BAB II
LANDASAN TEORI

2.1 Referensi

RFID adalah sebuah teknologi yang menggunakan frekuensi radio untuk mengindentifikasi suatu barang atau manusia. (Erwin, 2004 : 7). Menurut Henlia (2006 : 1), sejarah perkembangan *radio frequency identification* dimulai sejak tahun 1920, tetapi berkembang menjadi IFF *transponder* pada tahun 1939. Yang waktu itu berfungsi sebagai alat identifikasi pesawat musuh, dipakai oleh militer Inggris pada perang dunia II. Sejak tahun 1945 beberapa orang berfikir bahwa perangkat pertama *RFID* ditemukan oleh Leon Theremin sebagai suatu *tool spionase* untuk pemerintahan Rusia. Menurut Kania (2011 : 16), sistem *RFID* terbagi menjadi 3 komponen, yaitu : *RFID Tag, RFID Terminal Reader*, dan *Middleware*. Sedangkan untuk jenisnya *RFID* terbagi, berdasarkan frekuensi, berdasarkan sumber energi, dan berdasarkan bentuk.

Penerapan RFID sudah digunakan di berbagai jenis perpustakaan. Mulai dari perpustakaan perguruan tinggi, perpustakaan daerah, perpustakaan sekolah dan jenis perpustakaan lainnya. Adapun kelebihan dari sistem RFID tersebut adalah sistem inventori berkecepatan tinggi, proses sirkulasi yang cepat, penanganan buku-buku secara otomatis. (Kania, 2011). *RFID* mampu membaca suatu objek data dengan ukuran tertentu tanpa melalui kontak langsung (*contacless*) dan tidak harus sejajar dengan objek yang dibaca, selain dapat menyimpan informasi pada tag RFID sesuai dengan kapasitasnya penyimpanan (tarigan,2004).

Tabel 2.1 Kelebihan Kekurangan *RFID* pada beberapa Aplikasi

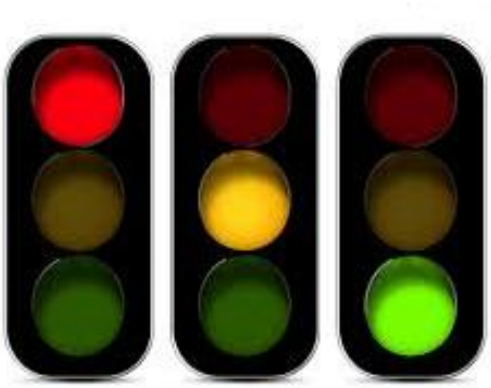
Nama	Judul	Kelebihan	Kekurangan
Hamid 2010	Pengembangan Sistem Parkir Terkomputerisasi dengan Otomatisasi Pembiayaan dan Penggunaan *RFID* sebagai Pengenal Unik Pengguna	1. Pengelolaan lahan parkir lebih mudah dan efisien. 2. Input pelat kendaraan secara otomatis 3. Pembayaran biaya parkir secara otomatis 4. *Card* dapat diisi ulang.	Kurangnya teknologi pengolahan citra dalam pencocokan identitas kendaraan.
I Wayan Kemara Giri, S.Sos.,M.si	Optimalisasi Utilitas Gudang Unilever-PT.POS Indonesia melalui penataan Layout Gudang dan Aplikasi Sistem Informasi Manajemen Inventory Pergudangan berupa sistem *RFID*	Otomatisasi pencatatan yang keluar masuk dari gudang	Semua barang harus dipasang *tag RFID*
Supriyono	Penerapan Aplikasi *RFID* dibidang Perpustakaan	1. sistem inventori berkecepatan tinggi, proses sirkulasi yang cepat, penanganan buku-buku secara otomatis	Dibutuhkan *middleware* dalam jumlah besar, yaitu penghubung antara *tag dan reader*.

		2. Memudahkan pelacakan buku yang pernah dipinjam anggota perpustakaan yang pernah dilayani oleh pustakawan. 3. Meningkatkan proses peminjaman (*check out)* buku sekaligus secara bersamaan	

2.2 Lampu Lalu Lintas

Menurut UU no. 22/2009 tentang Lalu lintas dan Angkutan Jalan: **alat pemberi isyarat lalu lintas** atau **APILL**) Lampu lalu lintas adalah lampu yang mengendalikan arus lalu lintas yang terpasang di persimpangan jalan, tempat penyeberangan pejalan kaki (*zebra cross*), dan tempat arus lalu lintas lainnya. Lampu ini yang menandakan kapan kendaraan harus berjalan dan berhenti secara bergantian dari berbagai arah. Pengaturan lalu lintas di persimpangan jalan dimaksudkan untuk mengatur pergerakan kendaraan pada masing-masing kelompok pergerakan kendaraan agar dapat bergerak secara bergantian sehingga tidak saling mengganggu antar-arus yang ada.

Warna lampu lalu lintas ada 3 (tiga), merah yang menandakan kendaraan harus berhenti. Kuning, yang menandakan kendaraan untuk hati-hati atau bersiap-siap untuk warna merah maupun warna hijau. Hijau, yaitu menandakan kendaraan harus berjalan.



Gambar 2.1 Lampu Lalu Lintas

(Sumber : www.kaskus.co.id)

Jenis-jenis lampu lalu lintas

Berdasarkan cakupannya

1. Lampu lalu lintas terpisah — pengoperasian lampu lalu lintas yang pemasangannya didasarkan pada suatu tempat persimpangan saja tanpa mempertimbangkan persimpangan lain.
2. Lampu lalu lintas terkoordinasi — pengoperasian lampu lalu lintas yang pemasangannya mempertimbangakan beberapa persimpangan yang terdapat pada arah tertentu.
3. Lampu lalu lintas jaringan — pengoperasian lampu lalu lintas yang pemasangannya mempertimbangkan beberapa persimpangan yang terdapat dalam suatu jaringan yang masih dalam satu kawasan.

Berdasarkan cara pengoperasiannya

1. *Fixed time traffic signal* — lampu lalu lintas yang pengoperasiaannya menggunakan waktu yang tepat dan tidak mengalami perubahan.
2. *Actuated traffic signal* — lampu lalu lintas yang pengoperasiaannya dengan pengaturan waktu tertentu dan mengalami perubahan dari waktu ke waktu sesuai dengan kedatangan kendaraan dari berbagai persimpangan.

Tujuan adanya lampu lalu lintas :

1. Menghindari hambatan karena adanya perbedaan arus jalan bagi pergerakan kendaraan.

2. Memfasilitasi persimpangan antara jalan utama untuk kendaraan dan pejalan kaki dengan jalan sekunder sehingga kelancaran arus lalu lintas dapat terjamin.
3. Mengurangi tingkat kecelakaan yang diakibatkan oleh tabrakan karena perbedaan arus jalan.

2.3 *RFID (Radio Frequency Identification Device)*

Definisi menurut (Maryono, 2005) identifikasi dengan frekuensi radio adalah teknologi untuk mengidentifikasi seseorang atau objek benda menggunakan transmisi frekuensi radio, khususnya 125kHz, 13.65Mhz atau 800-900MHz. RFID menggunakan komunikasi gelombang radio untuk secara unik mengidentifikasi objek atau seseorang terdapat beberapa pengertian RFID menurut (Maryono, 2005) yaitu :

a. RFID (*Radio Frequency Identification*) adalah sebuah metode identifikasi dengan menggunakan sarana yang disebut label RFID atau *transponder* (*tag*) untuk menyimpan dan mengambil data jarak jauh.
b. Label atau transponder (*tag*) adalah sebuah benda yang bisa dipasang atau dimasukkan di dalam sebuah produk, hewan atau bahkan manusia dengan tujuan untuk identifikasi menggunakan gelombang radio. Label RFID terdiri atas *mikrochip silikon* dan antenna.

