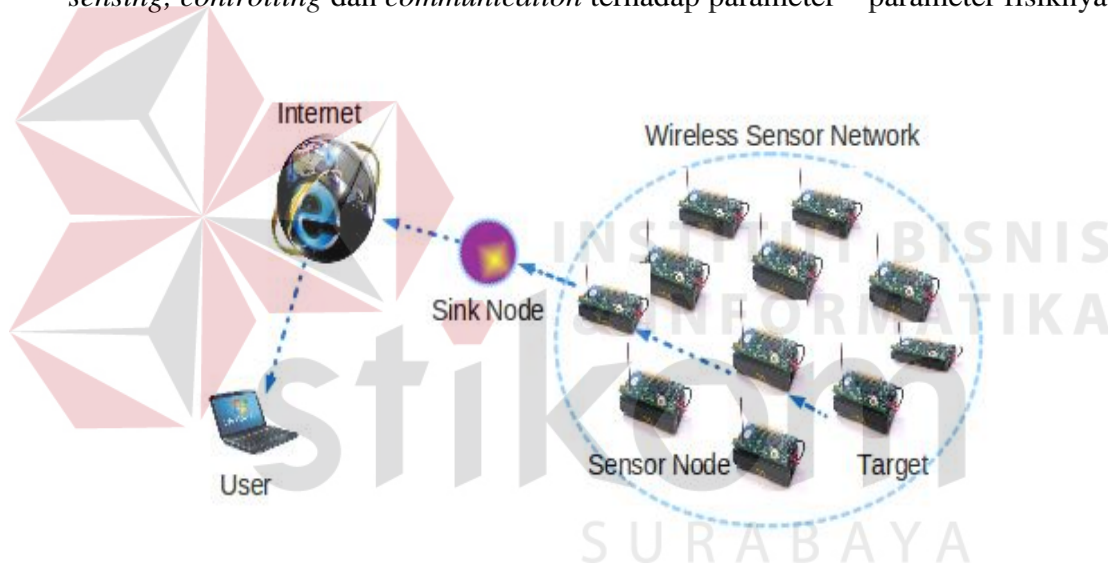


BAB II

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.



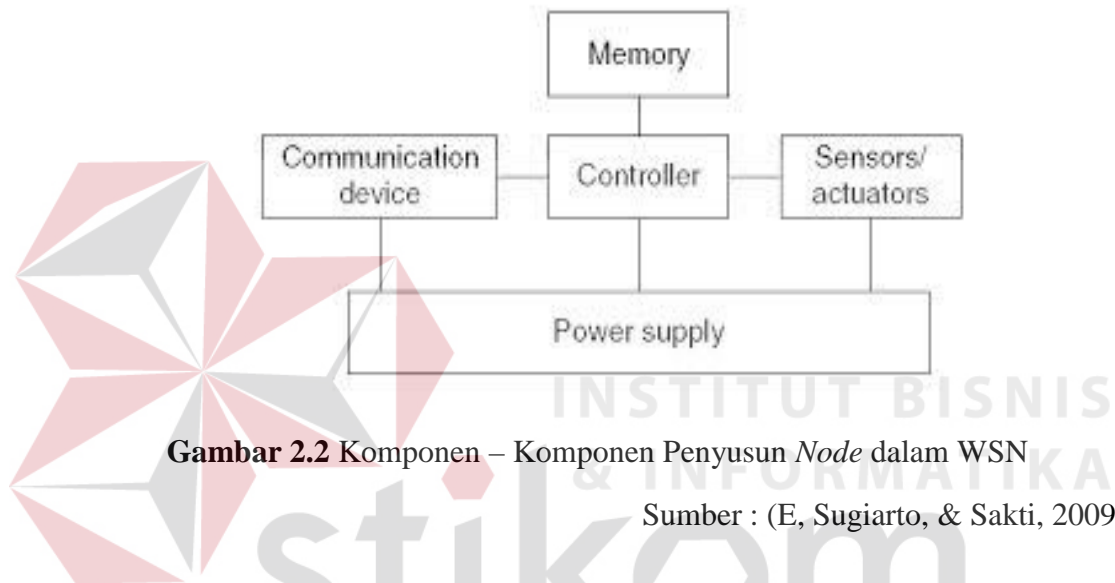
Gambar 2.1 Arsitektur WSN

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

Pada Gambar 2.1 menunjukkan gambaran umum WSN, dapat dilihat *node* sensor yang berukuran kecil tersebar 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 controller/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 dibawah ini.



