BAB III

METODE PENELITIAN

3.1 Metode Penelitian

Metode penelitian yang digunakan dalam penelitian ini adalah transmisi data *streaming* menggunakan Zigbee *wireless network* dengan teknik *scheduling* metode *circular* FIFO. Pada penelitian ini menggunakan data real yang didapat menggunakan *heart sound sensor*. Data ini berupa sinyal suara jantung. Blok diagram penelitian ditunjukkan pada Gambar 3.1.

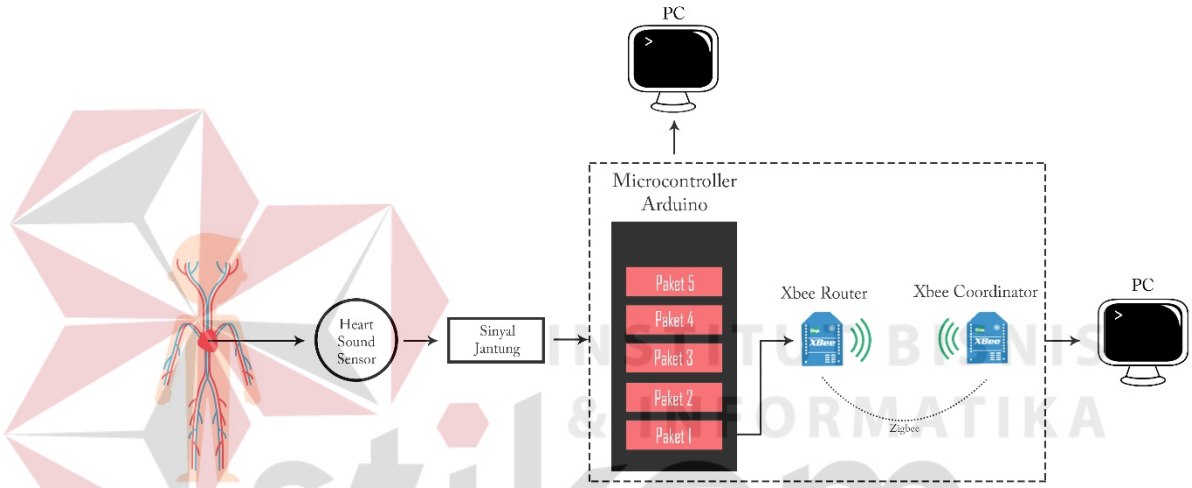
Gambar 3.1 Blok Diagram Penelitian

Penelitian diawali dengan melakukan pengambilan data secara langsung dari pasien dengan menggunakan *heart sound sensor*. Setelah pengambilan data sinyal suara jantung, langkah selanjutnya adalah melakukan data *processing* terhadap data yang diterima pada *microcontroller* Arduino. Data *processing* dilakukan untuk mengendalikan aliran data dengan membagi data kedalam elemen antrian. Proses pembagian data kedalam elemen antrian pada *microcontroller* Arduino dilakukan untuk menghindari *overflow* pada *buffer* Zigbee. Proses ini dilakukan dengan memperhitungkan

data sinyal suara jantung yang masuk. Sehingga proses *transmission* data dilakukan dengan *output* berupa paket per-paket.

3.2 Model Perancangan

Pada perancangan ini penulis menggambarkan perancangan sistem seperti ditunjukkan oleh Gambar 3.2.



Gambar 3.2 Skema Perancangan

Dari Gambar 3.2 didapatkan bahwa setiap *node* WSN memiliki tugas berbeda-beda seperti berikut:

a) Sensor

Bertanggung jawab sebagai pencatat hasil auskultasi sinyal suara jantung pada pasien dan kemudian menuju *microcontroller* Arduino seperti perancangan pada Gambar 3.2.

b) Microcontroller Arduino

Bertanggung jawab atas pembagian data sinyal suara jantung kedalam sebuah antrian sehingga dapat membentuk paket data dengan teknik *scheduling* menggunakan metode *circular* FIFO sehingga sinyal suara jantung dapat dikendalikan.

c) Zigbee

Bertanggung jawab atas proses transmisi paket data sinyal suara jantung mulai proses pengiriman dari *microcontroller* Arduino menuju penerima *end device* / PC.

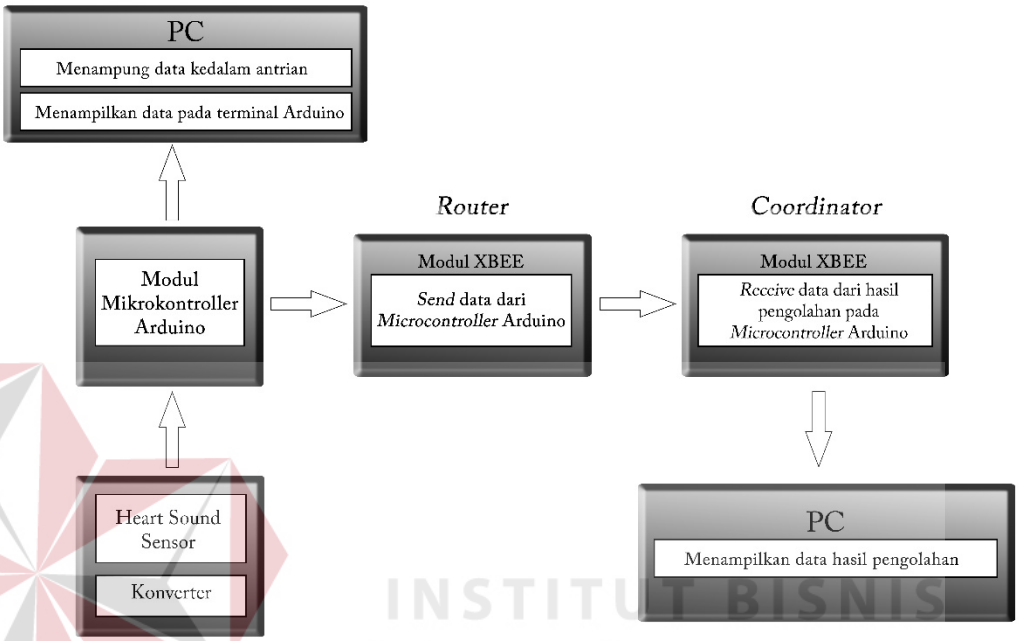
d) *End device* / PC (*Personal Computer*)

