SKRIPSI

PENGEMBANGAN APLIKASI UNTUK MENGANALISA SINYAL ELEKTROKARDIOGRAP (EKG) BERBASIS AKUISISI DATA ISYARAT ELEKTRIS JANTUNG

Disusun Oleh :

DWI CAHYA PRATOWO

06.12.597

JURUSAN TEKNIK ELEKTRO S-1

KONSENTRASI TEKNIK KOMPUTER DAN INFORMATIKA

FAKULTAS TEKNOLOGI INDUSTRI

INSTITUT TEKNOLOGI NASIONAL MALANG

2010

SKRIPSI

PENGEMBANGAN APLIKASI UNTUK MENGANALISA
SINYAL ELEKTROKARDIOGRAF (EKG) BERBASIS ANDROID
DATA ISYARAT ELEKTRIS JANTUNG

Disusun Oleh :

IDWI CAHYA PRATOMO
08.12.597

JURUSAN TEKNIK ELEKTRO S-1
KONSENTRASI TEKNIK KOMPUTER DAN INFORMATIKA
FAKULTAS TEKNOLOGI INDUSTRI
INSTITUT TEKNOLOGI NASIONAL MALANG

2010

LEMBAR PERSETUJUAN

PENGEMBANGAN APLIKASI UNTUK MENGANALISA SINYAL ELEKTROKARDIOGRAP (EKG) BERBASIS AKUISISI DATA ISYARAT ELEKTRIS JANTUNG

SKRIPSI

Disusun dan Diajukan Sebagai Salah Satu Syarat Untuk Memperoleh Gelar Sarjana Teknik Komputer Dan Informatika Strata Satu (S-1)

Disusun Oleh :

DWI CAHYA PRATOWO
NIM : 06.12.597

Diperiksa dan Disetujui

Dosen Pembimbing

I Komang Somawirata, ST. MT.
NIP.Y. 1030100361

Mengetahui

Ketua Jurusan Teknik Elektro S-1

Ir. Yusuf Ismail Nakhoda, MT.
NIP.Y. 1018800189

JURUSAN TEKNIK ELEKTRO S-1
KONSENTRASI TEKNIK KOMPUTER DAN INFORMATIKA
FAKULTAS TEKNOLOGI INDUSTRI
INSTITUT TEKNOLOGI NASIONAL MALANG
2010

PENGEMBANGAN APLIKASI UNTUK MENGANALISA SINYAL ELEKTROKARDIOGRAP (EKG) BERBASIS AKUISISI DATA ISYARAT ELEKTRIS JANTUNG

Dwi Cahya Pratowo

Jurusan Teknik Elektro S-1, Konsentrasi T.Komputer dan Informatika
Fakultas Teknologi Industri, Institut Teknologi Nasional Malang
Jln. Raya Karanglo Km 2 Malang
dwy_cah@yahoo.co.id

Dosen Pembimbing : I Komang Somawirata, ST. MT

Abstraksi

Jantung adalah organ penting dalam tubuh manusia yang difungsikan untuk memompa darah ke seluruh tubuh. Proses pemompaan darah ini terjadi karena otot jantung berkontraksi akibat mendapat rangsangan elektris atau impuls. Apabila terjadi ketidaknormalan pada kinerja jantung maka aliran darah pada tubuh akan mengalami gangguan. Untuk mengetahui keadaan ketidaknormalan jantung dapat dilakukan dengan pengukuran gelombang jantung yang dikenal dengan nama EKG. Hasil pengukuran gelombang jantung pada umumnya berupa kertas yang disebut elektrokardiogram. Dan diperlukannya pembacaan secara manual terhadap elektrokardiogram untuk mengetahui ketidaknormalan berdasarkan interval P-R, interval R-R, interval kompleks QRS dan interval segment ST. Aplikasi untuk menganalisa sinyal EKG berbasis akuisisi data isyarat elektris jantung di rancang untuk mengetahui secara otomatis ketidaknormalan hasil pengukuran gelombang jantung berdasarkan interval pada defleksi P, kompleks QRS dan defleksi T kedalam bentuk visual.

Kata Kunci: Jantung, EKG

KATA PENGANTAR

Puji syukur kehadirat Tuhan Yang Maha Esa yang telah memberikan rahmat dan hidayah-Nya, sehingga dapat diselesaikan skripsi yang berjudul **"PENGEMBANGAN APLIKASI UNTUK MENGANALISA SINYAL ELEKTROKARDIOGRAP (EKG) BERBASIS AKUISISI DATA ISYARAT ELEKTRIS JANTUNG"** ini dengan lancar. Skripsi ini merupakan persyaratan kelulusan Studi pada Jurusan Teknik Elektro S-1 Konsentrasi Teknik Komputer dan Informatika ITN Malang dan untuk mencapai gelar Sarjana Teknik.

Keberhasilan peyelesaian laporan skripsi ini tidak lepas dari dukungan dan bantuan berbagai pihak. Untuk itu penyusun menyampaikan terima kasih kepada :

1. Prof. Dr. Ir. Abraham Lomi, MSEE, selaku Rektor Institut Teknologi Nasional Malang.
2. Bapak Ir. Sidik Noertjahjono, MT, selaku Dekan Fakultas Teknologi Industri.
3. Bapak Ir. Yusuf Ismail Nakhoda, MT, selaku Ketua dan Sekretaris Jurusan Teknik Elektro S-1.
4. Bapak I Komang Somawirata, ST. MT, selaku Dosen Pembimbing.
5. Ayah dan Ibu serta saudara-saudara kami yang telah memberikan do'a restu, dorongan, semangat, dan biaya.
6. Rekan-rekan instruktur di Laboratorium Pemrograman Komputer dan Multimedia (PK&M) ITN Malang.

7. Semua yang telah membantu dalam penyelesaian penyusunan skripsi ini.

Penulis telah berusaha semaksimal mungkin dan menyadari sepenuhnya akan keterbatasan pengetahuan dalam menyelesaikan laporan ini. Untuk itu penyusun mengharapkan saran dan kritik yang membangun dari pembaca demi kesempurnaan laporan ini.