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, 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, 2013)



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

Sumber : (arduino.cc, 2013)

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, 2013)

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 *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, 2013)

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 dikonfigurasi 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 *mensupport* 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 rangnya 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, 2013)

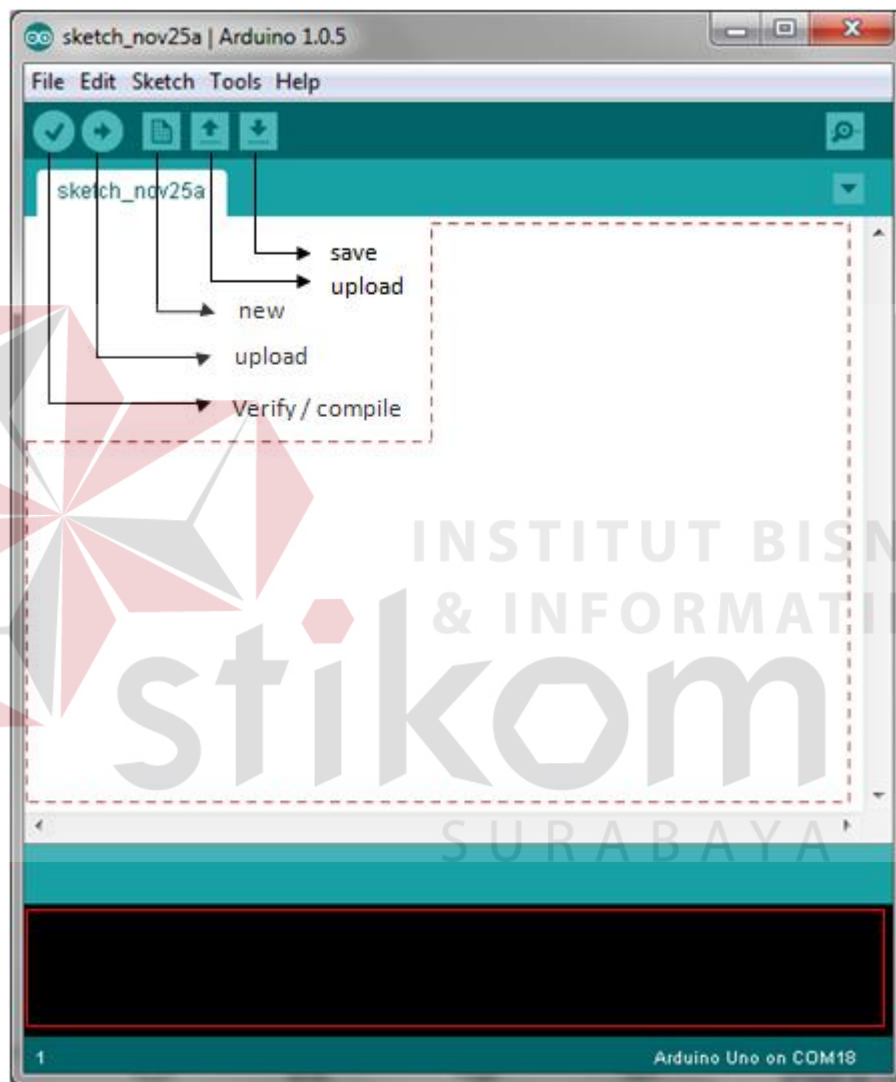
2.3 Software Arduino IDE

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 mikrokontroler adalah kode biner.

3. *Uploader*, sebuah modul yang memuat kode biner dari komputer ke dalam memori mikrokontroler 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 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 Pembandingan

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, 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 menerus 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 Sensor Ultrasonik PING)))

Parallax PING))) *ultrasonic range finder* merupakan sebuah sensor pengukur jarak tanpa kontak langsung, dengan kemampuan jarak ukur 2 cm (0.8 inches) sampai 3 m (3.3 yards) di udara yang memiliki cepat rambat 344 m/s. Sehingga untuk menempuh jarak 1 cm dibutuhkan waktu 29 μ s. Sensor ini hanya memerlukan 1 pin I/O dari mikrokontroler untuk mengontrolnya (Gambar 2.5). (Yunidar, 2009)



Gambar 2.5 Sensor Ultrasonik PING)))