2.3.1 RFID Tag

Tag RFID dapat berupa *stiker*, kertas atau plastik dengan beragam ukuran. Di dalam setiap *tag* ini terdapat *chip* yang mampu menyimpan sejumlah informasi tertentu. Memori pada *tag* secara dibagi menjadi sel-sel. Beberapa sel menyimpan data *Read Only*, misalnya serial number yang unik yang disimpan pada saat *tag* tersebut diproduksi. Selain pada RFID mungkin juga dapat ditulis dan dibaca secara berulang.

Sebuah *tag* RFID atau transponder, terdiri atas sebuah mikro (*microchip*) dan sebuah sistem. *Chip* mikro itu sendiri dapat berukuran sekecil butiran pasir, seukuran 0.4 mm. *Chip* tersebut menyimpan nomor seri yang unik atau informasi lainnya tergantung kepada tipe memorinya. Tipe memori itu sendiri dapat *read-*

only, read-write, atau *writeonceread-many*. Antena yang terpasang pada chip mikro mengirimkan informasi dari chip ke *reader*. Biasanya rentang pembacaan diindikasikan dengan besarnya sistem. Antena yang lebih besar mengindikasikan rentang pembacaan yang lebih jauh. *Tag* tersebut terpasang atau tertanam dalam obyek yang akan diidentifikasi. *Tag* dapat dis*can* dengan *reader* bergerak maupun stasioner menggunakan gelombang radio.

Gambar 2.2 RFID *Tag*

(Sumber : Maryono, 2005)

Berdasarkan catu daya *tag*, *tag RFID* dapat digolongkan menjadi :

a. *Tag* Aktif: yaitu *tag* yang catu dayanya diperoleh dari baterai, sehingga akan mengurangi daya yang diperlukan oleh pembaca RFID dan *tag* dapat mengirimkan informasi dalam jarak yang lebih jauh. Kelemahan dari *tipe tag* ini adalah harganya yang mahal dan ukurannya yang lebih besar karena lebih komplek. Semakin banyak fungsi yang dapat dilakukan oleh *tag* RFID maka rangkaiannya akan semakin komplek dan ukurannya akan semakin besar.

b. *Tag* Pasif: yaitu *tag* yang catu dayanya diperoleh dari medan yang dihasilkan oleh pembaca RFID. Rangkaiannya lebih sederhana, harganya jauh lebih murah, ukurannya kecil, dan lebih ringan. Kelemahannya adalah *tag* hanya dapat mengirimkan informasi dalam jarak yang dekat dan pembaca RFID harus menyediakan daya tambahan untuk *tag* RFID. *Tag* RFID telah sering dipertimbangkan untuk digunakan sebagai *barcode* pada masa yang akan datang. Pembacaan informasi pada *tag* RFID tidak memerlukan kontak sama sekali. Karena kemampuan rangkaian terintegrasi yang modern, maka *tag* RFID dapat menyimpan jauh lebih banyak informasi dibandingkan dengan *barcode.*

2.3.2 Reader RFID

Terminal *Reader* RFID, terdiri atas RFID *reader* dan antena yang akan mempengaruhi jarak optimal identifikasi. Terminal RFID akan membaca atau mengubah informasi yang tersimpan di dalam *tag* melalui frekuensi radio. Terminal RFID terhubung langsung dengan sistem *Host* Komputer

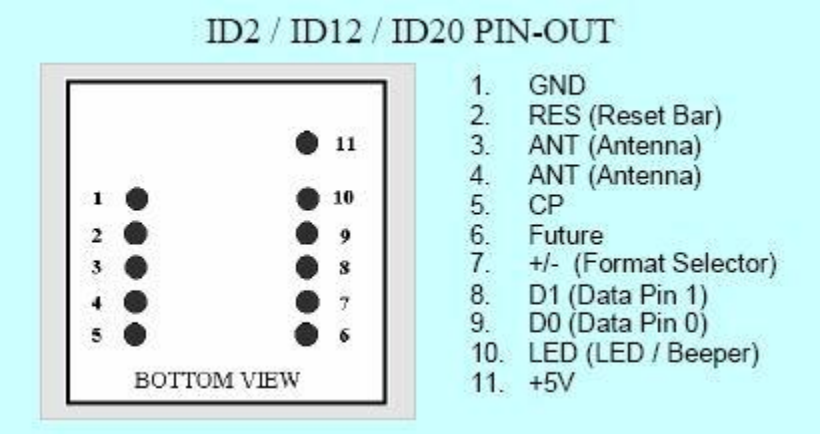
Gambar 2.3 RFID *Reader*

(Sumber : Maryono, 2005)

Tabel 2.2 Spesifikasi modul RFID *reader* ID-12

Parameter	ID-12
Jarak Baca	Sampai 2cm
Dimensi	26mm x 25mm x 7mm
Frekuensi	125kHz
Format Kartu	GK4001/EM 4001 atau yang *compatible*
Encoding	Manchaster 64-bit, modulus 64
Jenis Catudaya	5VDC pada 30mA nominal
Arus Output I/O	-
Jangkauan Catudaya	+4.6V-5.4V

Pemilihan keadaan untuk pin 5, pin 7, dan pin 8/pin 9 pada ID-12 digunakan untuk memilih keluaran data yang diinginkan. Pin 3 dan 4 digunakan untuk penambahan antena luar dan kapasitor tuning. Pin 10 digunakan untuk menyalakan buzzer atau led sebagai penanda sebuah tag terbaca. Konfigurasi pin ID-12 diberikan pada Gambar 2.4



Gambar 2.4 Spesifikasi pin pada ID-2, ID-12, dan ID-20
https://www.sparkfun.com/datasheets/Sensors/ID-12-Datasheet.pdf

RFID *Reader* ID-12 mempunyai spesifikasi:

1. Tegangan pada kaki 11 adalah +4,6 Volt hingga +5,5 Volt.
2. Frekuensi yang digunakan adalah 125 KHz.
3. Keluaran data digital dapat berupa format ASCII ataupun format Wiegand pada kaki 8 dan kaki 9.
4. Hanya dapat menangkap data dari RFID *Tag Card* yang berjenis EM 4001

2.4 Mikrokontroler

Mikrokontroler adalah sebuah komputer kecil ("*Special purpose computers*") di dalam satu IC yang berisi CPU, memori, timer, saluran komunikasi serial dan parallel, Port I/O,ADC. Mikrokontroler digunakan suntuk suatu tugas dan menjalankan suatu program.(Andrianto, 2013: 1)

Mikrokontroler, sebagai suatu terobosan teknologi mikroprosesor dan mikrokomputer, hadir memenuhi kebutuhan pasar (*market need*) dan teknologi baru. Sebagai teknologi baru, yaitu teknologi semikonduktor dengan kandungan transistor yang lebih banyak namun hanya membutuhkan ruang yang kecil serta dapat diproduksi secara masal (dalam jumlah banyak) membuat harganya menjadi lebih murah (dibandingkan mikroprosesor). (Andrianto, 2013: 2*)*

Adapun kelebihan dari mikrokontroller adalah sebagai berikut :

1. Penggerak pada mikrokontroler menggunakan bahasa pemograman assembly dengan berpatokan pada kaidah digital dasar sehingga pengoperasian sistem menjadi sangat mudah dikerjakan sesuai dengan logika sistem.
2. Mikrokontroler tersusun dalam satu chip dimana prosesor, memori, dan I/O terintegrasi menjadi satu kesatuan kontrol sistem.
3. Sistem running bersifat berdiri sendiri tanpa tergantung dengan komputer sedangkan parameter komputer hanya digunakan untuk download perintah instruksi atau program.
4. Pada mikrokontroler tersedia fasilitas tambahan untuk pengembangan memori dan I/O yang disesuaikan dengan kebutuhan sistem.
5. Harga untuk memperoleh alat ini lebih murah dan mudah didapat.