Harapan penyusun semoga laporan skripsi ini memberikan manfaat bagi perkembangan ilmu pengetahuan dan pembaca.

Malang, Agustus 2010

penyusun

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I

PENDAHULUAN

1.1. Latar Belakang

Pelayanan kesehatan terhadap penyakit jantung di Indonesia masih mengalami penurunan dibandingkan dengan negara negara berkembang ASEAN lainnya. Dokter spesialis jantung yang dimiliki negara Indonesia menurut Dr. H. Aulia Syawal, Pengurus Yayasan Jantung Indonesia (kompas.com, Rabu 4 Maret 2009, 16:56 WIB) mengatakan jumlah dokter yang dimiliki negara indonesia masih terbilang sangat kurang dibandingkan dengan jumlah dokter yang ada di luar negeri. Di luar negeri, satu orang dokter spesialis jantung melayani 250.000 penduduk atau sampai 500.000 penduduk, sedangkan di Indonesia hanya memiliki 400 orang dokter spesialis jantung dan melayani sebanyak lebih dari 200 juta jiwa penduduk[8].

Kendala utama yang dihadapi oleh pusat-pusat jantung di daerah adalah keterbatasan kuantitas serta kualitas sumber daya manusia (SDM) dalam intervensi transkater untuk penderita jantung. Kesemuanya ini berdampak pada banyaknya kasus penyakit jantung yang meninggal tanpa sempat terdiagnosis. Ditambahnya penyebaran dokter spesialis jantung, tidak terdistribusi merata di Indonesia[9]. Hanya terkonsentrasi dirumah sakit pusat pendidikan, sehingga rumah sakit di daerah pedesaan masih kekurangan dokter dan kekurangan alat alat medis yang digunakan didalam melakukan pemeriksaan terhadap penderita penyakit jantung.

Jantung adalah organ penting dalam tubuh manusia yang difungsikan untuk memompa darah ke seluruh tubuh. Proses pemompaan darah ini terjadi karena otot jantung berkontraksi akibat mendapat rangsangan elektris atau impuls. Rangsangan elektris berawal dari potensial aksi yang terjadi pada sel-sel otot jantung sendiri. Untuk mengetahui aktivitas elektris otot jantung diperlukan pencatatan atau perekaman dari permukaan tubuh.

Oleh karena itu diperlukan suatu sistem yang dapat digunakan untuk memberikan informasi mengenai hasil perekaman jantung serta mempercepat proses pradiagnosa dikarenakan informasi yang diberikan berupa hasil perhitungan interval R-R, interval QRS, interval P-R dan interval segment S-T. Bagi penderita jantung yang ada di daerah pedesaan dengan adanya sistem yang akan dikembangkan ini secara tidak langsung dapat memberikan informasi dan pengetahuan melalui visualisasi elektrokardiogram serta mengurangi kematian tanpa terdiagnosa yang disebabkan oleh penyakit jantung.

1.2. Rumusan Masalah

Berdasarkan latar belakang diatas, telah diambil permasalahan yang akan dibahas sebagai berikut :

1. Lambatnya pelayanan, penanganan dan proses pendiagnosaan bagi penderita penyakit jantung.
2. Data elektrokardiogram yang masih berupa kertas, mengharuskan melakukan pembacaan dilakukan dengan menghitung banyaknya kotak berdasarkan defleksi gelombang.
3. Kurangnya perhatian masyarakat, terhadap penderita penyakit jantung.

1.3. Tujuan Penelitian

Adapun tujuan yang diharapkan dalam penelitian ini adalah :

1. Mempercepat proses pendiagnosaan terhadap penderita penyakit jantung.
2. Mengubah data elektrokardiogram kedalam bentuk visualisasi grafis.
3. Memberikan informasi secara langsung kepada staf medis/non dokter serta pasien terhadap batasan-batasan normal dari masing-masing defleksi yang dihasilkan, setelah melakukan proses perekaman data.

1.4. Batasan Masalah

Agar permasalahan mengarah sesuai dengan tujuan yang diharapkan, maka pembahasan dibatasi oleh hal-hal sebagai berikut :

1. Lingkup dari pengembangan sistem hanya memanfaatkan data yang diterima dari EKG melalui Mic In yang ada pada port Audio.
2. Sinyal yang didapat diasumsikan ideal, yaitu tidak memperhitungkan noise yang berarti semua noise dianggap stasioner.
3. Tidak membahas spesifikasi dari alat yang digunakan.
4. Pengembangan Aplikasi elektrokardiograp (EKG) dibuat pada aplikasi berbasis windows dengan menggunakan bahasa pemrograman Borland Delphi 7.

1.5. Metode Penelitian

Adapun metode penelitian yang digunakan adalah sebagai berikut:

1. Studi literatur

   Pengumpulan data yang dilakukan dengan mencari bahan-bahan kepustakaan dan referensi dari berbagai sumber sebagai landasan teori yang ada hubunganya dengan permasalahan yang dijadikan objek penelitian.

2. Hipotesa

   Data dan informasi yang telah diperoleh dari hasil tanya jawab dan sharing kepada dokter akan dijadikan sebagai langkah awal dalam proses pendiagnosaan yang bertujuan untuk mendefinisikan batasan batasan normal yang digunakan sebagai acuan pendiagnosaan dalam perancangan sistem.

3. Perancangan Sistem

   Berdasarkan data dan informasi yang telah diperoleh, akan dibuat rancangan kerangka global yang menggambarkan mekanisme dari sistem yang akan dibuat.

4. Pembuatan Software

   Tahapan ini menerjemahkan hasil perancangan spesifikasi program dari tahapan sebelumnya kedalam baris–baris kode program yang dapat dimengerti oleh komputer.

5. Pengujian Sistem

   Tahapan ini melakukan pengujian program yang telah disesuaikan dengan batasan-batasan normal yang ditentukan kepada seorang pasien, untuk mengetahui secara detail bagaimana hasil yang didapatkan.