Sensor ini bekerja berdasarkan prinsip pantulan gelombang suara, dimana sensor akan memancarkan gelombang suara yang kemudian menangkap pantulannya kembali dengan perbedaan waktu sebagai dasar pengindraannya. Perbedaan waktu antara gelombang suara dipancarkan dengan ditangkapnya

kembali gelombang suara tersebut adalah berbanding lurus dengan jarak atau tinggi objek yang memantulkannya. Jenis objek yang dapat diindera diantaranya adalah: objek padat, cair, butiran maupun tekstil. Sensor ini memiliki 3 pin, yang masing - masingnya dihubungkan ke *Ground*, *Vcc (5V)* dan pin ketiga merupakan pin *I/O (SIG)* dihubungkan ke arduino uno. (Yunidar, 2009)

2.6 Real Time Clock (RTC)

Real Time Clock merupakan suatu *chip (IC)* yang memiliki fungsi sebagai penyimpan waktu dan tanggal. RTC DS1307 (Gambar 2.6) merupakan *Real-time clock (RTC)* yang dapat meyimpan data-data detik, menit, jam, tanggal, bulan, hari dalam seminggu, dan tahun valid hingga 2100. 56-byte, battery-backed, RAM *nonvolatile (NV)* RAM untuk penyimpanan. RTC DS1307 merupakan *Real-time clock (RTC)* dengan jalur data paralel yang memiliki Antarmuka *serial Two-wire (I2C)*, Sinyal luaran gelombang-kotak terprogram (*Programmable squarewave*), Deteksi otomatis kegagalan-daya (*power-fail*) dan rangkaian switch, Konsumsi daya kurang dari 500nA menggunakan mode baterai cadangan dengan operasional osilator. (Arrosyid, Tjahjono, & Epyk Sunarno, 2009)



Gambar 2.6 RTC DS1307

2.7 Xbee Series 2 Chip Antenna dan Xbee Pro Series 2 Wire 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 antenna, salah duanya adalah *chip antenna* dan *wire antenna*. *Chip antenna* merupakan suatu chip keramik yang terletak pada *board* modul Xbee, bentuknya lebih kecil. *Chip antenna* memiliki pola radiasi *cardoid*, yang artinya sinyal dilemahkan dalam berbagai arah dan sangat baik digunakan dalam area yang tidak terlalu besar atau kecil. Sedangkan *wire antenna* merupakan suatu antenna kawat yang terletak pada board modul Xbee, *wire antenna* memiliki pola radiasi *omndirectional* yang artinya jarak transmisi maksimum hampir sama pada semua arah ketika antenna tersebut tegak lurus terhadap modul. Gambar 2.7 merupakan gambar dari modul Xbee *series 2 chip antenna* dan Gambar 2.8 merupakan gambar dari modul Xbee *series 2 wire antenna*. (Faludi, 2011)



Gambar 2.7 Xbee Series 2 Chip Antenna



Gambar 2.8 Xbee Series 2 Wire Antenna

Berikut adalah spesifikasi dari modul Xbee pro *series 2 chip antenna* (Inc, XBee Series 2 OEM RF Modules, 2007):

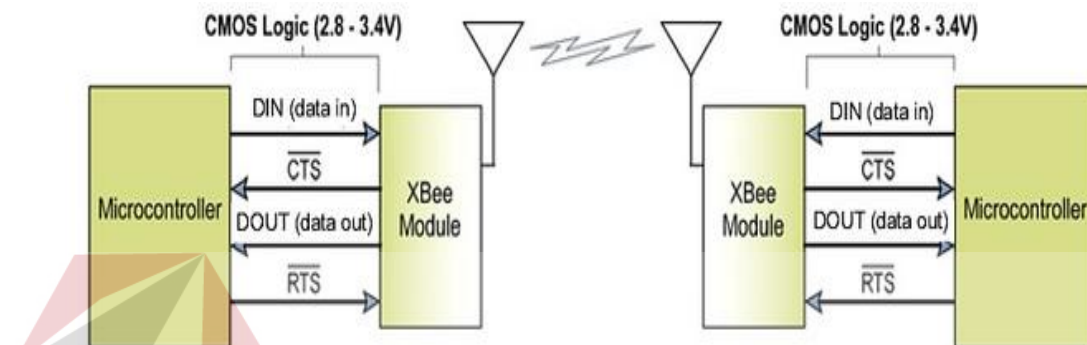
- a) Jarak jangkauan *indoor* 133 ft atau 40 meter.
- b) Jarak jangkauan *outdoor line of sight* 400 ft atau 120 meter.
- c) Transmit *power output* 2 mW (+ 3 dbm).
- d) Radio Frekuensi data *rate* 250 Kbps.
- e) Frekuensi 2.4 GHz.
- f) *Receiver sensitivity* -98 dbm (1 % *packet error rate*).
- g) Antena menggunakan *chip antenna*.