2.5 Mikrokontroler AVR Atmega16

AVR merupakan seri mikrokontroler CMOS 8-bit buatan Atmel, berbasis arsitektur RISC (*Reduced Instruction Set Computer*). Hampir semua instruksi dieksekusi dalam satu siklus clock. AVR mempunyai 32 register general-purpose, *timer/counter* fleksibel dengan *modecompare*, *interrupt internaldan eksternal*, serial UART, programmable Watchdog Timer, dan mode power saving, ADC dan PWM internal. (Andrianto, 2013: 1)

AVR juga mempunyai *In-System Programmable Flash on-chip* yang mengijinkan memori program untuk diprogram ulang dalam system menggunakan hubungan serial SPI. ATMega16 mempunyai *throughput* mendekati 1 MIPS per MHz membuat disainer sistem untuk mengoptimasi konsumsi daya versus kecepatan proses.

Berikut ini ringkasan berbagai macam fitur-fitur untuk Mikrokontroler AVR ATMega16 :

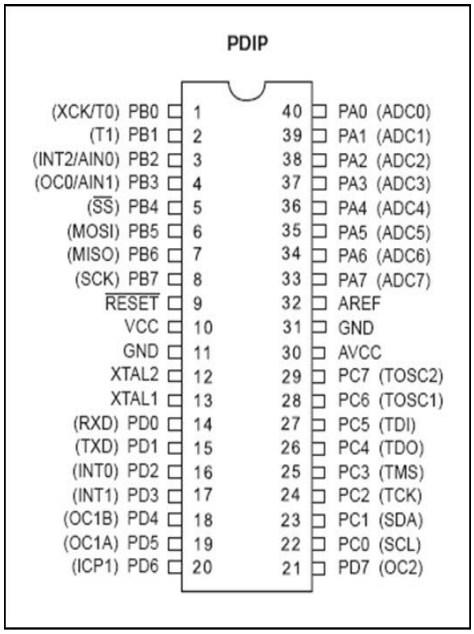
1. Mikrokontroler AVR 8-bit daya-rendah .
2. Arsitektur RISC tingkat lanjut
    a. 131 Instruksi yang ampuh (Hampir semuanya dieksekusi dalam satu detak (dock saja).
    b. 32 x 8 General Purpose Working Registers.

c. Operasi statis penuh.
d. Throughput hingga 16 MIPS pada 16 MHz .
e. Pengali On-chip 2-cycle.

3. High Endurance Non-volatile Memory segments
   a. 16K Bytes of In-System Self-programmable Flash program memory.
   b. 512 Bytes EEPROM .
   c. 1K Byte Internal SRAM.
   d. Write/Erase Cycles: 10,000 Flash/100,000 EEPROM - Data retention: 20 years at 85°C/100 years at 25°C - Optional Boot Code Section with Independent
   e. Lock Bits In-System Programming by On-chip Boot Program True Read-While-Write Operation.
   f. Programming Lock for Software Security.
4. Antarmuka ITAG (IEEE std. 1149.1 Compliant)
   a. Boundary-scan Capabilities According to the ITAG Standard
   b. Extensive On-chip Debug Support
   c. Programming of Flash, EEPROM, Fuses, and Lock Bits through the ITAG Interface
5. Fitur-fiturperiferal
   a. Dua Pewaktu/Pencacah 8-bit dengan Praskalar dan Mode Pembanding terpisah.
   b. Sebuah Pewaktu/Pencacah 16-bit Timer/Counter Dengan Praskalar, Mode Pembanding dan Capture yang terpisah.
   c. Pencacah Real Time denganOsilator terpisah
   d. Empat kanal PWM - 8-kanal, 10-bit ADC
6. Byte-oriented Two-wire Serial Interface
7. Programmable Serial USART

(Andrianto, 2013)

2.5.1 Konfigurasi Pin Atmega16



Gambar 2.5 Konfigurasi Pin Atmega16

(Sumber : share-pdf.com)

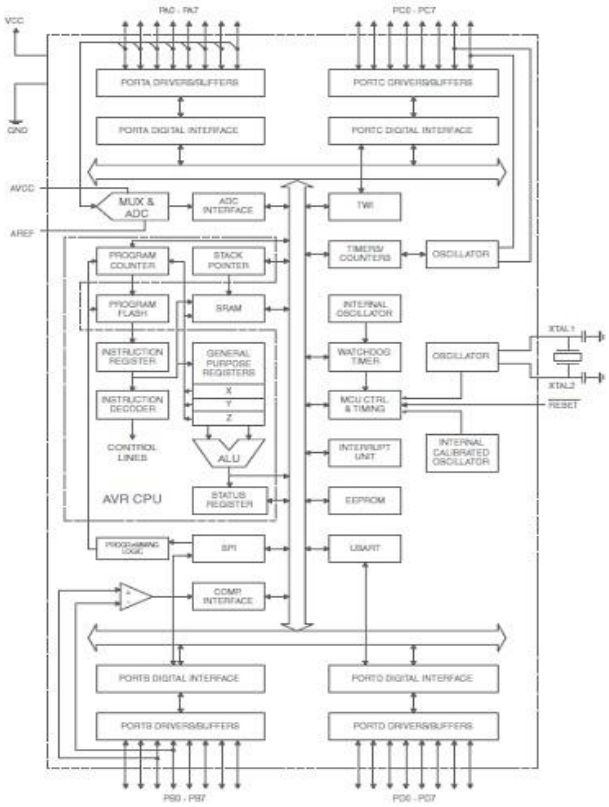
Gambar di atas merupakan susunan kaki standar 40 pin mikrokontroler AVR Atmega16. Berikut penjelasan umum susunan kaki Atmega16 tersebut:

a. VCC merupakan pin masukan positif catudaya. Setiap peralatan elektronika digital tentunya butuh sumber catu daya yang umumnya sebesar 5 V, itulah sebabnya di PCB kit rangkaian mikrokontroler selalu dipasang IC regulator 7805.
b. GND sebagai PIN ground.
c. Port A (PA0 ... PA7) merupakan pin I/O dua arah dan dapat deprogram sebagai pin masukan ADC.
d. Port B (PB0 ... PB7) merupakan pin I/O dua arah dan pin fungsi khusus, yaitu Timer/Counter, Komparator Analog, dan SPI.
e. Port C (PC0 ... PC7) merupakan pin I/O dua arah dan pin fungsi khusus, yaitu TWI, komparator analog, dan Timer Oscilator.
f. Port D (PD0 ... PD7) merupakan pin I/O dua arah dan pin fungsi khusus, yaitu komparator analog, interupsi eksternal, dan komunikasi serial.

g. Reset merupakan pin yang digunakan untuk me-reset mikrokontroler ke kondisi semula.
h. XTAL 1 dan XTAL 2 sebagai pin masukan *clock* eksternal. Suatu mikrokontroler membutuhkan sumber detak (*clock*) agar dapat mengeksekusi intruksi yang ada di memori. Semakin tinggi nilai kristalnya, maka semakin cepat pula mikrokontroler tersebut dalam mengeksekusi program.
i. AVCC sebagai pin masukan tegangan untuk ADC.
j. AREF sebagai pin masukan tegangan referensi.

