Aplikasi PC Dekstop sebagai Penerima Data pada Implementasi Jaringan Mesh Berbasis XBee

Sugondo Hadiyoso
Program Studi D3 Teknik Telekomunikasi
Fakultas Ilmu Terapan, Universitas Telkom
Bandung
sugondo.hadiyoso@gmail.com

***Abstrak*—Monitoring secara terpusat menjadi salah satu isu penting yang telah banyak diimplementasikan untuk berbagai aplikasi. Beberapa dilakukan secara *online* menggunakan internet sebagai bentuk penerapan Internet of Things (IoT) maupun secara *offline* untuk cakupan yang relatif kecil. Kegiatan monitoring dapat dilakukan secara *peer to peer* maupun secara bersamaan dari beberapa node sensor sehingga memerlukan aplikasi khusus yang dapat memonitor data secara bersamaan. Fokus pekerjaan yang dilakukan adalah membuat aplikasi untuk menerima data dan merepresentasikan data dalam bentuk grafik dari seluruh sensor node pada jaringan Mesh yang menggunakan *transceiver* XBee sebagai medianya. Data yang diterima dari setiap node akan dipisahkan berdasarkan penamaan node sehingga data dapat direpresentasikan dengan benar sesuai data yang dikirimkan oleh setiap node. Dari hasil yang diperoleh, program aplikasi yang direalisasiakn dapat memisahkan data antara node yang satu dengan node yang lain dengan nilai keberhasilan 100%.**

Kata kunci—Monitoring; Mesh; XBee; Node

I. PENDAHULUAN

Kontrol dan monitoring secara terpusat dan nirkabel menjadi salah satu tren aplikasi saat ini. Kemudahan akses dan fleksibilitas menjadi alasan aplikasi tersebut banyak dikembangkan. Permasalahan timbul ketika jumlah objek yang diamati lebih dari satu buah, dalam artian ada beberapa node yang dimonitor oleh sebuah perangkat.

Media komunikasi yang dapat digunakan diantaranya Wifi, Bluetooth dan Xbee. Dari ketiga perangkat tersebut XBee merupakan perangkat yang banyak digunakan. Kemudahan konfigurasi untuk berbagai tipe jaringan menjadi salah satu alasan. Selain itu kebutuhan power yang relatif kecil dan data *rate* yang rendah menjadikan perangkat ini menjadi lebih efisien untuk aplikasi monitoring yang membutuhkan data *rate* rendah.

Penelitian yang telah dilakukan Cai Ken adalah membuat sistem monitoring ECG menggunakan topologi mesh [1]. Pada penelitian tersebut direalisasikan sebuah aplikasi monitoring sinyal ECG dari beberapa pasien yang kemudian dapat dimonitor oleh user melalui jaringan internet. Data dari seluruh pasien dikirim melalui ZigBee ke *gateway*. Kemudian oleh *gateway* meneruskan informasi melalui internet. Pada penelitian ini tidak menggunakan ZigBee yang bertindak sebagai *router*. Sehingga dari dari sisi pasien langsung terkirim ke *gateway* tanpa melalui *router*. Penelitian lain yaitu mengimplementasikan jaringan mesh untuk monitoring EKG berbasis XBee [2]. Dimana pada penelitian yang dimaksud belum terdapat program aplikasi untuk menampilkan data dari masing-masing node sensor hanya sebatas pengujian pada sisi kordinator.

Dari pekerjaan yang telah dilakukan sebelumnya di atas, pada tulisan ini dibahas secara detail bagaimana membuat aplikasi yang mampu menerima data dari beberapa buah node, melakukan parsing data kemudian merepresentasikan data tersebut ke dalam grafik. Program aplikasi dibuat dengan menggunakan *open source* QT yang dilengkapi dengan GUI sebagai *interface*-nya. Tentunya, akan dibahas juga bentuk jaringan yang diimplementasikan. Untuk data yang dilakukan monitoring adalah data sinyal EKG.

Secara runtut pekerjaan yang dilakuakn akan dijelaskan pada bagian berikutnya terdiri dari: bagian 2 penjelasan teori XBee, demultipleksing, QT dan jaringan mesh, bagian 3 berisi penjelasan perancangan dan pembuatan aplikasi, bagian 4 berisi hasil dan diskusi, dan bagian 5 berisi kesimpulan.