Berikut adalah spesifikasi dari modul Xbee pro *series 2 wire antenna* (Inc, 2012):

- a) Jarak jangkauan *indoor* 300 ft (90 meter) dan 200 ft (60 meter)
- b) Jarak jangkauan *outdoor line of sight* 2 miles atau 3200 meter dan 5000 ft atau 1500 meter (variant lainnya).
- c) Transmit *power output* 50 mW (+ 17 dbm).
- d) Radio Frekuensi data *rate* 250 Kbps.
- e) Frekuensi 2.4 GHz.
- f) *Receiver sensitivity* -102 dbm.
- g) Antena menggunakan *wire antenna*.

2.8 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 Receiver transmitter*). Gambar 2.9 menunjukkan diagram sistem aliran data secara UART. (Inc, 2007)



Gambar 2.9 Diagram Sistem Aliran Data UART pada Xbee

2.9 AT / Transparent Mode

Dalam mode transparent/AT, modul Xbee bertindak sebagai pengganti *serial line*. Semua data UART (*Universal Asynchronous Receiver transmitter*) 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)

2.10 Coordinator, Router, dan End Device

Pada jaringan ZigBee terdapat tiga jenis *node*, yaitu:

1. Coordinator

Sebuah *node* yang berfungsi untuk membentuk suatu jaringan. *Coordinator* bertanggung jawab untuk membangun *operating channel* dan PAN

(*Personal Area Network*) ID untuk suatu jaringan. *Coordinator* dapat membentuk suatu jaringan dengan mengizinkan *router* dan *end device* untuk bergabung dalam jaringan tersebut. Setelah jaringan terbentuk, fungsi *coordinator* seperti *router* (dapat berpartisipasi dalam *routing* paket dan menjadi sumber atau tujuan untuk paket data). (Inc, 2007)

2. *Router*

Sebuah *node* yang membuat / menjaga informasi jaringan dan menggunakan informasi ini untuk menentukan rute untuk paket data. Agar suatu *router* dapat mengizinkan *router* lain dan *end device* untuk bergabung, sebelumnya *router* tersebut harus terlebih dahulu bergabung dengan jaringan yang ada. *Router* dapat berpartisipasi dalam *routing* paket dan menjadi sumber atau tujuan untuk paket data. (Inc, 2007)

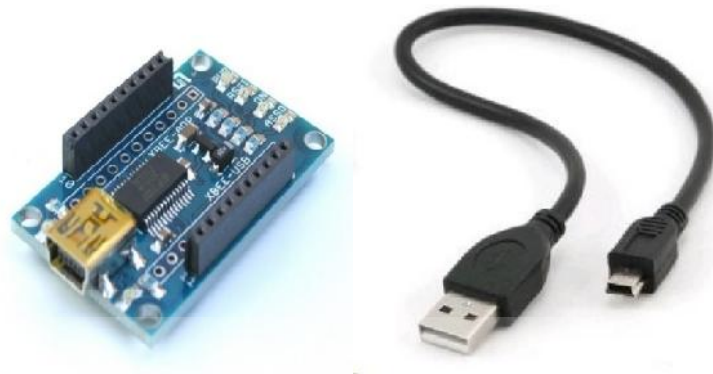
3. *End device*

End device harus selalu berinteraksi atau terhubung dengan *coordinator* atau *router* untuk dapat menerima dan mengirimkan data. *End device* juga dapat menjadi sumber atau tujuan untuk paket data tetapi tidak bisa untuk menentukan rute paket data. (Inc, 2007)

