BAB II
TINJAUAN PUSTAKA

2.1 RFID (*Radio Frequency Identification*)

Identifikasi suatu objek sangat erat hubungannya dengan mengambil data. Salah satu metoda identifikasi yang dianggap paling menguntungkan adalah *auto*-ID atau *automatic Identification*. Yaitu, metoda pengambilan data dengan identifikasi objek secara otomatis tanpa ada keterlibatan manusia. Auto-ID bekerja secara otomatis sehingga dapat meningkatkan efesiensi dan mengurangi kesalahan dalam memasukkan data. Karena Auto-ID tidak memerlukan tenaga manusia dalam operasinya, tenaga manusia yang ada dapat difokuskan pada bidang lain. *Barcode, smart, card, voice recognition* (OCR) dan *Radio Frequency Identification* (RFID) merupakan teknologi yang menggunakan metoda auto-ID. *Radio Frequency Identification* atau yang lebih dikenal dengan RFID merupakan suatu metoda identifikasi objek yang menggunakan gelombang radio. Proses identifikasi dilakukan oleh RFID *reader* dan RFID *transponder* (RFID *tag*). RFID *tag* diletakkan pada suatu benda atau suatu objek yang akan diidentifikasi. Tiap-tiap RFID *tag* memiliki data angka identifikasi (ID *number*) yang unik, sehingga tidak ada RFID *tag* yang memiliki ID *number* yang sama (Pratama, 2009: 6).

2.1.1 Sistem RFID

Secara umum, sistem RFID terdiri dari 2 bagian, yaitu:

1. **RFID *Tag***

   RFID transponder atau RFID *tag* terdiri dari *chip* rangkaian sirkuit yang terintegrasi dan sebuah antena. Rangkaian elektonik dari RFID *tag* umumnya memiliki memori. Memori ini memungkinkan RFID *tag* mempunyai kemampuan untuk menyimpan data. Memori pada *tag* dibagi berdasarkan frekuensi radio, RFID *tag* digolongkan mejadi:

   1. *Low frequency tag* (125 KHz – 134 KHz)
   2. *High frequency tag* (13,56 MHz)
   3. *Ultra high frequency tag* (868 Mhz- 956 MHz)

4. *Microwave tag* (2,45 GHz)

Untuk lebih jelasnya perbedaan dari *tag* aktif dan *tag* pasif dapat dilihat pada tabel dibawah ini :

5

Tabel 2.1 perbedaan kartu tag aktif dan kartu tag pasif

Jenis Kartu Tag	Spesifikasi
Tag Aktif	1. Read and write (dapat dibaca dan ditulis/diisi dengan program) 2. Memiliki internal baterai/catu daya sendiri 3. Dapat bekerja pada frekuensi tinggi sehingga RFID reader hanya membutuhkan daya yang kecil. Contohnya : Katu tag aktif bisa dijumpai pada kehidupan sehari-hari, seperti : Kartu ATM, e-KTP, dan SmartCard pada Bis Trans Musi.
Tag Pasif	1. Read Only (hanya di program pada saat tag dibuat,data dan kode tidak dapat diubah sama sekali) 2. Daya pada tag pasif didapat dari RFID reader 3. Hanya bekerja pada frekuensi rendah yaitu sekitar (125 kHz-134kHz) sehingga RFID reader memerlukan daya yang lebih besar untuk membantu tag ini. Contohnya : Kartu tag pasif biasanya digunakan untuk keperluan pendidikan, seperti pada tugas akhir ini.

(Pratama, 2009: 6)

RFID *tag* terdiri dari dua bagian, yaitu:

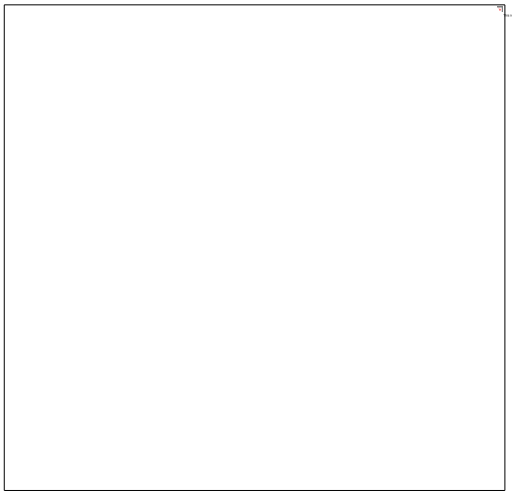
1. Inlay

Inlay merupakan bagian dari inti RFID *tag*, yang terdiri dari *chip* dimana informasi disimpan dan antena. Informasi yang dismpan terdiri dari :

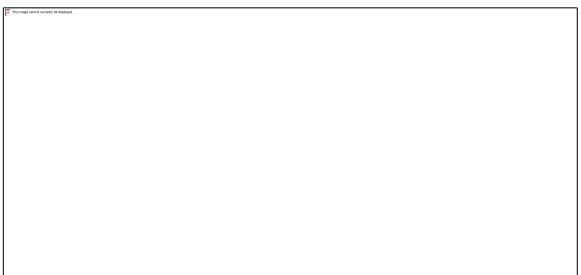
1. Informasi permanen yang berisi ID yang unik dari *tag* tersebut, sehingga setiap *tag* memiliki ID yang berbeda satu sama lainnya. Informasi juga tidak bisa diubah oleh aplikasi atau memakai RFID *reader*.
2. Informasi *non*-permanen yang dapat ditulis oleh aplikasi dengan bantuan RFID *reader* saat pengoperasian dilapangan.
3. *Inlay* ini berbentuk kecil, "halus", dan bentuknya mudah rusak, sehingga tidak praktis untuk pemakaian lapangan, sehingga RFID yang digunakan dilapangan selalu dalam bentuk *encapsulated*.