6. Pembahasan dan Analisa

   Pada tahap ini, sistem yang telah selesai dibuat akan dianalisa, yaitu berdasarkan fungsionalitas program, dan akan dilakukan koreksi dan penyempurnaan program jika diperlukan.

1.6. Sistematika Penulisan

Untuk mempermudah dan memahami pembahasan penulisan skripsi ini, maka sistematika penulisan disusun sebagai berikut :

Bab I : Pendahuluan

Berisi Latar Belakang, Rumusan Masalah, Tujuan Penelitian, Pembatasan Permasalahan, Metode Penelitian dan Sistematika Penulisan.

Bab II : Landasan Teori

Berisi tentang landasan teori mengenai permasalahan yang berhubungan dengan penelitian yang dilakukan.

Bab III : Perancangan Sistem

Dalam bab ini berisi mengenai analisa kebutuhan sistem baik software maupun hardware yang diperlukan untuk membuat

kerangka global yang menggambarkan mekanisme dari sistem yang akan dibuat.

Bab IV : Pengujian dan Analisa Sistem

Berisi tentang implementasi dari perancangan sistem yang telah dibuat serta pengujian terhadap sistem tersebut.

Bab V : Penutup

Merupakan bab terakhir yang memuat intisari dari hasil pembahasan yang berisikan kesimpulan dan saran yang dapat digunakan sebagai pertimbangan untuk pengembangan penulisan selanjutnya.

BAB II

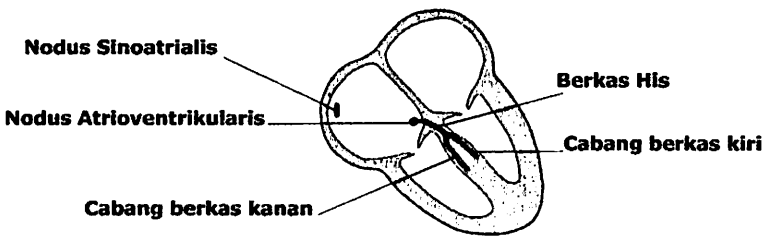
LANDASAN TEORI

2.1. Sistem Pengolahan Kelistrikan Jantung

2.1.1. Penghantaran Listrik Jantung (Wiring)[2]

Jantung adalah organ muskular yang berfungsi sebagai pompa ganda sistem kardiovaskular. Sisi kanan jantung memompa darah ke paru-paru sedangkan sisi kiri memompa darah ke seluruh tubuh. Kontraksi otot terjadi karena perubahan listrik yang disebut "depolarisasi", dan perubahan ini dapat dideteksi oleh elektrode yang ditempelkan pada permukaan tubuh. Perubahan listrik pada konstraksi otot jantung hanya akan tampak jelas jika pasien relaksasi penuh dan tidak ada otot skelet yang berkontraksi karena semua kontraksi otot akan terdeteksi.

Walaupun jantung mempunyai empat ruang, dari sudut pandang listrik, jantung hanya mempunyai dua ruang karena kedua atrium berkontraksi besamaan dan kemudian diikuti kontraksi kedua ventrikel secara bersamaan. Diagram penghantaran listrik jantung ditunjukan pada Gambar 2.1 berikut.



Gambar 2.1 Diagram penghantar listrik (wiring) jantung

Discharge listrik untuk setiap siklus jantung normalnya dimulai di daerah khusus di atrium kanan yang disebut "*nodus sinoatrialis (SA)*". Depolarisasi kemudian menyebar melalui serabut otot atrium. Terjadi pelambatan (*delay*) pada saat penyebaran depolarisasi melalui daerah khusus lain di atrium, yaitu "*nodus atrioventrikularis (AV)*". Setelah itu, listrik berjalan amat cepat turun melalui jaringan konduksi khusus: jalur tunggal, "berkas His", yang kemudian pada septum interventrikular becabang menjadi cabang berkas kanan dan kiri. Cabang berkas kiri sendiri bercabang menjadi dua. Di dalam masa otot ventrikel, konduksi menyebar agak lebih lambat, melalui jaringan khusus yang disebut "serabut Purkinje"

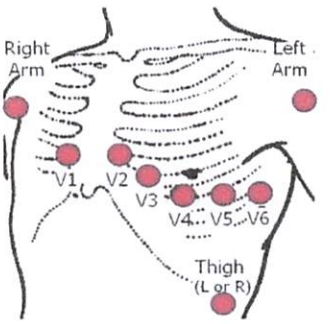
2.1.2. Irama Jantung

Kata "irama" digunakan untuk menyebut bagian jantung yang mengendalikan rangkaian aktivasi tersebut. Irama jantung normal, yang aktivasi listriknya mulai dari nodus SA, disebut "irama sinus".

2.2. Elektrokardiografi

2.2.1. Definisi Elektrokardiografi

Elektrokardiogram adalah grafik yang merekam aktivitas kelistrikan jantung. Grafik yang menggambarkan rekaman listrik jantung dihasilkan oleh sebuah alat yang disebut *electrocardiograf.* Elektrokardiografi adalah ilmu yang mempelajari aktifitas kelistrikan jantung[2]. Posisi peletakan elektrode dapat terlihat seperti Gambar 2.2 berikut.



Gambar 2.2 Posisi Elektrode pada tubuh

Untuk memperoleh *elektrokardiogram* beberapa elektrode dipasang pada permukaan tubuh pasien. Elektrode ini dihubungkan ke *elektrokardiograf (EKG)* melalui kabel. Dari grafik yang dihasilkan *elektrokardiograf,* dokter akan mendapatkan informasi tentang aktivitas kelistrikan jantung yang kemudian dilakukan analisa terhadap kondisi jantung. Sandapan/perekaman EKG dibagi menjadi 3 yaitu: sandapan Bipolar, sandapan Unipolar Ekstremitas dan sandapan Prekordial[5].

2.2.2. Sandapan Bipolar (Lead I, Lead II, Lead III)[5]

Sandapan bipolar hanya memerlukan 2 elektroda yaitu elektroda negatif dan elektroda positif, dimana elektroda di tempatkan pada tangan kanan (RA), tangan kiri (LA) dan kaki kiri (LL).