ATMega16 mempunyai empat buah *port* yang bernama *Port*A, *Port*B, *Port*C, dan *Port*D. Keempat *port* tersebut merupakan jalur *bidirectional* dengan pilihan *internal pull-up*. Tiap *port* mempunyai tiga buah register bit, yaitu DDxn, PORTxn, dan PINxn. Huruf 'x' mewakili nama huruf dari *port* sedangkan huruf 'n' mewakili nomor bit. Bit DDxn terdapat pada I/O *address* DDRx, bit PORTxn terdapat pada I/O *address* PORTx, dan bit PINxn terdapat pada I/O *address* PINx. Bit DDxn dalam register DDRx (*Data Direction Register*) menentukan arah pin. Bila DDxn diset 1 maka Px berfungsi sebagai pin *output*. Bila DDxn diset 0 maka Px berfungsi sebagai pin input. Bila *PORT*xn diset 1 pada saat pin terkonfigurasi sebagai pin input, maka resistor *pull-up* akan diaktifkan. Untuk mematikan resistor *pull-up*, *PORT*xn harus diset 0 atau pin dikonfigurasi sebagai pin *output*. Pin port adalah tri-state setelah kondisi reset.

Bila PORTxn diset 1 pada saat pin terkonfigurasi sebagai pin *output* maka pin *port* akan berlogika 1. Dan bila PORTxn diset 0 pada saat pin terkonfigurasi sebagai pin output maka pin *port* akan berlogika 0. Saat mengubah kondisi *port* dari kondisi *tri-state* (DDxn=0, PORTxn=0) ke kondisi *output high* (DDxn=1, PORTxn=1) maka harus ada kondisi peralihan apakah itu kondisi pull-up enabled (DDxn=0, PORTxn=1) atau kondisi *output low* (DDxn=1, PORTxn=0).

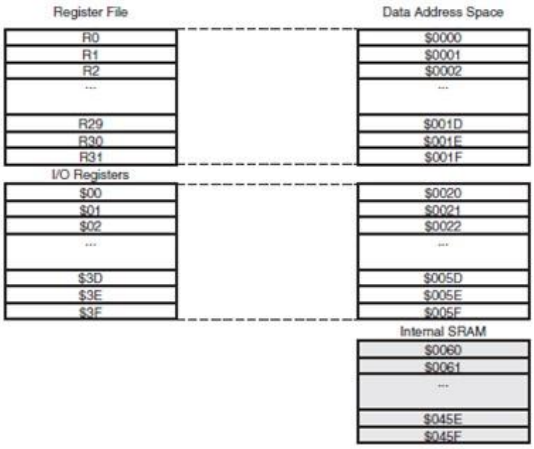


Gambar 2.6 Blok Diagram Atmega16

(Sumber : share-pdf.com)

AVR ATMega16 memiliki ruang pengalamatan memori data dan memori program yang terpisah. Memori data terbagi menjadi 3 bagian, yaitu 32 buah register umum, 64 buah register I/O, dan 1kb SRAM internal.

Register keperluan umum menempati space data pada alamat terbawah, yaitu $00 sampai $1F. Sementara itu, register khusus untuk menangani I/O dan kontrol terhadap mikrokontroler menempati 64 alamat berikutnya, yaitu mulai dari $20 hingga $5F. Register tersebut merupakan register yang khusus digunakan untuk mengatur fungsi terhadap berbagai peripheral mikrokontroler, seperti kontrol register, *timer*/*counter*, fungsi – fungsi I/O, dan sebagainya. Alamat memori berikutnya yang digunakan untuk SRAM 1kb, yaitu pada lokasi $60 sampai dengan $45F.

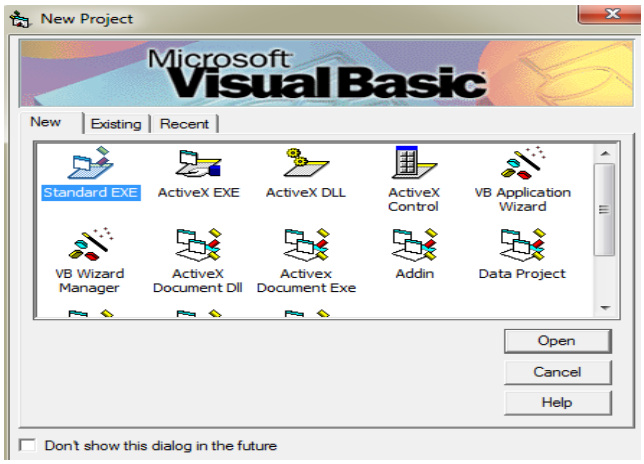


Gambar 2.7 Konfigurasi Memori Data Atmega16

(Sumber : share-pdf.com)

2.6 *Visual Basic 6.0*

Untuk memulai pemrograman dengan *visual basic*, jalankan program *visual basic*6.0. Selanjutnya pada tampilan awal akan ditampilkan kotak dialog *new project* seperti pada gambar 2.8 di bawah ini.



Gambar 2.8 Tampilan Kotak Dialog New Project

(Sumber : Fajrillah Hasballah, 2009)

Pada kotak dialog tersebut terdapat 3 buah tab yang terdiri dari:

a. ***New***(menampilkan daftar pilihan untuk membuat *project* baru)

b. ***Existing***(untuk *browsing* dan membuka *project*)

c.***Recent*** (untuk membuka *project* yang sering digunakan).

Visual Basic 6.0 menyediakan 13 jenis *project* yang bisa dibuat seperti terlihat pada gambar 1.3 di atas. Ada beberapa *project* yang biasa digunakan oleh banyak pengguna *Visual Basic*, antara lain:

a. *Standard EXE: Project* standar dalam Visual Basic dengan komponen-komponen standar. Jenis *project* ini sangat sederhana, tetapi memiliki keunggulan bahwa semua komponennya dapat diakui oleh semua unit komputer dan semua user meskipun bukan administrator. Pada buku ini akan digunakan *project Standard EXE* ini, sebagai konsep pemrograman visualnya.

b. *ActiveX EXE: Project* ini adalah *project ActiveX* berisi komponen-komponen kemampuan intuk berinteraksi dengan semua aplikasi di sistem operasi *windows*.

c. *ActiveX DLL: Project* ini menghasilkan sebuah aplikasi *library* yang selanjutnya dapat digunakan oleh semua aplikasi di sistem operasi *Windows.*

d. *ActiveX Control: Project* ini menghasilkan komponen-komponen baru untuk aplikasi Visual Basic yang lain.

e. *VB Application Wizard: Project* ini memandu pengguna untuk membuat aplikasi secara mudah tanpa harus pusing-pusing dengan perintah-perintah pemrograman.

