BAB III

METODE PENELITIAN DAN PERANCANGAN SISTEM

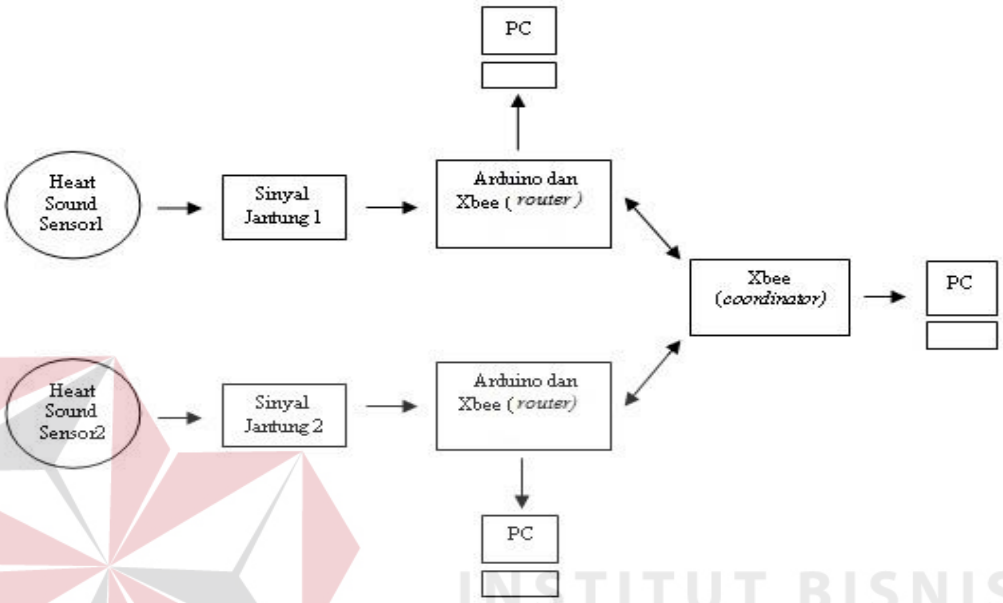
3.1 Metode Penelitian

Metode penelitian yang digunakan pada perancangan ini adalah komunikasi data multipoint *wireless sensor network*. Perancangan terdiri dari 2 buah *node* dengan 1 *base station* yang dikomuikasikan secara bersamaan dengan kecepatan data 115200bps. Dalam berkomunikasi, 2 node yang terdiri dari Xbee mengirim ke satu *base station* sehingga alamat *destination* pada masing - masing *node* nilainya sama dengan nomor seri Xbee bagian *base station*.

Untuk melakukan komunkasi *multipoint*, 2 buah *node* diset menjadi *router* (*router* 1 dan *router* 2) sementara pada *base station* diset sebagai *coordinator*, alamat *destination node* (DH (*Destinqtion High*) dan DL (*Destination Low*)) adalah alamat *source coordinator* (SH (*Source High*) dan SL (*Source Low*)). Data yang diterima oleh *coordiator* selanjutnya dipilah untuk mengetahui arah data dari *node* pengirim, dalam hal ini masing - masing *node* mengirimkan data dengan protokol data yang terdiri dari ID *node*, dan data yang dibawa sehingga pemilahan data dapat diketahui dengan memeriksa ID yang masuk pada deretan protokol pengirim (*node*). Adapun data yag dikirim berupa nilai tegangan dari sensor *heartbeat* (jantung) yang dipresentasikan dalam bentuk ASCII yang selanjutnya ditampilkan dalam bentuk grafik pada *form* perangkat lunak VB (Visual Basic). Data tersebut selanjutnya disimpan dalam file txt sebagai arsip pembacaan yang dapat dibuka sewaktu - waktu.

3.2 Model Perancangan

Pada perancangan ini penulis menggambarkan perancangan sistemnya seperti pada gambar 3.1 berikut.



Gambar 3.1 Gambar Perancangan

Dari Gambar 3.1 didapatkan bahwa setiap *node* WSN memiliki tugas berbeda-beda seperti berikut:

a) *Node Router* 1

Pada *node* ini, *node* bertanggung jawab sebagai pencatat hasil auskultasi sinyal jantung pada pasien pertama, dan mengirimkan data pada *node coordinator* sesuai dengan protokol yang telah dibuat melalui modul arduino.

b) *Node Router* 2

Pada *node* ini, *node* bertanggung jawab sebagai pencatat hasil auskultasi sinyal jantung pada pasien kedua, dan mengirimkan data pada *node coordinator* sesuai dengan protokol yang telah dibuat melalui modul Arduino.

c) *Node coordinator*

Pada *node* ini, *node* bertanggung jawab atas penerima data yang telah dikirimkan oleh kedua *node* sensor (*router).* Data yang diterima oleh *node* ini masih belum diolah, tapi data yang diterima sesuai dengan protokol pengiriman data. Pada *node coordinator* data langsung dikirimkan ke *end device* / PC tanpa pengolahan melalui modul arduino, hal ini dikarenakan proses pengolahan data dilakukan pada *end device.*

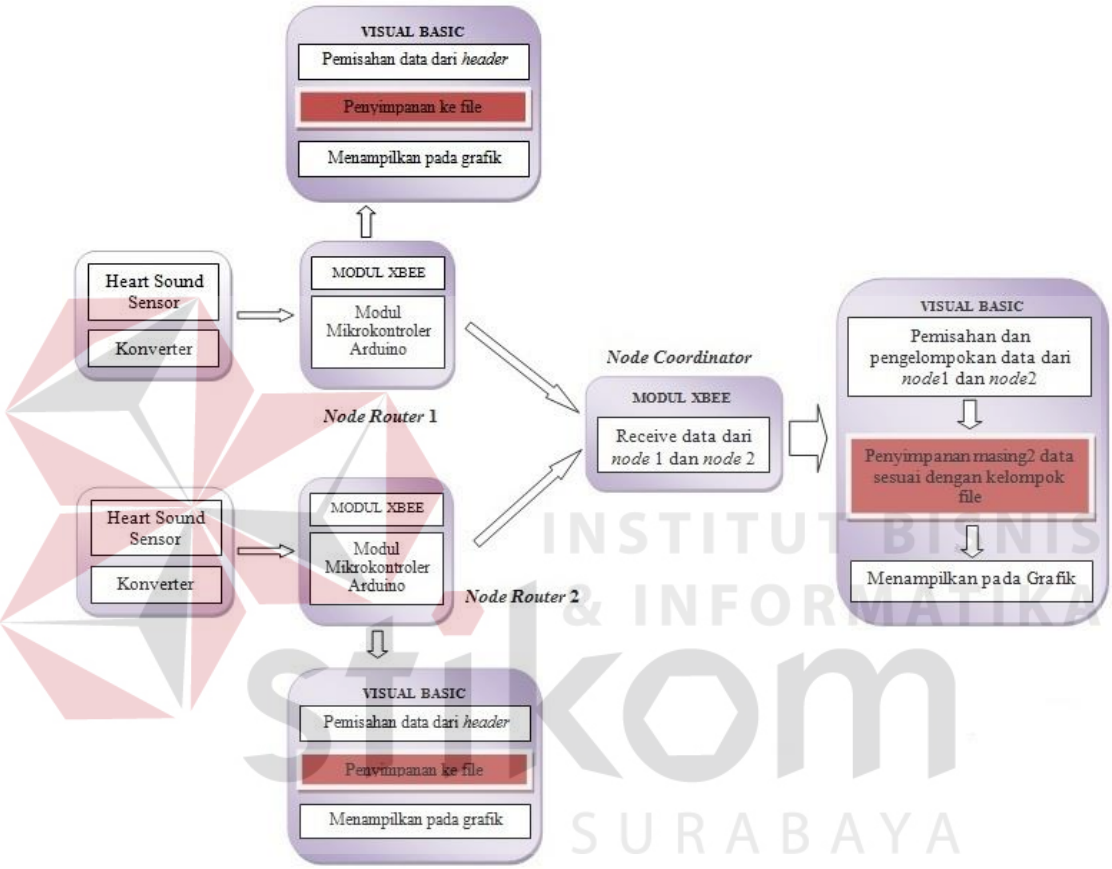
d) *End device* / PC (*Personal Computer*)

Terdapat 3 *end device* yang masing – masing berkomunikasi secara *unicast* (*point to point*) dengan *node coordinator,* dan dengan *router.* Pada *end device* yang tersambung dengan *node coordinator* digunakan oleh user untuk melihat hasil auskultasi dari *node* sensor 1 dan *node* sensor 2, dan data yang diperoleh falid (tidak tertukar). Dan dilakukannya pemisahan data dan pengelompokan data sinyal asukultasi berdasarkan asal data. Hal ini dapat dilakukan dengan melihat ID yang sudah diberikan pada saat pengiriman data. Sedangkan penyimpanan data pada masing – masing *end device* yang terhubung dengan *router* digunakan untuk data pembanding antara data yang dikirim dengan data

yang diterima oleh *node coordinator*. Agar dapat diketahui berapa besar *througput*, berapa data yang *loss*, dan *delay* ketika sistem ini dijalankan.