4. *Encapsulation*/ Bungkus *inclay*

Karena bentuk *inlay* yang rapuh, maka secara praktis perlu dibungkus sehingga sesuai dengan kondisi lapangan dimana RFID *tag* dipakai. Pemakaian *encapsulation* dapat disesuaikan dengan lingkungan yang exstrim, seperti temperatur maupun kelembapan yang tinggi. (Doni dkk. 2010: 17)



Gambar 2.1 RFID *Tag (Keychain)*



Gambar 2.2 RFID *Tag (Card Tag)*

Tugas akhir ini menggunakan modul RFID *reader* yang khusus mendeteksi RFID *tag* pasif dengan frekuensi rendah, RFID *tag* yang kompatibel dengan modul RFID *reader* ini adalah tipe GK4001 atau EM4001. Gambar 2.1 dan 2.2 memperlihatkan RFID *tag* akan digunakan. Tabel 2.2 memperlihatkan spesifikasi dari RFID *tag* tipe GK4001 atau EM4001.

Tabel 2.2 Spesifikasi RFID *tag* GK4001/EM4001

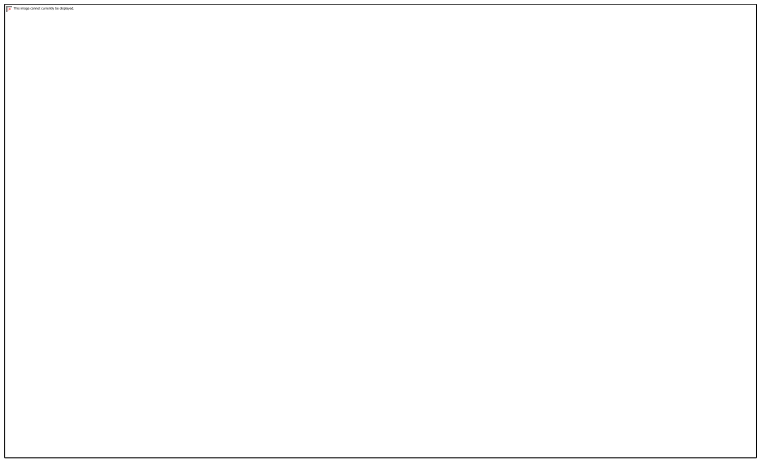
Parameter	Spesifikasi
Frekuensi	125 KHz
Jangkauan baca	Sampai 2 cm
Dimensi	86 x 54 x 1.9 mm
Kapasitas Data	1. Bit

(Pratama, 2009: 07)

2. RFID *Reader*

RFID *Reader* merupakan penghubung antara *software* aplikasi dengan antena yang akan meradiasikan gelombang radio ke RFID *tag*, RFID *reader* akan membaca ID *number* dan aplikasinya disimpan oleh RFID *Tag*. RFID *reader* harus kompatibel dengan RFID *tag* agar RFID *tag* dapat dibaca. Gelombang radio yang ditransmisikan oleh antena berpropagasi pada ruangan disekitarnya. Akibatnya data dapat berpindah secara *wireless* ke *tag* RFID yang berada berdekatan

dengan antena. ID-12 merupakan *reader* yang khusus mendeteksi RFID *tag* frekuensi 125 kHz. (Rahmat, 2010 : 9).

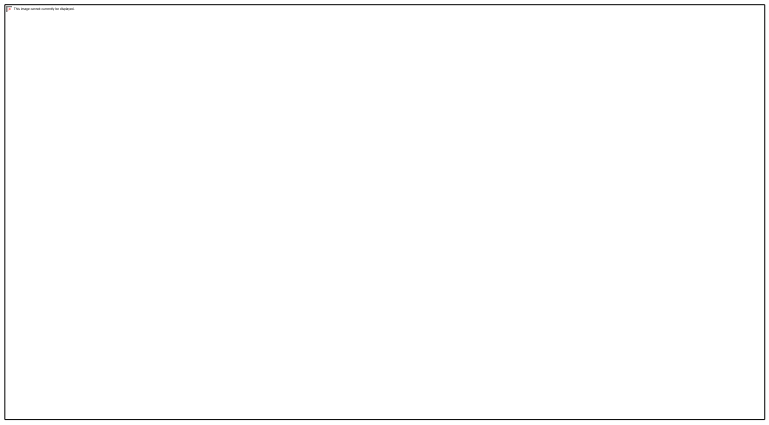


Gambar 2.3 RFID *Reader* ID-12

Tabel 2.3 Spesifikasi modul RFID *reader* ID 12

Parameter	RFID ID-12
Jarak Baca	Sampai 2 cm
Dimensi	26mm x 25mm x 7mm
Frekuensi	125 KHz
Format Kartu	GK4001 atau yang compatible
Encoding	Manchester 64-bit, modulus 64
Jenis Catu daya	5VDC pada 30 mA nominal
Arus	Output I/O -
Jangkauan Catu daya	+4.6 – 5.4 V

(Ahson dan Ilyas, 2008 : 7)



Gambar 2.4 Spesifikasi pin pada ID2/ ID12/ID20

(Ahson dan Ilyas, 2008 : 8)

2.1.2 Cara Kerja Perpindahan Data Pada RFID Reader

Perpindahan data yang terjadi ketika sebuah *tag* didekatkan pada sebuah *reader* dikenal sebagai *coupling*. Perbedaan frekuensi yang digunakan oleh RFID *tag* aktif dengan RFID *tag pasif* menyebabkan perbedaan metode perpindahan data yang digunakan pada kedua *tag* tersebut. Perpindahan data pada RFID *tag pasif* menggunakan metode *backcatter coupling*. *Inductive coupling* terjadi pada frekuensi rendah.

