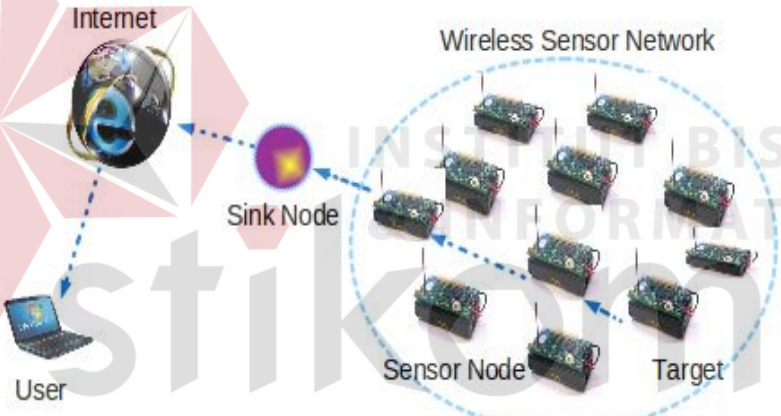
BAB II

LANDASAN TEORI

2.1 *Wireless Sensor Network* (WSN)

Wireless Sensor Network (WSN) atau jaringan sensor nirkabel merupakan suatu jaringan nirkabel yang terdiri dari beberapa sensor *node* yang bersifat individu yang diletakkan ditempat – tempat yang berbeda untuk memonitoring kondisi suatu tempat dan dapat berinteraksi dengan lingkungannya dengan cara *sensing, controlling* dan *communication* terhadap parameter – parameter fisiknya.



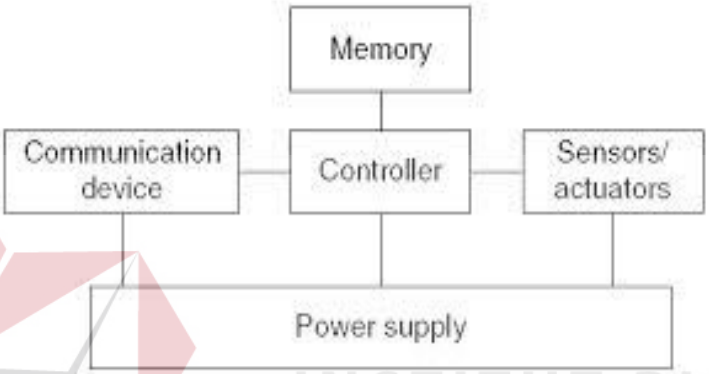
Gambar 2.1 Arsitektur WSN

Sumber : (http://virtual-labs.ac.in)

Pada Gambar 2.1 menunjukkan gambaran umum WSN 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*) atau *sink node* 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*. (E, Sugiarto, & Sakti, 2009)

Setiap *node* dalam WSN terdiri dari lima komponen yaitu kontroller/mikrokontroler, memori, sensor/akuator, perangkat komunikasi dan catu daya. Umumnya catu daya yang dipakai adalah baterai. Komponen – komponen dari sebuah *node* ditunjukkan pada Gambar 2.2.



Gambar 2.2 Komponen – Komponen Penyusun *Node* dalam WSN

Sumber : (E, Sugiarto, & Sakti, 2009)

1. *Communication device* (perangkat komunikasi)

   Berfungsi untuk menerima/mengirim data dengan menggunakan protokol IEEE 802.15.4 atau IEEE 802.11b/g kepada *device* atau *node* lainnya.

2. Mikrokontroler

   Berfungsi untuk melakukan fungsi perhitungan, mengontrol dan memproses *device – device* yang terhubung dengan mikrokontroler.

3. Sensor

   Berfungsi untuk men–sensing besaran–besaran fisis yang hendak diukur. Sensor adalah suatu alat yang mampu untuk mengubah suatu bentuk energi ke bentuk energi lain, dalam hal ini adalah mengubah dari energi besaran yang

diukur menjadi energi listrik yang kemudian diubah oleh ADC menjadi deretan pulsa terkuantisasi yang kemudian bisa dibaca oleh mikrokontroler.

4. *Memory*

   Berfungsi sebagai tambahan memori bagi sistem *Wireless* Sensor, pada dasarnya sebuah unit mikrokontroler memiliki unit memori sendiri.

5. *Power supply*

   Berfungsi sebagai sumber energi bagi sistem *Wireless Sensor* secara keseluruhan.

2.2 Arduino

Arduino adalah prototipe platform elektronik opensource yang terdiri dari mikrokontroler, bahasa pemrograman, dan IDE. Arduino adalah alat untuk membuat aplikasi interaktif, yang dirancang untuk mempermudah proyek bagi pemula tetapi masih cukup fleksibel bagi para ahli untuk mengembangkan proyek-proyek yang kompleks. (Banzi, Getting Started with Arduino, 2009)

2.2.1 Arduino Uno SMD R3

Arduino Uno adalah papan mikrokontroler berbasis ATmega328. Dalam bahasa Italy “Uno” berarti satu, maka peluncuran arduino ini diberi nama Uno. Arduino ini berisi semua yang diperlukan untuk mendukung mikrokontroler, untuk mengaktifkan cukup menghubungkannya ke komputer dengan sebuah kabel USB atau mensuplainya dengan sebuah adaptor AC ke DC atau menggunakan baterai. (arduino.cc)

Gambar 2.3 Arduino Uno SMD R3 Sisi Depan (Kiri) dan Belakang(Kanan)

Sumber : (arduino.cc)

Secara umum arduino terdiri dari dua bagian, yaitu:

1. *Hardware*: papan input/output (I/O)
2. *Software*: *software* arduino meliputi IDE untuk menulis program, driver untuk koneksi dengan komputer, contoh program dan *library* untuk pengembangan program. (Djuandi, 2011)

Berikut adalah Tabel 2.1 spesifikasi dari arduino uno smd R3:

Tabel 2.1 Spesifikasi Arduino Uno SMD R3

Mikrokontroler	ATmega328
Tegangan pengoperasian	5V
Tegangan input yang disarankan	7-12V
Batas tegangan input	6-20V
Jumlah pin I/O digital	14 (6 di antaranya menyediakan keluaran PWM)
Jumlah pin input analog	6
Arus DC tiap pin I/O	40 mA
Arus DC untuk pin 3.3V	50 mA
Memori Flash	32 KB (ATmega328), sekitar 0.5 KB digunakan oleh bootloader
SRAM	2 KB (ATmega328)
EEPROM	1 KB (ATmega328)
Clock Speed	16 MHz

2.2.2 Daya (*Power*)

Arduino Uno dapat disuplai melalui koneksi USB atau dengan sebuah power suplai eksternal. Suplai eksternal (non-USB) dapat diperoleh dari sebuah adaptor AC ke DC atau baterai. Adaptor dapat dihubungkan dengan mencolokkan sebuah *center-positive* plug yang panjangnya 2,1 mm ke power jack dari *board*. Kabel lead dari sebuah battery dapat dimasukkan dalam *header*/kepala pin *Ground* (Gnd) dan pin Vin dari konektor *POWER*.

