

BAB II

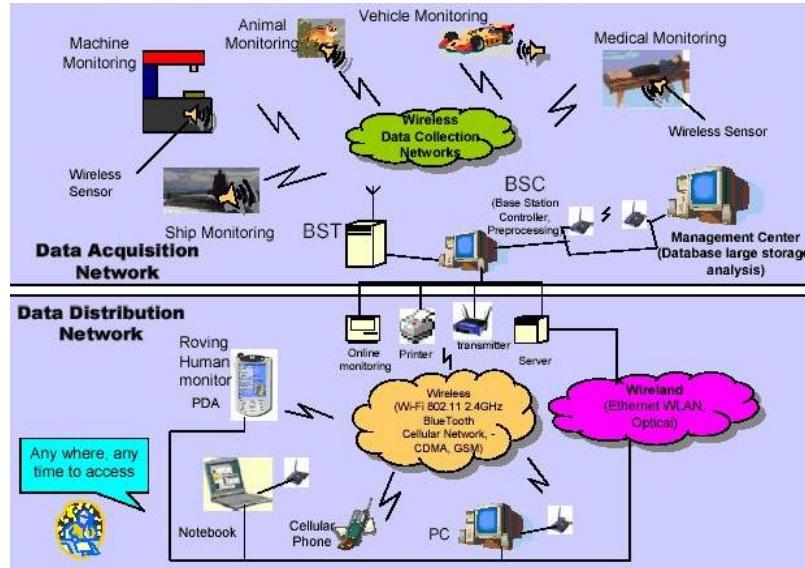
LANDASAN TEORI

2.1 Wireless Sensor Network (WSN)

WSN adalah suatu infrastruktur jaringan *wireless* yang menggunakan sensor untuk memantau kondisi fisik atau kondisi lingkungan yang dapat terhubung ke jaringan. Masing-masing *node* dalam jaringan sensor nirkabel biasanya dilengkapi dengan *radio tranciever* atau alat komunikasi *wireless* lainnya, mikrokontroler, dan sumber energi baterai.

Berdasarkan fakta di dunia, sekitar 98% *processor* bukan berada didalam sebuah komputer PC / laptop, namun terintegrasi dalam aplikasi militer, kesehatan, *remote control*, *chip robotic*, alat komunikasi, dan mesin-mesin industri yang didalamnya telah dipasang sensor.

Perkembangan WSN dan kemajuan teknologi dapat direpresentasikan pada Gambar 2.1. Bahwa dengan berjalanannya waktu, maka perkembangan teknologi semakin mengarah kepada konektivitas lingkungan fisik. Kebanyakan observasi yang dilakukan di lapangan melibatkan banyak faktor dan parameter – parameter untuk mendapatkan hasil yang maksimal dan akurat. Jika peneliti hendak mengambil informasi langsung di lapangan, maka kendalanya adalah dibutuhkan biaya yang besar dan waktu yang lama untuk mendeteksi fenomena yang muncul sehingga menyebabkan performansi yang tidak efisien dan tidak praktis.



Gambar 2.1 Arsitektur Global

Pada Gambar 2.1 diatas dapat dilihat bahwa dengan berjalananya waktu, maka perkembangan teknologi semakin mengarah kepada konektivitas lingkungan fisik. Kebanyakan observasi yang dilakukan di lapangan melibatkan banyak faktor dan parameter-parameter untuk mendapatkan hasil yang maksimal dan akurat. Jika peneliti hendak mengambil informasi langsung di lapangan, maka kendalanya adalah dibutuhkan biaya yang besar dan waktu yang lama untuk mendeteksi fenomena yang muncul sehingga menyebabkan performansi yang tidak efisien dan tidak praktis.

Dengan adanya teknologi WSN, memungkinkan peneliti untuk mendapat informasi yang maksimal tanpa harus berada di area sensor. Informasi dapat diakses dari jarak jauh melalui *gadget* seperti *laptop*, *remote control*, *server* dan sebagainya.

Beberapa keuntungan yang bisa diperoleh dari teknologi WSN adalah Simpel / praktis / ringkas karena tidak perlu ada instalasi kabel yang rumit dan dalam kondisi geografi tertentu sangat menguntungkan dibanding *Wired Sensor*.

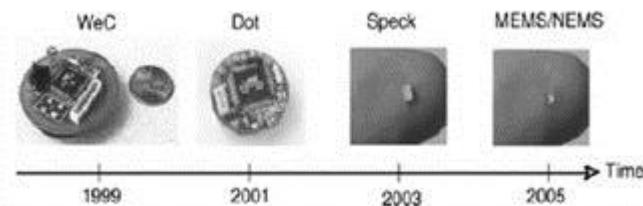
Sensor menjadi bersifat *mobile*, artinya pada suatu saat dimungkinkan untuk memindahkan sensor untuk mendapat pengukuran yang lebih tepat tanpa harus khawatir mengubah desain ruangan maupun susunan kabel ruangan.