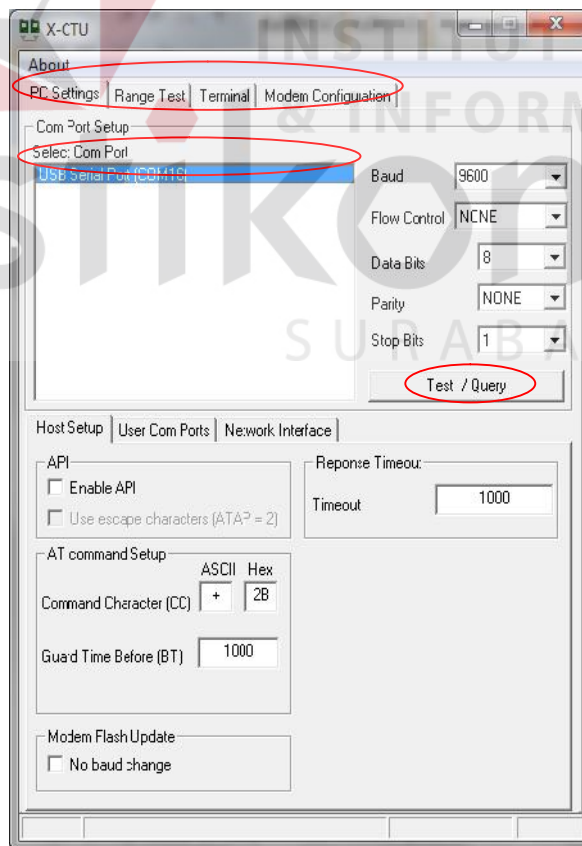
2.11 Xbee USB Adapter dan Software X-CTU

Xbee usb *adapter* (Gambar 2.10) merupakan alat untuk menghubungkan modul Xbee ke komputer dengan kabel mini usb (Gambar 2.10) dan selanjutnya dapat dikonfigurasi menggunakan *software* X-CTU (Gambar 2.11). *software* X-CTU merupakan *software* yang digunakan untuk mengkonfigurasi Xbee agar dapat berkomunikasi dengan Xbee lainnya. 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. Xbee dapat berkomunikasi point to point dan point to multipoint (broadcast).



Gambar 2.10 Xbee Usb Adapter dan Kabel Mini Usb



Gambar 2.11 Tampilan Software X-CTU

Pada Gambar 2.11 *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 2.12.



Gambar 2.12 Kotak Dialog Respon

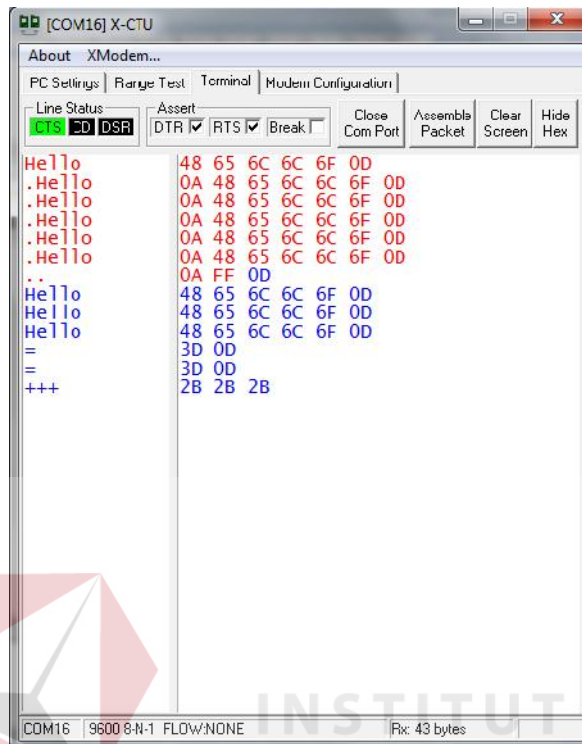
2. *Range Test*

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.

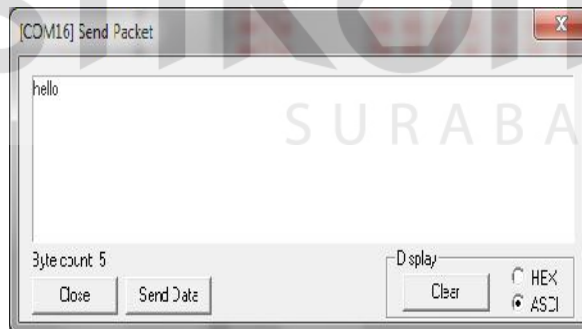
3. *Terminal*

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.2.14).