Board Arduino Uno dapat beroperasi pada sebuah suplai eksternal 6 sampai 20 Volt. Jika disuplai dengan yang lebih kecil dari 7 V, kiranya pin 5 Volt mungkin mensuplai kecil dari 5 Volt dan *board* Arduino Uno bisa menjadi tidak stabil. Jika menggunakan suplai yang lebih dari besar 12 Volt, *voltage* regulator bisa kelebihan panas dan membahayakan *board* Arduino Uno. *Range* yang direkomendasikan adalah 7 sampai 12 Volt. (arduino.cc)

Pin-pin dayanya adalah sebagai berikut:

a) VIN. Tegangan input ke Arduino *board* ketika *board* sedang menggunakan sumber suplai eksternal (seperti 5 Volt dari koneksi USB atau sumber tenaga lainnya yang diatur). Kita dapat menyuplai tegangan melalui pin ini, atau jika penyuplaian tegangan melalui *power jack*, aksesnya melalui pin ini.

b) 5V. Pin output ini merupakan tegangan 5 Volt yang diatur dari regulator pada *board*. *Board* dapat disuplai dengan salah satu suplai dari DC *power jack* (7-12V), USB *connector* (5V), atau pin VIN dari *board* (7-12). Penyuplaian tegangan melalui pin 5V atau 3,3V mem*bypass* regulator, dan dapat membahayakan *board*. Hal itu tidak dianjurkan.

c) 3V3. Sebuah suplai 3,3 Volt dihasilkan oleh regulator pada *board*. Arus maksimum yang dapat dilalui adalah 50 mA.

d) GND. Pin *ground*.

2.2.3 Memori

ATmega328 mempunyai 32 KB yang bersifat *non-volatile*, digunakan untuk menyimpan program yang dimuat dari komputer. (dengan 0,5 KB digunakan untuk *bootloader*). ATmega 328 juga mempunyai 2 KB SRAM yang *volatile* (hilang saat daya dimatikan), digunakan oleh variable-variabel di dalam program. dan 1 KB EEPROM (yang dapat dibaca dan ditulis (RW/*read* and *written*). (arduino.cc)

2.2.4 *Input dan Output*

Setiap 14 pin digital pada Arduino Uno dapat digunakan sebagai input dan output. Fungsi-fungsi tersebut beroperasi di tegangan 5 Volt. Setiap pin dapat memberikan atau menerima suatu arus maksimum 40 mA dan mempunyai sebuah resistor pull-up (terputus secara default) 20-50 kOhm. Selain itu, beberapa pin mempunyai fungsi-fungsi sebagai berikut:

a) **Serial: 0 (RX) dan 1 (TX)**. Digunakan untuk menerima (RX) dan memancarkan (TX) serial data TTL (*Transistor-Transistor Logic*). Kedua pin ini dihubungkan ke pin-pin yang sesuai dari chip Serial Atmega8U2 USB-ke-TTL.

b) **External Interrupts: 2 dan 3**. Pin-pin ini dapat dikonfigurasikan untuk dipicu sebuah interrupt (gangguan) pada suatu nilai rendah, suatu kenaikan atau penurunan yang besar, atau suatu perubahan nilai.

c) **PWM: 3, 5, 6, 9, 10, dan 11**. Memberikan 8-bit PWM *output* dengan fungsi *analogWrite*().

d) **SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK)**. Pin-pin ini men*support* komunikasi SPI menggunakan SPI *library*.

e) **LED: 13**. Ada sebuah LED yang terpasang, terhubung ke pin digital 13. Ketika pin bernilai *HIGH* LED menyala, ketika pin bernilai *LOW* LED mati.

Arduino UNO mempunyai 6 input analog, diberi label A0 sampai A5, setiapnya memberikan resolusi 10 bit. Secara default, 6 input analog tersebut mengukur tegangan dari *ground* sampai tegangan 5 Volt, dengan itu memungkinkan untuk mengganti batas atas dari rangenya dengan menggunakan pin AREF dan fungsi *analogReference*(). Di sisi lainnya, beberapa pin mempunyai fungsi spesifik yaitu pin A4 atau SDA dan pin A5 atau SCL. Mendukung komunikasi TWI dengan menggunakan *Wire library*. Ada sepasang pin lainnya pada board yaitu AREF referensi tegangan untuk input analog. Digunakan dengan *analogReference*(), dan reset untuk mereset mikrokontroler. (arduino.cc)