Terdapat 2 *end device*, pada *end device* yang tersambung dengan *microcontroller* arduino digunakan oleh *user* untuk memprogram sehingga data dari sensor bisa diterima dan kemudian diolah. Digunakan juga untuk data pembanding antara data yang dikirim dengan data yang diterima oleh *xbee coordinator* agar dapat diketahui berapa banyak data yang loss dan delay saat proses transmisi dilakukan. Sedangkan *end device* yang tersambung dengan Xbee *coordinator* digunakan oleh *user* untuk melihat data hasil pengolahan pada *microcontroller* Arduino sesuai dengan program yang dibuat.

INSTITUT BISNIS & INFORMATIKA stikom SURABAYA

3.3 Perancangan Sistem

Berikut perancangan blok diagram seperti pada Gambar 3.3.



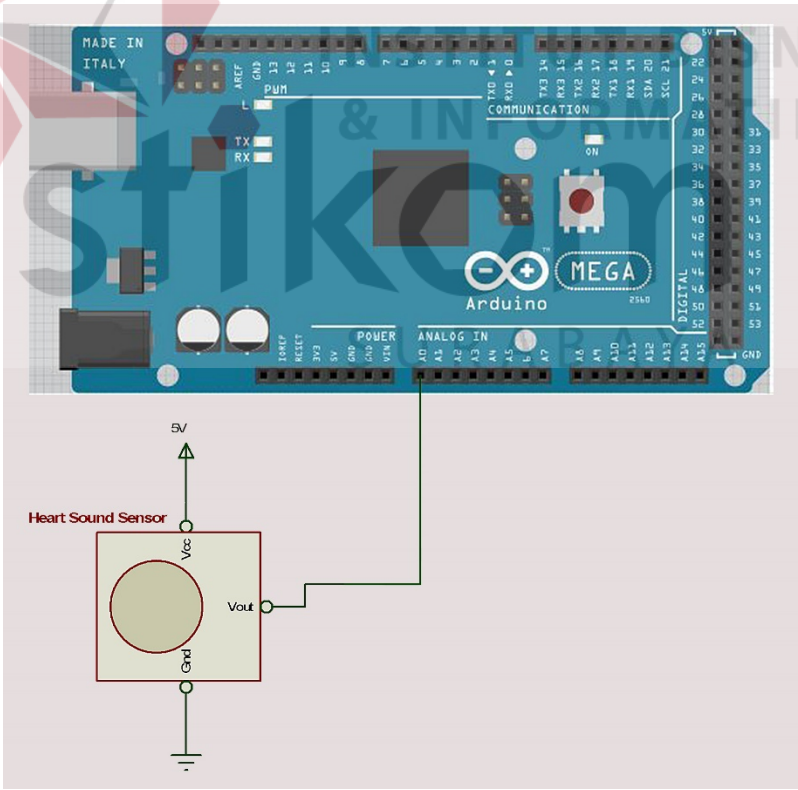
Gambar 3.3 Block Diagram Sistem

Dalam tugas akhir ini, penulis hanya akan memfokuskan pada proses pengendalian aliran data yaitu dengan membagi data kedalam elemen antrian untuk membentuk paket-paket data sehingga proses pengiriman melalui Zigbee *Wireless Network* dapat terkendali dan mencegah terjadinya *overflow* pada *buffer* Zigbee.

3.4 Perancangan Perangkat Keras

3.4.1 Perancangan Sensor Jantung

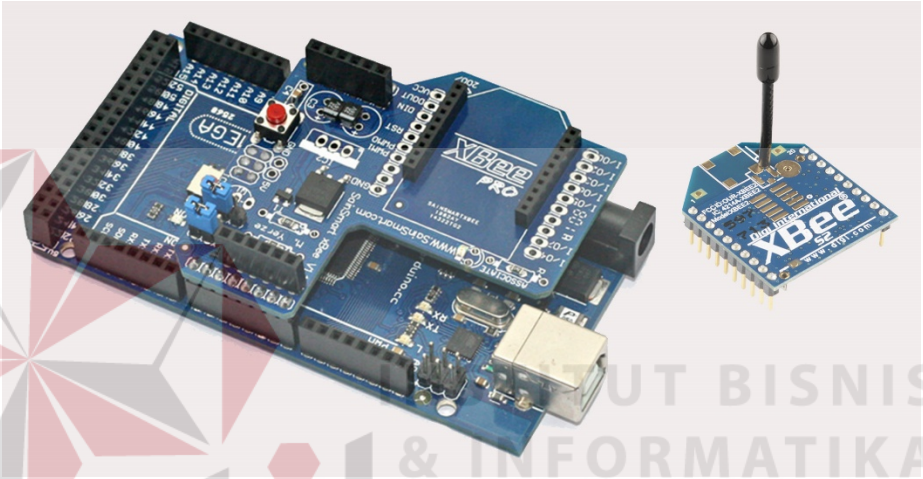
Pemeriksaan detak jantung secara elektronik harus membutuhkan alat yang dinamakan sensor. Sensor yang digunakan pada penelitian transmisi sinyal auskultasi jantung ini adalah *Heart Sound Sensor*. Sensor ini telah dilengkapi dengan pengkondisi sinyal dan filter yang bertugas meredam dan mengolah sinyal jantung dan mengkonversinya dalam bentuk tegangan. Dengan demikian keluaran dari *Heart Sound Sensor* dapat langsung dibaca melalui ADC internal pada modul Arduino Mega2560. Adapun perancangan rangkaian *Heart Sound Sensor* ditunjukkan pada Gambar 3.4.