II. TEORI DASAR

A. XBee

XBee pada penelitian ini digunakan sebagai media untuk mengirimkan data digital sinyal EKG melalui mikrokontroler. Terdapat 2 seri XBee yang ada dipasaran yaitu XBee S1 *(series 1)* dan XBee S2 *(series 2).* Modul XBee yang digunakan pada penelitian ini adalah XBee S2 karena pada jenis XBee tersebut mendukung untuk implementasi jaringan *mesh* ZigBee yang tidak ditemukan pada S1[3]. Tabel 1 berikut adalah perbandingan antara *series* 1 dan *series* 2. Konfigurasi parameter XBee untuk membentuk jaringan yang dimaksud menggunakan aplikasi XCTU.

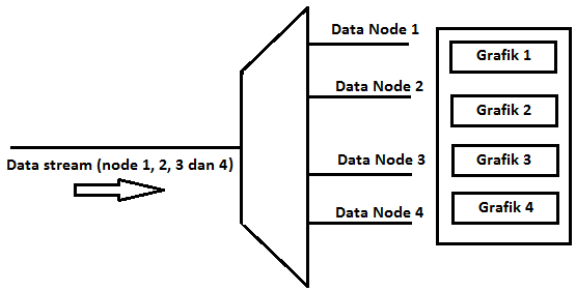
TABEL 1. PERBANDINGAN XBEE SERIES 1 DAN 2 [3]

Parameter	Series 1	Series 2
Firmware (typical)	802.15.4 point to point	ZB ZigBee mesh
Range (indoor)	30 m	40 m
Best (LOS) Range	100 m	120 m

Topologi Point to point, star	Ya	Ya
Topologi Mesh, cluster tree	Tidak	Ya

B. *Demultipleksing*

Demultipleksing adalah proses untuk mendapatkan kembali informasi sinyal hasil penggabungan oleh multiplekser. Data sinyal yang diterima dilakukan *parsing* untuk membedakan dari node mana data berasal. Proses tersebut dapat dilakukan pada sisi aplikasi. Untuk mempermudah demultipleksing, pada sisi pengirim untuk setiap node sudah memiliki kode masing-masing untuk membedakan antara node yang satu dengan yang lain. Gambar 1 berikut adalah ilustrasi demultipleksing yang diaplikasikan.



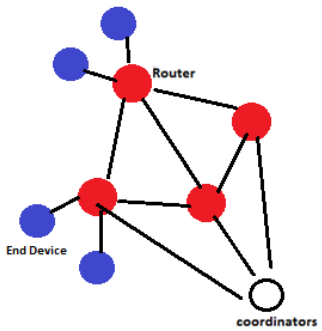
Gambar 1. Ilustrasi Demultipleksing

C. *QT Framework*

QT Framework merupakan *framework application cross platform*[4]. Dengan QT kita dapat membuat program untuk windows, Linux dan IOS. Menggunakan bahasa C++ sebagai basis pemrogramannya. QT dikembangkan oleh Nokia untuk memudahkan pembuatan aplikasi *mobile*. Karena QT lintas *platform*, maka kita dapat membuat aplikasi Dekstop menggunakan QT.

D. *Mesh Network*

Jaringan *mesh* atau jala memiliki ciri bahwa setiap node dapat saling berkomunikasi dengan node lain. Ciri lainnya adalah terdapat *router* sebagai alternatif rute untuk menyampaikan pesan ke kordinator [3]. Contoh jaringan mesh dapat dilihat pada Gambar 2 berikut.



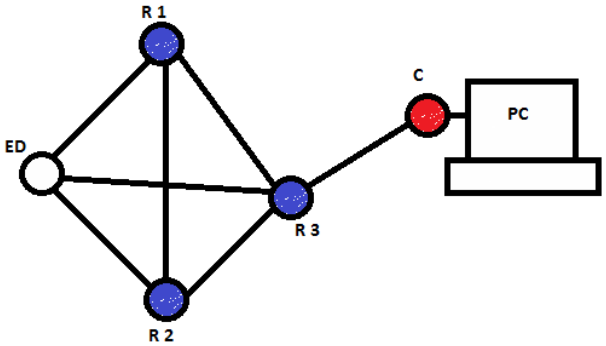
Gambar 2. Contoh Jaringan *Mesh*

III. RANCANGAN DAN IMPLEMENTASI

Perancangan *hardware* dan implementasi jaringan *mesh* telah dibahas pada tulisan sebelumnya [2]. Pada bagian ini, terdapat ulasan singkat untuk mengetahui konfigurasi yang telah diimplementasikan sebelumnya. Fokus pada bagian ini adalah perancangan algoritma untuk memisahkan data dari masing-masing node, rancangan GUI yang kemudian diimplementasikan dalam bentuk kode program. Detail penjelasan terdapat pada bagian berikut ini.