2.2.3. Sandapan Unipolar Ekstrimitas *aVR, aVF,Avl* [5]

Sandapan ini memutuskan elektroda negarif dengan central terminal, sehingga dihasilkan voltase defleksi yang lebih besar dibandingkan dengan sandapan bipolar.

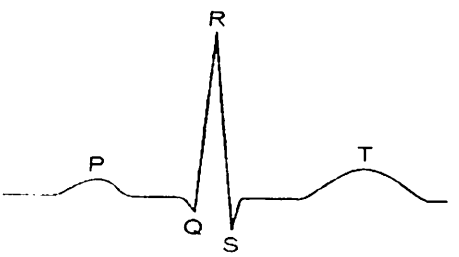
2.2.4. Sandapan Unipolar Prekordial V1, V2, V3, V4, V5 dan V6[5]

Sandapan ini merekam bioelektrikal jantung dari bidang hirizontal, dimana menggunakan 6 single positip elektroda yang ditempatkan pada permukaan tubuh manusia.

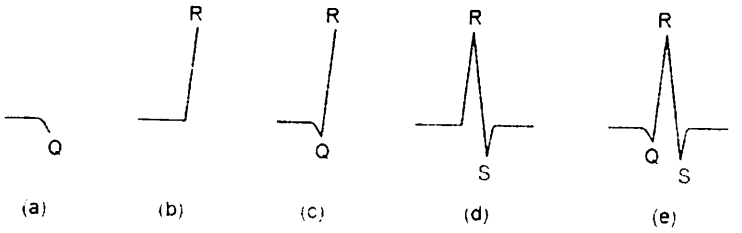
2.3. Bentuk EKG[2]

Massa otot atrium lebih kecil daripada massa otot ventrikel, sehingga perubahan listrik yang menyertai kontraksi atrium juga kecil. Kontraksi atrium menyebabkan gelombang "P". Karena massa ventrikel besar, pada saat ventrikel terdepolarisasi, terjadi defleksi EKG yang besar. Ini desebut kompleks "QRS". Gelombang "T" terjadi karena massa ventrikel kembali ke status listrik istirahat ("repolarisasi").

Huruf P,Q,R,S dan T dipilih dan ditentukan pada awal sejarah EKG. Defleksi P, Q, R, S, dan T disebut gelombang. Gelombang Q, R dan S membentuk suatu kompleks, dan interval antara gelombang S dan gelombang T disebut "segmen ST". Bentuk dasar EKG normal ditunjukan pada Gambar 2.3.



Gambar 2.3 Bentuk dasar EKG normal

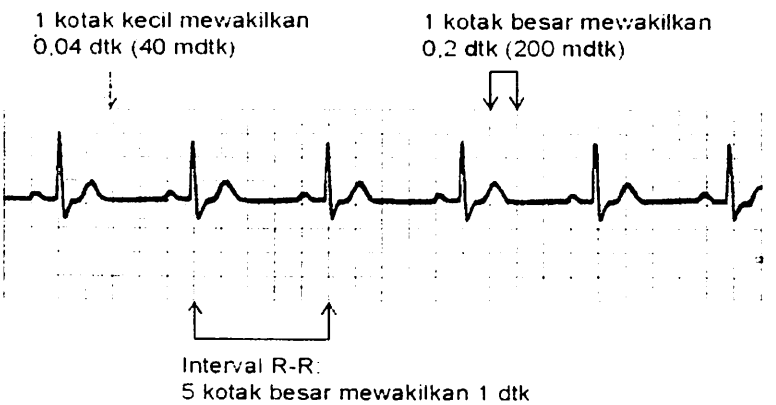


Gambar 2.4 Bagian-bagian pada kompleks QRS. (a) gelombang Q. (b,c) gelombang R. (d,e) gelombang S

Defleksi ke bawah pertama disebut gelombang Q (Gambar 2.4). Defleksi ke atas disebut gelombang R (Gambar 2.4), baik didahului oleh gelombang Q maupun tidak. Setiap defleksi dibawah garis dasar pascagelombang R disebut gelombang S (Gambar 2.4), baik didahului gelombang Q maupun tidak.

2.4. Waktu dan Kecepatan[2]

Semua mesin EKG berjalan pada kecepatan standar dan menggunakan kertas kotak-kotak berukuran standar. Satu kotak besar (5mm) mewakilkan 0,2 detik atau 200 milidetik (mdtk), sehingga ada lima kotak besar per detik atau 300 kotak besar per menit. Waktu dan kecepatan ditampilkan seperti Gambar 2.5.



Gambar 2.5 Hubungan antara kotak-kotak pada kertas EKG dan waktu.

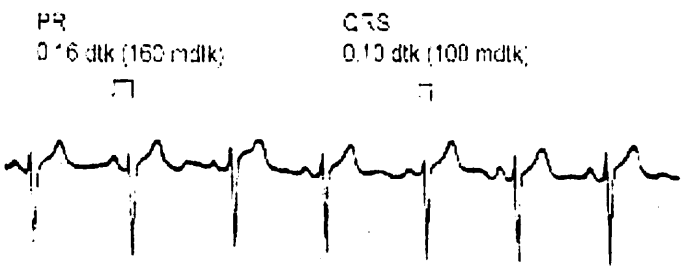
EKG, seperti kompleks QRS yang terjadi satu kali per kotak besar, berlangsung pada kecepatan 300/mnt (Gambar 2.5). Frekuensi jantung dapat cepat dihitung dengan mengingat rangkaian pada Tabel 2.1.