Gambar 3.4 Hubungan Rangkaian *Heart Sound Sensor* dan Arduino

3.4.2 Perancangan Xbee Zigbee S2

Agar modul Arduino dapat berkomunikasi secara serial *wireless* dengan perangkat lain, maka dibutuhkan perangkat *wireless* yang dalam perancangan ini menggunakan modul Zigbee S2. Seperti yang ditunjukkan pada Gambar 3.5.



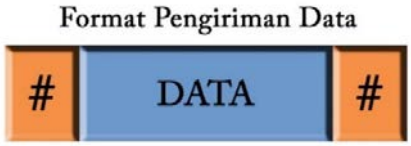
Gambar 3.5 Modul Arduino Mega2560, Xbee Shield dan Xbee S2

3.4.3 Arduino Mega2560

Pada Arduino Mega2560 memiliki fungsi yaitu membaca nilai analog berasal dari PORT A0 dengan menggunakan fungsi *readAnalog()* didalam modul Arduino.

Pada modul Arduino juga dilakukan pengolahan data hasil pembacaan sensor, yang berfungsi untuk mengendalikan aliran data. Pengolahan pada penilitian ini adalah membagi data hasil auskultasi sinyal jantung ke dalam elemen antrian. Pembagian data kedalam antrian dilakukan agar proses

transmisi melalui Zigbee dapat berjalan dengan baik, artinya tidak terjadi *overflow* pada *buffer* Zigbee. Format pembagian data dapat terlihat pada Gambar 3.6.



Gambar 3.6 Format Pengiriman Data

Berikut penjelasan dari Gambar 3.6 :

1. # : Penanda awal dan akhir dari data
2. Data : Data sinyal auskultasi jantung yang dikirimkan

Format diatas dibuat agar memudahkan dalam pemisahan data pada sisi penerima.

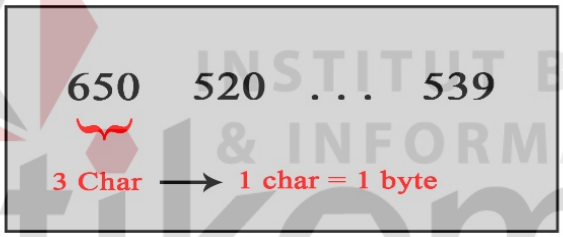
Data hasil pembacaan sensor yang diterima pada mikrokontroller arduino akan ditampung kedalam antrian untuk membentuk paket data, sehingga dalam setiap antrian mempunyai elemen sebanyak 30 data jantung. Terbatasnya kapasitas *buffer* pada Zigbee yang hanya 127 bytes membuat pembagian data kedalam antrian ini begitu penting. Sehingga pada saat proses pemeriksaan pasien, data dapat diolah dengan baik oleh dokter.

INSTITUT BISNIS & INFORMATIKA stikom SURABAYA

Data Jantung
650
520
537
519
539

Gambar 3.7 Contoh Data Sinyal Jantung yang diterima (Oktarina, Sari. 2015)

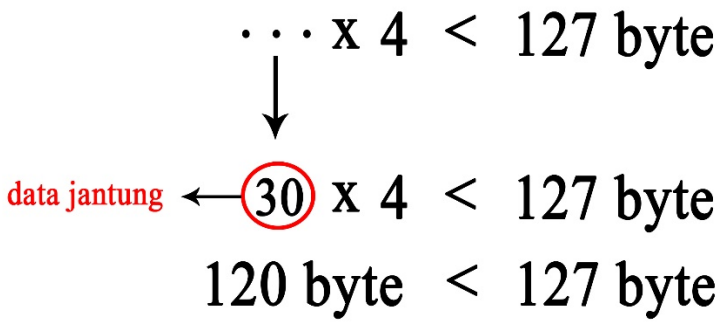
Setiap data hasil pembacaan sensor yang diterima memiliki 3 karakter, seperti yang ditunjukkan pada Gambar 3.7.



Gambar 3.8 Ukuran Tiap Data Jantung

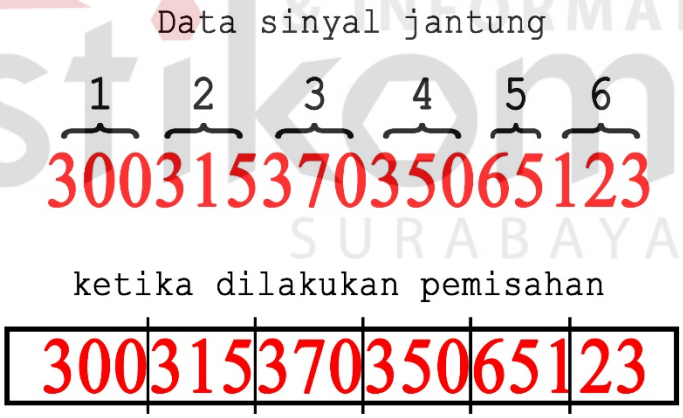
Seperti yang terlihat pada Gambar 3.8 1 data terdapat 3 karakter, artinya dalam 1 data jantung mempunyai ukuran 3 byte. Format pengiriman data seperti pada Gambar 3.6 menunjukkan data ditambah dengan (#) atau *header* sebagai pemisah serta penanda awal dan akhir data. Kapasitas *buffer* Zigbee adalah 127 byte, untuk mencukupi kemampuan Zigbee tersebut terdapat perhitungan seperti pada Gambar 3.9.