Ketika medan gelombang radio dari *reader* didekati oleh *tag pasif*, koil antena yang terdapat pada *tag pasif* ini akan membentuk suatu medan magnet. Medan magnet ini akan menginduksi suatu tegangan listrik yang memberi tenaga pada *tag pasif*. Pada saat yang sama akan terjadi suatu tegangan jatuh pada beban *tag*. Tegangan jatuh adalah tegangan yang digunakan pada beban dan ditimbulkan oleh arus yang mengalir melalui tahanan kawat, tegangan ini akan terbaca oleh *reader* dan berlaku sebagai amplitudo modulasi untuk bit data. Ilustrasi untuk *inductive coupling* diberikan oleh gambar 2.5 berikut ini :

Gambar 2.5 *Inducting Coupling*

(Mardiyono dkk, 2011: 8)

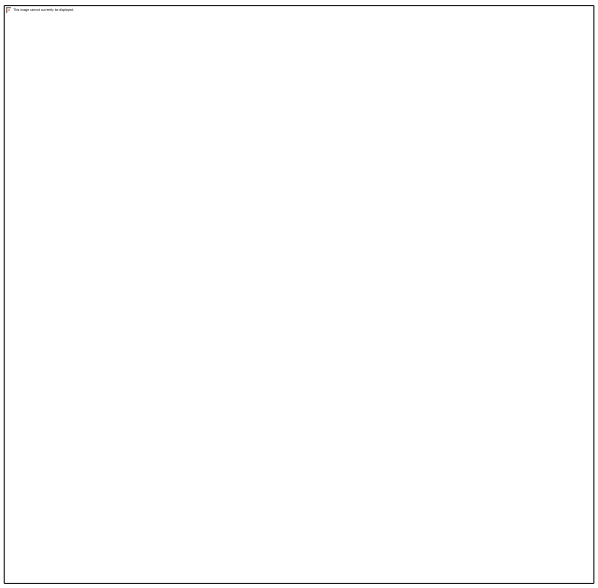
2.2 Mikrokontroller

Mikrokontroler adalah sebuah sistem komputer lengkap dalam satu serpih *(chip).* Mikrokontroler lebih dari sekedar sebuah mikroprosesor karena sudah terdapat atau berisikan ROM (*Read-Only Memory*), RAM (*Read-Write Memory*), beberapa Port masukan maupun keluaran, dan beberapa peripheral seperti pencacah/pewaktu, ADC (*Analog to Digital converter*), DAC (*Digital to Analog converter*) dan serial komunikasi. Salah satu mikrokontroler yang banyak digunakan saat ini yaitu mikrokontroler AVR. AVR adalah mikrokontroler RISC (*Reduce Instuction Set Computer*) 8 bit berdasarkan arsitektur *Harvard*. Secara umum mikrokontroler AVR dapat dapat dikelompokkan menjadi 3 kelompok, yaitu keluarga AT90Sxx, ATMega dan ATtiny. Pada dasarnya yang membedakan masing-masing kelas adalah memori, peripheral, dan fiturnya Seperti mikroprosesor pada umumnya, secara internal mikrokontroler ATMega16 terdiri atas unit-unit fungsionalnya *Arithmetic and Logical Unit* (ALU), himpunan register kerja, register dan dekoder instruksi, dan pewaktu beserta komponen kendali lainnya. Berbeda dengan mikroprosesor, mikrokontroler menyediakan

memori dalam serpih yang sama dengen prosesornya (*in chip*). (Radianujinugraha, 2008: 1)

2.2.1 Mikrokontroler ATMega8 AVR

Mikrokontroler ATMega8 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* atau *counter* fleksibel dengan *mode compare*, *interrupt* internal dan eksternal, serial USART, Programmable Watchdog Timer, dan *mode power saving.* Beberapa diantaranya mempunyai ADC dan PWM internal. AVR juga mempunyai *In-System Programmable Flash on-chip* yang mengijinkan memori program untuk diprogram ulang dalam sistem menggunakan hubungan serial SPI. ATMega8 adalah mikrokontroler CMOS 8-bit daya rendah berbasis arsitektur RISC yang ditingkatkan. Kebanyakan instruksi dikerjakan pada satu siklus *clock,* ATMega8 mempunyai *throughput* mendekati 1 MPS per MHz membuat disain dari sistem untuk mengoptimasi konsumsi daya versus kecepatan proses. Susunan pin-pin dari IC mikrokontroler ATMega8 diperlihatkan pada gambar dibawah ini. IC ini tersusun dari 28 pin yang memiliki beberapa fungsi tertentu.



Gambar 2.6 Susunan Pin Mikrokontroller ATMega8

(Bagus, 2012: 18)

Penggunaan rangkaian mikrokontroler ATMega8 ada dua pilihan,dengan menggunakan *board* ATMega8 *develompment board* yang sudah ada dipasaran atau dengan membuat sendiri rangkaian mikrokontroler tersebut. Jika menggunakan rangkaian mikrokontroler yang sudah tersedia dipasaran maka akan mempersingkat waktu pembuatan sistem, karena hanya tinggal membeli rangkaian berupa kit dan hanya tinggal menggunakannya. *Chip* yang dijelaskan di sini menggunakan kemasan PDIP, untuk kemasan yang lain (TQPF,QFN atau MLF) tidak jauh berbeda. Untuk lebih jelasnya silahkan merujuk ke data sheet. Nama-nama pin di atas usahakan lebih sering dikenal, hal ini berguna untuk penggunaan *pheripheral internal*.

ATMega8 memiliki 28 pin yang masing-masing pin nya memiliki fungsi yang berbeda-beda baik sebagai port ataupun sebagai fungsi yang lain. Berikut akan dijelaskan tentang kegunaan dari masing-masing kaki pada ATMega8.

1. VCC

   Merupakan *supply* tegangan untuk digital

2. GND

   Merupakan *ground* untuk smua komponen yang membutuhkan *grounding*

3. Port B

   Adalah 8 buah pin mulai dari pin B.0 sampai dengan pin B.7. Tiap pin dapat digunakan sebagai input dan juga *output*. Port B merupakan sebuah 8-bit bit-directional I/O port dengan *internal pull-up* resistor. Sebagai input, pin-pin yang terdapat pada port B yang secara eksternal diturunkan, maka akan mengeluarkan arus jika *pull-up* resistor diaktifkan. Jika ingin menggunakan tambahan kristal, maka cukup untuk menghubungkan kaki dari kristal ke kaki pada pin port B. Namun jika tidak digunakan, maka cukup untuk dibiarkan saja. Pengguna kegunaan dari masing-masing kaki ditentukan dari *clock fuse setting*-nya.

4. Port C

Port C merupakan sebuah 7-bit bi-directional I/O yang di dalam masing-masing pin terdapat *pull-up* resistor. Jumlah pin-nya hanya 7 buah mulai dari C.0 sampai dengan pin C.6. Sebagai keluaran atau *output*, port C memiliki karakteristik yang sama dalam hal kemampuan menyarap arus (*sink*) ataupun mengeluarkan arus (*source*).