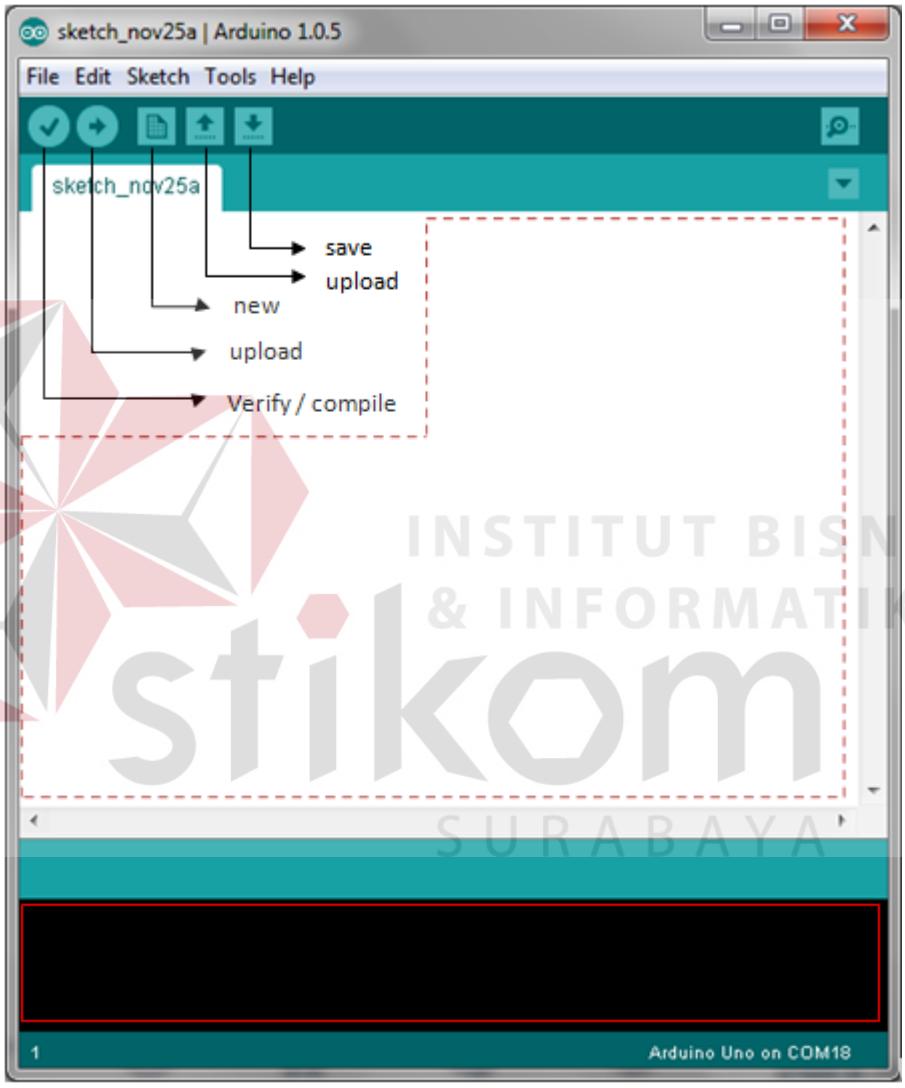
2.3 *Software* Arduino IDE

Arduino IDE 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 2.4 Tampilan *Software* Arduino IDE

Pada Gambar 2.4 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*".

2.4 Bahasa Pemograman 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)

1. Struktur

Setiap program Arduino (biasa disebut *sketch*) mempunyai dua buah fungsi yang harus ada (arduino.cc). 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). 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)

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) **;**(titk 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)

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)

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 != 10 adalah *TRUE* (benar) atau 12 != 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)

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 yang *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 yang *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 yang *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).

2.5 Xbee *Series 2 Chip Antenna*

Xbee *series* 2 modul RF dirancang untuk beroperasi dalam protokol ZigBee dengan biaya yang murah dan jaringan sensor nirkabel menggunakan daya yang rendah. Modul ini membutuhkan daya yang rendah dan dapat melakukan pengiriman data yang handal antara perangkat dengan jarak yang jauh. Modul ini beroperasi pada frekuensi 2.4 GHz. (Inc, XBee Series 2 OEM RF Modules, 2007)

Xbee series 2 ini mempunyai beberapa model antena, salah satunya adalah *chip antenna*. *Chip antenna* merupakan suatu chip keramik yang terletak pada board modul Xbee, bentuknya lebih kecil. *Chip antenna* memiliki pola radiasi cardoid, yang berarti sinyal dilemahkan dalam berbagai arah dan sangat baik digunakan dalam area yang tidak luas atau kecil. Pada Gambar 2.5 merupakan gambar dari modul Xbee *series* 2 *chip antenna.*

Gambar 2.5 Xbee Series 2 Chip Antenna

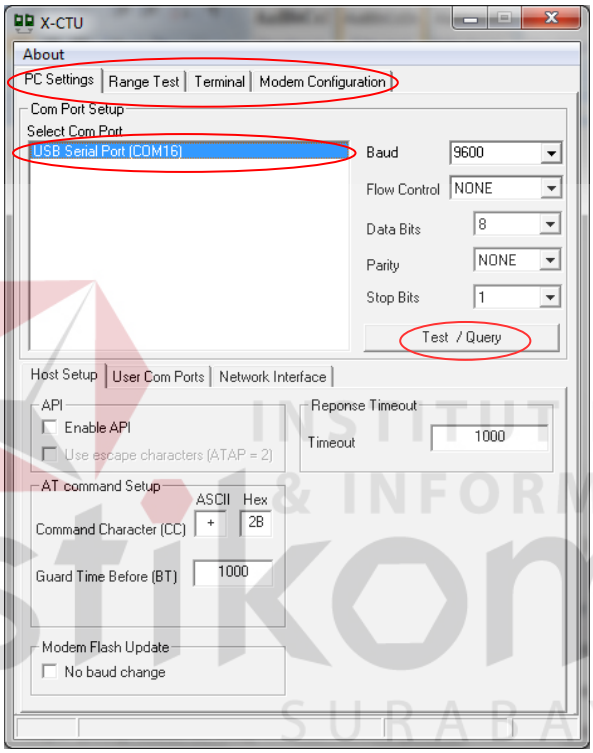
Berikut adalah spesifikasi dari modul Xbee *series* 2 *chip antenna* :

1. Jarak jangkauan indoor 133 ft atau 40 meter.
2. Jarak jangkauan outdoor line of sight 400 ft atau 120 meter.
3. Transmit power output 2 mW (+ 3 dbm).
4. Radio Frekuensi data rate 250 Kbps.
5. Frekuensi 2.4 GHz.
6. Receiver sensitivity -98 dbm (1 % pakcet *error* rate).
7. Antena menggunakan *chip antenna.*

2.6 Xbee Usb Adapter dan *Software* X-CTU

Xbee usb adapter (Gambar 2.6) merupakan alat untuk menghubungkan modul Xbee ke komputer dengan kabel mini usb dan selanjutnya dapat dikonfigurasi menggunakan *software* X-CTU (Gambar 2.7). *software* X-CTU merupakan *software* yang digunakan untuk mengkonfigurasi Xbee agar dapat berkomunikasi dengan Xbee lainya. Parameter yang harus diatur adalah PAN ID (Personal Area Network) ID yaitu parameter yang mengatur radio mana saja yang dapat berkomunikasi, agar dapat berkomunikasi PAN ID dalam satu jaringan harus sama. Parameter yang harus diatur adalah PAN ID (Personal Area Network) ID yaitu parameter yang mengatur Xbee mana saja yang dapat berkomunikasi, agar dapat berkomunikasi PAN ID dalam satu jaringan harus sama. Xbee dapat berkomunikasi point to point dan point to multipoint (*broadcast*).