- a) Meningkatkan efisiensi secara operasional.
- b) Mengurangi total biaya sistem secara signifikan.
- c) Dapat mengumpulkan data dalam jumlah besar.
- d) Konfigurasi *software* mudah.
- e) Memungkinkan komunikasi digital 2 arah.
- f) Menyediakan konektivitas internet yang secara global, kapanpun dimanapun informasi tersebut dapat diakses melalui server, laptop dan sebagainya.

2.1.1. Arsitektur WSN

Pada WSN, *node* sensor disebar dengan tujuan untuk menangkap adanya gejala atau fenomena yang hendak diteliti. Jumlah *node* yang disebar dapat ditentukan sesuai kebutuhan dan tergantung beberapa faktor misalnya luas area, kemampuan *sensing node*, dan sebagainya. Tiap *node* memiliki kemampuan untuk mengumpulkan data dan meroutingkannya kembali ke *Base Station*. *Node* sensor dapat mengumpulkan data dalam jumlah yang besar dari gejala yang timbul dari lingkungan sekitar.

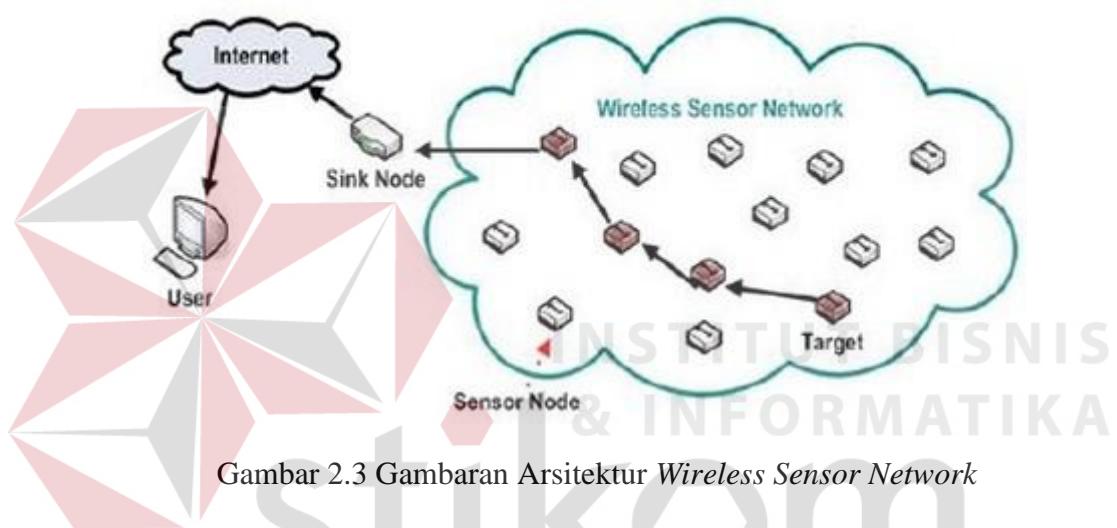
Perkembangan *node* sensor mengikuti trend teknologi nano, dimana ukuran *node* sensor menjadi semakin kecil dari tahun ke tahun. *Node* sensor dapat direpresentasikan oleh Gambar 2.2 berikut:



Gambar 2.2 Perkembangan Dimensi *Node* Sensor Terhadap Waktu

Dan untuk arsitektur WSN secara umum dapat direpresentasikan oleh

Gambar 2.3 sebagai berikut:



Gambar 2.3 Gambaran Arsitektur *Wireless Sensor Network*

Pada Gambar 2.3 dapat dilihat, *node* sensor yang berukuran kecil disebar dalam di suatu area sensor. *Node* sensor tersebut memiliki kemampuan untuk merutekan data yang dikumpulkan ke *node* lain yang berdekatan. Data dikirimkan melalui *transmisi radio* akan diteruskan menuju BS (*Base Station*) yang merupakan penghubung antara *node* sensor dan *user*. Informasi tersebut dapat diakses melalui berbagai *platform* seperti koneksi internet atau satelit sehingga memungkinkan *user* untuk dapat mengakses secara *realtime* melalui *remote server*.

2.2 Perangkat Keras

2.2.1 Arduino Uno

Arduino Uno adalah papan mikrokontroler berbasis ATmega328. Dalam bahasa Italy “Uno” berarti satu, maka peluncuran arduino ini diberi nama Uno.