5. Reset / PC6

Jika RSTDISBL *Fuse* diprogram, maka PC6 akan berfungsi sebagai pin I/O. Untuk diperhatikan juga bahwa pin ini memiliki karakteristik yang berbeda dengan pin-pin yang tedapat pada port C. Namun jika RSTDISBL *Fuse* tidak diprogram, maka pin ini akan berfungsi sebagai *input reset*. Dan jika level tegangan yang masuk ke pin ini rendah dan pulsa yang ada lebih pendek dari pulsa minimum, makan akan menghasilkan suatu kondisi reset meskipun *clock*-nya tidak berkerja.

6. Port D

Port D merupakan 8-bit bi-directional I/O *dengan internal pull-up* resistor. Fungsi dari port ini sama dengan port-port yang lain. Hanya saja pada port ini tidak terdapat kegunaan-kegunaan yang lain. Pada port ini hanya berfungsi sebagai masukan dan keluaran saja atau biasa disebut dengan I/O.

7. AVCC

Pada pin ini memiliki fungsi sebagai *power supply* tegangan untuk ADC. Untuk pin ini harus dihubungkan secara terpisah dengan VCC karena pin ini digunakan untuk analog saja. Bahkan jika ACD pada AVR tidak digunakan, tetap saja disarankan untuk menghubungkan secara terpisah dengan VCC. Cara menghubungkan AVCC adalah melewati *low-pass filter* setelah itu dihubungkan dengan VCC.

8. AREF

Merupakan pin referensi analog jika menggunakan ADC. Pada AVR status Register mengandung beberapa informasi mengenai hasil dari kebanyakan hasil eksekusi intruksi aritmatik. Informasi ini dapat digunakan untuk altering arus program sebagai kegunaan untuk meningkatkan performa pengoperasian. Perlu

diketahui bahwa register ini di-*update* setelah semua operasi ALU (*Arithmetic Logic Unit*). Hal tersebut seperti yang telah tertulis dalam data sheet khususnya pada bagian *Intruction Set Reference*. Dalam hal ini untuk beberapa kasus dapat membuang kebutuhan penggunaan instruksi perbandingan yang telah didedikasikan serta dapat menghasilkan peningkatan dalam hal kecepatan dan kode yang lebih sederhana dan singkat. Register ini tidak secara otomatis tersimpan ketika memasuki sebuah rutin interupsi dan juga ketika menjalankan sebuah perintah setelah kembali dari interupsi. Namun hal iini harus dilakukan melalui *software*.

9. Bit 7 (1)

   Merupakan bit Global *Interrupt Enable.* Bit ini harus di-*set* supaya semua perintah interupsi dapat dijalankan. Untuk fungsi interupsi individual akan dijelaskan pada bagian yang lain. Jika bit ini di-*reset*, maka semua perintah interupsi baik yang secara individual maupun yang secara umum akan diabaikan. Bit ini akan dibersihkan atau *cleared* oleh *hardware* setelah sebuah interupsi dijalankan dan akan di-*set* kembali oleh perintah RETI. Bit ini juga dapat di-*set* dan di-*reset* melalui aplikasi dengan instruksi SEI dan CLI.

10. Bit 6 (T)

    Merupakan bit *Copy Storage*. Instruksi bit *Copy Instruction* BLD (*Bit LoaD*) dan BST (*Bit Store*) menggunakan bit ini sebagai asal atau tujuan untuk bit yang telah dioperasikan. Sebuah bit dari sebuah register dan *Register File* dapat disalin ke dalam bit ini dengan menggunakan intruksi BST, dan sebuah bit di dalam bit ini dapat disalin ke dalam sebuah bit di register pada Register File dengan menggunakan perintah BLD.

11. Bit 5 (H)

    Merupakan bit *Half Carry Flag*. Bit ini menandakan sebuah *Half Carry* dalam beberapa operasi aritmatika. Bit ini berfungsi dalam aritmatik BCD

12. Bit 4 (S)

Merupakan *Sign* bit. Bit ini selalu merupakan sebuah eksklusif di antara *Negative Flag* (N) dan *Two's Complement Overflow Flag (V).*

13. Bit 3 (V)

    Merupakan bit *Two's Complement Overflow Flag.* Bit ini menyediakan fungsi aritmatika dua komplemen.

14. Bit 2 (N)

    Merupakan bit *Negative Flag*. Bit ini menyediakan sebuah hasil *negative* di dalam sebuah fungsi logika atau aritmatika.

15. Bit 1 (Z)

    Merupakan bit *Zero Flag*. Bit ini mengindikasikan sebuah hasil nol " 0 " dalam sebuah fungsi aritmatika atau logika.

16. Bit 0 (C)

    Merupakan bit *Carry Flag*. Bit ini mengindikasikan sebuah *Carry* atau sisa dalam sebuah fungsi aritmatika atau logika. (Bagus, 2012: 21)

2.3 *Basic Complier*

Compiler adalah suatu program yang menerjemahkan bahasa program kedalam bahasa objek (*obyek code*). *Compiler* menggabungkan keseluruhan bahasa program, mengumpulkannya dan kemudian menyusunnya kembali. *complier* memerlukan waktu untuk membuat suatu program dapat di eksekusi oleh komputer, program yang dieksekusi oleh *compiler* adalah dapat berjalan lebih cepat disbanding program yang diperoduksi oleh *interpreter,* disamping itu juga bersifat independen. Contoh program yang menggunakan compiler adalah *Visual Basic, Visual Delvi, dan Pascal*. Tahap Visualisasi yaitu :

1. Pertama *source code* (program yang ditulis) dibaca kememori computer).

2. *Source code* tersebut diubah menjadi objek code (bahasa *Assembly).*
3. *Objek code* di hubungkan dengan liberary yang dibutuhkan untuk membentuk file yang bisa dieksekusi.