3.3 Perancangan Sistem

Adapun perancangan blok diagram ditunjukkan sebagaimana gambar 3.2:



Gambar 3.2 Blok Diagram Sistem

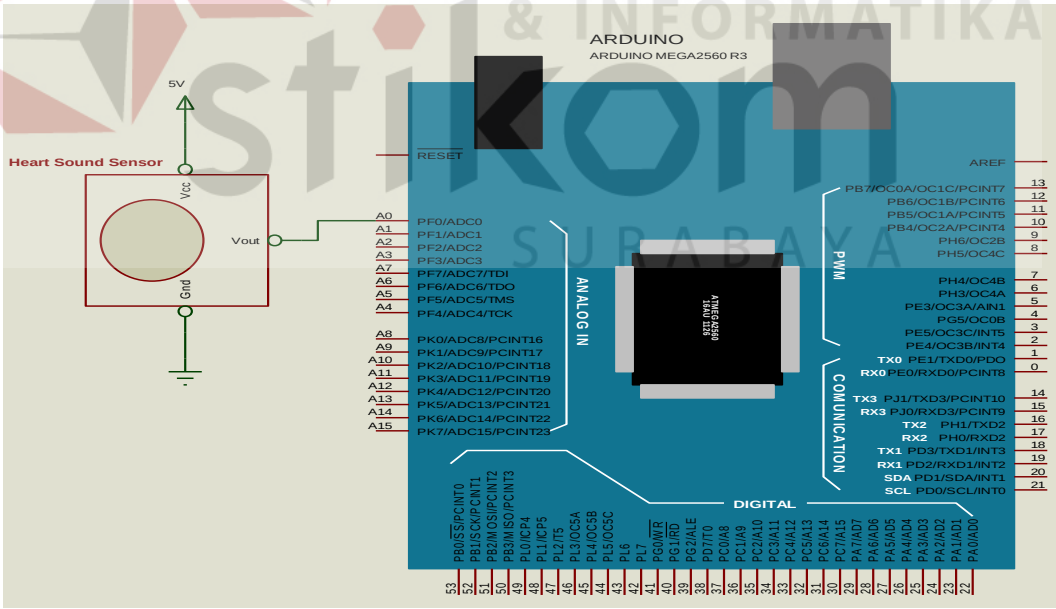
Dalam tugas akhir ini, penulis hanya akan memfokuskan penjelasan data yang dikirim dari dua *node router* ke *node coordinator*. Dan juga hasil unjuk kerja

jaringan pada transmisi sinyal auskultasi jantung dari dua *node* ke satu *node coordinator*.

3.4 Peracangan perangkat keras

3.4.1 Perancangan sensor jantung

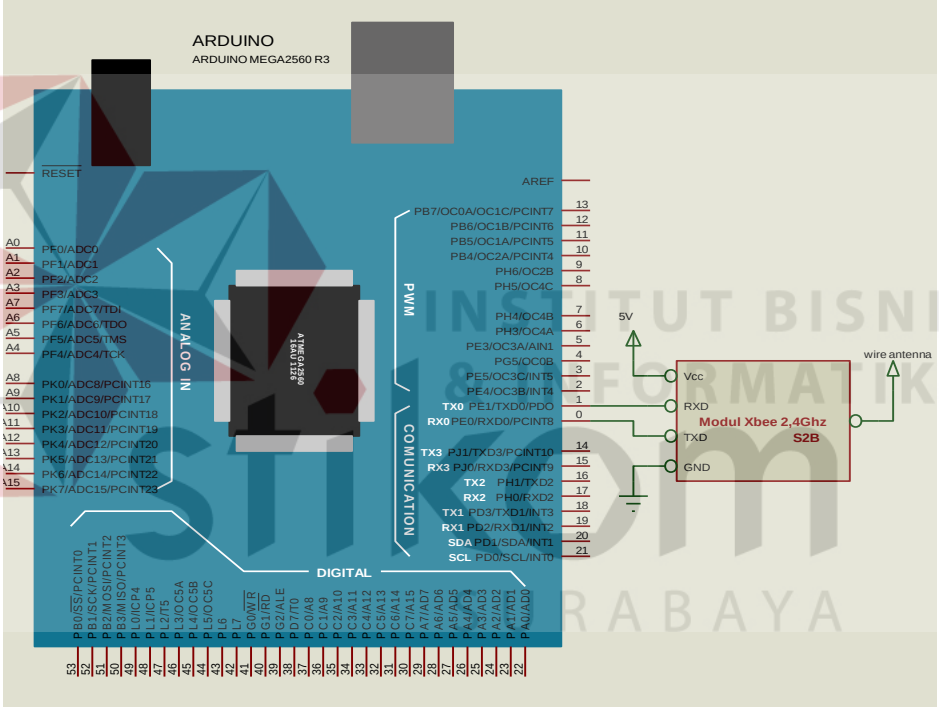
Untuk dapat mendeteksi adanya detak jantung pasien secara elektronik, maka dibutuhkan sensor. Sensor yang digunakan pada penelitan transimsi sinyal auskultasi jantung ini adalah *Heart Sound Sensor*. Sensor ini telah dilengkapi degan pengkondisi sinyal dan fiter yang bertugas meredam dan mengolah sinyal jantung dan mengkonversinya dalam bentuk tegangan. dengan demikian keluaran dari sensor *Heart Sound sensor* dapat langsung dibaca melalui ADC internal pada modul Arduino Mega2560. Adapun perancangan rangkaian *heart sound sensor* ditunjukkan pada gambar 3.3



Gambar 3.3 Hubungan Rangkaian Heart sound sensor dan Arduino

3.4.2 Perancangan rangkaian Xbee Zigbee S2B

Agar modul arduino dapat berkomunikasi secara serial *wireless* dengan perangkat lain, maka dibutuhkan rangkaian *wireless* yang dalam perancangan ini menggunakan modul Zigbee S2B. modul zigbee dapat berkomunikasi *wireless* dan diakses menggunakan komunikasi serial TTL (*Time to Live*). Adapun port serial yang digunakan untuk pengendalian dan pembacaan modul Xbee adalah TX0 dan RX0 pada modul arduino sebagaimana ditunjukkan pada gambar 3.4:

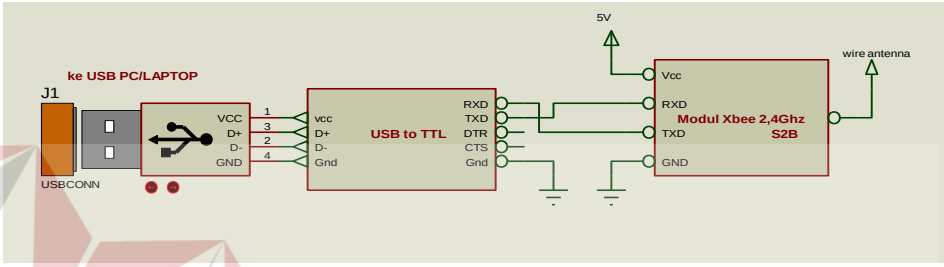


Gambar 3.4 Hubungan Rangkaian Xbee dan arduino

3.4.3 Perancangan rangkaian USB to serial Xbee

Untuk dapat menerima data serial hasil pengiriman dari *node* 1 dan *node* 2 atau perangkat *wireless* Xbee pengirim, maka pada bagian penerima juga

dibutuhkan Xbee. Sementara itu agar hasil pembacaan dan pengiriman data pada Xbee dapat diproses menggunakan laptop atau PC, maka dibutuhkan konverter USB to serial. Untuk itu pada perancangan ini digunakan modul USB to serial Xbee yang difungsikan khusus untuk menjembatani antarmuka UART antara komputer dengan Xbee. Adapun rangkaian modul USB to serial Xbee ditunjukkan pada gambar 3.4:



Gambar 3.5 Rangkaian modul USB to Serial Xbee pada *base station*

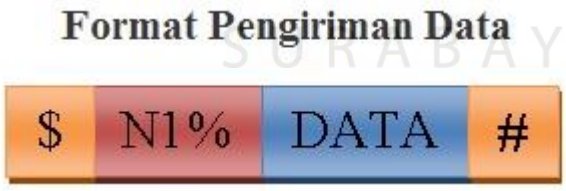
Sementara itu bentuk fisik dari modul USB to serial Xbee ditunjukkan pada gambar 3.6

Gambar 3.6 modul USB to serial Xbee

3.4.4 Arduino 2560

Pada gambar 3.2 terdapat 2 arduino 2560 yang memiliki fungsi yang sama yaitu membaca sensor yang memiliki nilai analog, pembacaan data dilakukan dengan cara inputan yang berasal dari sensor diletakkan pada PORT A0, untuk membaca nilai dari sinyal analog tersebut digunakan fungsi *ReadAnalog* didalam modul arduino.