INSTITUT BISNIS & INFORMATIKA stikom SURABAYA



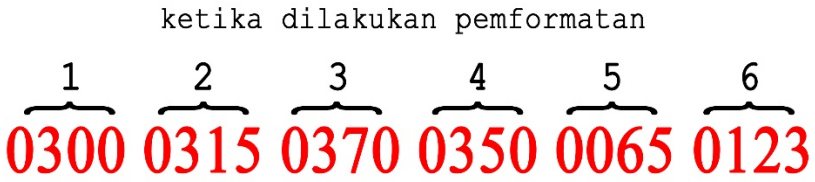
Gambar 3.9 Ukuran untuk Setiap Paket Pengiriman

Nilai 4 pada gambar diatas adalah format pengiriman data yang ditetapkan agar saat proses analisa data dapat dikelola dengan baik. Ketika data yang dikirim tanpa dilakukan pemformatan terlihat seperti pada Gambar 3.10.



Gambar 3.10 Tanpa Dilakukan Pemformatan Data

Pada Gambar 3.10 menjelaskan bahwa ketika data yang diterima dianalisa akan terjadi kekacauan saat proses pemisahan data satu per satu. Maka pemformatan pada proses transmisi ini perlu dilakukan sehingga data akan aman akan ketika dianalisa. Dapat terlihat seperti Gambar 3.11.



Gambar 3.11 Pemformatan Data

Karena resolusi ADC pada penelitian ini adalah 10 bit maka dalam 1 data jantung di format menjadi 4 karakter. Didapat bahwa setiap kali pengiriman terdapat 30 data jantung dengan ukuran 120 byte + *header*. Maka dengan perhitungan yang telah dibuat proses transmisi diharapkan dapat berjalan tanpa terjadi kehilangan paket.

3.4.4 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.

INSTITUT BISNIS & INFORMATIKA stikom SURABAYA

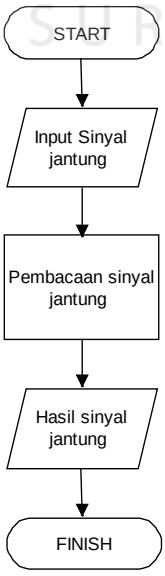
Langkah pertama untuk dapat berkomunikasi dalam satu jaringan, maka PAN ID antar Xbee harus diisi dengan nama atau nilai yang sama. Langkah kedua yaitu mengisi DH dengan ID yang terdapat pada Xbee dan DL dengan nilai yang sesuai dengan nilai DL pada Xbee yang digunakan sebagai *node coordinator*. Hal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

3.5 Perancangan Perangkat Lunak

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

3.5.1 Pembacaan Sinyal Jantung

Berikut adalah *flowchart* pembacaan sinyal jantung dapat terlihat seperti pada Gambar 3.12.



Gambar 3.12 *Flowchart* Pembacaan Sinyal Jantung

INSTITUT BISNIS & INFORMATIKA stikom SURABAYA

Hasil keluaran dari sensor jantung adalah berupa sinyal analog. Pada modul Arduino dilakukan pembacaan melalui fungsi yang dimiliki oleh Arduino Mega2560 yaitu *readAnalog*. Resolusi pembacaan tegangan analog di pin analog (A0 s/d A5) adalah sebesar 10 bit. Fungsi tersebut terlihat pada Gambar 3.13.

```
sensorValue     = analogRead(A0);
```

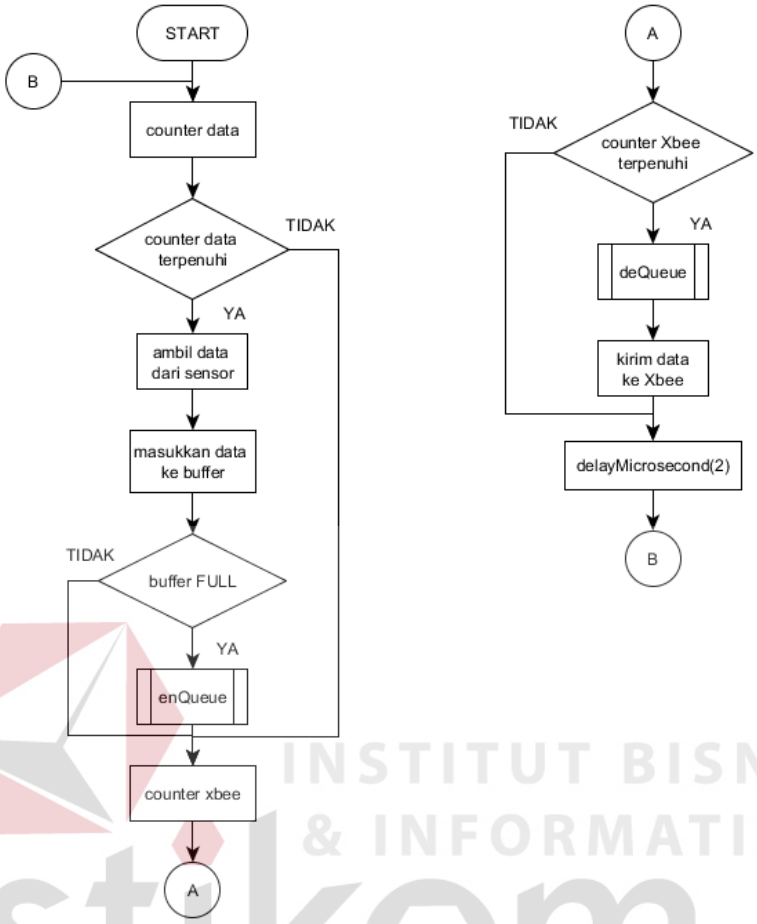
Gambar 3.13 Pembacaan Sinyal Jantung pada Modul Arduino Mega2560

