO, *master* membacanya pada jalur yang sama.

Transmisi dapat menghasilkan beberapa siklus *clock.* Jika tidak ada data yang dikirim lagi maka *master* menghentikan *clock* tersebut dan kemudian menon-aktifkan *slave.*



Gambar 2.10 *Format Penerimaan Karakter (CPHA = 0)*



Gambar 2.11 *Format Pengiriman Karakter (CPHA = 1)*

Diagram pewaktuan (*timing diagram*) SPI dimulai pada saat SS diaktifkan (*low*). Pada saat tersebut siklus *clock* dimulai, pada contoh diatas dalam satu siklus terdapat 8 bit pengiriman data. Saat SS aktif, MISO/MOSI mulai mengirimkan data mulai dari *Most Significant Bit* (MSB) data tersebut. Pada saat *clock* berubah maka proses pengiriman data dilanjutkan pada bit yang lebih rendah. Proses tersebut berlangsung sampai pengiriman data selesai dengan mengirimkan bit *Least Significant Bit* (LSB) dan siklus *clock* berakhir serta SS kembali dinon-aktifkan (*high*). Pada saat ini biasanya *slave* mengirimkan *interrupt* ke *master* yang mengindikasikan bahwa pengiriman data telah selesai dan siap untuk melakukan pengiriman data selanjutnya. Dalam diagram pewaktuan, *clock* mempunyai beberapa mode pengaturan pada polaritas (CPOL) dan fasa (CPHA).

Pengaturan kedua mode tersebut jelas terlihat pada ilustrasi di atas (Gambar 2.10 dan Gambar 2.11).