Gambar 2.13 Tab Terminal



Gambar 2.14 Assemble Packet

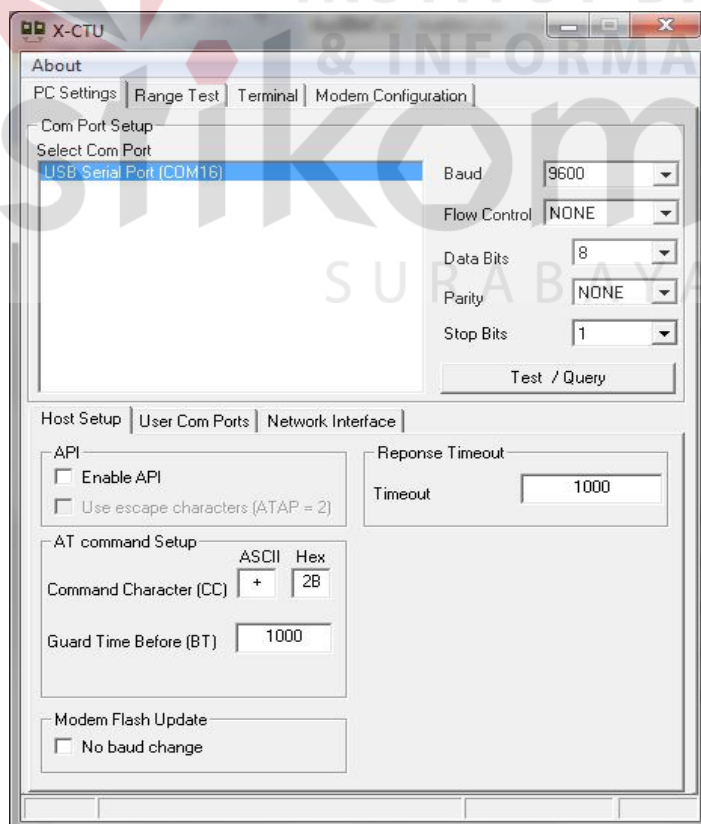
Pada Gambar 2.14 area putih dalam terminal *tab* ini berisi data informasi komunikasi yang terjadi antara 2 Xbee atau lebih. Teks yang berwarna biru merupakan teks yang telah diketik pengguna dan diarahkan ke *port* serial Xbee sedangkan teks merah merupakan data yang masuk dari *port* serial Xbee.

4. Modem Configuration Tab

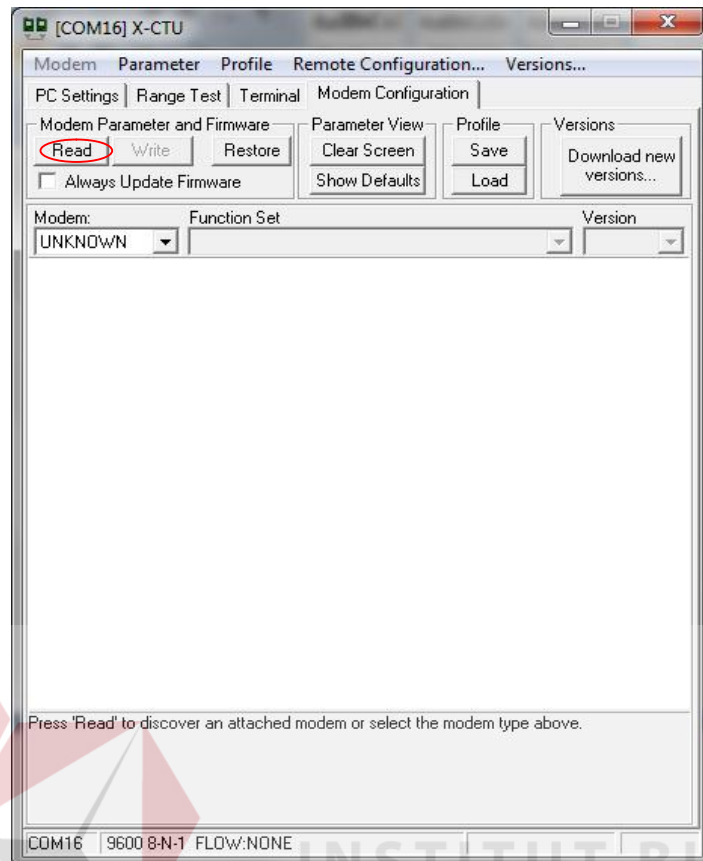
Pada tab ini, pengguna dapat melakukan pemrograman pada pengaturan *firmware* Xbee dan merubah versi *firmware*nya melalui *Graphical User Interface* (GUI). Terdapat 4 fungsi dasar dalam *modem configuration tab*, yaitu:

- a) Menyediakan fasilitas GUI untuk mengatur *firmware* pada Xbee.
- b) *Read* dan *write* *firmware* ke mikrokontroller 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 2.15. Pada *tab Modem Configuration*, klik “*Read*” pada bagian *Modem Parameter and Firmware* (Gambar 2.16).