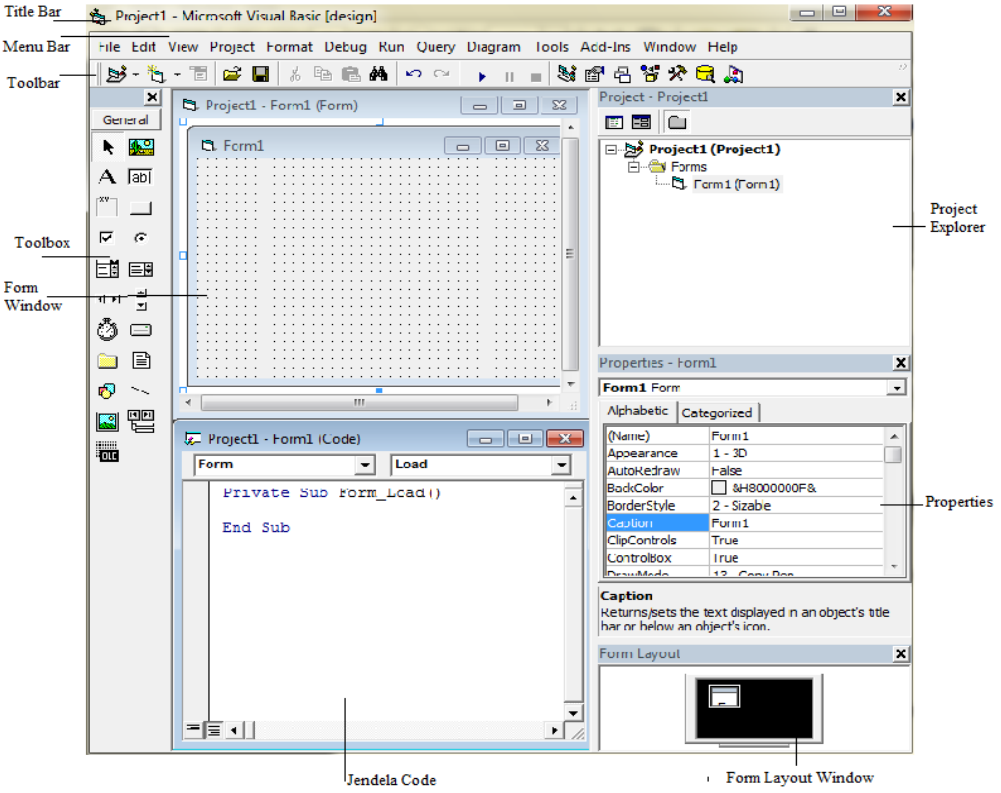
f. *Addin: Project* seperti *Standard EXE* tetapi dengan berbagai macam komponen tambahan yang memungkinkan kebebasan kreasi dari pengguna.

g. *Data project: Project* ini melengkapi komponennya dengan komponen-komponen database. Sehingga bisa dikatakan *project* ini memang disediakan untuk keperluan pembuatan aplikasi database.

h. *DHTML Application: Project* ini digunakan untuk membuat aplikasi internet pada sisi *client* (*client side*) dengan fungsi-fungsi DHTML.

i. *IIS Application: Project* ini menghasilkan aplikasi internet pada sisi *server* (*server side*) dengan komponen-komponen CGI (*Common Gateway Interface*).

Untuk pembuatan program pertama kali pilih tab New, pilih Standard EXE lalu klik Open. Selanjutnya muncul tampilan utama Visual Basic 6.0 seperti pada gambar



Gambar 2.9 IDE Microsoft Visual Basic 6.0

(Sumber : Fajrillah Hasballah, 2009)

1. *Title Bar*

   Title bar merupakan batang jendela dari program visual basic 6.0 yang terletak pada bagian paling atas dari jendela program yang berfungsi untuk menampilkan judul atau nama jendela. Selain itu juga berfungsi untuk memindahkan posisi jendela dengan menggunakan *drag* dan *drop* pada posisi *title bar* tersebut dan untuk mengatur ukuran jendela dari ukuran *minimize* ke ukuran *restore* ataupun sebaliknya dengan melakukan klik ganda pada posisi *title bar* tersebut.

2. *Menu Bar*

   Menu bar merupakan batang menu yang terletak di bawah *title bar* yang berfungsi untuk menampilkan pilihan menu atau perintah untuk mengoperasikan program visual basic. Saat pertama kali jendela program visual basic terbuka dapat dilihat tiga belas menu utama yaitu :

a. *File* : terdiri dari perintah-perintah untuk membuka, menutup, menyimpan, mencetak, mengcompile *Project* yang sedang kita kerjakan
b. *Edit* : kumpulan perintah yang membantu kita memanipulasi penulisan *code* (*listing* program)
c. *View* : terdiri dari perintah untuk melihat/berpindah antar *window* didalam lingkungan pengembangan
d. *Project* : sekumpulan perintah untuk menambah komponen *Project* (*Form, Module*, dll) dan tempat dimana kita akan men-*set-up properties Project*
e. *Format* : perintah-perintah untuk memanipulasi kontrol yang terdapat pada sebuah *Form*
f. *Debug* : Sekumpulan perintah yang berguna untuk melacak *bugs* ataupun melacak/*trace listing* program yang sedang kita kerjakan *line*-per-*line*
g. *Run* : perintah untuk menjalankan, *pause* dan menghentikan jalannya Program/aplikasi
h. *Tools* : terdiri dari *wizard-wizard* yang sangat membantu pengembangan aplikasi
i. *Add-Ins* : sekumpulan *component/wizard* yang bisa digunakan dalam pembuatan aplikasi
j. *Window* : kumpulan *window-window* yang terdapat didalam *Project*
k. *Help* : berisi *file-file* bantuan/*help* dan tentang Visual Basic itu sendiri

3. *Toolbars*

Toolbars merupakan sebuah batang yang berisi kumpulan tombol yang terletak dibagian bawah menu bar atau terdapat didalam Menu Bar *(shortcut)* yang dapat digunakan untuk menjalankan perintah memanipulasi *Project*. Pada kondisi *default* program visual basic hanya menampilkan *toolbars* standar. Namun dapat pula di-*set* sesuai dengan keinginan kita sendiri.

4. *Project Explorer*

Project Explorer merupakan suatu kumpulan *module* atau merupakan program aplikasi itu sendiri. Dalam visual basic, *file project* disimpan dengan nama *file* berakhiran vbp, dimana *file* ini berfungsi untuk menyimpan seluruh komponen program. Apabila membuat suatu program aplikasi baru maka secara otomatis

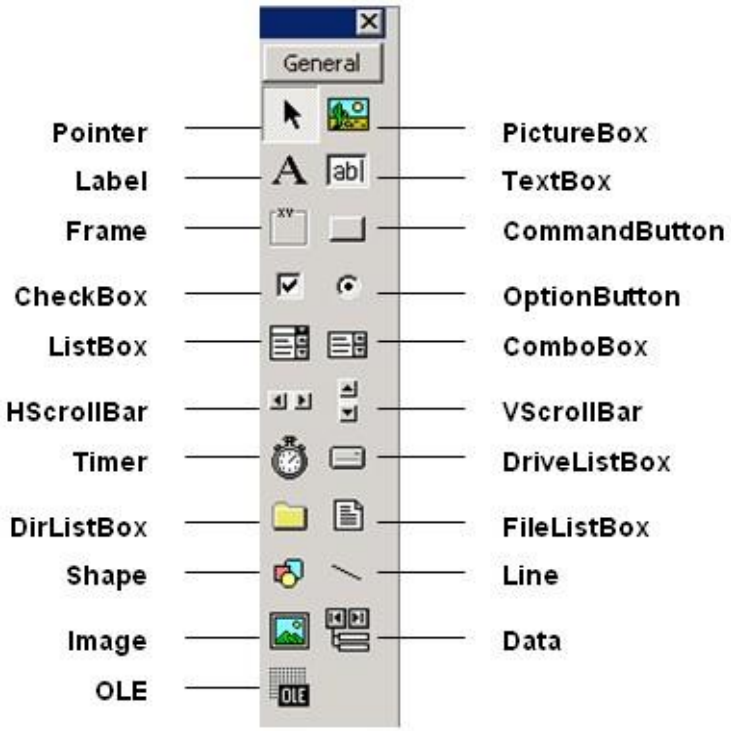
project tersebut akan diisi dengan *form* baru. Dalam jendela *project explorer* ditampilkan suatu struktur hirarki dari sebuah *project* itu sendiri yang berisi semua item yang terkandung di dalamnya.