Tabel 2.1 Hubungan antara jumlah kotak besar
interval R-R dan frekuensi jantung

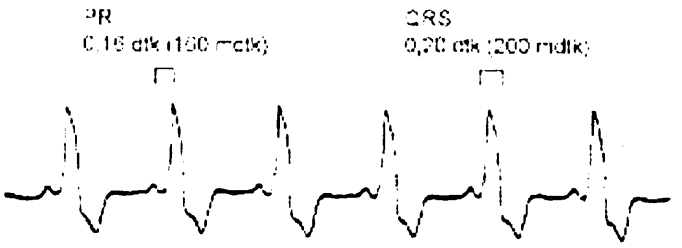
Interval R-R (kotak besar)	Frekuensi Jantung (denyut/mnt)
1	300
2	150
3	100
4	75
5	60
6	50

Seperti halnya jarak antara 2 gelombang R menunjukkan frekuensi jantung, demikian juga jarak antara bagian-bagian kompleks P-QRS-T menunjukkan waktu yang diperlukan oleh konduksi *discharge* listrik untuk menyebar melalui berbagai bagian jantung.

Interval PR diukur dari permulaan gelombang P sampai permulaan kompleks QRS, dan merupakan waktu yang diperlukan oleh eksitasi untuk menyebar dari nodus SA melalui otot atrium dan nodus AV, turun melalui berkas His, dan masuk kedalam otot ventrikel. Durasi interval PR ditampilkan pada Gambar 2.6, dan durasi kompleks QRS pada Gambar 2.7.



Gambar 2.6 Durasi Interval PR



Gambar 2.7 Durasi Kompleks QRS

Interval PR normal adalah 0,12-0,2 dtk (120-200 mdtk), atau tiga sampai lima kotak kecil. Kebanyakan waktu tesebut terpakai oleh pelambatan di nodus AV (Gambar.2.6) jika interval PR amat pendek, berarti atrium terdepolarisasi di dekat nodus AV atau terjadi konduksi cepat yang abnormal dari atrium ke ventrikel.

Durasi kompleks QRS menunjukkan waktu yang diperlukan oleh eksitasi untuk menyebar melalui ventrikel. Durasi QRS normal adalah 0,12 dtk (120 mdtk = tiga kotak kecil atau kurang). Pada gangguan konduksi, durasi ini lebih lama dan menyebabkan kompleks QRS melebar Gambar.2.7.

2.5. Algoritma dan Pemrograman Delphi

2.5.1. Algoritma[3]

Algoritma memegang peranan penting dalam bidang pemrograman. Sebegitu pentingnya suatu algoritma, sehingga perlu dipahami konsep dasar algortima. Algoritma adalah kumpulan instruksi yang dibuat secara jelas untuk menunjukan langkah-langkah penyelesaian suatu masalah.

Dalam bidang komputer, misalnya EDP(Elektronik Data Processing) atau MIS (Management Informatio System), algoritma sering dimanfaatkan untuk menyelesaikan suatu masalah atau untuk proses pengambilan keputusan seorang system analisis (analisist system) tentunya menggunakan algoritma untuk merancang suatu system. Bagi seorang programmer, algoritma digunakan untuk membuat modul-modul program. Guna memahami suatu algoritma, harus dimiliki pengetahuan dasar matematika karena pada dasarnya algoritma lahir dari konsep logika matematika. Disini yang perlu dilatih adalah kemampuan logikanya agar benar-benar bias menyusun langkah-langkah penyelesaian masalah dengan baik.

2.5.2. Konsep Dasar Algoritma[3]

Algoritma adalah kumpulan instruksi/perintah yang dibuat secara jelas dan sistematis berdasarkan urutan yang logis (logika) untu penyelesaian suatu masalah. French, C.S (1984) menyatakan sejumlah konsep yang mempunyai relevansi dengan masalah rancangan program yaitu kemampuan komputer, kesulitan dan ketepatan.

Penerapan dari konsep tersebut biasanya digunakan dalam rancangan algoritma. Dalam merancang sebuah algoritma, Fletcher (1991) memberikan beberapa cara atau metode yaitu kumpulan perintah, ekspresi, table instruksi, program komputer, kode semu dan flowchart. Untuk keperluan matematika dan program komputer metode yang sering digunakan yaitu;

1. Diagram Alir (Flow Chart)
2. Kode Semu (Pseudo Code)
3. Algoritma Fundamental

Didalam merancang sebuah algoritma ada 3 (tiga) komponen yang harus ada yaitu;

a. Komponen Masukan (Input)

Komponen ini biasanya terdiri dari pemiliha variable, jenis variable, tipe variable, konstanta dan parameter (dalam fungsi).

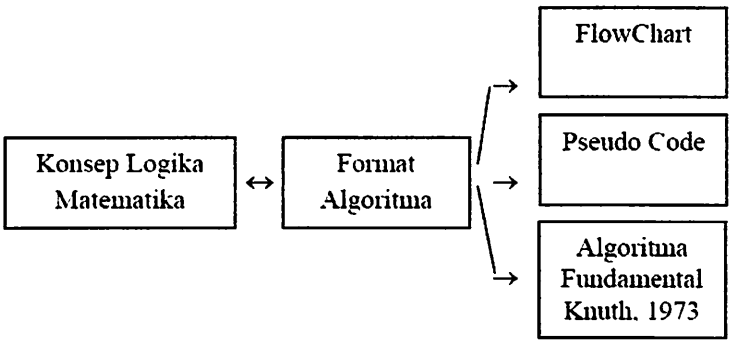
b. Komponen Keluaran (Output)

Komponen ini merupakan tujuan dari perancangan algoritma dan program.

c. Komponen proses (Processing)

Komponen ini merupakan bagian utama dan terpenting dalam merancang sebuah algoritma. Dalam bagian ini terdapat logika masalah, logika algoritma (sintaksis dan semantik), rumusan, metode (rekursi, perbandingan, penggabungan, pengurangan dan lain-lain).