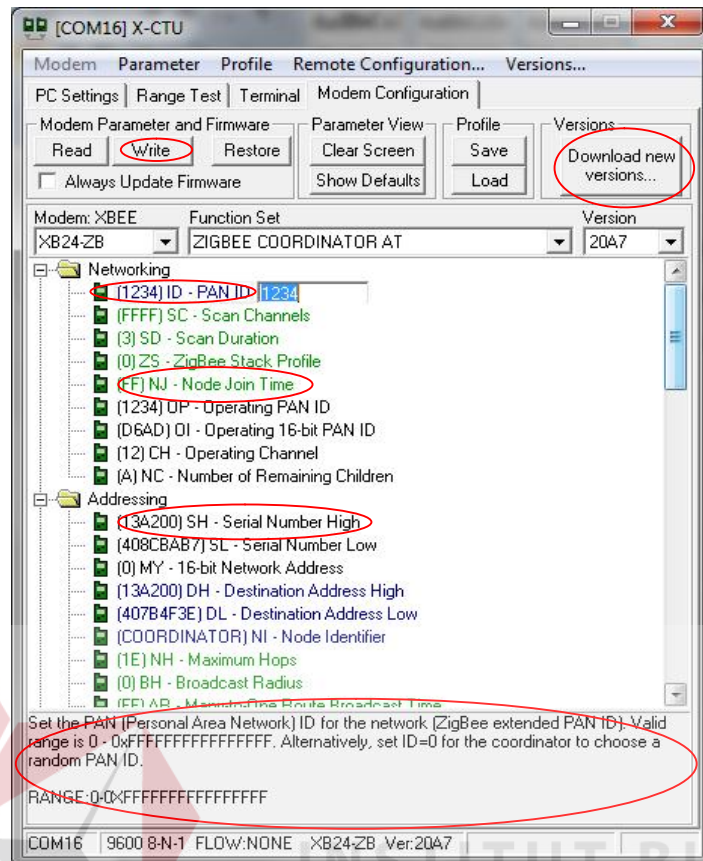


Gambar 2.15 *Tab PC Settings*



Gambar 2.16 Tab Modem Configuration

Setelah *firmware* Xbee telah terbaca, terdapat tiga warna dalam pengaturan konfigurasi (Gambar 2.17) 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 parameter 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 2.17).