A. *Realisasi Jaringan Mesh*

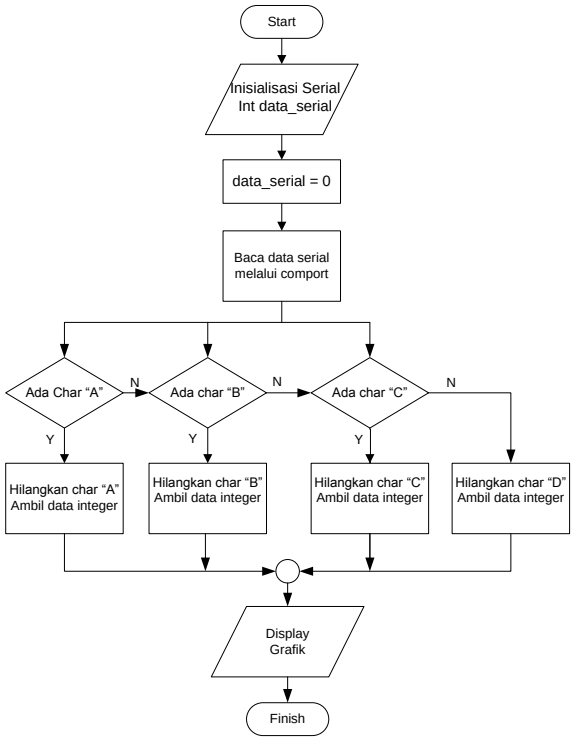
Topologi jaringan mesh yang direalisasikan terdiri dari 1 buah *end device*, 3 buah *router* dan 1 buah kordinator dengan XBee sebagai media pengirimnya. *End device* (ED) dan *router* (R1, R2 dan R3) masing-masing terhubung dengan perangkat akuisisi sinyal EKG. Dengan demikian terdapat 4 data sinyal EKG yang dapat dikirim secara bersamaan. Pada *end device* hanya dapat mengirimkan data, dimana fungsi untuk mencari jalur ketika terdapat perangkat dalam kondisi "off" dilakukan sepenuhnya oleh *router*. Lebih jelasnya dapat dilihat pada Gambar 3 berikut.



Gambar 3. Jaringan *Mesh* yang Direalisasikan

B. *Algoritma Program*

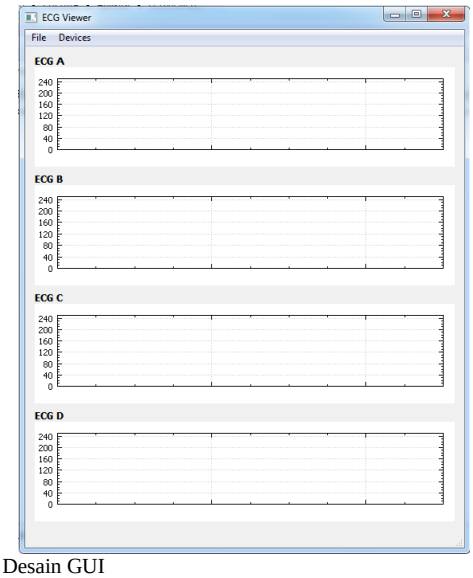
Secara sederhana, program aplikasi pada dekstop bertugas untuk menerima data serial dari XBee kordinator Via komunikasi serial USB kemudian melakukan pemilahan data dengan melihat karakter yang disisipkan pada setiap data integer yang dikirim. Sebagai contoh, A20 berarti data bersumber dari node A dan angka 20 adalah nilai integernya. Data hasil *parsing* selanjutnya di plot dalam bentuk grafik. Alur program dapat dilihat pada Gambar 4.



Gambar 4. *Flowchart* Program

C. *GUI Aplikasi*

GUI yang direalisasikan digunakan untuk menampilkan grafik sinyal EKG secara *real time*. Terdiri dari 4 buah grafik yang untuk menampilkan sinyal dari semua *node*. Aplikasi juga dilengkapi dengan fasilitas untuk penyimpanan data dalam bentuk file *text*. Desain GUI dapat ditunjukkan pada Gambar 5.