Gambar 2.6 Xbee Usb *Adapter* dan Kabel Mini Usb



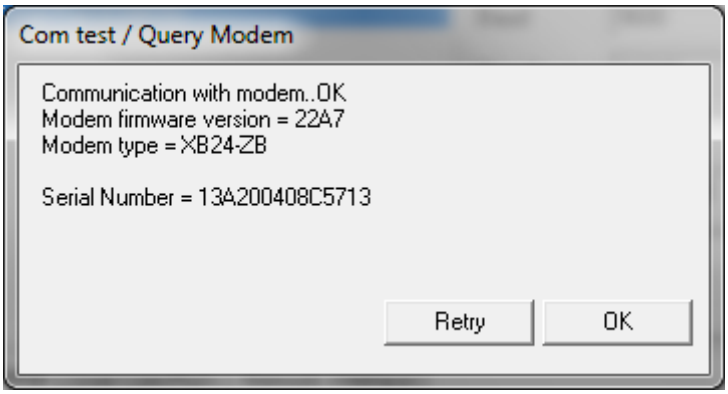
Gambar 2.7 Tampilan *Software* X-CTU

Pada Gambar 2.7 *software* X-CTU, terdapat empat tab di bagian atas program. Masing – masing tab mempunyai fungsi yang berbeda – beda (Inc, 2008). Berikut adalah ke empat tab tersebut:

1. PC Settings

Pada tab ini mengijinkan pengguna untuk memilih *COM port* yang diinginkan dan mengkonfigurasi *port* tersebut sesuai pengaturan Xbee yang

diinginkan. Terdapat tombol *Test* / *Query* pada tab *PC Settings*, tombol ini digunakan untuk menguji *COM port* yang telah dipilih, jika pengaturan dan *COM port* benar, maka akan muncul kotak dialog respon seperti pada Gambar 2.8.



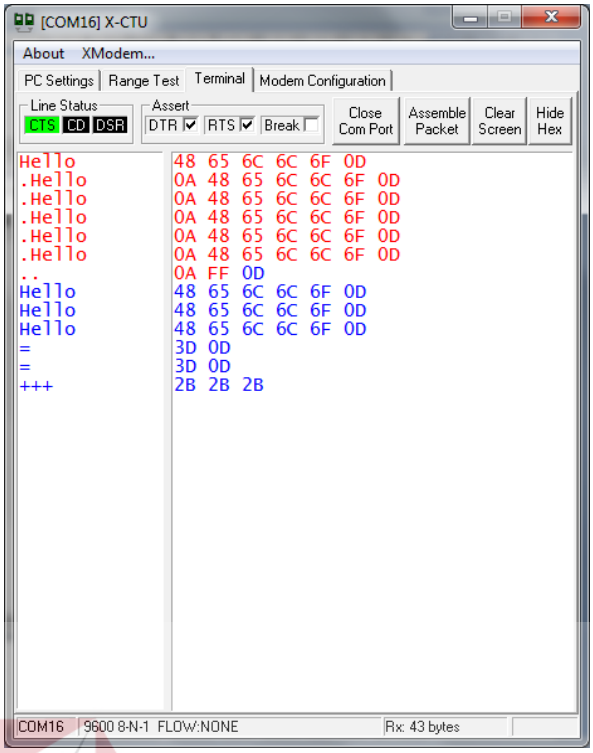
Gambar 2.8 Kotak Dialog Respon

2. Range Test

Pada tab *range test* ini, pengguna dapat melakukan pengujian *range test* antara dua Xbee dengan mengirimkan paket data yang ditentukan pengguna dan memverifikasi apakah paket data yang dikirim sama dengan yang diterima.

3. Terminal

Pada *tab* ini, pengguna memungkinkan akses ke *COM port* komputer dengan program terminal *emulation*. Tab ini juga memungkinkan untuk mengakses *firmware* Xbee menggunakan AT *commands* dan dapat mengirim atau menerima data dalam format Hex dan ASCII dengan memilih *Assemble Packet* (Gambar.2.9).



Gambar 2.9 *Tab* Terminal

Gambar 2.10 Assemble Packet

Pada Gambar 2.10 area putih dalam terminal *tab* ini berisi data informasi komunikasi yang terjadi antara 2 Xbee atau lebih. Teks yang berwarna biru merupakan teks yang telah diketik pengguna dan diarahkan ke *port* serial Xbee sedangkan teks merah merupakan data yang masuk dari *port* serial Xbee.

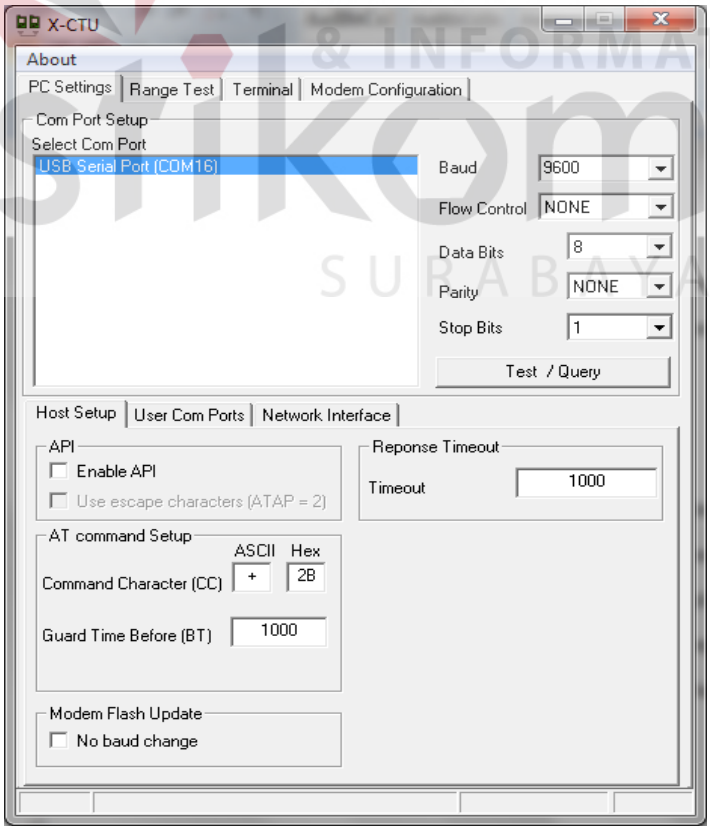
4. Modem Configuration Tab

Pada tab ini, pengguna dapat melakukan pemrograman pada pengaturan *firmware* Xbee dan merubah versi *firmware*nya melalui *Graphical User Interface* (GUI). Terdapat 4 fungsi dasar dalam *modem configuration tab*, yaitu:

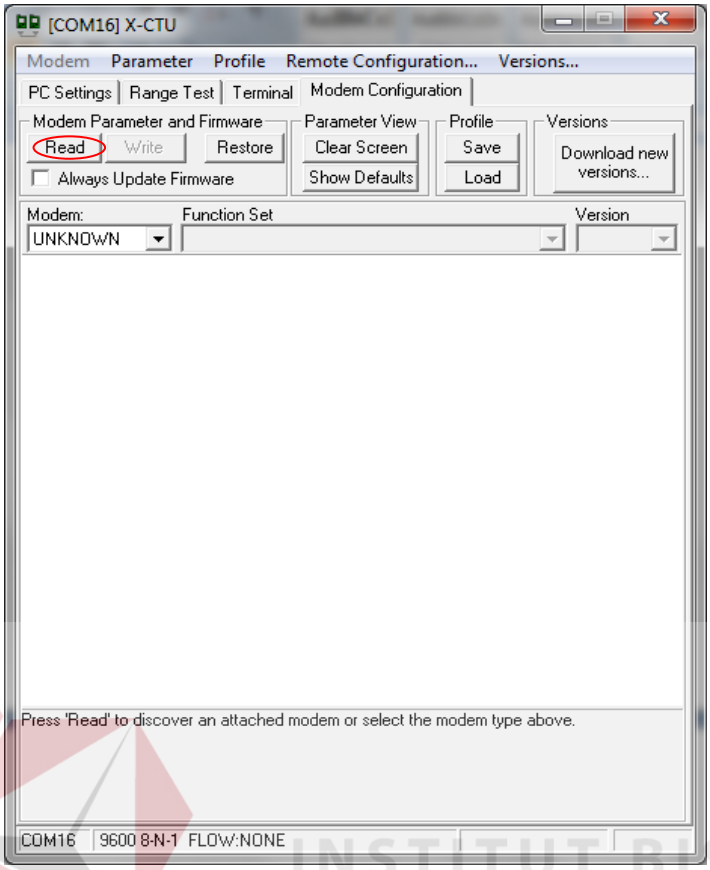
a) Menyediakan fasilitas GUI untuk mengatur *firmware* pada Xbee.

b) *Read* dan *write* firmware ke mikrokontroler Xbee.

Untuk dapat membaca (*read*) firmware Xbee, pertama hubungkan Xbee dengan Xbee usb adapter yang telah terhubung dengan kabel mini usb, kemudian hubungkan kabel tersebut ke komputer melalui *interface* usb. Jalankan aplikasi X-CTU pada komputer, selanjutnya atur *com port* pada tab PC *Settings* seperti Gambar 2.11. Pada *tab Modem Configuration*, klik "*Read*" pada bagian *Modem Parameter and Firmware* (Gambar 2.12).