Selanjutnya data sinyal jantung yang diterima oleh mikrokontroller Arduino akan dilakukan pengolahan dengan membagi data kedalam elemen antrian.

3.5.2 Pengolahan Sinyal Jantung

Berikut adalah *flowchart* pengolahan sinyal jantung dapat terlihat seperti pada Gambar 3.14.

INSTITUT BISNIS & INFORMATIKA stikom SURABAYA



Gambar 3.14 *Flowchart* Pengolahan Data Sinyal Jantung

INSTITUT BISNIS & INFORMATIKA stikom SURABAYA

Penjelasan dari *flowchart* pada Gambar 3.14 adalah sebagai berikut :

Bagian pertama :

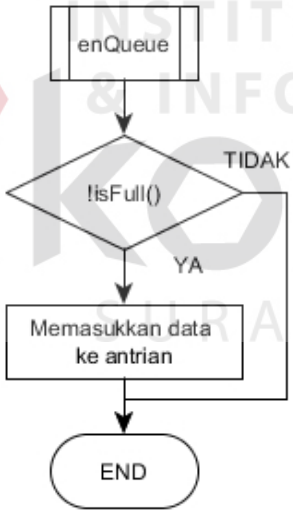
1. Mula-mula *counter* sensor di *update*
2. Mengecek apakah *counter* sensor sudah *full*
3. Ketika counter sensor *full* maka melakukan proses pengambilan data dari sensor
4. Selanjutnya data akan dimasukkan kedalam *buffer*
5. Mengecek apakah *buffer* sudah *full*

6. Ketika *buffer full* maka data dalam *buffer* dimasukkan kedalam *queue* atau disebut prose *enqueue*

Bagian kedua :

1. Mula-mula counter Xbee di *update*
2. Mengecek apakah counter Xbee *full*
3. Ketika *counter* Xbee *full* maka melakukan proses *dequeue* atau mengambil data yang sudah masuk kedalam antrian.
4. Selanjutnya melakukan proses pengiriman data ke Xbee

Berikut adalah *flowchart* pemasukan data kedalam *queue* dapat terlihat seperti pada Gambar 3.15.



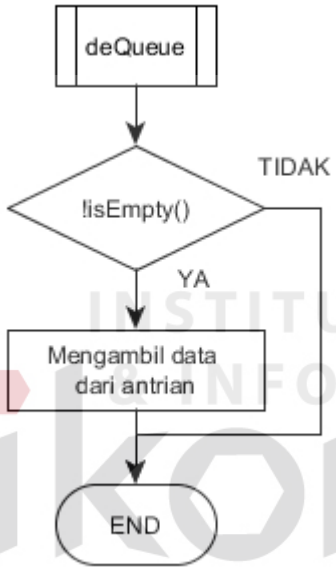
Gambar 3.15 *Flowchart* Proses *Enqueue*

Penjelasan dari *flowchart* pada Gambar 3.15 adalah sebagai berikut :

1. Masuk ke dalam fungsi enqueue
2. Mengecek apakah antrian yang disediakan penuh atau tidak penuh

3. Ketika kondisi antrian tidak penuh maka dilakukan proses memasukkan data ke dalam *queue*
4. Proses enqueue dilakukan selama kondisi *queue* tidak dalam keadaan penuh

Berikut adalah *flowchart* pengambilan data yang sudah masuk ke dalam *queue* dapat terlihat seperti pada Gambar 3.16.



Gambar 3.16 *Flowchart* Proses *Dequeue*

Penjelasan dari *flowchart* pada Gambar 3.16 adalah sebagai berikut :

1. Masuk ke dalam fungsi dequeue
2. Mengecek apakah antrian yang disediakan dalam keadaan kosong atau tidak kosong
3. Ketika kondisi antrian tidak kosong maka dilakukan proses pengambilan data yang berada didalam *queue*

4. Proses dequeue dilakukan selama kondisi *queue* tidak dalam keadaan kosong

Berikut adalah *flowchart* pemformatan data jantung sesuai dengan ketetapan yang dibuat dapat terlihat seperti pada Gambar 3.17.

Gambar 3.17 *Flowchart* Pemformatan Data

INSTITUT BISNIS & INFORMATIKA stikom SURABAYA

Penjelasan dari *flowchart* pada Gambar 3.17 adalah sebagai berikut :

1. Masuk ke dalam fungsi format
2. Mengecek apakah panjang data kurang dari 4
3. Ketika panjang data kurang dari 4 maka dilakukan penambahan jumlah digit pada data

Data jantung akan ditransmisikan melalui Zigbee *wireless network* setelah melewati pengolahan pada di Arduino. Pada dasarnya konsep dari sensor jantung adalah menerima setiap suara, maka apabila sensor jantung

mendapatkan tegangan, secara langsung data apapun itu akan ditransmisikan, sehingga mengakibatkan banyaknya noise yang diterima oleh sisi penerima penerimaan data. Maka diusahakan pada saat proses pengambilan data jantung sensor ditempatkan pada posisi yang benar, sehingga data yang ditransmisikan benar berupa data jantung walaupun terdapat noise dari sensor jantung itu sendiri.