Pada modul arduino juga dilakukan pemberian identitas pada data yang akan ditransmisikan. Yang artinya data yang dikirim mendapatkan tambahan identitas *node*. Dari tambahan itu yang akan membuat *node* dapat mengenali asal data tersebut. Contoh pemberian identitas yaitu diberi identitas N1 untuk *node router* 1, dan N2 untuk *node router* 2. Identitas ini akan dikirimkan bersama dengan inti data yang *node router* kirimkan kepada *node coordinator*. Pada isi data yang diterima *router coordinator* nantinya terdapat simbol N1 (untuk data dari *node router* 1) atau N2 (untuk data dari *node router* 2) selanjutnya diikuti inti pesan yang dikirim masing – masing *node router*.



Gambar 3.7 *Form*at Pengiriman Data

Berikut penjelasan dari gambar 3.7 :

1. $: penanda awal pengiriman data
2. NI : ID darimana data berasal

3. DATA : data sinyal auskultasi jantung yang dikirimkan
4. # : penanda akhir pengiriman data
5. % : digunakan sebagai pemisah data dengan header

Hal tersebut dibuat untuk memudahkan dalam pemisahan data pada saat penerimaan data pada *coordinator*. Selanjutnya arduino mengirimkan in*form*asi yang dipancar melalui pemancar data zigbee.

3.4.5 Xbee

Untuk mengirimkan data dari masing – masing *node* ke *coordinator* diperlukan sebuah pemancar data. dalam penelitian ini penulis menggunakan Xbee Series 2 untuk pemancar data. Konfigurasi yang dilakukan pada Xbee sangat penting, agar data dapat dikirimkan ke alamat yang sesuai.

Untuk mengkonfigurasi Xbee tersebut dibutuhkan sebuah *software*. *Software* yang biasa digunakan untuk mengkonfigurasi Xbee salah satunya ialah X-CTU.

Xbee dikonfigurasi untuk menjadi *end device* dalam mode AT untuk Xbee yang terdapat pada *node router* dan *coordinator* dalam mode AT. Dalam mengkonfigurasi Xbee *series* 2 hal yang terpenting ialah mengisi nilai PAN ID, DH dan DL.

Langkah pertama untuk dapat berkomunikasi dalam satu jaringan, maka PAN ID antar Xbee harus diisi dengan nilai yang sama. Langkah kedua yaitu mengisi DH dengan ID yang terdapat pada Xbee dan DL dengan nilai yang sesuai dengan nilai DL pada Xbee yang digunakan sebagai *node coordinator*. Hal ini

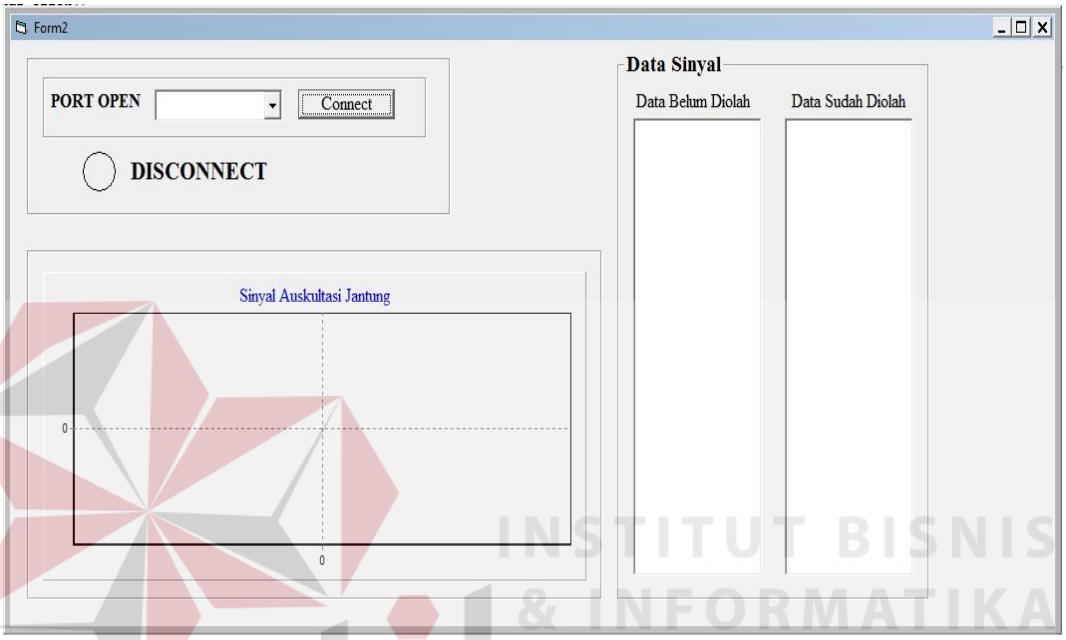
dilakukan agar Xbee yang digunakan pada *node router* hanya berkomunikasi dengan Xbee *coordinator*.

3.4.6 Visual Basic

Visual basic pada komputer atau *end device* berfungsi untuk mengolah data yang dikirimkan oleh *node coordinator*. Data yang diterima tersebut masih berupa sekumpulan in*form*asi dan kode yang masih lengkap yang berup *header* dan data (sesuai dengan protokol), sehingga diperlukan pemisahan data serta pengelompakan pada data tersebut agar didapatkan sebuah data beserta in*form*asi yang diinginkan dari data tersebut. Seperti yang dijelaskan pada gambar 3.3 pengelompokan data sesuai dengan kode yang terdapat pada satu paket data (N1 atau N2), selanjutnya data yang sudah dipisah di simpan sesuai dengan pengelompokan data. Hal ini dilakukan agar data yang diperoleh nantinya dapat dianalisa, sehingga dapat diketahui kemapuan algoritma dari sistem transmisi auskultasi ini. Selanjutnya dari data yang telah dikelompokkan dan dipisah ditampilkan pada sebuah grafik agar dapat dilihat oleh *user*. Pada *node coordinator* terdapat 2 penelitian yang akan dilakukan, yaitu penelitian untuk penerimaan data secara *real time* dan tidak *real time,* agar dapat dibandingkan keakuratan data saat diterima secara *real time* dan tidak *real time* dan nantinya dapat dijadikan acuan saat dibangun sebuah aplikasi pengiriman data auskultasi jantung. Maka dibuatlah sebuah desain dari Visual Basic.

Terdapat 3 desain yang harus dibuat, yaitu desain untuk penyimpanan data pada *end device router* dan desain pada *end device coordinator* (*real time* dan tidak *real time*) untuk menampilkan data secara *real time* dan tidak *real time.*

Karena nantinya akan dibandingkan antara data pada *router* dan *coordinator*, apakah data yang dikirimkan *node router* sesuai dengan data yang diterima *node coordinator* dan baik mana data yang diterima secara *real time* dan tidak *real time.*