5. *Form*

   Form merupakan *windows* atau jendela di mana akan dibuat *user interface*. Kita dapat menambahkan sebanyak mungkin *form* kedalam aplikasi kita sesuai dengan kebutuhan.

6. *Toolbox* atau kontrol

   Merupakan tampilan berbasis grafis yang dimasukkan pada *form* untuk membuat interaksi dengan pemakai. Bentuk *toolbox* visual basic adalah sebagai berikut.



Gambar 2.10 Toolbox Pada Visual Basic 6.0

(Sumber : Fajrillah Hasballah, 2009)

Adapun secara garis besar fungsi dari masing-masing control tersebut adalah sebagai berikut :

a. Pointer

   Pointer bukan merupakan suatu kontrol tapi *icon* ini digunakan ketika memilih kontrol yang sudah berada pada *form*.

b. Picturebox

   Picturebox adalah kontrol yang digunakan untuk menampilkan *image* dengan *form*at *BMP, DIB, (Bitmap), ICO (Icon), CUR (Cursor), WMF (Metafile), CMF (Enhanced Metafile), GIF, JPEG.*

c. Label

   Label adalah kontrol yang digunakan untuk menampilkan teks yang tidak dapat diperbaiki.

d. Textbox

   TextBox adalah kontrol yang mengandung *string* yang dapat dipakai oleh pemakai, dapat berupa satu baris tunggal atau banyak baris.

e. Frame

   Frame adalah kontrol yang digunakan sebagai kontainer bagi kontrol lainnya.

f. CommandButton

   CommandButton merupakan kontrol hampir ditemukan pada setiap *form* dan digunakan untuk membangkitkan *event* proses tertentu ketika pemakai melakukan klik padanya.

g. CheckBox

   CheckBox digunakan untuk pilihan yang isinya bernilai *yes* atau *no, true* atau *false*.

h. OptionButton

   OptionButton sering digunakan lebih dari satu sebagai pilihan terhadap beberapa *option* yang hanya dapat dipilih satu.

i. ListBox

   ListBox mengandung sejumlah item dan pemakai dapat memilih lebih dari satu.

j. ComboBox

   ComboBox merupakan kombinasi dari *TextBox* dan suatu *ListBox* di mana pemasukan data dapat dilakukan dengan pengetikan maupun pemilihan.

k. HScrollbar/VScrollbar

HScrollbar/VScrollbar digunakan untuk membentuk *scrollbar* berdiri sendiri.

l. Timer

Timer digunakan untuk proses *background* yang diaktifkan berdasarkan interval waktu tertentu. Ini merupakan kontrol non visual.

m. DriveListBox, DirListBox, dan *File*ListBox

DriveListBox, DirListBox, dan *FileListBox* sering digunakan untuk membentuk dialog *box* yang berkaitan dengan *file*.

n. Shape dan Line

Shape dan *Line* digunakan untuk menampilkan bentuk seperti garis, persegi, bulatan, oval.

o. Image

Image berfungsi menyerupai *image box*, tetapi tidak dapat digunakan sebagai kontainer bagi kontrol lainnya. Sesuatu yang perlu diketahui bahwa kontrol *image* menggunakan *resource* yang lebih kecil dibandingkan dengan *Picture Box.*

p. Data dan Adodc

Data dan Adodc digunakan untuk menampilkan database pada suatu *form*.

q. OLE

OLE dapat digunakan sebagai tempat bagi program eksternal seperti *Microsoft Excel, Microsoft Word,* dan lain-lain.

7. Properties

Properties merupakan nilai yang dimiliki oleh sebuah objek visual basic, merupakan sebuah jendela yang digunakan untuk menampung nama properti dari kontrol yang dipilih.

8. Jendela Code

Jendela Code adalah salah satu jendela yang paling penting dalam visual basic, yang berisi kode-kode program yang merupakan instruksi-instruksi untuk aplikasi visual basic. Setiap objek pada visual basic dapat ditambahai kode-kode program untuk melaksanakan tugas-tugas tertentu, misalnya membatalkan perintah, menutup aplikasi dan sebagainya.

9. *Form Layout Window*

   Form Layout Window merupakan sebuah jendela yang digunakan untuk mengatur posisi dari *form* pada *form* saat program dijalankan. Pada saat mengarahkan *pointer mouse* ke bagian *form*, maka *pointer mouse* akan berubah menjadi anak panah empat arah (pointer mengatur posisi) untuk memindah posisi *form* pada *layer monitor* dapat dilakukan dengan proses *drag* dan *drop*.

(Fajrillah Hasballah, 2009)

2.7 *Database*

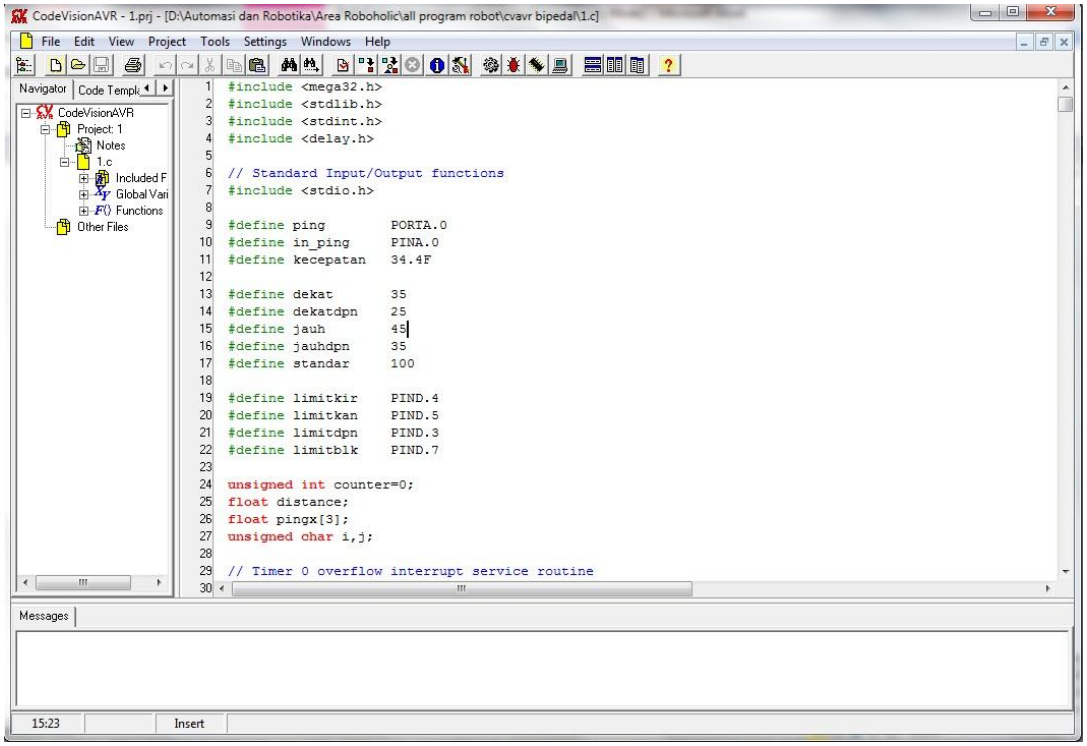
Database adalah kumpulan data yang terdiri atas satu atau lebih tabel yang terintegrasi satu sama lain, dimana setiap pemakai (*user*) diberi wewenang (otorisasi) untuk dapat mengakses (mengubah, menghampus, menganalisis, menambah, memperbaiki) data dalam tabel-tabel tersebut. *Database* merupakan salah satu komponen yang penting dalam sistem informasi yang merupakan basis dalam menyediakan informasi bagi para pemakai. Penerapan basis data dalam sistem informasi tersebut dengan *database* sistem. (Yung, 2002: 2)