2.3.1 Tipe Data

Tipe data merupakan bagian program yang paling penting karena sangat berpengaruh setiap instruksi yang akan dilaksanakan oleh komputer pada program. Pemilihan tipe data yang tepat maka operasi data menjadi lebih efeisian dan efektif. (Iswanto, 2009:31)

Tabel 2.4 Tipe data pada Bascom AVR

No	Tipe	Jangkauan
1	*Bit*	0 atau 1
2	*Byte*	0 – 255
3	*Integer*	-32,768 –32,767
4	*Word*	0 – 65535
5	*Long*	-2147483648 – 2147483647
6	*Single*	1.5 x 10^–45 – 3.4 x 10^38
7	*Double*	5.0 x 10^–324 to 1.7 x 10^308

(Sumber : Iswanto, 2009: 31)

2.3.2 Konstanta

Konstanta merupakan suatu nilai yang tidak dapat diubah selama proses program berlangsung. Nilai konstanta selalu tetap. Konstanta harus didefinisikan terlebih dahulu di awal program. Konstanta dapat bernilai integer, pecahan, karakter ataupun string. Contoh konstanta : A=5 c=1.1.
(Iswanto, 2009: 31)

2.3.3 Variabel

Variabel adalah suatu pengenal (*identifier)* yang digunakan untuk mewakili suatu nilai tertentu di dalam proses program yang dapat diubah-ubah sesuai dengan kebutuhan. Nama dari *variable* terserah sesuai dengan yang diinginkan namun hal yang terpenting adalah setiap variabel diharuskan :

1. Terdiri dari gabungan huruf dan angka dengan karakter pertama

harus berupa huruf, max 32 karakter.

2. Tidak boleh mengandung spasi atau symbol-simbol khusus seperti : *$, ?, %, #, !,* &, *, (,),* -, +, = dan lain sebagainya kecuali *underscore*.
3. Deklarasi, sangat diperlukan bila akan menggunakan pengenal (*identifier*) dalam suatu program. (Fahmiza, 2010: 1)

1. Operator

2.3.4.1 Operator Penugasan

Operator Penugasan (*Assignment operator*) dalam Bahasa Basic berupa "=".

2.3.4.2 Operator Aritmatika

* : untuk perkalian

/ : untuk pembagian

\+ : untuk pertambahan

\- : untuk pengurangan

% : untuk sisa pembagian (modulus)

1. **Operator Hubungan (Perbandingan)**

Operator hubungan digunakan untuk membandingkan hubungan dua buah operand atau sebuah nilai / variable, misalnya :

= '*Equality* X = Y

< '*Less than* X < Y

> '*Greater than* X > Y

<= '*Less than or equal to* X <= Y

>= '*Greater than or equal to* X >= Y

2. **Operator Logika**

Operator logika digunakan untuk membandingkan logika hasil dari operator-operator hubungan. Operator logika ada empat macam, yaitu :

NOT '*Logical complement*

AND '*Conjunction*

OR ‘*Disjunction*

XOR ‘*Exclusive or*

3. **Operator *Bitwise***

Operator *bitwise* digunakan untuk memanipulasi bit dari data yang ada di memori. Operator bitwise dalam Bahasa Basic :

Shift A, Left, 2 : Pergeseran bit ke kiri

Shift A, Right, 2 : Pergeseran bit ke kanan

Rotate A, Left, 2 : Putar bit ke kiri

Rotate A, right, 2 : Putar bit ke kanan

2.3.5 Pernyataan Kondisional

Pernyataan ini digunakan untuk melakukan pengambilan keputusan terhadap dua buah bahkan lebih kemungkinan untuk melakukan suaatu blok pernyataan atau tidak. Konstruksi penulisan pernyataan

IF-THEN-ELSE-END IF pada bahasa BASIC sebagai berikut:

IF **pernyataan kondisi 1** *THEN*

‘blok pernyataan 1 yang dikerjakan bila kondisi 1 terpenuhi

IF **pernyataan kondisi 2** *THEN*

‘blok pernyataan 2 yang dikerjakan bila kondisi 2 terpenuhi

IF **pernyataan kondisi 3** *THEN*

‘blok pernyataan 3 yang dikerjakan bila kondisi 3 terpenuhi

Setiap penggunaan pernyataan *IF-THEN* harus diakhiri dengan perintah *END IF* sebagai akhir dari pernyatan kondisional (Iswanto, 2009: 31)

2. Visual Basic .NET

2.4.1 Sejarah VB .NET

Pada zaman dahulu ada sebuah bahasa pemograman yang diberi nama Basc (*Beginner’s All-purpose Symbolic Instruction Code*). Sesuai dengan namanya, *basic* ditujukan sebagai bahasa yang paling sederhana bagi mereka yang tidak terlalu familiar dengan dunia pemograman.

Pada tahun 1991 *Microsoft* mengeluarkan Visual Basic, pengembangan dari Basic yang berubah dari sisi pembuatan antarmukanya. *Visual Basic* sampai sekarang masih menjadi salah satu bahasa pemograman terpopuler di dunia.

Pada akhir tahun 1999, Teknologi tersebut sebagai *platform* untuk membangun XML *Web Services*. XML *Web Services* memungkinkan aplikasi tipe apapun dapat berjalan pada system komputer dengan tipe manapun dapat mengambil data yang tersimpan pada s*erver* dengan tipe apapun melalui internet. (Priyanto, 2014: 3)

2.4.2 Pengertian Visual Basic .NET

Visual basic .NET adalah Visual Basic yang direkayasa kembali untuk digunakan pada platform.NET sehingga aplikasi yang dibuat menggunakan Visual Basic.NET dapat berjalan pada sistem komputer apa pun, dan dapat mengambil data dari *server* dengan tipe apa pun asalkan terinstal .NET Framework.

Berikut ini perkembangan Visual Basic .NET :

1. Visual Basic .NET 2002 (VB 7.0)
2. Visual Basic .NET 2003 (VB 7.1)
3. Visual Basic 2005 (VB 8.0)
4. Visual Basic 2008 (VB 9.0)
5. Visual Basic 2010 (VB 10.0)
6. Visual Basic 2012 (VB 11.0)
7. Visual Basic 2013

Pada umumnya Visual Basic .NET terpaket dalam Visual Studio .NET. pada distribusinya, terdapat berbagai versi Visual Studio .NET yaitu versi Professional, Permium dan yang paling lengkap adalah versi Ultimate. Semua versi ini adalah versi berbayar dengan harga yang sangat mahal. (Priyanto, 2014: 3)

2.4.3 Kelebihan Visual Basic .NET

Aplikasi-aplikasi pemograman visual yang ada saat ini mempunyai kelebihan dan kelemahan masing-masing. Untuk suatu kasus, bisa jadi

menggunakan Delphi lebih bagus, tapi untuk kasus yang lain bisa jadi aplikasi VB .NET yang lebih baik. Namun, VB .NET layak untuk dijadikan pilihan karena mempunyai cukup banyak kelebihan.