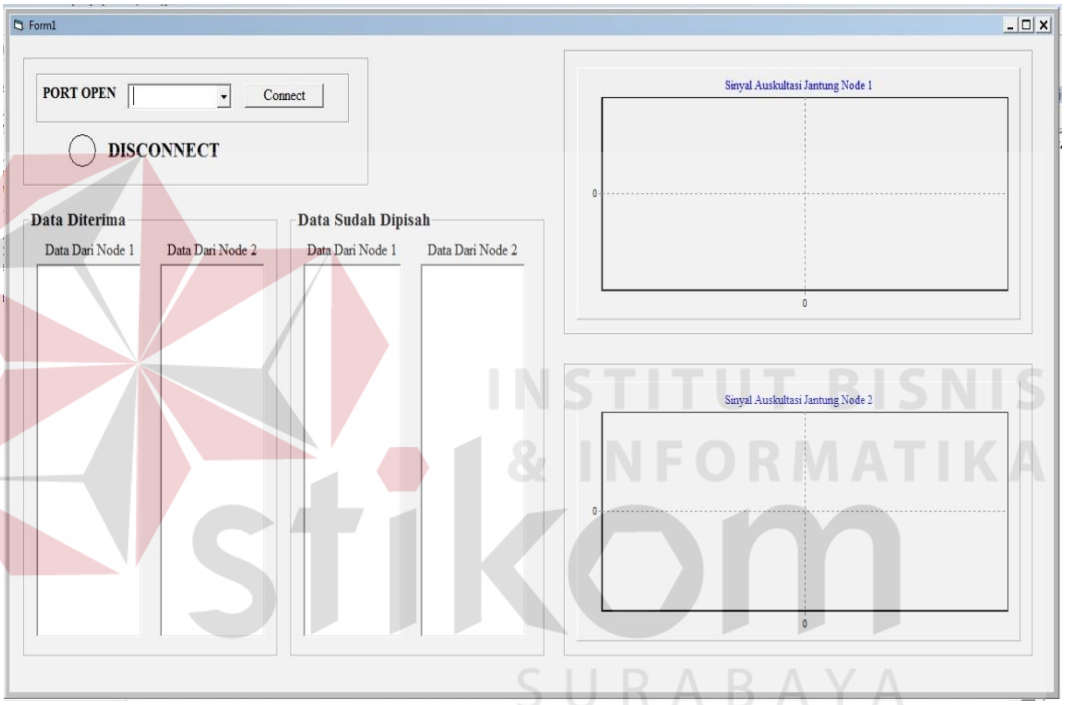


Gambar 3.8 Desain pada *end device router*

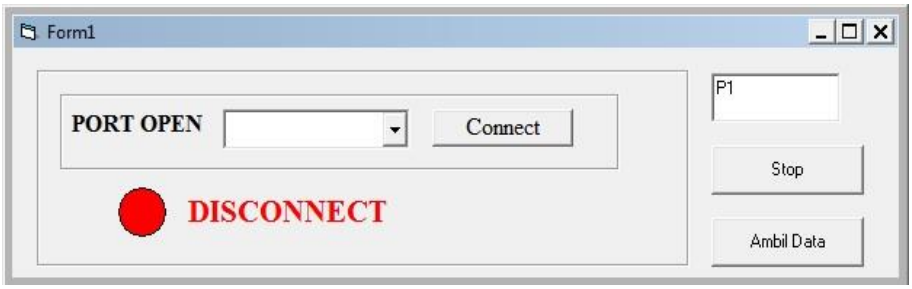
Dari desain diatas *user* dapat melihat secara langsung hasil sinyal jantung, sehingga dapat mengetahui benar tidaknya posisi *heart sound sensor* pada jantung. Hal ini dikarenakan penempatan posisi sensor sangat berpengaruh terhadap hasil yang didapat, dimana jika posisi sensor tidak valid, maka akan menyebabkan hasil pembacaan tidak akurat. Dan dapat dilihat langsung nilai dari sensor.

Dari gambar 3.8 terdapat pemilihan PORT, hal ini digunakan untuk memilih PORT yang telah terhubung dengan mikrokontroler. Selanjutnya ketika

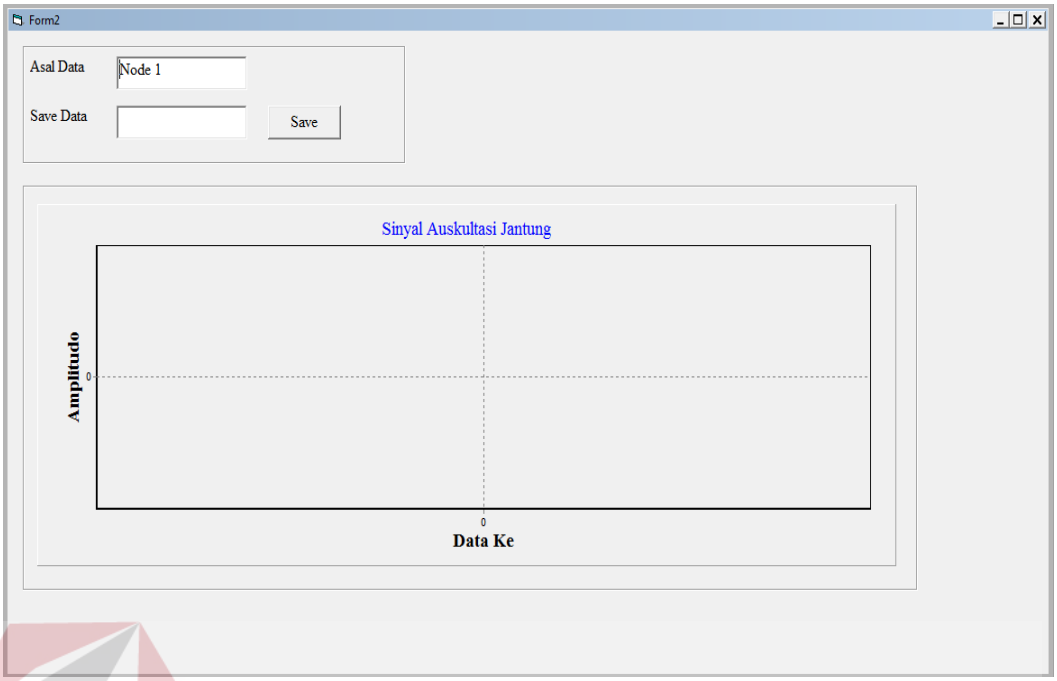
mikrokontroler mengirimkan data pada pemancar, maka secara otomatis data juga akan terkirim secara serial ke komputer. Terdapat kolom data yang diterima dan data yang telah dipisah agar dapat terlihat bahwa data dari masing – masing *router* tidak tertukar pada saat proses pemisahan data. Grafik digunakan untuk dapat melihat apakah data yang diterima adalah data yang bersal dari sinyal jantung, karena sifat dari sensor jantung yang digunakan adalah menangkap suara.



Gambar 3.9 Desain pada *end device coordinator* (*Real Time*)



Gambar 3.10 Desain pada *end device coordinator* (tidak *real time*)



Gambar 3.11 Desain pada *end device coordinator* untuk melihat grafik (tidak *real time*)

Sama halnya dengan desain pada *end device router* yang mengharuskan *user* memilih PORT yang telah tersambung dengan komputer, pada *end device coordinator* juga mengharuskan *user* melakukan hal yang sama. Hanya saja berbeda dengan desain pada *end device router* desain *end device coordinator* terdapat dua grafik karena pada *end device coordinator* digunakan untuk melihat data auskultasi dari dua jantung.

Terdapat perbedaan desain antara *node coordinator real time* dengan *coordinator* tidak *real time*, karena pada *coordinator* tidak *real time* membutuhkan perintah untuk mengambil data dari *router* yang telah disimpan pada drive sedangkan pada *coordinator real time* tidak dibutuhkan perintah tersebut karena data disimpan setelah data ditampilkan pada grafik

Pada gambar 3.9 terdapat dua kolom data yang diterima dan data yang telah dipisah agar dapat terlihat bahwa data dari masing – masing *router* tidak tertukar pada saat proses pemisahan data. Selain itu juga terdapat dua grafik yang menampilkan sinyal jantung dari masing – masing *node.* Dan nantinya juga akan terdapat dua file penyimpanan yang menyimpan data dari masing – masing *node.*

Pada gambar 3.10 hanya terdapat perintah untuk koneksi dengan serial dan perintah mengambil data, karena grafik tidak langsung ditampilkan, melaikna disimpan terlebih dulu kedalam suatu file yang nantinya akan di baca setelah data selesai terkirim. Pemisahan grafik dengan tampilan utama dimaksudkan agar program dapat menampilkan lebih dari 2 data pada grafik. Sehingga analisa dapat lebih fokus pada setiap *node* karena grafik setiap *node* ditampilkan dalan sebuah *form.*

3.5 Perancangan Perangkat Lunak

Dari perancangan sistem diatas, selain perancangan *hardware*, juga dibutuhkan perancangan perangkat lunak untuk menjalankan perancangan *hardware* yang telah dibuat.

Perangkat lunak terdiri dari beberapa algoritma perancangan dari sistem yang ditangani oleh pengontrol.