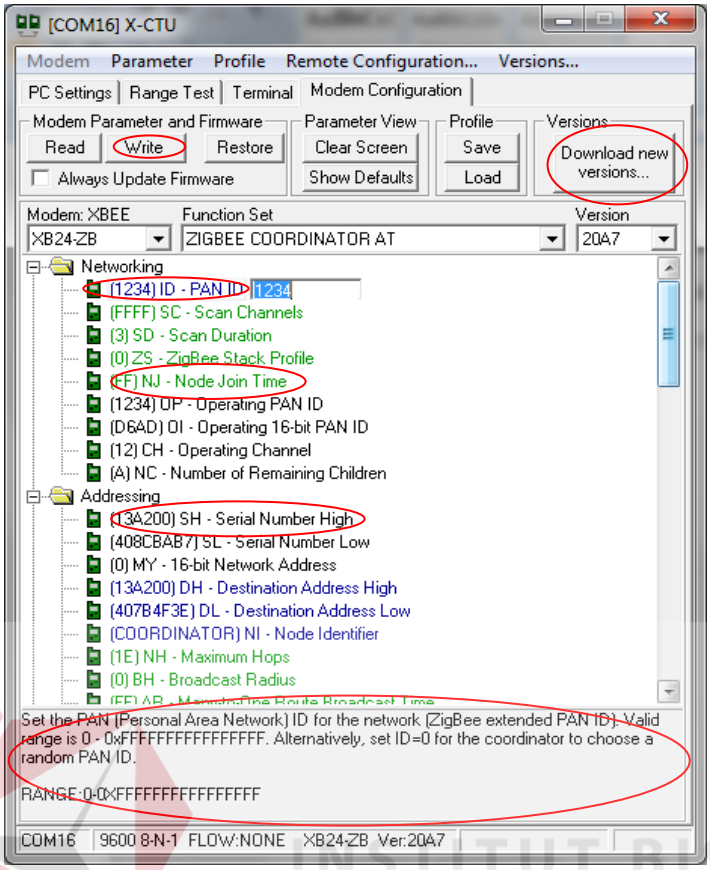


Gambar 2.11 Tab PC Settings



Gambar 2.12 Tab Modem Configuration

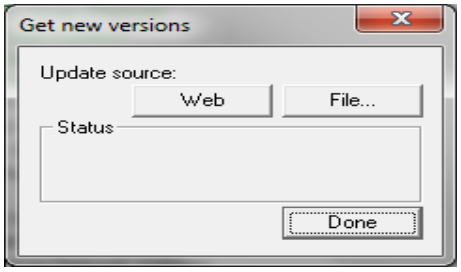
Setelah *firmware* Xbee telah terbaca, terdapat tiga warna dalam pengaturan konfigurasi (Gambar 2.13) yaitu hitam yang berarti *read-only* dan tidak bisa dirubah nilainya, kemudian hijau yang berarti nilai *default* Xbee dan biru yang berarti nilai yang pengguna tentukan sesuai keinginan. Untuk mengubah nilai parameter yang bisa dirubah, klik paramater tersebut kemudian ketik nilai yang baru sesuai keinginan pengguna. Untuk memudahkan pengisian nilai, terdapat deskripsi atau keterangan dalam pengisian nilai pada setiap parameter yang berada pada bagian bawah. Setelah semua nilai – nilai yang baru masuk, maka nilai tersebut dapat disimpan ke memori *non-volatile* pada Xbee. Klik tombol "*Write*" pada bagian *Modem Parameter and Firmware* untuk menyimpan konfigurasi parameter ke memori pada Xbee (Gambar 2.13).



Gambar 2.13 Firmware Xbee Telah Terbaca

c) Meng-*update firmware* pada Xbee

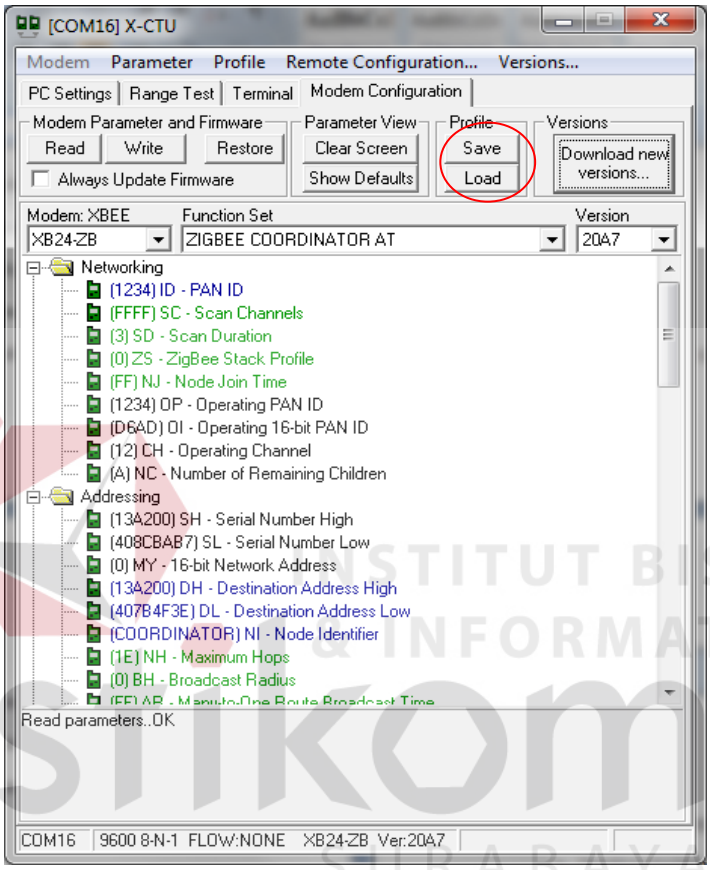
User atau pengguna dapat meng-*update firmware* pada Xbee baik melalui *web* atau menginstalnya dari *file.zip* atau *disk*. Klik "*Download New Versions*" pada bagian *Versions* (Gambar 2.13). Klik tombol *Web* jika ingin meng-*update file firmware* melalui *web* atau klik tombol *File* jika ingin meng-update melalui *file.zip* atau *disk* (Gambar 2.14).



Gambar 2.14 Kotak Dialog *Get New Versions*

d) *Save* atau *load modem profile*

X-CTU dapat menyimpan dan men-*load* profil modem atau konfigurasi yang telah disimpan, ini sangat berguna ketika paramater konfigurasi yang sama ditetapkan pada beberapa Xbee yang berbeda (Gambar 2.15)

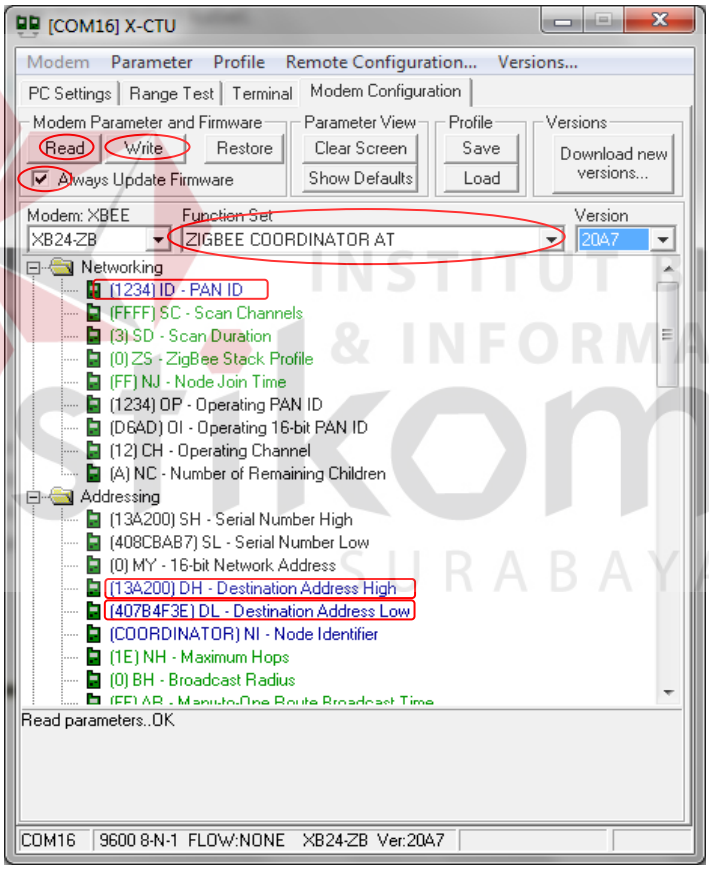


Gambar 2.15 Save dan Load Modem Profile

2.5 Topologi *Point to Point*

Topologi *point to point* adalah topologi yang membangun hubungan langsung antara dua *node* jaringan. Dalam jaringan sensor nirkabel atau WSN ini dapat menggunakan topologi *point to point*. Jenis *node* dan parameter yang harus dikonfigurasi agar dua Xbee dapat berkomunikasi secara *point to point* adalah salah satu *node* harus menjadi *coordinator* dan lainnya menjadi *router* atau *end device*. Klik "*Read*" dan "*Always Update Firmware*" pada *tab modem*