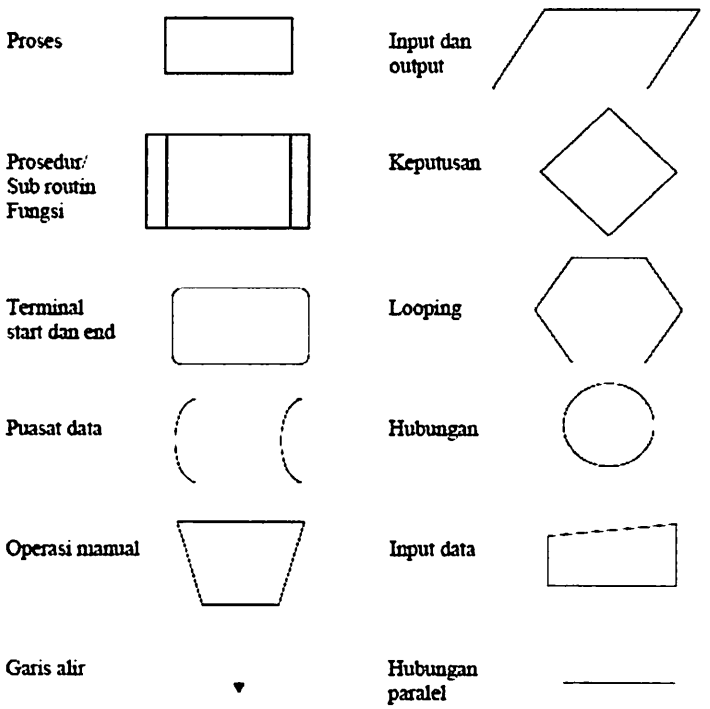
Struktur hubungan dan jenis algoritma dapat terlihat jelas seperti Gambar 2.8 berikut;



Gambar 2.8 Struktur Hubungan dan Jenis Algoritma

2.5.3. Diagram Alir[3]

Algoritma ini menggunakan sejumlah symbol untuk menyelesaikan kegiatan-kegiatan secara keseluruhan. Symbol dan artinya dalam diagram alir dapat terlihat seperti pada gambar 2.9 berikut;



Gambar 2.9 Simbol dan arti Diagram Alir

2.5.4. Kode Semu[3]

Dalam merancang sebuah algoritma menggunakan kode semu, komponen-komponen input, output dan proses harus terdefinisi secara jelas.

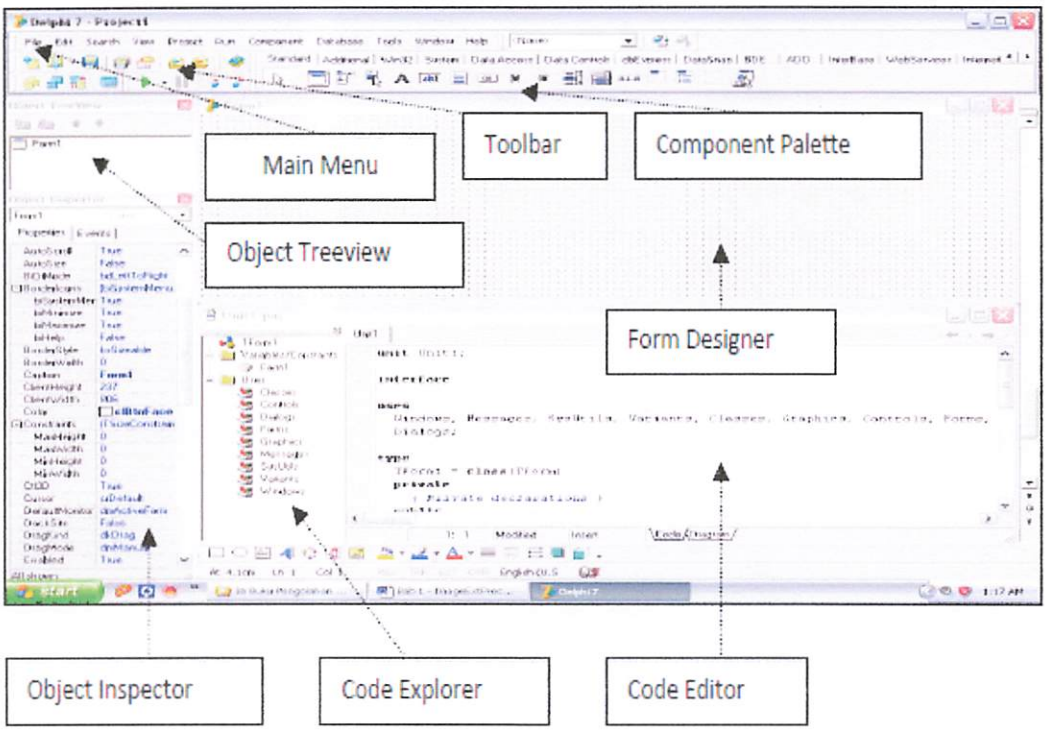
2.5.5. Algoritma Fundamental[3]

Knuth (1973) menyajikan format algoritma yang dapat digunakan secara bebas untuk berbagai bahasa pemrograman, artinya dapat dengan mudah diimplementasikan menggunakan Pascal, C, Fortran, PL atau BASIC.

2.5.6. Pengenalan Delphi[6]

Delphi merupakan suatu bahasa pemrograman yang memberikan berbagai fasilitas pembuatan aplikasi visual. Keunggulan bahasa pemrograman ini terletak pada produktivitas, kualitas, software development, kecepatan kompilasi, pola desain yang menarik serta diperkuat dengan pemrogramannya yang terstruktur.

IDE atau Integrated Development Environment merupakan lingkungan kerja yang disediakan oleh Delphi untuk para user dalam mengembangkan proyek aplikasi. IDE dalam program Delphi (versi 7) terbagi menjadi delapan bagian utama : Main Menu, ToolBar, Componen Palette, Form Designer, Code Editor, Object Inspector, Code Explorer, dan Object Treeview. Integrated Development Environment ditampilkan pada Gambar 2.10 dan penjelasan mengenai bagian-bagian dari IDE ditampilkan pada Tabel 2.2 berikut.