Pengiriman data dilakukan sesuai dengan protokol yang sudah ditetapkan pada Gambar 3.6, bahwa pengiriman data berupa *string*. Data akan ditransmisikan melalui modul pemancar yang sudah disediakan.

Proses pengiriman dilakukan dengan *interval sampling* 1ms dan 2ms dengan tujuan agar sistem dapat mewakili rentang frekuensi sinyal jantung yang terletak diantara 250-500 Hz. Sesuai dengan teori sampling (lynn, 1994) maka hubungan antara frekuensi sinyal jantung dengan *interval sampling* ditunjukkan dalam perhitungan sebagai berikut.

Frekuensi jantung (*fn*) = 250 Hz – 500 Hz

Frekuensi sampling (*fn*) = 2 x *fn*

= 2 x 250 Hz

= 500 Hz

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

Frekuensi jantung (*fn*) = 250 Hz – 500 Hz

Frekuensi sampling (*fn*) = 2 x *fn*

= 2 x 500 Hz

= 1000 Hz

Periode Sampling = $\frac{1}{1000} = 1\ ms$

Pada Tugas Akhir ini, mikrokontroller yang digunakan adalah Arduino Mega2560. *Software* yang digunakan untuk memprogram adalah *software* Arduino IDE. Dari *flowchart* yang dibuat diatas maka dibuatlah program seperti pada Gambar 3.18.

Gambar 3.18 Tampilan Program pada *Software* Arduino IDE

Berikut contoh *script* yang dibuat pada penelitian ini pada modul Arduino Mega2560.

a. Inisialisasi variable

Inisialisasi variabel digunakan untuk proses pengisian nilai awal ke dalam sebuah variabel. Dalam program yang dibuat inisialisasi variabel dapat digunakan secara global oleh program.

```
int front = 0;
int rear  = 0;
int count = 0;
int in = 0;
int out = 0;
```

```
int counter_sensor = 0;
int counter_xbee = 0;
```

b. #define

Digunakan untuk mendefinisikan suatu nilai tertentu kepada suatu nama konstanta.

```
#define BARIS 97
#define KOLOM 30
```

c. Pembuatan array dua dimensi dan satu dimensi

Digunakan untuk menyimpan sekumpulan data yang memiliki tipe data yang sama dan elemen yang akan diakses melalui dua indeks yaitu indeks baris dan indeks kolom.

```
short Q[BARIS][KOLOM];
short k[KOLOM];
```

d. Fungsi isFull()

Fungsi isFull() ini digunakan untuk mengecek apakah antrian yang disediakan penuh atau tidak penuh.

```
int isFull()
{
  if(count == BARIS) return 1;
  else return 0;
}
```

e. Fungsi isEmpty()

Fungsi isEmpty() ini digunakan untuk mengecek apakah antrian dalam keadaan kosong atau tidak.

```
int isEmpty()
{
  if(count == 0)return 1;
  else return 0;
}
```

INSTITUT BISNIS & INFORMATIKA stikom SURABAYA

f. Fungsi enQueue()

Fungsi ini digunakan untuk memasukkan data kedalam antrian. Sebelum memasukkan data kedalam antrian, harus mengecek terlebih dahulu apakah antrian sudah penuh atau belum.

```
void enQueue(int data_masuk[KOLOM])
{
  if(!isFull())
  {
    for(int i=0; i<KOLOM; i++)
    {
      Q[rear][i] = data_masuk[i];
    }
    rear = rear + 1;
    if(rear == BARIS)rear = 0;
    count = count + 1;
   }
}
```

g. Fungsi deQueue()

Fungsi ini digunakan untuk mengambil data yang sudah masuk kedalam antrian. Sebelum mengambil data dalam antrian, harus mengecek terlebih dahulu apakah antrian kosong atau tidak.

```
void deQueue(int data_keluar[KOLOM])
{
  if(!isEmpty())
  {
    for(int i=0; i<KOLOM; i++)
    {
      data_keluar[i] = Q[front][i];
    }
    front = front + 1;

    if(front==BARIS)front = 0;
    count = count - 1;
  }
}
```

INSTITUT BISNIS & INFORMATIKA stikom SURABAYA

h. Fungsi format()

Fungsi ini digunakan untuk menggabungkan dua buah string menjadi satu string baru. Dalam penelitian ini untuk memformat data yang masuk menjadi 4 karakter, ketika kurang dari ketetapan itu maka data yang masuk akan di tambah dengan “0”.

```
String format(int number, int digit)
{
  String s = String(number);
  String s1 = "0";
  while(s.length()<digit)
  {
    s1.concat(s);
    s = s1;
    s1 = "0";
  }
  return(s);
}
```

i. Fungsi setup

Dalam fungsi setup perintah akan dibaca satu kali setelah program berjalan. Penulis mengisikan *baudrate* sehingga proses komunikasi serial dapat terjadi.

```
void setup()
{
  Serial.begin(115200);
}
```

j. Fungsi void loop

Dalam fungsi loop perintah akan dibaca berulang kali selama mikrokontroller tersambung dengan tegangan. Dalam tugas akhir ini penulis mengisi perintah mulai dari proses pembacaan data, data diolah dan akhirnya dikirimkan. Semua perintah ditulis pada void loop ini.

```
void loop()
```