Beberapa kelebihan VB .NET antara lain :

1. Sederhana dan mudah dipahami

Seperti pada VB, bahasa yang digunakan pada VB .NET sangat sederhana sehingga lebih mudah dipahami bagi mereka yang masih awam terhadap dunia pemograman.

2. Mendukung GUI

VB .NET bisa membuat *software* dengan antarmuka grafis yang lebih *user friendly*

3. Menyederhanakan *deployment*

VB .NET mengatasi masalah *deployment* dari aplikasi berbasis windows yaitu DLL Hell dan Registrasi COM (*Component object model*). Selain itu tersedia *wizard* yang memudahkan dalam pembuatan *file setup*.

4. Menyederhanakan pengembangan perangkat lunak

Ketika terjadi kesalahan penulisan kode dari sisi sintaks (bahasa), maka VB .NET langsung menuliskan kesalahannya pada bagian *message windows* sehingga *programmer* dapat memperbaiki kode dengan lebih cepat. Editor menu bersifat WYSIWYG (*What you see is what you get*). Adanya berbagai wizard yang memandu *programmer* dalam membuat *software*. Tersedianya *crystal report* (CR) untuk membuat laporan (pada Visual Studio 2012, *Crystal report* gratis namun harus diinstal secara terpisah). Adanya *code snippets* yaitu fitur untuk menyisipkan kode-kode koleksi kita pada program yang sedang kita buat. Di atas adalah hal-hal yang membuat pengembangan perangkat lunak menjadi lebih mudah.

5. Mendukung penuh OOP

Memiki fitur bahasa pemograman berorientasi objek seperti *inheritance* (pewarisan), *encapsulation* (pembungkusan), dan *polymorphism* (banyak bentuk).

6. Mempermudah pengembangan aplikasi berbasis WEB

Didesainkan *desainer form web*. Selain itu disediakan layanan web XML sehingga memungkinkan suatu aplikasi "berkomunikasi" dengan aplikasi lainnya dari berbagai *platform* menggunakan protokol internet terbuka.

7. Migrasi ke VB .NET dapat dilakukan dengan mudah

Jika anda sudah mengembangkan aplikasi di VB, maka konversi ke VB .NET dapat anda jalankan dengan mudah

8. Banyak digunakan oleh *programmer-programmer* di seluruh dunia. Salah satu keuntungannya dalah jika kita memiliki masalah atau pertanyaan, maka kita bisa tanyakan kepada *programmer-progammer* lain di seluruh dunia melalui forum-forum di internet. (Priyanto, 2014: 7)

1. IDE (*Integrated Development Environment*) Visual Basic 2010

Gambar 2.7 Visual Basic 2010

Secara rinci, tampilan layer pada Visual Basic 2010 seperti yang dapat dilihat pada gambar 2.7 adalah terdiri dari:

1. *Menu bar* adalah bagian dari IDE yang terdiri atas perintah-perintah untuk mengatur IDE, mengedit kode, mengeksekusi program. Di dalam menu bar, perintah-perintah dikelompokkan ke dalam beberapa baian sessuai jenis perintah tersebut. Menu bar pada Visual Studio 2010 terlihat pada gambar 2.8

Gambar 2.8 *Menu Bar*

Untuk menggunakan menu atau pilihan pada menu bar, kita tinggal mengklik pada menu atau pilihan yang akan dijalankan. Sebagi contoh, untuk membuat sebuah proyek baru, pilih menu File > New > Project.

Pada IDE visual studio 2010, terdapat sepuluh menu utama. Menu utama tersebut antara lain sebagai berikut :

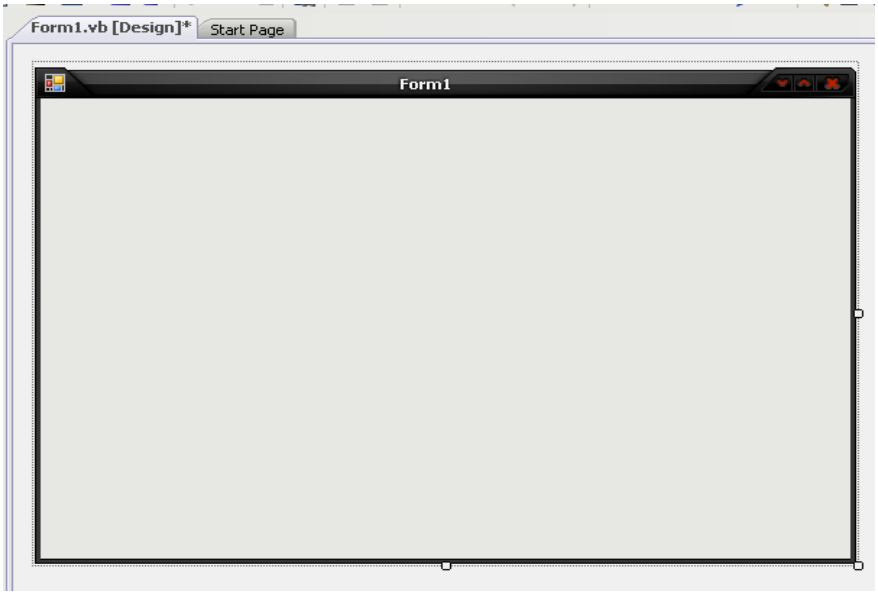
1. Menu **file** berisi perintah-perintah untuk membuat pproyek, membuka proyek, menutup proyek, mencetak data dari proyek dan lain-lain
2. Menu **edit** berisi perintah-perintah untuk *undo, cut, paste,* dan lain-lain
3. Menu **view** berisi perintah-perintah untuk menampilkan window-window dari IDE dan toolbar
4. Menu **project** berisi perintah-perintah untuk mengatur proyek dan file-file
5. Menu **build** berisi perintah-perintah untuk meng-*compile* program
6. Menu **debug** berisi perintah-perintah untuk men-debug dan menjalankan program
7. Menu **data** berisi perintah-perintah untuk menghubungkan dengan basis data
8. Menu **tools** berisi perintah-perintah untuk mengakses komponen IDE tambahan dan mengubah IDE