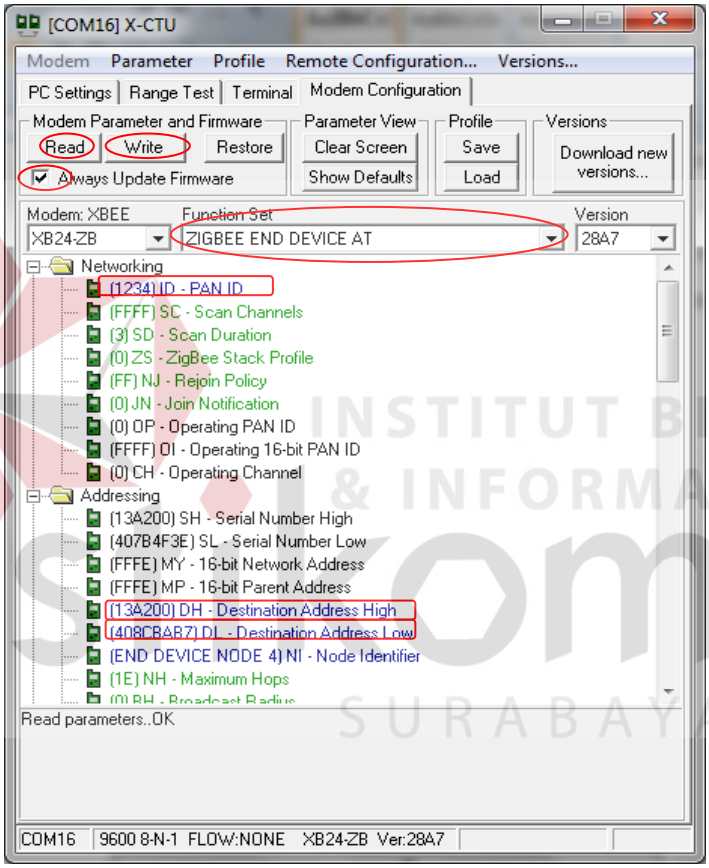
configuration dalam *software* X-CTU untuk dapat membaca modul Xbee, pada bagian *function set* diatur menjadi ZIGBEE *COORDINATOR* AT, selanjutnya parameter pada *node coordinator* yang harus diatur adalah parameter PAN ID, PAN ID dalam satu jaringan (antara dua Xbee) harus sama agar dapat berkomunikasi. Kemudian DH dan DL yang harus diatur sesuai dengan SH dan SL milik *router* atau *end device* yang merupakan *source address*. Jika jenis *node* dan parameter sudah diatur, setelah itu klik "*Write*" untuk menyimpan konfigurasi yang telah diatur ke dalam Xbee. (Gambar 2.16).



Gambar 2.16 Konfigurasi *Node Coordinator*

Pada *node* yang akan dijadikan *router* atau *end device*, pada bagian *function set* diatur menjadi ZIGBEE ROUTER AT ATAU *END DEVICE* AT. Selanjutnya parameter pada *node router* atau *end device* yang harus diatur adalah

parameter PAN ID, PAN ID dalam satu jaringan (antara dua Xbee) harus sama agar dapat berkomunikasi dalam hal ini mengikuti PAN ID *coordinator*. Kemudian DH dan DL yang harus diatur sesuai dengan SH dan SL milik *coordinator* yang merupakan *source address*. Jika jenis *node* dan parameter sudah diatur, setelah itu klik "*Write*" untuk menyimpan konfigurasi yang telah diatur ke dalam Xbee. (Gambar 2.17).

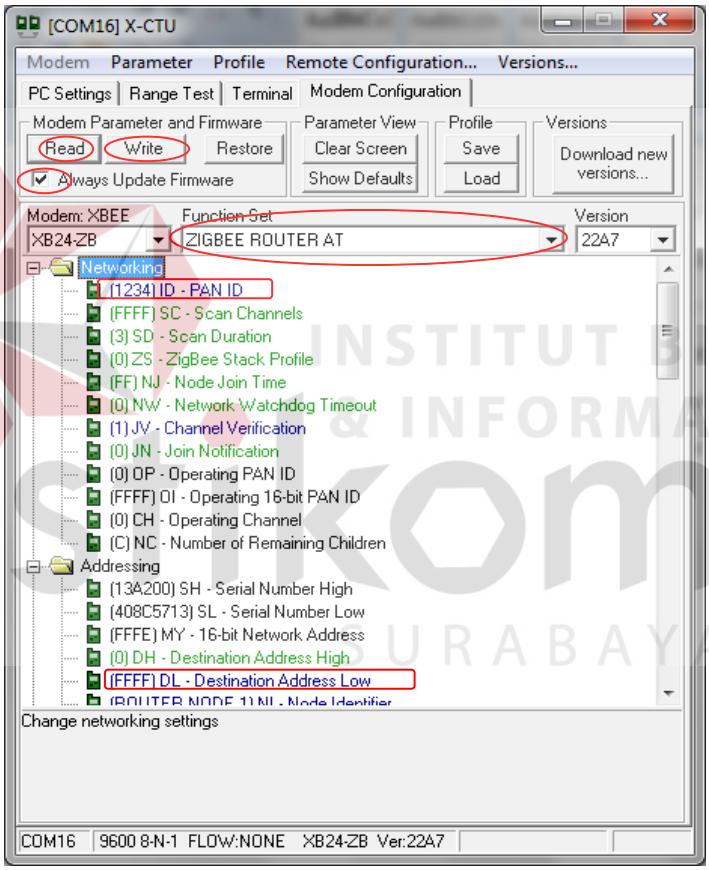


Gambar 2.17 Konfigurasi *Node End device*

2.6 Topologi *Point to Multipoint*

Topologi *point to multipoint* adalah topologi yang membangun hubungan antara beberapa *node*, yang dimana suatu *node* dapat berkomunikasi dengan beberapa *node* secara *broadcast*. Jenis *node* dan parameter yang harus dikonfigurasi agar suatu Xbee dapat berkomunikasi secara *point to multipoint*