INSTITUT BISNIS & INFORMATIKA stikom SURABAYA

```
{
      counter_sensor++;
      if(counter_sensor >= 1000)
      {
        k[in] = analogRead(A0);
        if(in == KOLOM)
        {
          enQueue(k);
          in = 0;
        }
        counter_sensor = 0;
      }
      counter_xbee++;
      if(counter_xbee >= 30000)
      {
        deQueue(k);

        Serial.print("#");
        for(out = 0; out<KOLOM; out++)
        {
          Serial.println(format(k[out],4));
        }
        Serial.print("#");
        Serial.println("");
        counter_xbee = 0;
      }
   delayMicroseconds(2);
}
```

INSTITUT BISNIS & INFORMATIKA stikom SURABAYA

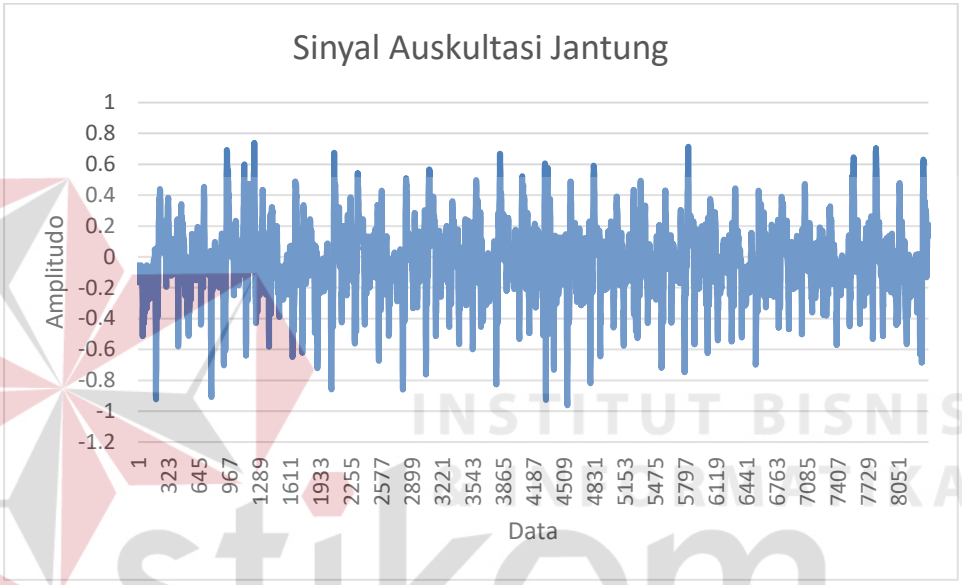
3.6 Metode Analisa

Pada proses pengiriman data hasil auskultasi sinyal jantung ini, selain membuat algoritma pengolahan dan pengiriman data, hal terpenting lainnya adalah analisa dari hasil pengiriman itu sendiri agar dapat diketahui apakah sistem yang sudah dibuat dapat berjalan dengan sesuai yang diharapkan.

3.6.1 Pengambilan Data Auskultasi Sinyal Jantung

Proses pengambilan data jantung dilakukan saat alat terpasang dan proses pembacaan serta pengiriman data jantung berlangsung. Letak posisi penempatan sensor pada jantung berada pada *left verticularr area.* Untuk

memastikan apakah data yang diambil merupakan data sinyal jantung, maka melalui grafik dapat dilihat apakah sensor sudah berada pada posisi yang tepat. Pengambilan sinyal jantung dilakukan selama 30 detik untuk mendapatkan hasil transmisi sinyal jantung. Contoh data sinyal auskultasi jantung dapat terlihat seperti pada Gambar 3.19.



Gambar 3.19. Grafik Hasil Auskultasi Sinyal Jantung

Hasil auskultasi sinyal yang dipresentasikan kedalam grafik merupakan hasil sinyal yang telah dirubah kedalam nilai tegangan. Cara merubah data menjadi tegangan adalah dengan cara memasukkan rumus sebagai berikut :

$$X = \left(\left(\left(\frac{data}{1024} \right) * 5 \right) - 2,5 \right)$$

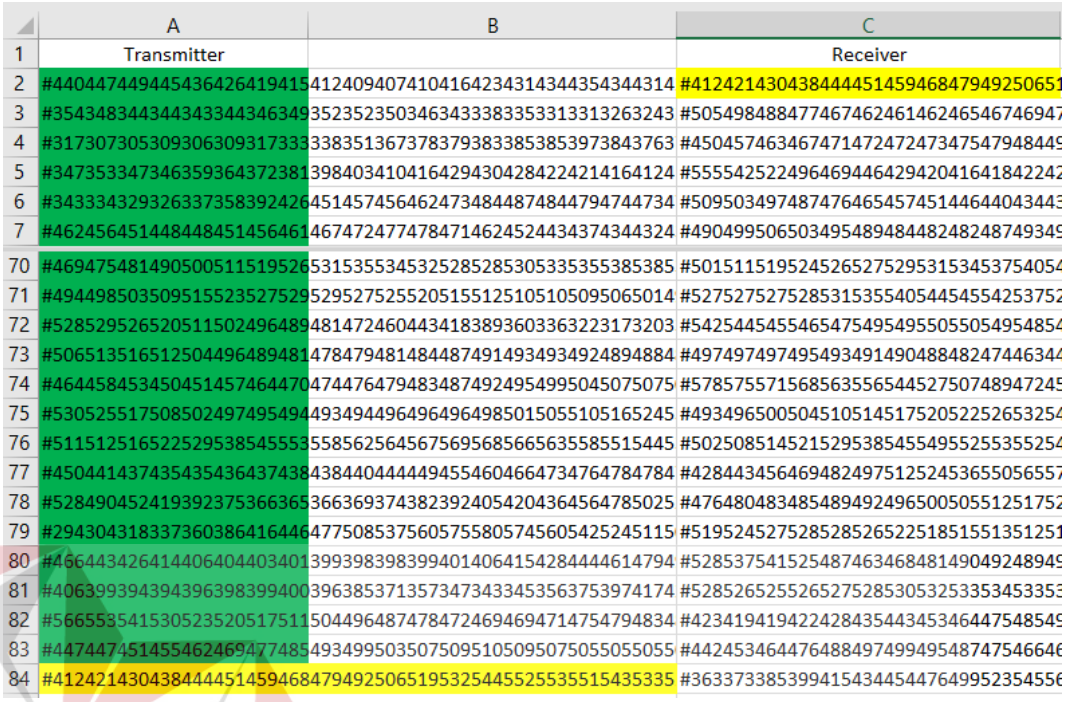
Berikut adalah penjelasan dari rumus diatas :