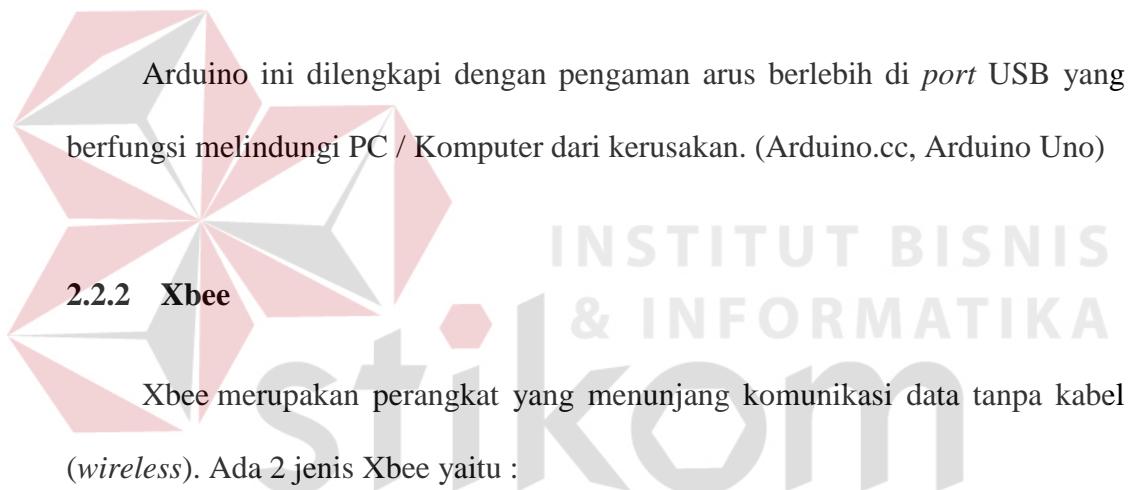


Gambar 2.4 Arduino Uno R3 Sisi Depan (Kiri) Dan Belakang (Kanan)

Arduino ini berisi semua yang diperlukan untuk mendukung mikrokontroler, untuk mengaktifkan cukup menghubungkannya ke komputer dengan kabel USB dengan *power supply* atau baterai. Berikut adalah spesifikasi dari Arduino Uno :

- 1) Mikrokontroler : ATMEGA328
- 2) Tegangan Operasi : 5V
 - a) Tegangan *Input (recommended)* : 7 - 12 V
 - b) Tegangan *Input (limit)* : 6 - 20 V
- 3) Pin *digital I/O* : 14 (6 diantaranya pin PWM)
- 4) Pin *Analog input* : 6
- 5) Arus DC per pin I/O : 40 mA
- 6) Arus DC untuk pin 3.3 V : 150 mA
- 7) *Flash Memory* : 32 KB dengan 0.5KB digunakan untuk *bootloader*
- 8) SRAM : 2 KB
- 9) EEPROM : 1 KB

- 10) Kecepatan Pewaktuan : 16 Mhz
- 11) Memiliki koneksi USB (menggunakan ATmega8U2 sebagai konverter *USB to Serial*)
- 12) Antamuka : UART TTL, I2C, SPI dan USB (Virtual Com)
- 13) Pemograman menggunakan Arduino *Software* (berbasiskan bahasa C yang telah dilengkapi dengan *library* yang kompatibel dengan desain *hardware* Arduino)
- 14) Pengisian kode program dapat menggunakan koneksi USB



- 1) Xbee 802.15.4 (*Xbee Series 1*)

Xbee *series 1* hanya dapat digunakan untuk komunikasi *point to point* dan topologi *star* dengan jangkauan 30 meter *indoor* dan 100 meter *outdoor*.

- 2) Xbee ZB *Series 2*

Xbee *series 2* dapat digunakan untuk komunikasi *point to point*, *point to multipoint* dan topologi *star*, dan topologi *mesh* dengan jangkauan 40 meter *indoor* dan 100 meter *outdoor*.

Xbee *series 1* maupun *series 2* tersedia dalam 2 bentuk berdasarkan kekuatan transmisinya yaitu Xbee reguler dan Xbee-pro. Xbee reguler biasa disebut dengan Xbee saja. Xbee-PRO mempunyai kekuatan transmisi lebih kuat, ukuran perangkatnya lebih besar, dan harganya lebih mahal. Xbee-PRO mempunyai jangkauan *indoor* mencapai 60 meter dan *outdoor* mencapai 1500 meter. Xbee ini dapat digunakan sebagai pengganti *serial / usb* atau dapat memasukkannya ke dalam *command mode* dan mengkonfigurasinya untuk berbagai macam jaringan *broadcast* dan *mesh*. *Shield* membagi setiap pin Xbee. Xbee juga menyediakan *header pin female* untuk penggunaan pin *digital* 2 sampai 7 dan *input analog*, yang *discover* oleh *shield* (pin *digital* 8 sampai 13 tidak tercover oleh *shield*, sehingga dapat menggunakan *header* pada papan itu sendiri). (Arduino.cc, Arduino Xbee Shield)