3.5.1 Algoritma Pembacaan Sinyal Jantung

Gambar 3.12 *Flowchart* pembacaan Heart Sound Sensor

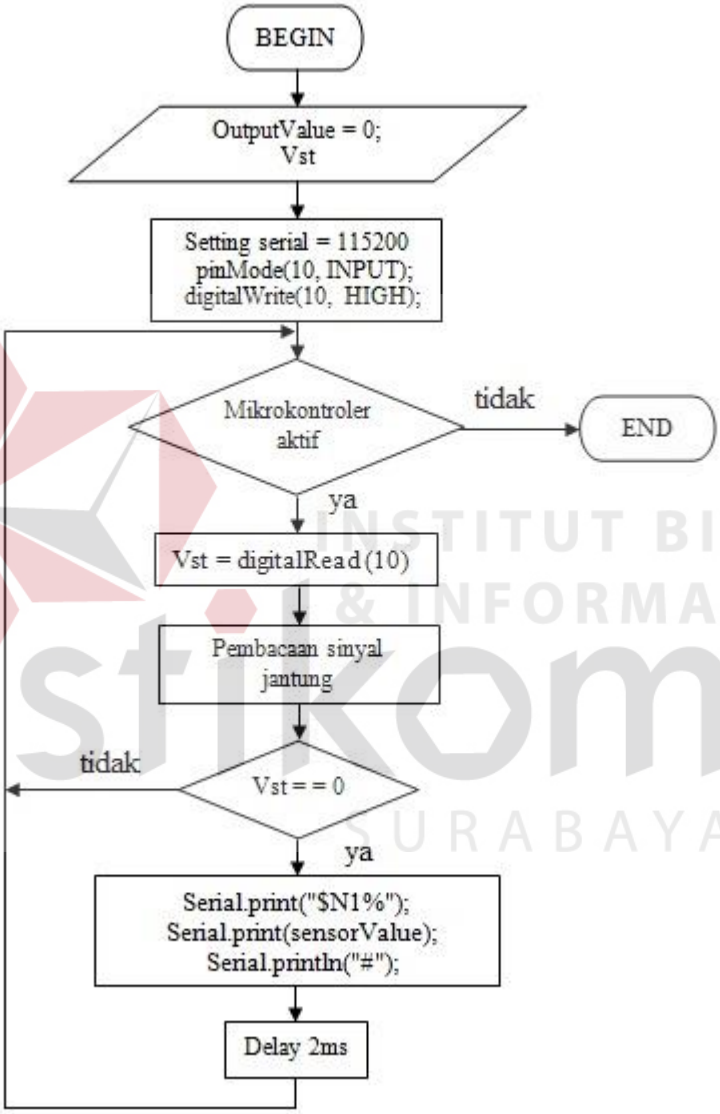
Seperti yang sudah dijelaskan diatas, hasil keluaran dari sensor jantung adalah berupa sinyal analog. Maka pada modul arduino dilakukan pembacaan melalui salah satu fungsi yang dimiliki oleh Arduino Mega2560, fungsi tersebut adalah *readAnalog*. Pada pemrograman modul Arduino Mega2560, *user* dimudahkan dengan beberapa fungsi yang sudah dimilikinya. Sinyal analog pada sensor diubah menjadi data ADC dengan resolusi10 bit. Hal ini ditujukan agar sinyal analog yang dibaca lebih presisi saat dikonversi ADC. Fungsi yang terdapat pada Arduino untuk mengubah data analog menjadi data desimal dengan ukuran 10 bit adalah :

```
sensorValue = analogRead(A0);
outputValue = map(sensorValue, 0, 1023, 0, 255);
analogWrite(9, outputValue);
```

Gambar 3.13 Program pada modul Arduino Mega2560

Selanjutnya data yang diperoleh akan ditransmisikan oleh pemancar sesuai dengan data yang telah diolah. Proses pengiriman data tidak langsung dikirim, karena data yang ada akan diberi ID untuk pengidentifikasian asal data sesuai dengan asal *router*.

3.5.2 Algoritma Pengiriman Sinyal Jantung



Gambar 3.14 *Flowchart* Pengiriman Data Auskultasi Jantung

Pada dasarnya konsep dari sensor jantung adalah menerima setiap suara, maka apabila ketika sensor jantung mendapatkan tegangan, secara langsung data apapun akan ditransmisikan ke *node coordinator*, sehingga mengakibatkan banyaknya noise yang diterima oleh *node coordinator* diawal penerimaan data. Selain itu, karena pada transmisi sinyal jantung ini data yang didapat dari dua *node*, maka untuk mempermudah dalam pengolahan data, maka dibuatlah algoritma seperti gambar 3.14.

Pemrograman Ardunino Mega 2560 merupakan sebuah pemrograman modul, maka pemrograman langsung dilakukan pada setiap pin. Pin 10 digunakan sebagai saklar. PIN 10 digunakan untuk memulai pengiriman data, dimana data dikirim ketika PIN 10 berlogika LOW. Pengiriman dianggap selesai ketika *user* menonaktifkan PIN 10.

Pengiriman data dilakukan sesuai dengan protokol yang sudah ditetapkan pada gambar 3.7, sehingga dapat disimpulkan bahwa pengiriman data berupa *string* atau *caracter*. Pengiriman data akan dipancarkan oleh modul pemancar yang sudah disediakan.

Pengiriman dilakukan dengan *delay* 2ms sesuai dengan ketetapan yang ada, yaitu pengiriman dilakukan minimal dengan 2 kali frekuensi sampling jantung normal (Teori Sampling). (Lynn, 1994)

$$frekuensi\ jantung(fn) = 250Hz - 500Hz$$

$$frekuensi\ sampling = 2x\ fn$$

$$= 2\ x\ 250Hz$$

$$= 500Hz$$

$$Periода\ Sampling = \frac{1}{500} = 2\ ms$$

Pada tugas akhir ini, penulis menggunakan Arduino Mega2560 sebagai mikrokontrolernya. *Software* yang digunakan untuk memprogam arduino tersebut ialah *software* Arduino IDE. Dan dari algoritma yang dibuat diatas maka dibuatlah program seperti gambar 3.15

Gambar 3.15 Tampilan program arduino pada *software* Arduino IDE

Berikut contoh pemrogaman modul arduino Mega 2560 pada *node end device* yang diprogam pada Arduino IDE

a. Pembuatan variabel

Dalam pembuatan variabel, terdapat beberapa variabel yang digunakan oleh penulis seperti pada algoritma diatas. ini penulis menggunakan variabel tipe string yang bernama “sensorValue” untuk menampung data dari sensor, “outputValue” untuk penampung data setelah dijadikan 10 bit. Pembuatan variabel ini diletakkan diluar fungsi void agar variabel ini dapat digunakan secara global. Berikut sebagai contoh :

```
int Vst;
int sensorValue = 0;
int outputValue = 0;
```

b. fungsi void setup

Dalam fungsi void setup perintah akan dibaca 1 kali setelah progam berjalan. Dalam tugas akhir ini penulis mengisikan *baudrate* dan variabel - variabel dalam kondisi kosong, begitu juga pemberian nilai awal pada PIN – PIN yang digunakan. Berikut sebagai contoh :

```
void setup()
{
  Serial.begin(115200);
  pinMode(10, INPUT);
  digitalWrite(10, HIGH);
}
```

c. fungsi void loop

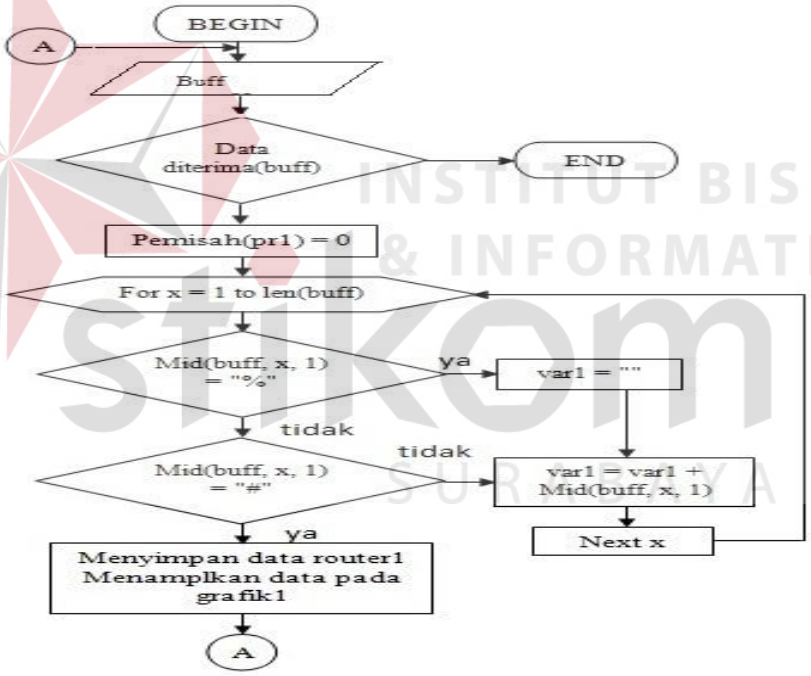
Dalam void loop perintah akan dibaca berulang kali selama mikrokontroler teersambung dengan tegangan. Dalam tugas akhir ini penulis mengisi perintah bagaimana data diolah dan akhirnya dikirimkan,semua perintah ditulis pada void loop ini. Dan progam ditulis sesuai dengan algoritma yang telah dibuat seperti paga gambar 3.14. Berikut contohnya :

```
void loop()
{
  Vst = digitalRead(10);

  sensorValue = analogRead(A0);
  outputValue = map(sensorValue, 0, 1023, 0, 255);
  analogWrite(9, outputValue);

  if (Vst == 0)
  {
    Serial.print("$N1%");
    Serial.print(sensorValue);
    Serial.println("#");
  }
  delay(2);
}
```

3.5.3 Algoritma Penerimaan dan Pemisahan Data pada *Router*



Gambar 3.16 *Flowchart* Pemisahan dan penyimpanan data pada *end device router*

Pada transmisi sinyal jantung auskultasi juga diperlukan pemisahan data pada *router* dikarenakan data yang dikirim berupa data bersama dengan header,

sedangkan yang akan di analisa adalah data sinyal, maka pada *router* juga dilakukan pemisahan data, sehingga mempermudah penulis nantinya dalam menganalisa data.