9. Menu **architecture** berisi tools-tools pemodelan data menggunakan er-diagram, UML, dsb
10. Menu **test** mengelola pengujian yang akan dilakukan pada proyek
11. Menu **analyze**, menganalisis kode untuk mengetahui performansinya
12. Menu **window** berisi perintah-perinbtah untuk mengatur dan menampilkan window
13. Menu **help** berisi perintah-perintah untuk mengakses fasilitas bantuan

14. *Toolbar* fungsinya sama seperti menu. Bedanya pada toolbar pilihan berbentuk icon. Untuk memilih suatu proses yang akan dilakukan, bagian toolbar terlihat seperti gambar 2.9. icon-icon pada toolbar adalah pilihan-pilihan pada menu yang sering digunakan dalam program aplikasi. Pada toolbar terdapat beberapa tombol untuk window dari IDE, Visual Basic .NET, antara lain solution Explorer, properties, dan toolbox

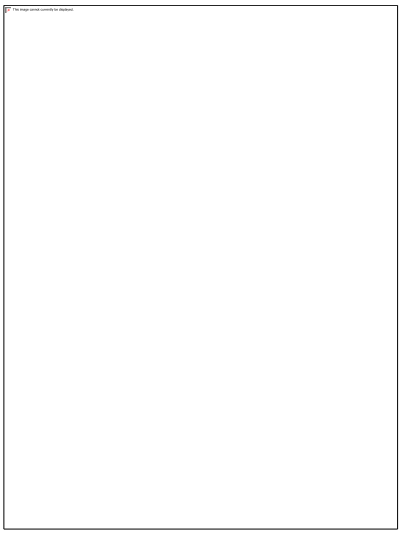
Gambar 2.9 *Toolbar*

15. *Form* merupakan tempat diman kontrol-kontrol diletakkan. *Form* juga berfungsi sebagai tempat pembuatan tampilan antarmuka (*user interface*) dari sebuah aplikasi Windows. Untuk lebih jelasnya dapat dilihat pada gambar 2.9. (Priyanto, 2014: 31)



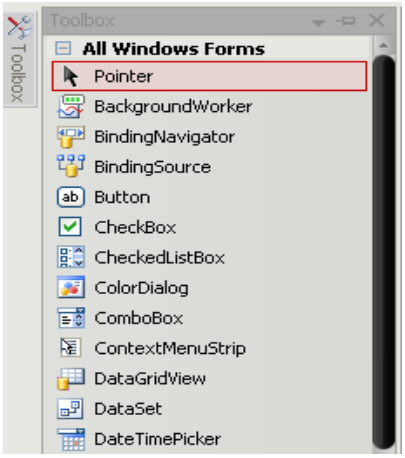
Gambar 2.10 *Form*

16. *Properties Windows* adalah tempat penyimpanan property dari setiap objek kontrol dan komponen. *Properties window* juga dipakai untuk mengatur *property* dari objek kontrol dan komponen yang dipakai. Tampilan dari *Properties Windows* dapat dilihat pada gambar 2.11. (Priyanto, 2014: 29)



Gambar 2.11 *Properties Windows*

17. *Windows Form Layout*, yaitu *windows* yang menampilkan letak dari *form* (posisi *form*) pada *layer monitor* pada saat program dijalankan.
18. *Toolbox*, yaitu sebuah *windows* yang mengandung samua objek atau *control* yang akan digunakan untuk membentuk sebuah program aplikasi dengan semua objek yang diletakkan di jendela *form*. Tampilan dari sebuah *toolbox* pada *Visual Basic 2010* dapat dilihat pada gambar 2.12. (Priyanto, 2014: 31)



Gambar 2.12 *Toolbox*

Keterangan mengenai Toolbox:

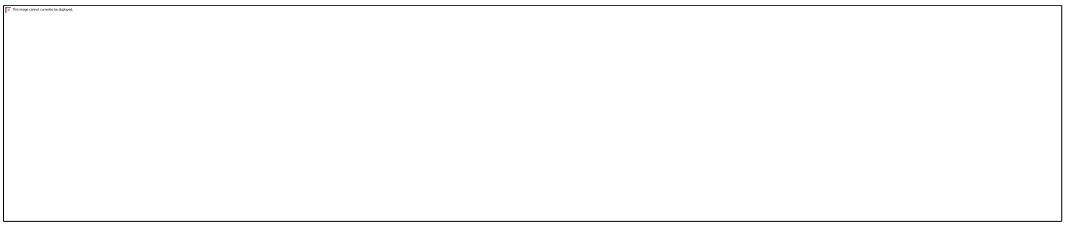
1. *Label*, menampilkan tulisan pada *form*. Pemakai tidak dapat mengubah isi tulisan tersebut secara langsung.
2. *Text box*, sebagai tempat input atau untuk menampilkan teks dan pemakai mengubah-ubah tulisan yang terdapat pada *control*.
3. *List box*, menampilkan beberapa item. Dari *control* ini item-item tersebut dipilih. *Scroll bar* dapat digunakan untuk mengulang pilihan yang tidak dapat ditampilkan seluruhnya.
4. *Command button*, menjalankan suatu tindakan jika pemakai telah melakukan pilihan, denga menekan tombol ini.
5. *Combo box*, merupakan kombinasi dari *text box* dan *list box*. Dengan demikian item ini dapat dilakukan dari list box atau dengan mengetik langsung pada *text box*.

6. *Check box*, menampilkan keadaan *true or false*. Beberapa *control* ini memiliki keadaan yang sama pada saat yang bersamaan.
7. *Picture box*, menampilkan gambar. Gambar-gambar dalam format *bitmap* atau *metafile* dapat ditampilkan oleh *control* ini.
8. *Option button*, sama dengan *control check box*, perbedaanya hanya satu *control* dari beberapa pilihan yang dapat dipilih.
9. *Frame*, mengelompokkan kontrol-kontrol secara visual (tergambar) atau secara fungsional (tindakkan).