Gambar 2.5 Xbee *Shield*

Berikut parameter untuk mengkonfigurasi modul Xbee. Pastikan untuk menambahkan AT ke nama parameter ketika mengirimkan perintah ke modul (misalnya untuk membaca parameter ID, harus mengirim perintah ATID). Langkah-langkah tentang membaca (*read*) dan menulis (*write*) Xbee, sebagai berikut:

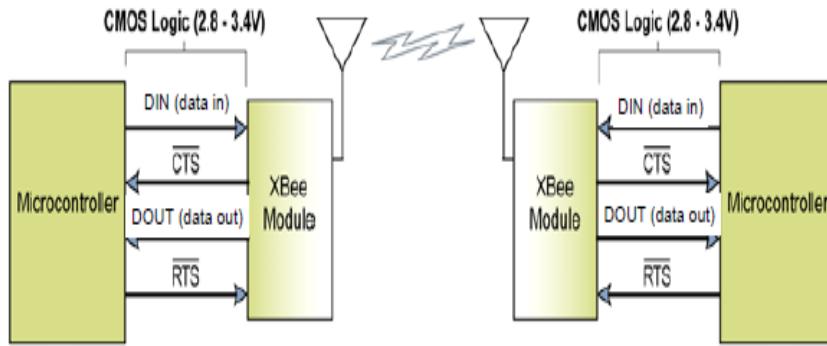
Tabel 2.1 Parameter Xbee

PERINTAH	KETERANGAN	NILAI VALID	Nilai Default
ID	ID jaringan modul Xbee	0-0Xffff	3332
CH	Saluran dari modul Xbee	0-0B-0x1A	0x0C
SH dan SL	Nomor seri modul Xbee (SH memberikan 32 bit tinggi, SL 32 bit rendah). <i>Read-only</i>	0-0xFFFFFFFF	Setiap modul berbeda
MY	Alamat 16 bit dari modul	0-0xFFFF	0
DH dan DL	Alamat tujuan untuk komunikasi nirkabel (DH adalah 32 bit <i>high</i> , DL 32 bit <i>low</i>)	0-0xFFFFFFFF (untuk kedua DH dan DL)	0 (untuk kedua DH dan DL)
BD	Baudrate yang digunakan untuk komunikasi <i>serial</i> dengan arduino atau komputer	0 (1200 bps) 1 (2400 bps) 2 (4800 bps) 3 (9600 bps) 4 (19200 bps) 5 (38400 bps) 6 (57600 bps) 7 (115200 bps)	3 (9600 baudrate)

Catatan: meskipun nilai-nilai yang *valid* dan standar dalam tabel di atas ditulis dengan awalan "0x" (untuk menunjukkan bahwa mereka adalah nomor heksadesimal), modul tidak akan mencakup "0x" ketika melaporkan nilai parameter, dan Anda harus menghilangkan ketika menetapkan nilai-nilai.(Arduino.cc, Arduino Xbee *Shield*)

a) Komunikasi Serial Xbee *Series 2*

Xbee series 2 merupakan sebuah modul yang terdiri dari receiver dan transmitter melalui *port* serial. Melalui *port* serial ini Xbee dapat berkomunikasi secara UART (*Universal Asynchronous Reciever transmiter*). Gambar 2.6 menunjukkan diagram sistem aliran data secara UART. (Inc, 2007)



Gambar 2.6 Diagram Sistem Aliran Data UART pada Xbee

b) AT / Transparent Mode

Dalam mode transparent/AT, modul Xbee bertindak sebagai pengganti *serial line*. Semua data UART (*Universal Asincrhounus Recivier transmiter*) diterima melalui pin *input* akan ditransmisikan. Ketika data tersebut diterima maka data akan dikirimkan keluar (Xbee lainnya) melalui pin *output*. Data atau paket yang diterima bisa ditujukan ke satu sasaran (*point to point*) atau ke beberapa sasaran (*star/broadcast*). (Inc, 2007) Dalam mode transparent/AT, modul Xbee bertindak sebagai pengganti *serial line*. Semua data UART (*Universal Asincrhounus Recivier transmiter*) diterima melalui pin *input* akan ditransmisikan. Ketika data tersebut diterima maka data akan dikirimkan keluar (Xbee lainnya) melalui pin *output*. Data atau paket yang diterima bisa ditujukan ke satu sasaran (*point to point*) atau ke beberapa sasaran (*star/broadcast*). (Inc, 2007)

2.3 Perangkat lunak

2.3.1 Zigbee

ZigBee adalah spesifikasi untuk jaringan protokol komunikasi tingkat tinggi, menggunakan *radio digital* berukuran kecil dengan daya rendah, dan berbasis pada standar IEEE 802.15.4-2003 untuk jaringan personal nirkabel tingkat rendah, seperti saklar lampu nirkabel dengan lampu, alat pengukur listrik dengan inovasi *In-Home Display* (IHD), serta perangkat-perangkat elektronik konsumen lainnya yang menggunakan jaringan *radio* jarak dekat dengan daya transfer data tingkat rendah.