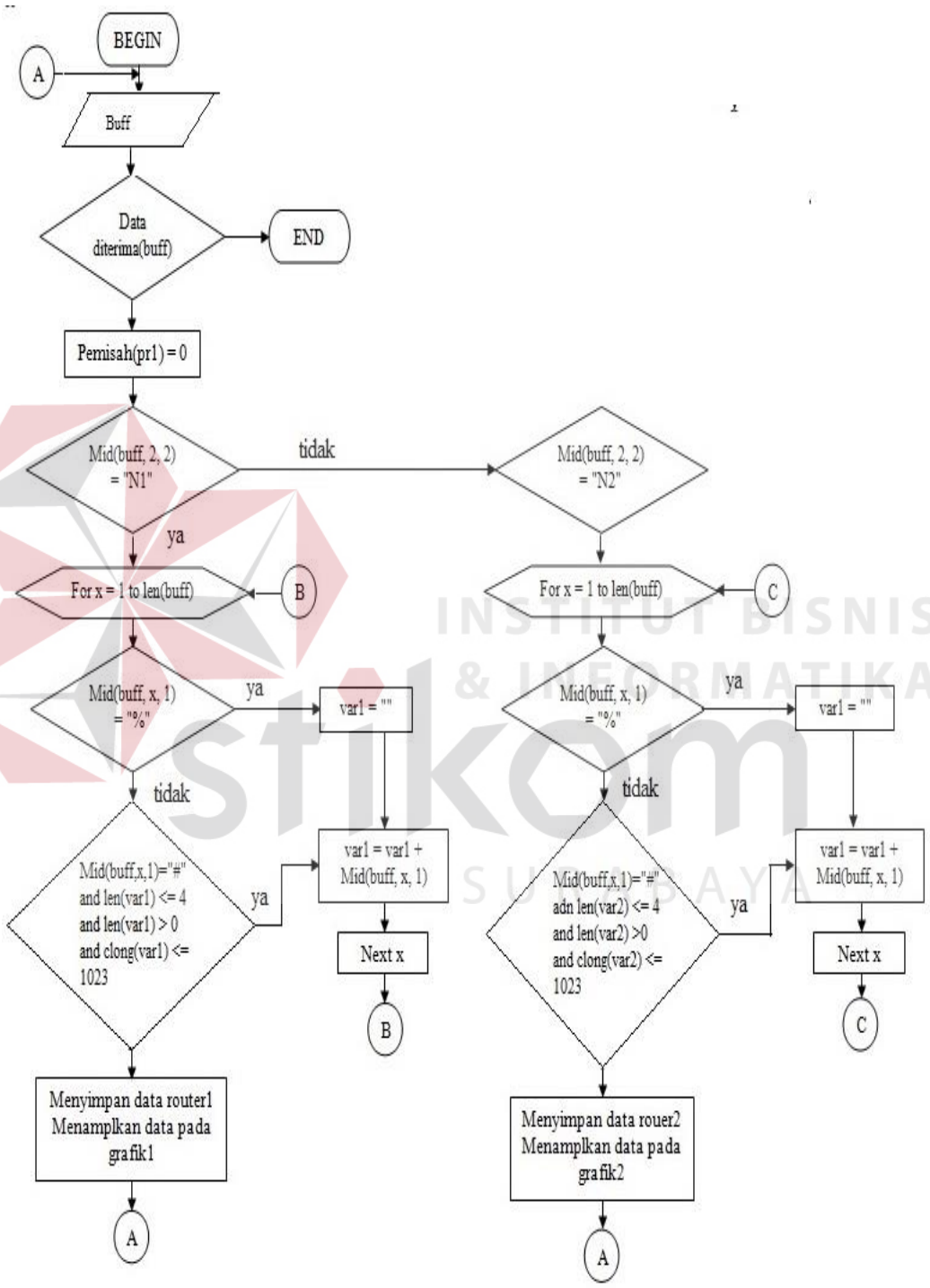
Data yang dikirimkan pada *end device* secara serial, dalam pemisahan dan penyimpanan data dilakukan pada saat data yang terambil sesuai dengan *form*at yang telah ditentukan, maka data akan diambil dengan baik. Apbila data yang diterima tidak lengkap maka pemisahan data tidak dapat dilakukan. Hal ini dikarenakan pemisahan data disesuaikan dengan *form*at yang sudah ditentukan pada saat data dikirim. Data – data yang tidak diperlukan akan diabaikan, dengan mengosongkan variable yang digunakan untuk menyimpan data.

3.5.4 Algoritma Penerimaan Data pada *End Device* (*Real Time*)

Berbeda dengan algoritma penerimaan data pada *end device router*, penerimaan data pada *end device coordinator* terdapat dua data yang diterima yaitu data dari *node* 1 dan *node* 2. Maka selain pemisahan data juga dilakukan pengelompokan data. Proses pemisahan data dilakukan sama dengan pemisahan data yang dilakukan pada *end device router*, hanya saja sebelum dilakukan pemisahan data dilakukan dulu pengelompokan data sesuai asal data.

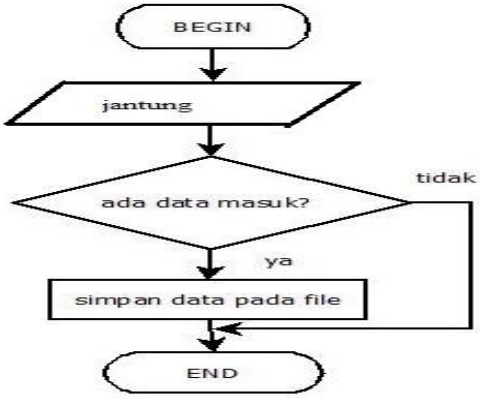
Pengelompokan data dapat dilihat dari ID *node* yang dikirimkan bersamaan dengan data yang dikirim. Ketika data sudah dikelompokkan, selanjutnya data baru dipisah. Hal ini akan memudahkan penulis dalam menganalisa data yang telah terkumpul. Karena data sudah otomatis dalam satu kelompok, dan data tidak tertukar antara satu *node* dengan *node* yang lain. dan

penulis tidak perlu memisah data secara manual. *Flowcart* pemisahan dan penyimpanan data dapat dilihat pada gambar 3.17.