Gambar 2.10 IDE Delphi

Tabel 2.2 Bagian Utama dalam Delphi

No	Bagian	Keterangan
1	**Main Menu**	-
2	**ToolBar**	Memuat sejumlah icon untuk keperluan pengoperasian dengan cepat
3	**Component Palette**	Memuat sekumpulan representative icon berbagai komponen pada VCL (Visual Component Library) 
4	**Object Inspector**	Memuat tab property dan events
5	**Code Editor**	Tempat penulisan listing program/syntax
6	**Form Designer**	Tempat perancangan tampilan program

7	**Object Treeview**	Diagram pohon berbagai komponen yang digunakan.
8	**Code Explorer**	Memudahkan pemakai untuk berpindah antar file unit Memuat diagram pohon yang merepresentasikan semua type, calss, property, method, global, dan routine global variable yang telah didefinisikan didalam unit

Awal Delphi dijalankan secara default Code Editor akan menampilkan kode program sebagai berikut dalam bagian unit;

```
unit Unit1;
interface
uses
Windows, Messages, SysUtils, Variants, Classes,
Graphics, Controls, Forms,
Dialogs;
type
TForm1 = class(TForm)
private
{ Private declarations }
public
{ Public declarations }
end;
var
Form1: TForm1;
implementation
{$R *.dfm}
end.
```

Penjelasan code diatas sebagai berikut;

Header unit

Dinyatakan dengan kata unit yang diikuti dengan nama unit yang juga merupakan nama file unit yang disimpan dengan ekstensi .pas

Interface

Merupakan bagian yang dapat berisi deklarasi tipe data (termasuk kelas), konstanta, variable, procedure atau function. Segala sesuatu yang dideklarasikan pada bagian ini dapat diakses oleh unit lain.

Uses

Merupakan klausa yang menyatakan library yang akan dikompilasi menjadi file eksekusi.

Type

Merupakan bagian yang digunakan untuk mendeklarasikan variable.

Private

Modul dalam suatu private tidak dapat dipanggil dari modul lain. Property dalam suatu private tidak dapat dibaca atau dituliskan pada modul lain.

Public

Modul dalam suatu public dapat dipanggil dari modul lain. Properti dalam suatu public dapat dibaca atau dituliskan pada modul lain.

Var

Bagian yang dapat digunakan untuk mendeklarasikan variable

Implementation

Bagian yang berisikan implementasi kelas, procedure dan function yang telah dideklarasikan pada bagian interface. Bagian ini juga dapat berisikan deklarasi tipe data, variable, konstanta, procedure atau function yang bersifat internal terhadap unit.

{$R*. DFM}

Direktif ini berfungsi sebagai pengikat form ke file ***.dfm***

2.6. Windows Application Programming Interface[4]

2.6.1. Windows API

Windows API (Application Programming Interface) merupakan kumpulan fungsi-fungsi eksternal yang terdapat dalam file-file perpustakaan windows (selanjutnya sering disebut sebagai libary windows) atau file libary lainnya yang dapat digunakan oleh program kita.

Fungsi ini dapat menangani semua yang berhubungan dengan Windows, seperti pengaksesan disk, interface printer, grafik windows, kotak dialog (buka file, simpan file, memilih font, memilih warna, dan lain-lain), windows shell, settings sistem komputer, penanganan file, mengakses sistem registry, memainkan musik, dan sebagainya. Fungsi ini menyediakan fitur-fitur standar untuk semua program yang berbasis windows.

Semua fungsi windows API hampir terdapat dalam direktor sistem milik windows (biasanya terdapat dalam direktori C:\Windows\System dan C:\Windows, bergantung pada setting pertama installasi Windows) dan paling banyak berekstensi .DLL yang digunakan oleh sistem operasi windows.

Selain itu fungsi ini juga untuk memastikan secara konsisten penggunaan semua sumber yang terdapat dalam Windows. File-file itulah yang disebut dengan **Windows API**.

Disebabkan fungsi Windows API merupakan fungsi eksternal, maka menggunakan fungsi tersebut terlebih dahulu dideklarasikan pada ***uses*** (yang menyatakan library yang akan dikompilasi menjadi file eksekusi). Setelah pendeklarasian selesai, selanjutnya kita bebas untuk menggunakan fungsi Windows API tersebut pada program layaknya bahasa Borland Delphi.

2.6.2. File DLL

File DLL (Dynamic Link Library) adalah file library windows, merupakan kode yang sudah dikompilasi dan dapat digunakan oleh program lain. Jika kita meletakan fungsi sub rutin ke dalam dll, berarti fungsi tersebut dapat diakses oleh semua program pada saat yang bersamaan. DLL biasanya ditulis dengan bahasa C/C++, Delphi atau bahasa lainnya yang mendukung sistem operasi windows.

Dengan memanggil fungsi yang terdapat dalam DLL, kita dapat mengakses ribuan fungsi yang berhubungan dengan sistem windows, dengan kualitas sebaik yang kita gunakan dalam bahasa yang kita gunakan.

Berikut adalah nama-nama library milik Windows pada Tabel 2.3 yang sering dan paling banyak digunakan dalam Windows API.

Tabel 2.3 Dynamic Link Library Windows

No	Nama File	Deskripsi File
1	Advapi32.dll	Library yang mendukung fungsi-fungsi keamanan dan rutin-rutin registry
2	Xomdilg32.dll	Standart kotak dialog windows
3	Gdi32.dll	Penanganan grafik windows
4	Kernel32.dll	Fungsi system operasi windows 32 bit
5	Lz32.dll	Fungsi kompresi File
6	Mpr.dll	Fungsi Internet
7	Netapi32.dll	Fungsi Jaringan
8	Shel32.dll	Library shell 32 bit
9	User32.dll	Penanganan rutin user interface

10	Version.dll	Versi windows
11	Winmm.dll	Fungsi fungsi multimedia windows
12	Winspool.drv	Fungsi Fungsi printer spooler

Yang perlu kita lakukan untuk menangani fungsi-fungsi dalam file library windows yaitu dengan menspesifikasikan di mana fungsi tersebut ditemukan dan menyediakan informasi yang dibutuhkan fungsi pada bagian pendeklarasian fungsi Windows API.