Teknologi yang memenuhi spesifikasi dari ZigBee adalah perangkat dengan pengoperasian yang mudah, sederhana, membutuhkan daya sangat rendah serta biaya yang murah jika dibandingkan dengan WPANs lainnya, yakni Bluetooth. ZigBee fokus pada aplikasi *Radio Frequency* (RF) yang membutuhkan data tingkat rendah, baterai tahan lama, serta jaringan yang aman. (faludi, 2011)

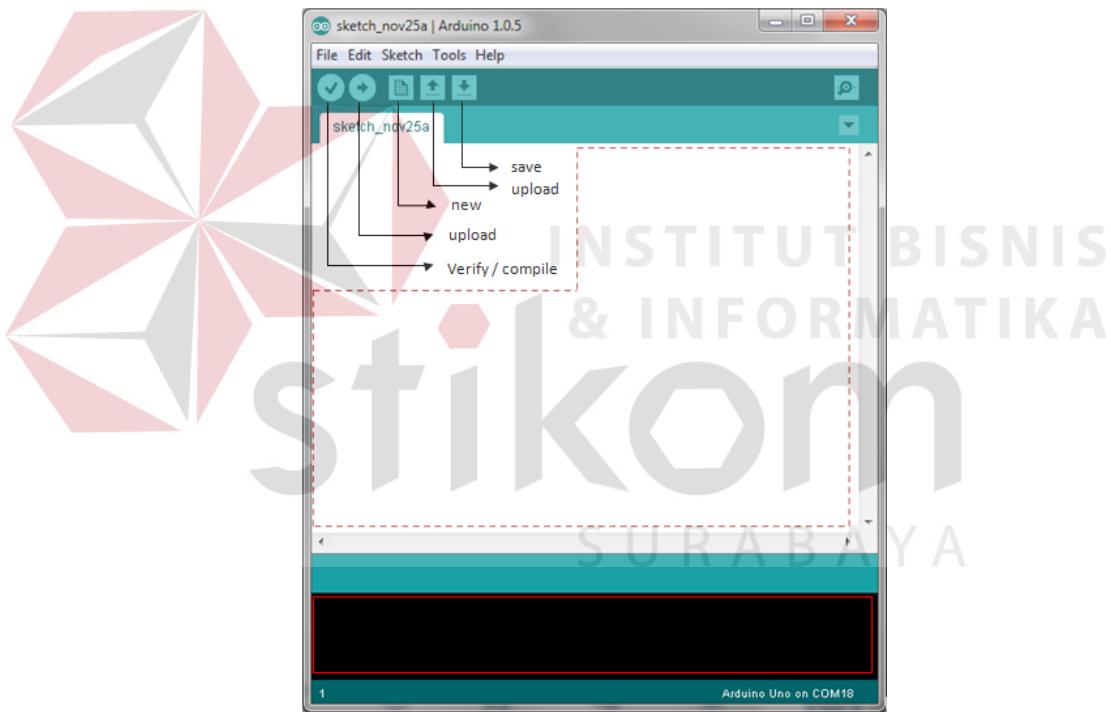
2.3.2 X-CTU

X-CTU adalah sebuah aplikasi yang disediakan oleh DIGI, dimana program ini dirancang oleh Digi untuk berinteraksi dengan Xbee. pada aplikasi ini *user* bisa mengupdate *firmware* xbee dari *coordinator* menjadi *Router / End device* ataupun sebaliknya.(www.digi.com)

2.3.3 IDE Arduino

IDE Arduino adalah *software* yang ditulis menggunakan java dan berdasarkan pengolahan seperti, avr-gcc, dan perangkat lunak *open source* lainnya (Djuandi, 2011). Arduino IDE terdiri dari:

1. *Editor program*, sebuah window yang memungkinkan pengguna menulis dan mengedit program dalam bahasa processing.
2. *Verify / Compiler*, sebuah modul yang mengubah kode program (bahasa processing) menjadi kode biner. Bagaimanapun sebuah mikrokontroller tidak akan bisa memahami bahasa processing, yang dipahami oleh mikrokontroller adalah kode biner.
3. *Uploader*, sebuah modul yang memuat kode biner dari komputer ke dalam memori mikrokontroller di dalam papan arduino.