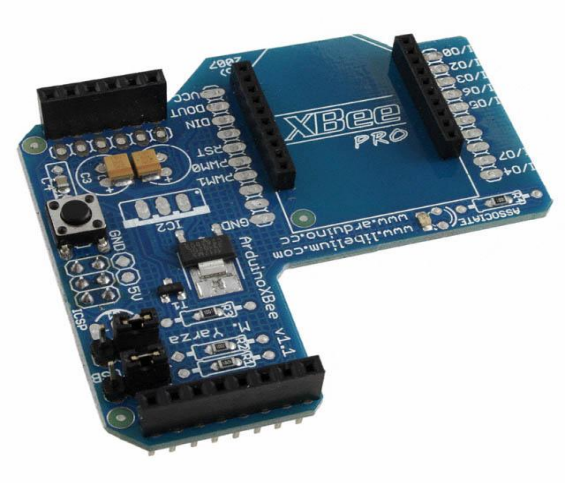
adalah salah satu *node* harus menjadi *coordinator* dan lainnya menjadi *router* atau *end device*. Parameter yang harus diatur hampir sama dengan parameter dalam komunikasi *point to point*, perbedaannya terletak pada DH dan DL *node* yang bersangkutan. Dalam *point to multipoint* ini, DH dan DL diatur nilainya menjadi DH = 0 dan DL = FFFF, FFFF mempunyai arti bahwa data akan dikirim secara *broadcast* sehingga beberapa *node* dalam PAN ID yang sama akan mendapat data tersebut. (Gambar 2.18).



Gambar 2.18 Konfigurasi *Point to Multipoint*

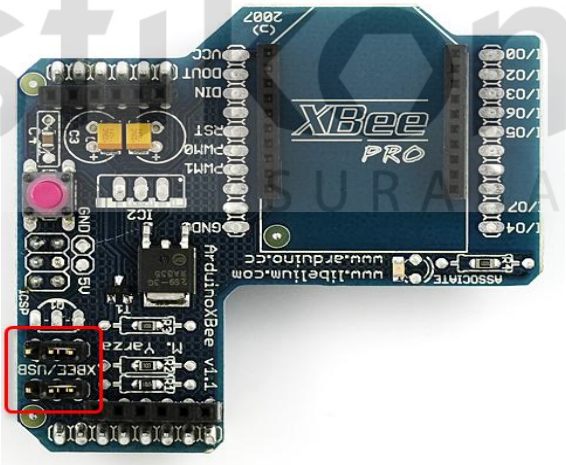
2.7 Xbee *Shield*

Xbee *shield* merupakan suatu *board* yang dapat menghubungkan *board* arduino untuk berkomunikasi secara nirkabel atau *wireless* menggunakan modul Xbee atau Zigbee (arduino.cc). (Gambar 2.19)



Gambar 2.19 Xbee *Shield*

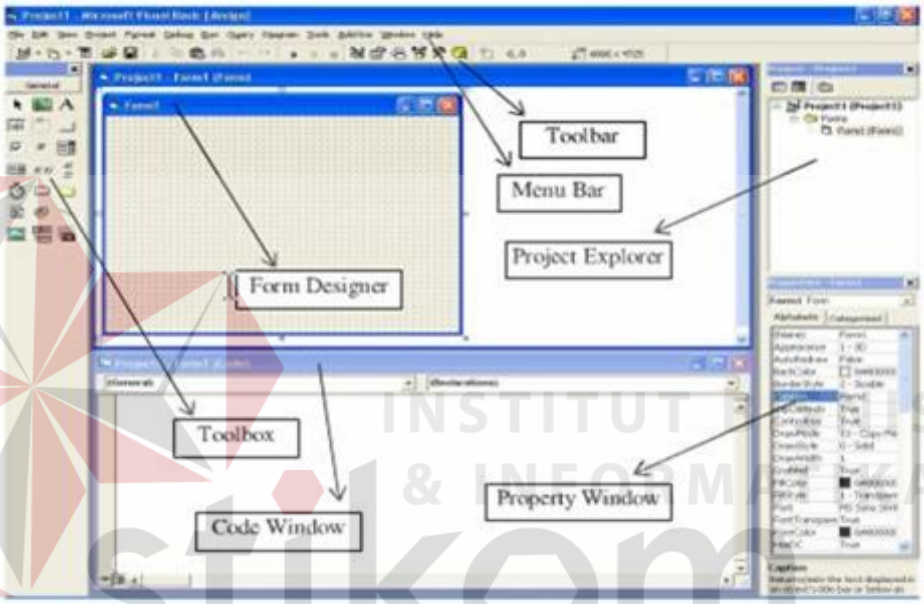
Xbee *shield* memiliki dua jumper terbuat dari plastik yang dapat di *removeable* dari tiga pin pada shied yang berlabel Xbee/USB (Gambar 2.20). Jumper ini menentukan komunikasi serial Xbee agar terhubung pada komunikasi serial antar mikrokontroller atau USB pada *board* arduino. (arduino.cc)



Gambar 2.20 Jumper pada Xbee *Shield*

2.8 Visual Basic

Visual Basic adalah salah suatu *development tools* untuk membangun aplikasi dalam lingkungan *Windows*. Dalam pengembangan aplikasi, *Visual Basic* menggunakan pendekatan *Visual* untuk merancang *user interface* dalam bentuk *form*. Tampilan *Visual Basic* terdapat pada *Integrated Development Environment* (IDE) seperti pada Gambar 2.21.



Gambar 2.21 Tampilan Utama Visual Basic 6.0

Adapun pejelasan 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 anda gunakan untuk mengembangkan user interface.

f) Jendela *Code*, merupakan tempat bagi anda untuk menulis koding. Anda dapat menampilkan jendela dengan menggunakan kombinasi Shift-F7.

g) Jendela *Properties*, merupakan daftar properti-properti object yang sedang terpilih. Sebagai contohnya anda dapat mengubah warna tulisan (*foreground*) dan warna latar belakang (background). Anda dapat menggunakan F4 untuk menampilkan jendela properti.

h) Jendela *Color Palette*, adalah fasilitas cepat untuk mengubah warna suatu object.

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