Gambar 5. Desain GUI

D. *Pemograman*

Program utama terdiri dari pengaturan komunikasi serial, pembacaan data, parsing data dan plot data dalam bentuk grafik. Berikut ini adalah potongan sub-program utama tersebut,

- Pengaturan komunikasi serial

```
PortSettings settings = {(BaudRateType)baudrate,
DATA_8, PAR_NONE, STOP_1, FLOW_OFF, 10};
    port     =     new     QextSerialPort(settings,
QextSerialPort::EventDriven);
    connect(port,            SIGNAL(readyRead()),
SLOT(onReadyRead()));
    connect(ui->actionExit,   SIGNAL(triggered()),
SLOT(onExit()));
    connect(ui->actionConnect,
SIGNAL(triggered()),
SLOT(onConnectDisconnect()));
```

- Pembacaan Data

```
bufReadData=bufReadData+QString::fromLatin1(port-
>readAll());
QStringList list =bufReadData.split("\n");
```

- Parsing Data

```
        QString readData = list.value(i);
           if(readData.contains("A")) {
               readData                          =
readData.replace("A","").trimmed();

               int in = readData.toInt();
               qDebug() << in;
               if(in >9) plotA.plot(in);
           } else if(readData.contains("B")) {
               readData                          =
readData.replace("B","").trimmed();
               int in = readData.toInt();
               if(in >9) plotB.plot(in);
           } else if(readData.contains("C")) {
               readData                          =
readData.replace("C","").trimmed();
               int in = readData.toInt();
               if(in >9) plotC.plot(in);
           } else if(readData.contains("D")) {
               readData                          =
readData.replace("D","").trimmed();
               int in = readData.toInt();
               if(in >9) plotD.plot(in);
           }
```

- Plot Grafik

```
plotA.plot(in); //in adalah variabel data
plotB.plot(in);
plotC.plot(in);
plotD.plot(in);
```

IV. HASIL DAN DISKUSI

Setelah rancangan program direalisasikan, selanjutnya dilakukan pengujian performansi dan verifikasi terhadap program aplikasi yang dibuat. Skenario pengujian yang dilakukan diantaranya: pengujian performa *parsing* data dan pengujian aplikasi untuk melakukan *plotting* grafik. Detail masing-masing skenario pengujian tersebut dijelaskan pada bagian berikut.

A. *Pengujian Parsing Data*

Keberhasilan aplikasi sangat ditentukan oleh performansi sub rutin *parsing* data. Pada bagian inilah, seluruh data yang diterima akan dipisahkan sesuai dengan *node* sumbernya. Pengujian dilakukan dengan melihat file *text* hasil penyimpanan data untuk setiap skenario. Skenario yang dimaksud, diantaranya: 1 node aktif, 2 node aktif, 3 node aktif dan seluruh node aktif. Pengujian dimaksudkan untuk melihat apakah terdapat data yang tidak di-*parsing* secara tepat. Hasil pengujian dapat dilihat pada Tabel 2.

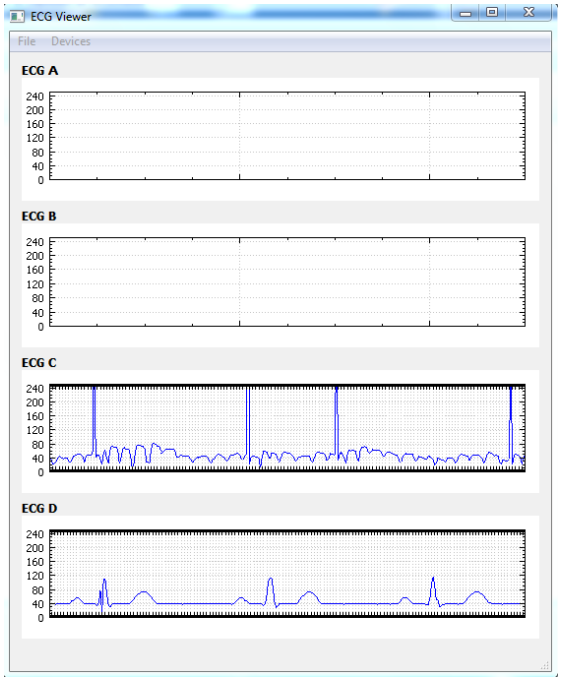
TABEL 2. HASIL PENGUJIAN *PARSING* DATA DALAM BENTUK TEXT

Node Aktif	File text			
	Data node A	Data node B	Data node C	Data node D
D	Null	Null	Null	**Any**
C, D	Null	Null	**Any**	**Any**
B , C , D	Null	**Any**	**Any**	**Any**
A, B, C, D	**Any**	**Any**	**Any**	**Any**

Dari tabel di atas dapat diketahui bahwa program untuk memisahkan karakter penanda dengan data integer berjalan dengan baik. Seluruh data dapat tersimpan secara tepat sesuai dengan *node*-nya masing-masing.