Gambar 2.7 Tampilan *Software IDE* Arduino

Pada Gambar 2.7 terdapat *menu bar*, kemudian *toolbar* dibawahnya, dan sebuah area putih untuk *editing sketch*, area hitam dapat kita sebut sebagai *progress area*, dan paling bawah dapat kita sebut sebagai "*status bar*".

2.3.4. Bahasa Pemrograman Arduino

Arduino ini bisa dijalankan di komputer dengan berbagai macam *platform* karena didukung atau berbasis Java. *Source* program yang dibuat untuk aplikasi mikrokontroler adalah bahasa C/C++ dan dapat digabungkan dengan assembly. (arduino.cc, 2013)

1. Struktur

Setiap program Arduino (biasa disebut *sketch*) mempunyai dua buah fungsi yang harus ada (arduino.cc, 2013). Antara lain:

a) **void setup()** { }

Semua kode didalam kurung kurawal akan dijalankan hanya satu kali ketika program Arduino dijalankan untuk pertama kalinya.

b) **void loop()** { }

Fungsi ini akan dijalankan setelah *setup* (fungsi *void setup*) selesai. Setelah dijalankan satu kali fungsi ini akan dijalankan lagi, dan lagi secara terus menerus sampai catu daya (*power*) dilepaskan.

2. Serial

Serial digunakan untuk komunikasi antara arduino *board*, komputer atau perangkat lainnya. Arduino *board* memiliki minimal satu *port* serial yang berkomunikasi melalui pin 0 (RX) dan 1 (TX) serta dengan komputer melalui USB. Jika menggunakan fungsi – fungsi ini, pin 0 dan 1 tidak dapat digunakan untuk *input* digital atau *output* digital (arduino.cc, 2013). Terdapat beberapa fungsi serial pada arduino, antara lain:

a) *Serial.begin()*

Fungsi ini digunakan untuk transmisi data serial dan mengatur data *rate* dalam *bits per second (baud)*. Untuk berkomunikasi dengan komputer gunakan salah satu dari angka ini: 300, 600, 1200, 2400, 4800, 9600, 14400, 19200, 28800, 38400, 57600, atau 115200.

b) `Serial.available()`

Fungsi ini digunakan untuk mendapatkan jumlah data *byte (characters)* yang tersedia dan membacanya dari *port* serial. Data tersebut adalah data yang telah tiba dan disimpan dalam *buffer* serial yang menampung sampai 64 *bytes*.

c) `Serial.read()`

Fungsi digunakan untuk membaca data serial yang masuk.

d) `Serial.print()` dan `Serial.println()`

Fungsi ini digunakan untuk mencetak data ke *port* serial dalam format text ASCII. Sedangkan fungsi `Serial.println()` sama seperti fungsi `Serial.print()` hanya saja ketika menggunakan fungsi ini akan mencetak data dan kemudian diikuti dengan karakter *newline* atau *enter*.

3. Syntax

Berikut ini adalah elemen bahasa C yang dibutuhkan untuk format penulisan. (arduino.cc, 2013)

a) `//(komentar satu baris)`

Kadang diperlukan untuk memberi catatan pada diri sendiri apa arti dari kode-kode yang dituliskan. Cukup menuliskan dua buah garis miring dan apapun yang kita ketikkan dibelakangnya akan diabaikan oleh program.

b) `/* */(komentar banyak baris)`

Jika anda punya banyak catatan, maka hal itu dapat dituliskan pada beberapa baris sebagai komentar. Semua hal yang terletak di antara dua simbol tersebut akan diabaikan oleh program.

c) **{ }**(kurung kurawal)

Digunakan untuk mendefinisikan kapan blok program mulai dan berakhir (digunakan juga pada fungsi dan pengulangan).

d) **;**(titik koma)

Setiap baris kode harus diakhiri dengan tanda titik koma (jika ada titik koma yang hilang maka program tidak akan bisa dijalankan).