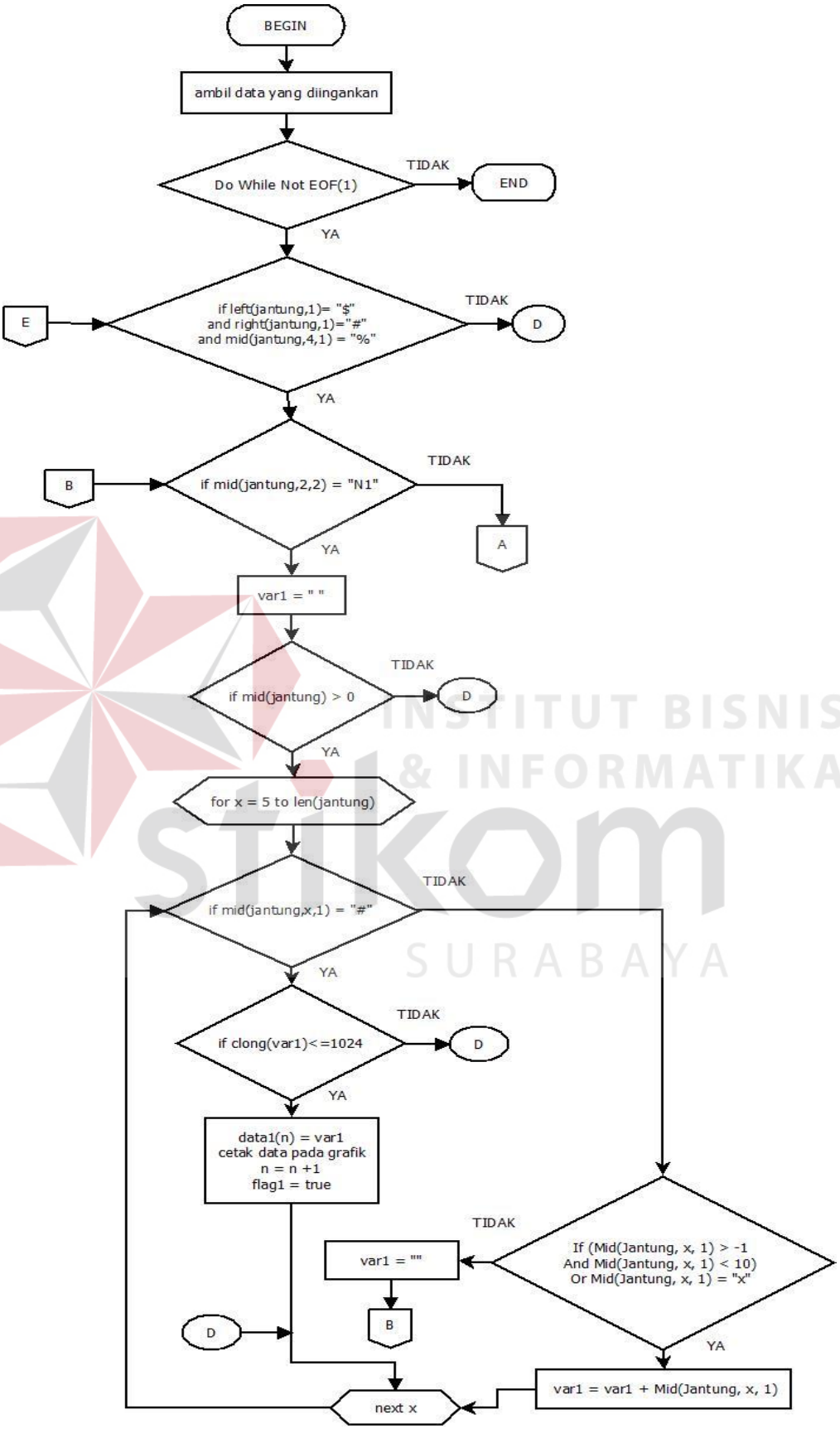
Gambar 3.17 *Flowchart* Pemisahan dan penyimpanan data pada *end device coordinator* (*Real Time*)

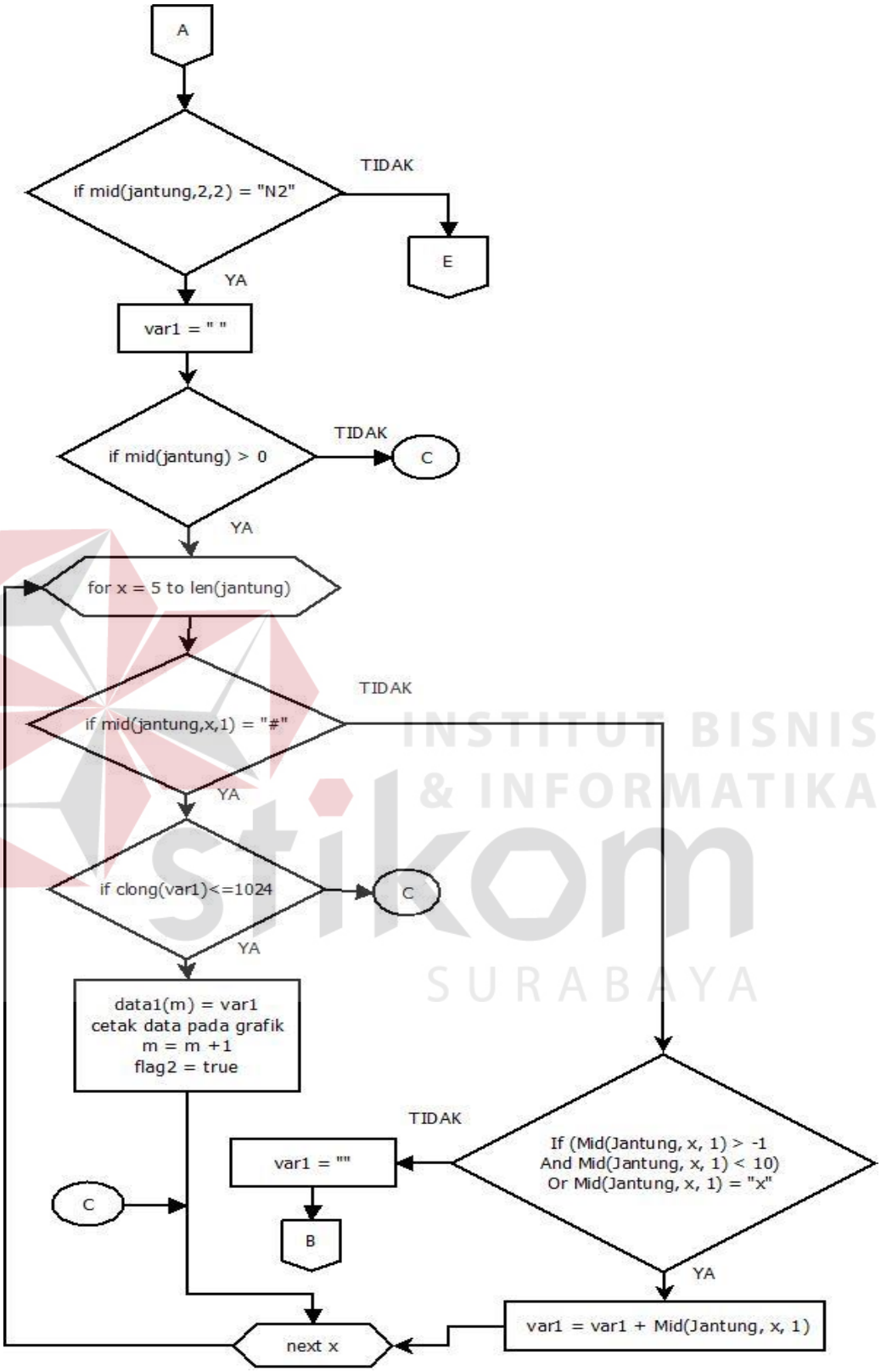
3.5.5 Algoritma Penerimaan Data pada *End Device* (tidak *Real Time*)



Gambar 3.18 *Flowchart* penyimpanan data pada *end device coordinator* (tidak *Real Time*)

Pemisahan data pada *coordinator real time* dengan *coordinator* tidak *real time* pada dasarnya sama, hanya saja berbeda dengan algoritma pada *coordinator real time* karena data terlebih dahulu disimpan kedalam sebuah file, selanjutnya data akan ditampilkan ketika *user* ingin mengambil data dari file. Dan grafik akan muncul sesuai dengan penerimaan data. Misalnya saja pada file tersebut terdapat data yang mengindikasikan berasal dari 2 *node*, tetapi apabila pada file tersebut hanya mengindikasikan bersal dari *node* maka hanya akan ada 1 *form* grafik yang terbuka. *Flowcart* pemisahan dan penyimpanan data pada aplikasi *offline* dapat dilihat pada gambar 3.19.





Gambar 3.19 *Flowchart* Pemisahan dan penyimpanan data pada *end device coordinator* (tidak *Real Time*)

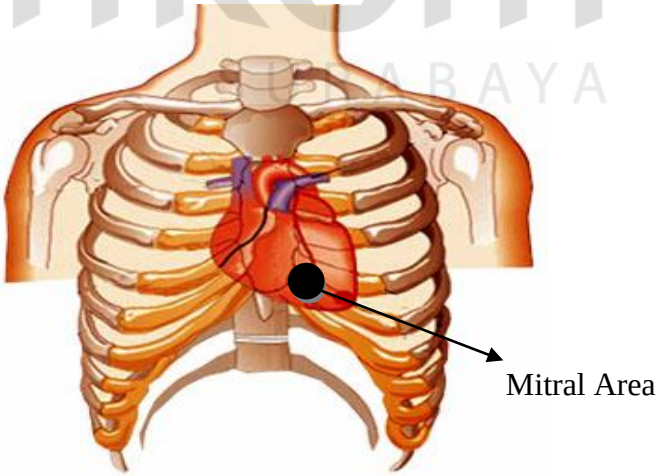
3.6 Metode Analisa

Pada transmisi sinyal auskultasi ini, selain pembuatan algoritma pengiriman data, hal terpenting lainnya adalah analisa dari hasil pengiriman itu sendiri agar dapat diketahui seberapa baik sistem yang telah dibangun.