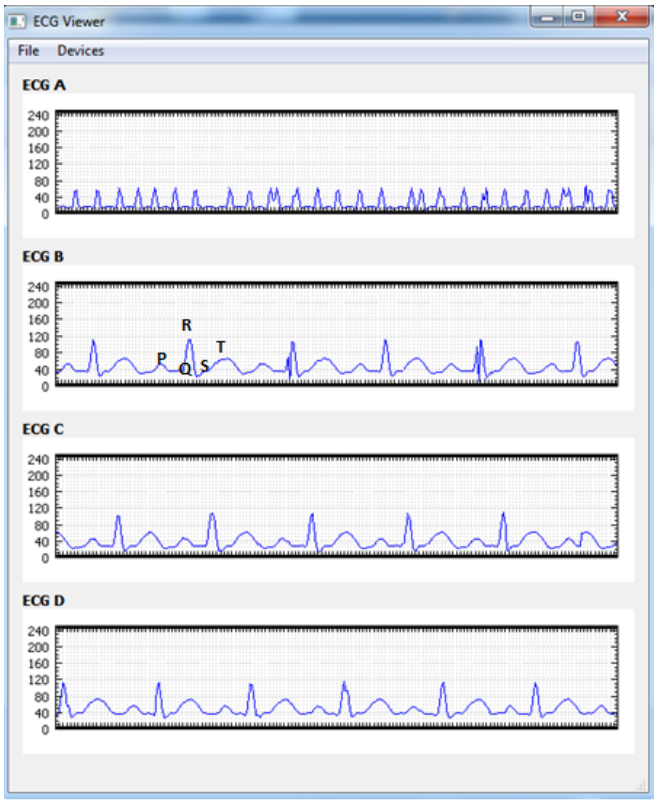
B. *Grafik Sebagai Hasil Representasi Data*

Data integer hasil *parsing* selanjutnya direpresentasikan dalam bentuk grafik dengan maksud untuk mengetahui apakah bentuk grafik merupakan sinyal EKG. *Sample* pengujian dilakukan saat 2 *node* aktif (C dan D) dan 4 *node* aktif. Beriku adalah Gambar masing-masing hasil pengujian tersebut.



Gambar 6. Grafik 2 *Node* Aktif

Dari grafik yang tergambar untuk masing-masing skenario, diperoleh kesimpulan bahwa aplikasi akan menampilkan grafik sesuai dengan *node* yang sedang aktif. Bentuk grafik menunjukkan representasi data sinyal EKG normal yang terdiri dari gelombang P-QRS-T(Gambar 7), ini berarti program untuk mendapatkan nilai *integer* berjalan dengan baik.



Gambar 7. Grafik Seluruh *Node* Aktif

V. KESIMPULAN

Dari pekerjaan yang telah dilakukan dapat diperoleh beberapa kesimpulan antara lain:

1. Demultipleksing data dapat dilakukan pada program aplikasi dengan teknik manipulasi data dalam prakteknya menggunakan *data parsing.*
2. Program pemisah data integer dan karakter dapat bekerja dengan baik. Data *integer* menempati variabel yang sesuai dengan *node*-nya masing-masing.
3. *Interface* GUI dalam bentuk grafik dapat menampilkan grafik sinyal EKG dari *node* yang aktif.
4. Tingkat keberhasilan program aplikasi yang dibuat untuk memisahkan data adalah 100%.

ACKNOWLEDGMENT

Tulisan ini merupakan salah satu bentuk keluaran dari penelitian dengan Judul "Monitoring EKG dengan Jaringan Mesh" yang didanai oleh Direktorat Penelitian dan Pengabadian Masyarakat (PPM) Universitas Telkom

REFERENSI

[1] Hadiyoso, Sugondo dan Mayasari, Ratna. 2014. Monitoring Elektrokardiograf Menggunakan Topolgi Mesh. Jurnal Elektro dan Telekomunikasi Terapan (JETT). Hal 75-82

[2] Faludi, Robert. 2010. Building Wireless Sensor Network. O'Reilly Media. Canada.

[3] QT Framework, "Aplikasi QT SDK" [online] tersedia di http://www.mediatutorial.web.id/2013/01/c-plus-plus-mengenal-qt-framework.html