4. Variabel

Sebuah program secara garis besar dapat didefinisikan sebagai instruksi untuk memindahkan angka dengan cara yang cerdas. Variabel inilah yang digunakan untuk memindahkannya. (arduino.cc, 2013)

a) **int** (integer)

Digunakan untuk menyimpan angka dalam 2 byte (16 bit). Tidak mempunyai angka desimal dan menyimpan nilai dari -32,768 dan 32,767.

b) **long** (long)

Digunakan ketika integer tidak mencukupi lagi. Memakai 4 byte (32 bit) dari memori (RAM) dan mempunyai rentang dari -2,147,483,648 dan 2,147,483,647.

c) **boolean** (boolean)

Variabel sederhana yang digunakan untuk menyimpan nilai *TRUE* (benar) atau *FALSE* (salah). Sangat berguna karena hanya menggunakan 1 bit dari RAM.

d) **float** (float)

Digunakan untuk angka desimal (*floating point*). Memakai 4 *byte* (32 bit) dari RAM dan mempunyai rentang dari -3.4028235E+38 dan 3.4028235E+38.

e) **char** (character)

Menyimpan 1 karakter menggunakan kode ASCII (misalnya ‘A’ = 65). Hanya memakai 1 *byte* (8 bit) dari RAM.

5. Operator Matematika

Operator yang digunakan untuk memanipulasi angka (bekerja seperti matematika yang sederhana). (arduino.cc, 2013)

a) **=** (sama dengan)

Membuat sesuatu menjadi sama dengan nilai yang lain (misalnya: $x = 10 * 2$, x sekarang sama dengan 20).

b) **% (persen)**

Menghasilkan sisa dari hasil pembagian suatu angka dengan angka yang lain (misalnya: $12 \% 10$, ini akan menghasilkan angka 2).

c) **+** (penjumlahan)

d) **-** (pengurangan)

e) ***** (perkalian)

f) **/** (pembagian)

6. Operator Pembanding

Digunakan untuk membandingkan nilai logika.

a) **==**

Sama dengan (misalnya: $12 == 10$ adalah *FALSE* (salah) atau $12 == 12$ adalah *TRUE* (benar)).

b) **!=**

Tidak sama dengan (misalnya: $12 \neq 10$ adalah *TRUE* (benar) atau $12 \neq 12$ adalah *FALSE* (salah)).

c) $<$

Lebih kecil dari (misalnya: $12 < 10$ adalah *FALSE* (salah) atau $12 < 12$ adalah *FALSE* (salah) atau $12 < 14$ adalah *TRUE* (benar)).

d) $>$

Lebih besar dari (misalnya: $12 > 10$ adalah *TRUE* (benar) atau $12 > 12$ adalah *FALSE* (salah) atau $12 > 14$ adalah *FALSE* (salah)).

7. Struktur Pengaturan

Program sangat tergantung pada pengaturan apa yang akan dijalankan berikutnya, berikut ini adalah elemen dasar pengaturan (banyak lagi yang lain dan bisa dicari di internet). (arduino.cc, 2013)

a) **If else**, dengan format seperti berikut ini:

```
if (kondisi) { }
else if (kondisi) { }
else { }
```

Dengan struktur seperti diatas program akan menjalankan kode yang ada di dalam kurung kurawal jika kondisinya *TRUE*, dan jika tidak (*FALSE*) maka akan diperiksa apakah kondisi pada *else if* dan jika kondisinya *FALSE* maka kode pada *else* yang akan dijalankan.

b) **While**, dengan format seperti berikut ini:

```
While(kondisi) { }
```

Dengan struktur ini, *while* akan melakukan pengulangan terus menurus dan tak terbatas sampai kondisi didalam kurung () menjadi *false*.

c) **for**, dengan format seperti berikut ini:

```
for (int i = 0; i < #pengulangan; i++) { }
```

Digunakan bila ingin melakukan pengulangan kode di dalam kurung kurawal beberapa kali, ganti `#pengulangan` dengan jumlah pengulangan yang diinginkan. Melakukan penghitungan ke atas dengan `i++` atau ke bawah dengan `i--`.

8. Operator Boolean

Operator ini dapat digunakan dalam kondisi if, antara lain:

a) **&&** (*logika and*), dengan format seperti berikut ini:

```
if (digitalRead(2) == HIGH && digitalRead(3) == HIGH) {}
```

Digunakan bila ingin mendapatkan nilai *true* hanya jika kedua input bernilai *HIGH*.

b) **||** (*logika or*), dengan format seperti berikut ini:

```
if (x > 0 || y > 0) {}
```

Digunakan bila ingin mendapatkan nilai *true* hanya jika nilai x dan y lebih besar dari 0.

c) **!** (*not*), dengan format seperti berikut ini:

```
if (!x) {}
```

Digunakan bila ingin mendapatkan nilai *true* hanya jika nilai tidak sama dengan x.

9. Digital

a) **pinMode(pin, mode)**

Digunakan untuk menetapkan mode dari suatu pin, *pin* adalah nomor pin yang akan digunakan dari 0-19 (pin analog 0-5 adalah 14-19). Mode yang bisa digunakan adalah *INPUT* atau *OUTPUT*.

b) **digitalWrite(pin, value)**

Ketika sebuah pin ditetapkan sebagai *OUTPUT*, pin tersebut dapat dijadikan *HIGH* (5 volts) atau *LOW* (diturunkan menjadi *ground*).

c) **digitalRead(pin)**

Ketika sebuah pin ditetapkan sebagai *INPUT* maka anda dapat menggunakan kode ini untuk mendapatkan nilai pin tersebut apakah *HIGH* (5 volts) atau *LOW* (diturunkan menjadi *ground*).