3.6.1 Peletakan Sensor pada Jantung

Dalam transmisi sinyal auskultasi, komponen terpenting adalah data yang diambil dari auskultasi jantung. Maka tahapan yang pertama dilakukan adalah mengambil data auskultasi dengan meletakkan sensor pada posisi jantung *user* dengan tepat.

Posisi jantung manusia adalah pada tulang iga manusia ke 6 di sebelah kiri dada manusia, atau 5 cm diatas ulu hati di sebelah kiri. Peletakan sensor sangat berpengaruh, karena apabila sensor tidak diletakkan pada bagian jantung yang tepat maka data yang akan diterima berupa data *noise*. Posisi Mitral area dapat dilihat pada gambar 3.20.



Gambar 3.20 Letak posisi penempatan sensor pada jantung (Mitral Area / *Left Verticularr Area)*

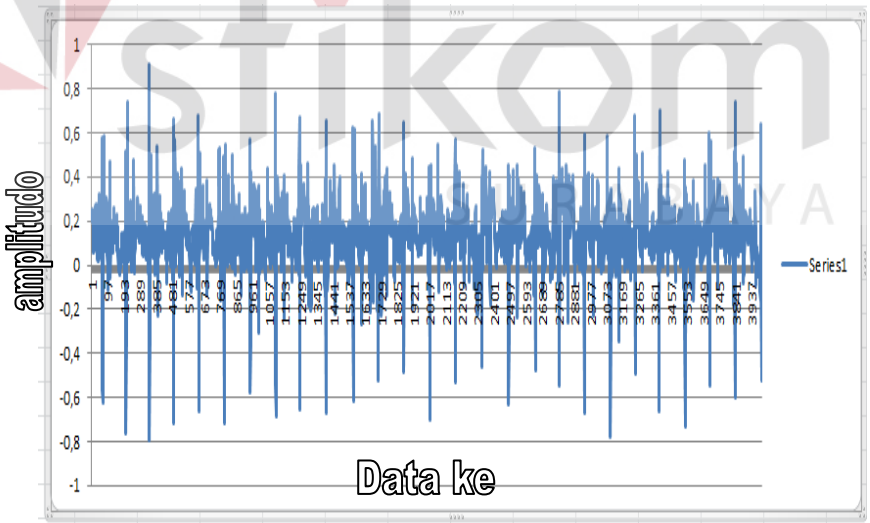
(sumber : Shank, 2014)

3.6.2 Pengambilan Sinyal Auskultasi Jantung

Proses pengambilan data dilakukan saat semua alat terpasang, dan proses pengiriman data berlangsung. Seperti yang dijelaskan di atas, melalui grafik kita dapat melihat apakah sensor sudah berada pada posisi yang tepat.

Data pada masing – masing *router* dan *coordinator* akan tersimpan pada sebuah file. File inilah yang nantinya digunakan untuk menganalisa seberapa baik sistem dapat mentransmisikan sinyal auskultasi jantung dari 2 *node* ke 1 titik *coordinator* secara bersamaan dan *streaming.*

Pengambilan sinyal jantung dilakukan selama 30 detik untuk mendapatkan hasil transmisi sinyal jantung. Hal ini dikarenakan penerimaan data pada titik *coordinator* lebih lama karena adanya proses pemisahan data yang terjadi pada titik *coordinator*. Contoh sinyal jantung hasil auskultasi terlihat dalam gambar 3.21.



Gambar 3.21 Hasil Sinyal Askultasi Jantung

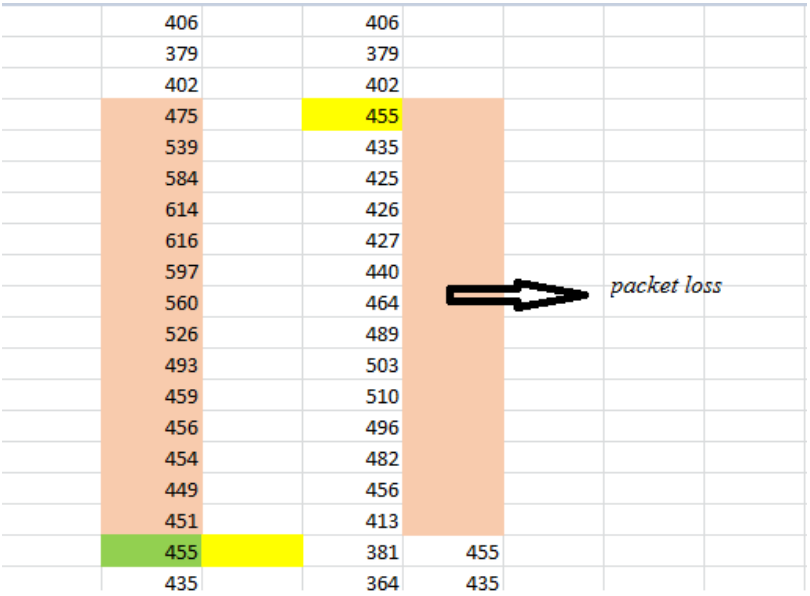
3.6.3 Analisa Transmisi Sinyal Auskultasi Jantung

Cara menganalisa hasil transmisi jantung adalah dengan memindah data dari file penyimpanan ke file excel. Lalu data dibandingkan antara data inputan dengan data yang berasal dari *receiver*. Posisi data pertama yang sesuai dengan data yang ada pada *transmiter* adalah data yang berhasil dikirim dan diterima. Sehingga dapat diketahui *delay* penerimaan data dari data yang telah dikirim.

Dari semua data yang dikirim terdapat *loss* data, *loss* data dapat diketahui dari data yang tidak sesuai dengan data yang terdapat pada pengiriman, dan dengan mengurangkan jumlah data yang terkirim dengan jumlah data yang telah diterima. Maka data yang *loss* akan dapat diketahui.

	T (s)	T Signal	T(s)	R Signal		Legend	
3	0	602	0,04	602			Start awal data T dan R sama
4		650		650			Titik awal data T dan R tidak sama atau data T hilang
5		520		520			Jumlah data hilang
6		537		537			
7		519		519		Jumlah data hilang=	308
8		539		539		Jumah data =	11971
9		553		553			
10		564		564		% Data lost =	2,57
11		606		606			
12		631		631			
13		624		624			
14		620		620			

Gambar 3.22 Data yang diterima pada waktu 0,04 dari pengiriman



Gambar 3.23 Data yang *loss*

Dari pencarian *delai* dan data *loss* seperti pada gambar 3.21 dan gambar 3.22 maka dapa dilakukan perhitungan untuk mengetahui berapa *packet loss* yang diterima, berapa lama pengiriman data serta berapa besar *througput* rata – rata setiap pengiriman data.

a. *Delay*

Setelah menyamakan data antara *transmitter* dan *receiver* maka terdapat selisih urutan antara kedua data tersebut, selisih urutan tersebutlah yang disebut *delay* dalam pentransmisian sinyal auskultasi jantung.

Karena data dikirim setiap 2ms maka jarak antara data tersebut dikalikan dengan waktu pengiriman data, dan akan ditemukan berapa lama data yang dikirimkan oleh *transmitter* diterima oleh *receiver.*

$$Delay = selisih\ urutan\ data\ x\ 0{,}002$$

b. *Packet Loss*

Pada pencarian *packet loss* seperti pada gambar 3.21 maka akan ditemukan banyak data yang tidak dapat diterima dengan baik oleh *receiver,* jumlah paket yang tidak diterima dengan sempurna tersebut adalah *packet loss* yang digunakan untun mecari berapa besar persentase data yang hilang.

$$packet\ loss = \frac{jumlah\ paket\ hilang}{jumlah\ data\ masuk}\ x\ 100\ \%$$

c. *Througput*

Seperti yang telah dijelaskan pada bab 3, *througput* adalah besar kecepatan data terkirim secara *real*. Maka untuk menemukan *througput* dilakukan dengan cara memsukkan jumlah data diterima selanjutnya dibagi dengan lama waktu pengamatan.

$$Throughput = \frac{jumlah\ data\ masuk\ x\ jumlah\ tiap\ paket\ data\ x\ \ besar\ pengiriman\ data}{lama\ pengamatan}$$

Maksud dari rumus diatas adalah :

- Jumlah data masuk : keseluruhan data yang masuk dari ke 2 *node* sebelum melalui proses pengelompokan data

- Jumlah tiap *packet data* : dalam ngeritimkan 1 buah paket data terdapat ± 8 *charakter*
- Besar pengiriman data : sebuah karakter terbentuk dari 10 bit data, yaitu 8 bit untuk setiap *charatcter,* 1 bit prmnbuka data dan juga 1 bit penutup.