2.8 *Microsoft Acces*

Microsoft Acces adalah salah satu program dari *Microsoft Office* yang dijalankan dengan menggunakan sistem operasi *Windows* yang berguna untuk penanganan data dan informasi secara structural : membuat, menyimpan, merubah, dan megaksesnya kembali dalam sebuah *Database.*

2.9 CodeVisionAVR

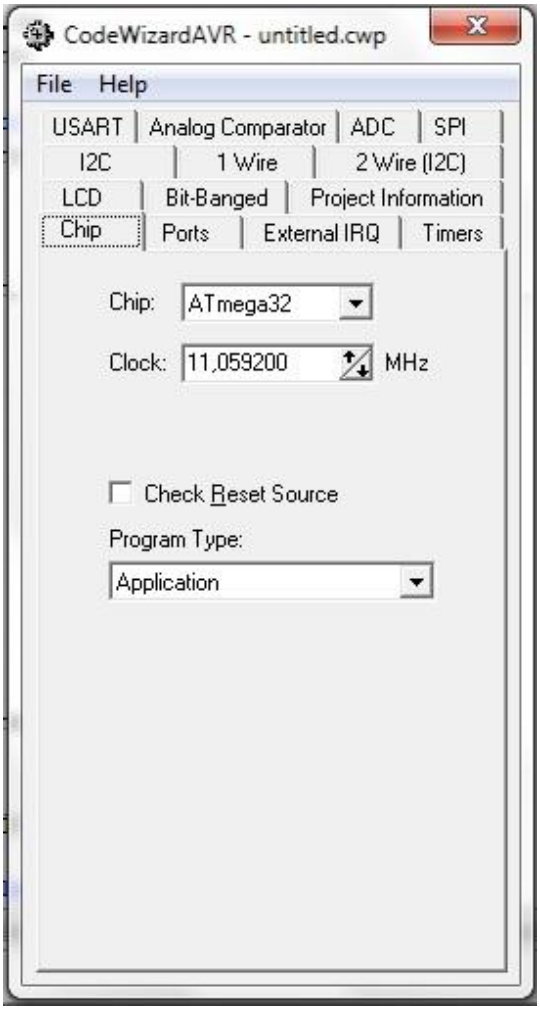
CodeVisionAVR merupakan salah satu jenis program yang dapat digunakan untuk memprogram semua jenis mikrokontroler dari keluarga AVR. Program ini menggunakan bahasa C sebagai bahasa pemrogramannya. Dalam tugas akhir ini, bahasa pemrograman yang digunakan adalah bahasa C.



Gambar 2.11 *Interface* CodeVisionAVR

Gambar di atas menunjukkan jendela *interface* dari program CodeVisionAVR, dimana pada *interface* tersebut terbagi dalam beberapa bagian yaitu *toolbar*, *messages*, *navigator* dan jendela tempat penulisan program.

CodeVisionAVR menyediakan suatu fasilitas yang bernama Code Wizard AVR, dimana fasilitas ini mempermudah dalam pemilihan jenis mikrokontroler, serta pengaktifan fasilitas-fasilitas dari mikrokontroler seperti *timer*, LCD, *input/output*, *external* IRQ, dan lain-lainnya. Gambar di bawah ini menunjukkan fasilitas dari Code Wizard AVR.



Gambar 2.12 Code Wizard AVR

Keterangan *tool* pada *toolbar* CodeVisionAVR dapat dilihat pada tabel berikut :

Tabel 2.3 Keterangan *tool* pada *toolbar* CodeVisionAVR

Icon	Nama	Fungsi
	Create New File	Membuat *file* baru
	Open File	Menbuka *file*
	Print File	Mencetak *file*
	Run the CodeWizardAVR	Menjalankan fasilitas CodeWizardAVR
	Check Syntax	Memeriksa kesalahan penulisan bahasa pemrograman
	Run the Chip Programmer	Menjalankan fasilitas Chip Programmer
	Make the Project	Men-*download* program ke mikrokontroler

2.10 Kontrol program

Keunggulan sebuah program terletak pada kontrol program ini.. Kontrol program dapat mengendalikan alur dari sebuah program dan menentukan apa yang harus dilakukan oleh sebuah program ketika menemukan suatu kondisi tertentu. Di dalam bahasa C terdapat beberapa kontrol program yang sering digunakan untuk menguji sebuah kondisi (pemilihan), perulangan dan peloncatan.

1) Statemen if

Dengan pernyataan ini kita dapat menguji sebuah kondisi tertentu dan kemudian menentukan tindakan yang sesuai dengan kondisi yang diinginkan. Penggunaan statemen/pernyataan if dapat kita klasifikasikan ke dalam tiga bagian yaitu untuk pemilihan yang didasarkan atas satu kondisi, dua kondisi, dan lebih dari dua kondisi.

– Untuk satu kondisi

Bentuk umum dari statemen if untuk satu kodisi adalah:

```
if(kondisi){
      tindakan_yang_akan_dieksekusi1;
      tindakan_yang_akan_dieksekusi2;
}
```

– Untuk dua kondisi

Bentuk umum dari statemen if untuk dua kondisi adalah:

```
if(kondisi){
      tindakan_jika_kondisi_benar1;
      tindakan_jika_kondisi_benar2;
}
else{
      tindakan_jika_kondisi_salah1;
      tindakan_jika_kondisi_salah2;
}
```

– Untuk lebih dari dua kondisi

Bentuk umum dari statemen if untuk lebih dari dua kondisi adalah:

```
if(kondisi1){
      tindakan_jika_kondisi1_terpenuhi;}
else if(kondisi2){
      tindakan_jika_kondisi2_terpenuhi;
}
else{
      tindakan_alternatif;
}
```

2) Statemen switch

Statemen switch digunakan untuk melakukan pemilihan terhadap kondisi yang memiliki nilai-nilai konstan. Oleh karena itu kondisi yang didefenisikan harus menghasilkan nilai yang bertipe bilangan bulat atau karakter. Untuk mendefenisikan nilai-nilai konstan tersebut adalah dengan menggunakan kata kunci case.