10. Analog

Arduino adalah mesin digital tetapi mempunyai kemampuan untuk beroperasi di dalam analog. Berikut ini cara untuk menghadapi hal yang bukan digital.

a) **analogWrite(pin, value)**

Beberapa pin pada Arduino mendukung PWM (*pulse width modulation*) yaitu pin 3, 5, 6, 9, 10, 11. Ini dapat merubah pin hidup (*on*) atau mati (*off*) dengan sangat cepat sehingga membuatnya dapat berfungsi layaknya keluaran analog. *Value* (nilai) pada format kode tersebut adalah angka antara 0 (0% *duty cycle* ~ 0V) dan 255 (100% *duty cycle* ~ 5V).

b) **analogRead(pin)**

Ketika pin analog ditetapkan sebagai *INPUT* anda dapat membaca keluaran voltase-nya. Keluarannya berupa angka antara 0 (untuk 0 volts) dan 1024 (untuk 5 volts).

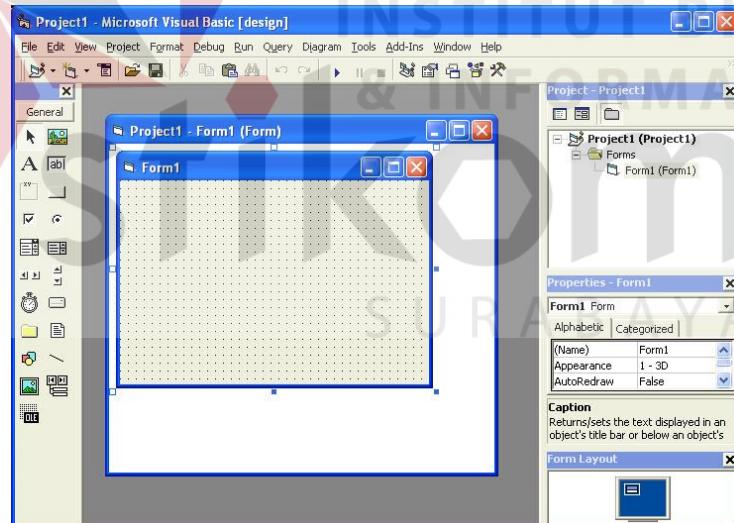
2.3.5. Visual Basic 6.0

Bahasa pemrograman visual basic 6.0 dapat digunakan untuk menyusun dan membuat program aplikasi pada sistem operasi windows. Program aplikasi dapat berupa program *database*, program grafis dan lain sebagainya. Didalam visual basic 6.0 terdapat komponen - komponen yang sangat membantu dalam pembuatan program aplikasi. Dalam pembuatan program aplikasi pada visual basic 6.0 dapat didukung oleh *software* seperti Microsoft Access, Microsoft Exel, Seagate Criystal Report, dan lain sebagainya. Tampilan visual basic terdapat pada *Integrated Development Environment* (IDE) pada Gambar 2.6. Adapun penjelasan jendela-jendela adalah sebagai berikut :

- a) Menu *Bar*, digunakan untuk memilih tugas-tugas tertentu seperti menyimpan *project*, membuka *project*, dll.
- b) Main *Toolbar*, digunakan untuk melakukan tugas-tugas tertentu dengan cepat.
- c) Jendela *Project*, jendela berisi gambaran dari semua modul yang terdapat dalam aplikasi.
- d) Jendela *Form Designer*, jendela merupakan tempat anda untuk merancang *user interface* dari aplikasi.
- e) Jendela *Toolbox*, jendela berisi komponen-komponen yang dapat digunakan untuk mengembangkan *user interface*.

- f) Jendela *Code*, merupakan tempat untuk menulis *coding* , yang dapat menampilkan jendela dengan menggunakan kombinasi Shift + F7.
- g) Jendela *Propertis*, merupakan daftar properti-properti object yang terpilih, sebagai contohnya anda dapat mengubah warna tulisan (*foreground*) dan warna latar belakang (*background*), untuk menampilkan jendela tekan F4.
- h) Jendela *colour palette*, adalah fasilitas cepat untuk mengubah warna suatu object.

- i) Jendela *Form Layout*, akan menunjukkan bagaimana form bersangkutan ditampilkan ketika *runtime*.



Gambar 2.8 Tampilan Utama Visual Basic 6.0.