a. Pembagian 1024 : karena auskultasi sinyal jantung telah dikonversi menjadi data ADC dengan resolusi 10 bit.
b. Perkalian 5 : karena data diambil dari tegangan antara 0V-5V.
c. Pengurangan 2,5 : agar data yang terambil berada pada posisi tengah.

3.6.2 Analisa Transmisi Data Auskultasi Sinyal Jantung

Proses menganalisa hasil transmisi data sinyal jantung dilakukan dengan cara mengambil data dari sisi *transmitter* dan sisi *receiver* lalu data dibandingkan. Sehingga dapat diketahui *delay* dan data *loss* dengan mengamati data yang tidak sesuai dengan data yang terdapat pada pengirim. Selain itu dengan mengurangkan jumlah data yang terkirim dengan jumlah data yang diterima sehingga data yang *loss* dapat diketahui. Contoh data pada sisi pengirim dan penerima dapat terlihat pada Gambar 3.20.

INSTITUT BISNIS & INFORMATIKA stikom SURABAYA



	A	B	C
1	Transmitter		Receiver
2	#4404474494454364264194154124094074104164234314344354344314		#41242143043844445145946847949250651
3	#3543483443443433443463493523523503463433383353313313263243		#5054984884774674624614624654674694
4	#3173073053093063093173333383513673783793833853853973843763		#4504574634674714724724734754794844
5	#3473533473463593643723813984034104164294304284224214164124		#5555425224964694464294204164184224
6	#3433343293263373583924264514574564624734844874844794744734		#5095034974874764654574514464404344
7	#4624564514484484514564614674724774784714624524434374344324		#4904995065034954894844824824874934
70	#4694754814905005115195265315355345325285285305335355385385		#5015115195245265275295315345375405
71	#4944985035095155235275295295275255205155125105105095065014		#5275275275285315355405445455425375
72	#5285295265205115024964894814724604434183893603363223173203		#5425445455465475495495505505495485
73	#5065135165125044964894814784794814844874914934934924894884		#4974974974954934914904884824744634
74	#4644584534504514574644704744764794834874924954995045075075		#5785755715685635565445275074894724
75	#5305255175085024974954944934944964964964985015055105165245		#4934965005045105145175205225265325
76	#5115125165225295385455535585625645675695685665635585515445		#5025085145215295385455495525535525
77	#4504414374354354364374384384404444494554604664734764784784		#4284434564694824975125245365505655
78	#5284904524193923753663653663693743823924054204364564785025		#4764804834854894924965005055125175
79	#2943043183373603864164464775085375605755805745605425245115		#5195245275285285265225185155135125
80	#4664434264144064044034013993983983994014064154284444614794		#5285375415254874634684814904924894
81	#4063993943943963983994003963853713573473433453563753974174		#5285265255265275285305325335345335
82	#5665535415305235205175115044964874784724694694714754794834		#4234194194224284354434534644754854
83	#4474474514554624694774854934995035075095105095075055055055		#4424534644764884974994954874754664
84	#4124214304384444514594684794925065195325445525535515435335		#3633733853994154344544764995235455

Gambar 3.20 Paket Data pada Sisi Pengirim dan Penerima

Pencarian data *loss* seperti pada Gambar 3.20 maka dapat dilakukan perhitungan untuk mengetahui berapa *packet loss* yang terdapat dalam sekali percobaan.

a. *Delay*

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

Setelah disamakan paket data antara *transmitter* dan *receiver* maka terdapat selisih urutan antara kedua paket data tersebut, selisih urutan tersebutlah yang disebut *delay* dalam proses mentransmisikan sinyal auskultasi jantung.

Karena data dikirim setiap 2ms maka jarak antara data tersebut dikalikan dengan waktu pengiriman paket data dan akan ditemukan

berapa lama paket data yang dikirimkan oleh *transmitter* diterima oleh *receiver*.

b. *Packet Loss*

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

Jumlah paket yang tidak diterima dengan sempurna tersebut adalah *packet loss* yang digunakan untuk mecari berapa besar persentase paket data yang hilang. Kategori *packet loss* ditunjukkan pada Tabel 3.1.

Tabel 3.1 Kategori *Packet Loss*

Kategori Degredasi	Packet Loss (%)
Sangat Bagus	0
Bagus	3
Sedang	15
Jelek	25

Sumber : (Wulandari, 2016)

INSTITUT BISNIS & INFORMATIKA stikom SURABAYA