Adapun bentuk umum dari statemen switch adalah:

```
switch(kondisi){
      case nilai_konstan1:
      {
            tindakan_yang_akan_dieksekusi;
            break;
      }
      case nilai_konstan2:
      {
            tindakan_yang_akan_dieksekusi;
            break;
      }
      default:
      {
            tindakan_alternatif;
            break;
      }
}
```

3) Statemen for

Statemen for ini digunakan untuk menuliskan jenis pengulangan yang banyaknya sudah pasti atau telah diketahui sebelumnya. Oleh karena itu kita harus melakukan inisialisasi nilai untuk kondisi awal pengulangan dan juga harus menuliskan kondisi untuk menghentikan proses pengulangan.

Adapun bentuk umum dari statemen switch adalah:

```
for(kondisi_awal;batas_akhir;proses){
      tindakan_yang _akan_diulang1;
      tindakan_yang _akan_diulang2;
}
```

Kata proses pada bentuk umum di atas memiliki arti bahwa proses perubahan kondisi awal ke kondisi akhir bisa berupa bertambah dan juga bisa berupa berkurang.

4) Statemen while

Pada pengulangan jenis ini, kondisi akan diperiksa dibagian awal. Hal ini tentu menyebabkan kemungkinan bahwa apabila ternyata kondisi yang kita defenisikan tidak terpenuhi, maka proses pengulangan pun tidak akan dilakukan lagi.

Bentuk umum dari statemen while adalah:

```
while(kondisi){
      tindakan_yang_akan_diulang1;
      tindakan_yang_akan_diulang2;
}
```

Jika pengulangan yang diinginkan tanpa ada batasan, maka kata kondisi pada bentuk umum while harus diganti dengan 1 yang berarti bahwa kondisi statemen while selalu terpenuhi.

5) Statemen do-while

Berbeda dengan statemen while dimana kondisinya terletak di awal blok pengulangan, pada statemen do-while kondisi diletakkan di akhir blok pengulangan. Hal ini menyebabkan bahwa statemen yang terdapat di dalam blok pengulangan ini pasti akan dieksekusi minimal satu kali, walaupun kondisinya bernilai salah sekali pun. Maka dari itu statemen do-while ini banyak digunakan untuk kasus-kasus pengulangan yang tidak mempedulikan benar atau salahnya kondisi pada saat memulai proses pengulangan.

Bentuk umum statemen do-while adalah :

```
do{
      tindakan_yang_akan_diulang1;
      tindakan_yang_akan_diulang2;
}
while(kondisi);
```

Jika pengulangan yang diinginkan tanpa ada batasan, maka kata kondisi pada bentuk umum do-while harus diganti dengan 1 yang berarti bahwa kondisi statemen while selalu terpenuhi.

6) Statemen break

Statemen “break” digunakan untuk menghentikan sebuah pengulangan dan program akan langsung meloncat ke statemen yang berada di bawah blok

pengulangan. Ini biasanya dilakukan karena alasan efisiensi program, yaitu untuk menghindari proses pengulangan yang sebenarnya sudah tidak diperlukan lagi.

```
while(kondisi){
        tindakan_yang_akan_diulang1;
        tindakan_yang_akan_diulang2;
        break;
}
```

2.11 Flow Chart

Menurut Muhammad Ridwan Sidiq Bahas dalam jurnalnya yang berjudul "Aplikasi Absensi Mengajar Guru Pada SMA NEGERI 11 MAKASAR " Pengertian Flowchart adalah penggambaran secara grafik dari langkah-langkah dan urut-urutan prosedur dari suatu program. Flowchart menolong analis dan programmer untuk memecahkan masalah kedalam segmen-segmen yang lebih kecil dan menolong dalam menganalisis alternatif-alternatif lain dalam pengoperasian. Flowchart biasanya mempermudah penyelesaian suatu masalah khususnya masalah yang perlu dipelajari dan dievaluasi lebih lanjut.

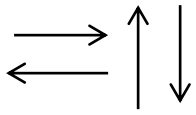
Bila seorang analis dan programmer akan membuat flowchart, ada beberapa petunjuk yang harus diperhatikan, seperti:

1. Flowchart digambarkan dari halaman atas ke bawah dan dari kiri ke kanan
2. Aktivitas yang digambarkan harus didefinisikan secara hati-hati dan definisi ini harus dapat dimengerti oleh pembacanya.
3. Kapan aktivitas dimulai dan berakhir harus ditentukan secara jelas.
4. Setiap langkah dari aktivitas harus berada pada urutan yang benar.
5. Lingkup dan range dari aktivitas yang sedang digambarkan harus ditelusuri dengan hati-hati. Percabangan-percabangan yang memotong aktivitas yang sedang digambarkan tidak perlu digambarkan pada flowchart yang sama. Symbol konektor harus digunakan dan percabangannya diletakkan pada halaman yang terpisah atau hilangkan seluruhnya bila percabangannya tidak berkaitan dengan system.
6. Gunakan symbol-symbol flowchart yang standar.

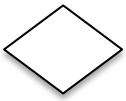
2.11.1 Simbol-simbol Flow Chart

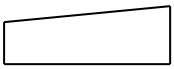
Simbol – simbol yang dipakai dalam flowchart dibagi menjadi 3 kelompok yaitu :

1. Flow Direction Simbols
   - Digunakan untuk menghubungkan symbol satu dengan yang lain
   - Disebut juga connecting line.

	Simbol *arus* / *flow*, yaitu menyatakan jalannya arus suatu proses
	Simbol *communication link*, yaitu menyatakan transmisi data dari satu lokasi ke lokasi lain
	Simbol *connector* berfungsi menyatakan sambungan dari proses ke proses lainnya dalam halaman sama
	Simbol *offline connector*, menyatakan sambungan dari proses ke proses lainnya dalam halaman berbeda

2. Processing Simbols
   - Menunjukkan jenis operasi pengolahan dalam suatu proses atau prosedur

	Simbol *process*, yaitu menyatakan suatu tindakan (proses) yang dilakukan oleh komputer
	Simbol manual, yaitu menyatakan suatu tindakan (proses) yang tidak dilakukan oleh komputer
	Simbol *decision* , yaitu menunjukkan suatu kondisi tertentu yang menghasilkan dua kemungkinan jawaban : ya/tidak
	Simbol *predefined process*, yaitu menyatakan penyediaan tempat penyimpanan suatu pengolahan untuk memberi harga awal

	Simbol terminal, yaitu menyatakan permulaan atau akhir suatu program
	Simbol *keying operation*, menyatakan segala jenis operasi yang diproses dengan menggunakan suatu mesin yang mempunyai *keyboard*
	Simbol *off-line storage*, menunjukkan bahwa data dalam symbol ini akan disimpan ke suatu media tertentu
	Simbol manual *input*, memasukkan data secara manual dengan menggunakan *offline keyboard*

3. Input / Output Simbols
   - Menunjukkan jenis peralatan yang digunakan sebagai media input atau output.

	Simbol *input / output*, menyatakan proses *input* atau *output* tanpa tergantung jenis peralatannya
	Simbol *punched card*, menyatakan input berasal dari kartu atau *output* ditulis ke kartu
	Simbol *magnetic tape*, menyatakan input berasal dari pita magnetis atau output disimpan ke pita magnetis
	Simbol *disc storage*, menyatakan pita berasal dari *disk* atau *output* disimpan ke *disk*
	Simbol *document*, mencetak keluaran dalam bentuk dokumen melalui printer
	Simbol *display*, mencetak keluaran dalam layar monitor