Gambar 2.17 Firmware Xbee Telah Terbaca

c) *Meng-update firmware pada Xbee*

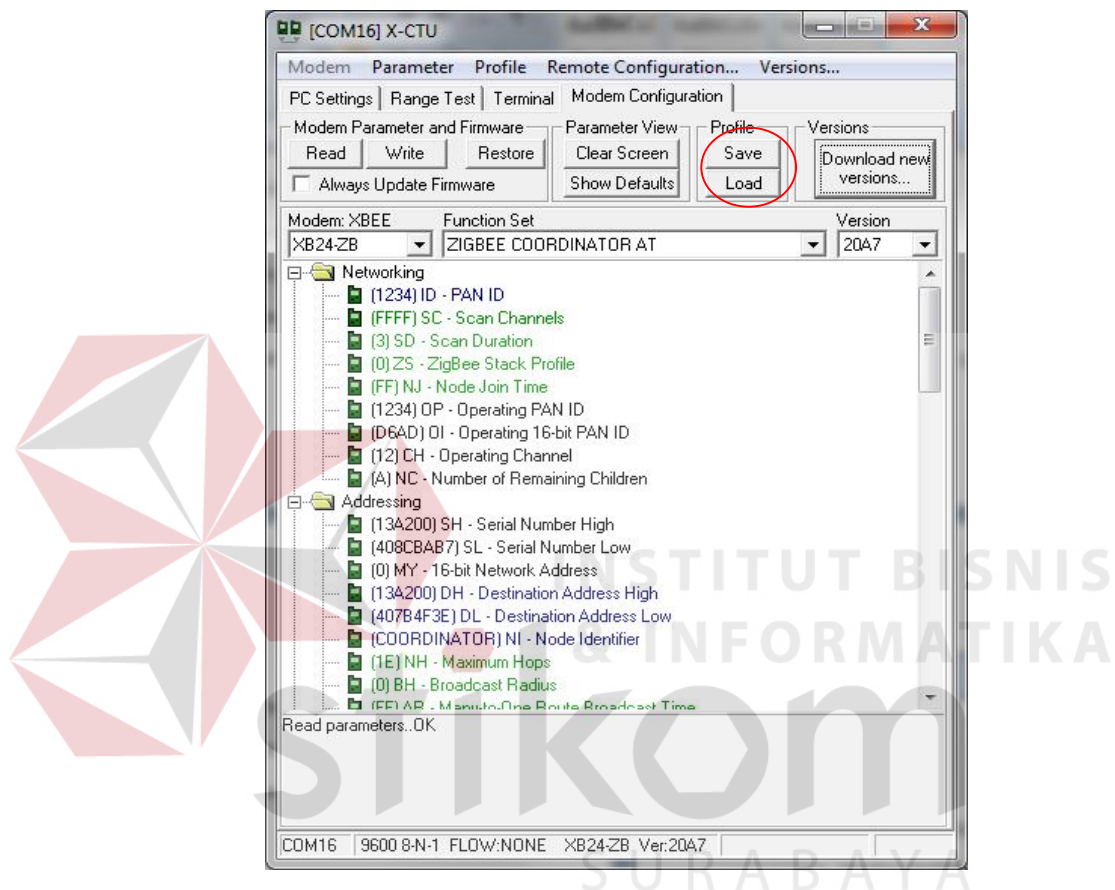
User atau pengguna dapat meng-update firmware pada Xbee baik melalui web atau menginstalnya dari file.zip atau disk. Klik “Download New Versions” pada bagian *Versions* (Gambar 2.17). Klik tombol *Web* jika ingin meng-update file firmware melalui web atau klik tombol *File* jika ingin meng-update melalui file.zip atau disk (Gambar 2.18).



Gambar 2.18 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 parameter konfigurasi yang sama ditetapkan pada beberapa Xbee yang berbeda (Gambar 2.19).



Gambar 2.19 *Save dan Load Modem Profile*

2.12 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*. Topologi *point to point* ini merupakan topologi jaringan yang paling sederhana tetapi dapat digunakan secara luas.

2.13 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*. Karena topologi ini membangun hubungan antara beberapa *node* yang dapat berkomunikasi dengan beberapa *node* lainnya secara *broadcast*, maka topologi ini dapat digunakan juga sebagai jaringan *mesh*.

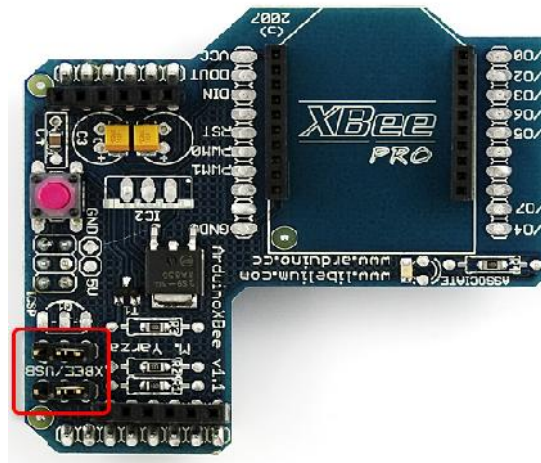
2.14 Xbee Shield

Xbee *shield* merupakan suatu *board* yang dapat menghubungkan *board* arduino untuk berkomunikasi secara nirkabel atau *wireless* menggunakan modul Xbee atau Zigbee (Gambar 2.20). (arduino.cc, 2013)



Gambar 2.20 Xbee Shield

Xbee *shield* memiliki dua jumper terbuat dari plastik yang dapat di *removeable* dari tiga pin pada *shield* yang berlabel Xbee/USB (Gambar 2.21). Jumper ini menentukan komunikasi serial Xbee agar terhubung pada komunikasi serial antar mikrokontroller atau USB pada *board* arduino. (arduino.cc, 2013)



Gambar 2.21 Jumper pada Xbee Shield