10. Project windows, yaitu windows yang menampilkan semua file di dalam visual basic yang sedang aktif. *Project* merupakan kumpulan dari *modul form*, *modul class* dan *modul standard* yang membentuk suatu aplikasi. Tampilan dari sebuah project windows dapat dilihat pada gambar 2.13. (Priyanto, 2014: 32)

Gambar 2.13 *Project Windows*

11. *Windows code*, yaitu jendela yang berisi kode-kode program yang merupakan instruksi-instruksi (perintah-perintah) untuk aplikasi-aplikasi *Visual Basic 2010*. Tampilan dari sebuah windows code dapat dilihat pada gambar 2.14.



Gambar 2.14 *Windows Code*

12. *Windows immediate*, yaitu *windows* yang berguna untuk mencoba beberapa instruksi program. Pada saat program diuji, windows ini dapat digunakan sebagai *windows debug* (pencarian kesalahan atau error). (Priyanto, 2014: 32)

2.5 Microsoft SQL Server

Database management system (DBMS) adalah aplikasi yang dipakai untuk mengelola basis data. DBMS biasanya menawarkan bebarapa kemampuan yang terintegrasi seperti :

1. Membuat, menghapus, menambah, dan memodifikasi basis data
2. Pada beberapa DBMS pengelolaanya berbasis windows (berbentuk jendela-jendela) sehingga lebih mudah digunakan
3. Tidak semua orang bisa mengakses basis data yang ada sehingga memberikan keamanan bagi data
4. Kemampuan berkomunikasi dengan program aplikassi yang lain. Misalnya dimungkinkan untuk mengakses basis data SQL Server menggunakan aplikasi yang dibuat menggunakan VB .NET
5. Kemampuan pengaksesan melalui komunikasi antarkomputer (*client server*)

Microsoft SQL Server adalah salah satu aplikasi DBMS yang sudah sangat banyak digunakan oleh para pemogram aplikasi basis data, contoh DBMS lainnya adalah: MySQL (*freeware*), postgreSQL (*freeware*), MS access dari microsoft, DB2 dari IBM, oracle dan oracle corp, Dbase, serta Foxpro (Priyanto, 2014: 175)

2.5.1 Arsitektur Sistem

Arsitektur sistem maksudnya adalah konfigurasi sistem secara keseluruhan yang menjadi “tempat hidup” dari DBMS, basis data dan aplikasi yang memanfaatkan.

Beberapa jenis arsitektur sistem yang digunakan adalah:

1. Sistem tunggal atau mandiri (*stand alone*)

Pada sistem ini DBMS, basis data serta aplikasi basis data ditempatkan pada komputer yang sama. Arsitektur ini paling sederhana dan murah. Arsistektur ini

dapat kita pilih jika basis data yang dikelola tidak terlalu besar dan lebih bersifat untuk membantu pekerjaan administratif. Sistem mandiri dapat digunakan di rumah masing-masing (untuk latihan), di apotik kecil, wartel, dan hotel kecil.

2. Sistem tersentralisasi (*centralized system*)

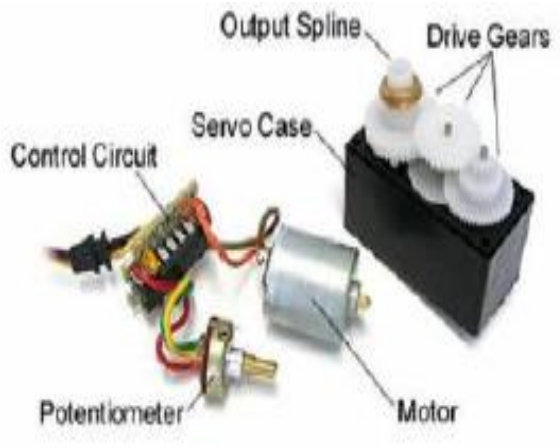
Jika yang tersentralisasi adalah DBMS-nya maka *server*-nya disebut *application server*. Jika yang disentralisasi adalah basis datanya, maka *server*-nya disebut *file server*. Sistem tersentralisasi menggunakan *application server* kurang baik untk spesifikasi *server* yang rendah. Karena pekerjaan atau proses pada *server* sangat berat. Sistem tersentralisasi menggunakan *file server* kurang baik untuk jaringan yang terlalu luas karena transaksi datanya bisa sangat berat dan keamanan kurang terjaga.

3. Sistem *client-server*

Arsitektur ini dibuat untuk menutupi kelemahan pada sistem tersentralisasi. Beban *server* jadi tidak terlalu berat karena *client* juga memiliki DBMS sehingga proses yang bisa dilakukan di *client* akan dilakukan di *client*. Lalu lintas data antara *server* dan *workstation* dibuat lebih efisien. (Priyanto, 2014: 176)

2.6 Motor DC Servo

Motor servo adalah sebuah motor dengan sistem umpan balik tertutup di mana posisi dari motor akan diinformasikan kembali ke rangkaian kontrol yang ada di dalam motor servo. Motor ini terdiri dari sebuah motor DC, serangkaian gear, potensiometer dan rangkaian kontrol. Potensiometer berfungsi untuk menentukan batas sudut dari putaran servo. Sedangkan sudut dari sumbu motor servo diatur berdasarkan lebar pulsa yang dikirim melalui kaki sinyal dari kabel motor. Servo terdiri dari rangkaian pengontrol, gear, potensiometer dan DC motor. Potensiometer terhubung dengan gear demikian pula DC motor. Ketika DC motor diberi *signal* oleh rangkaian pengontrol maka dia akan bergerak demikian pula potensiometer dan otomatis akan mengubah resistansinya. Rangkaian pengontrol akan mengamati perubahan resistansi dan ketika resistansi mencapai nilai yang diinginkan maka motor akan berhenti pada posisi yang diinginkan. (Rois'Am dkk, 2010: 2)



Gambar 2.15 Bentuk Motor Servo

(Rois'Am dkk, 2010: 2)