2.7. SoundCard[11]

Soundcard adalah peralatan tambahan dalam system PC (personal computer) untuk memasukan dan mengeluarkan sinyal suara. Komponen utama soundcard adalah ADC (Analog-to-Digital Converter) dan DAC (Digital-to-Analog Converter).

1. Driver CD-ROM dengan keluaran CD-DA dan control volume
2. DAC 16-bit dengan karaktersitik:
   - Pencuplikan PCM (Pulse Code Modulation) yang linear.
   - Kemampuan transfer DMA atau FIFO tersangga dengan interupsi pada kekosongan penyangga.
   - Laju pencuplikan sebesar 44,1 KHz, 22,05 KHz dan 11,025 KHz.
3. ADC 16-bit dengan karakteristik sebagai berikut;
   - Pencuplikan PCM yang linear
   - Kemampuan transfer DMA atau FIFO tersangga dengan interupsi pada kekosongan penyangga.
   - Laju pencuplikan sebesar 44,1 KHz; 22,05 KHz dan 11,025 KHz

- Masukkan dari Mikrofon

4. Pencampuran internal dengan kemampuan sebagai berikut;
   - Dapat mengkombinasikan tiga sumber suara dan menampilkannya sebagai keluaran stereo, dengan taraf line-out pada panel belakang.
   - Sumber pencampuran adalah CD audio, synthesizer, dan DAC.
   - Tiap sumber pencampuran mempunyai control volume 3 bit dengan kenaikan logaritmik.

BAB III

PERANCANGAN SISTEM

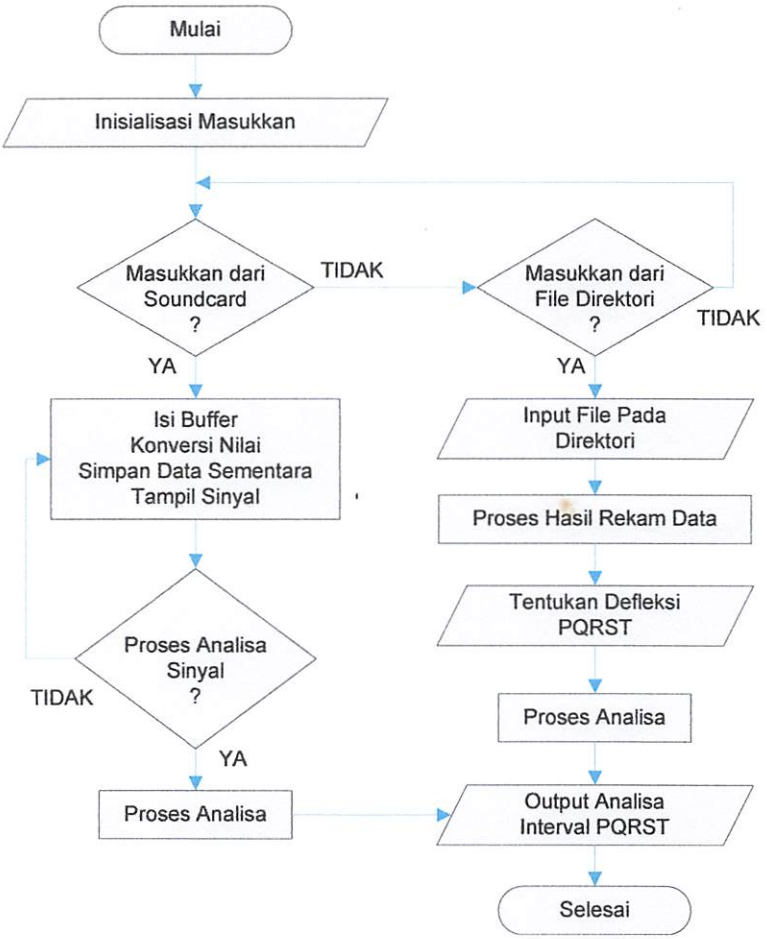
Perancangan sistem diawali dengan merumuskan tuntutan perancangan, kemudian merumuskan konsep dasar solusinya. Solusi yang dirumuskan yaitu merancang aplikasi untuk menampilkan data yang tersimpan didalam buffer kedalam bentuk gelombang dan membuat kode antar muka.

3.1. Tuntutan Perancangan

Tuntutan perancangan adalah bagaimana agar pengguna dapat mengakses akuisisi data berbasis data isyarat elektris jantung dengan mudah dalam penulisan perangkat lunak. Bagi pengguna, cara kerja sistem akuisisi data tidaklah penting, sehingga pengguna tidak perlu mempelajari bagaimana mendapatkan data dari mic-in pada soundcard menjadi data yang diperlukan.

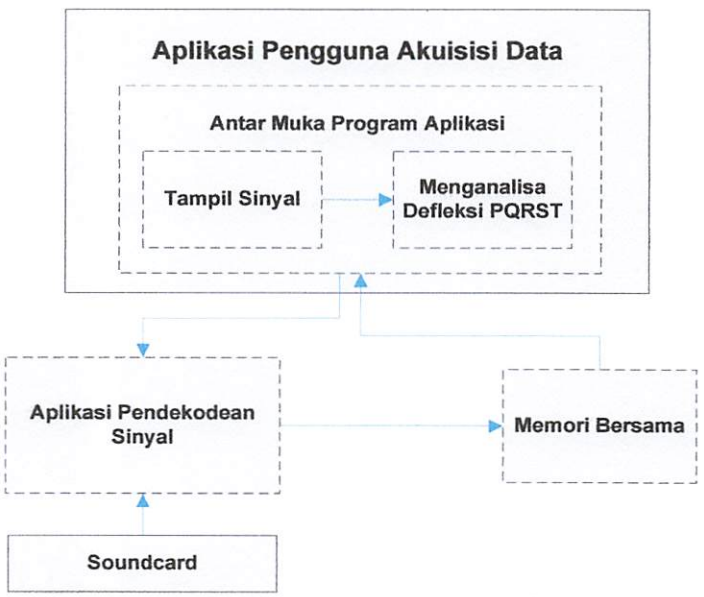
3.2. Konsep Dasar Solusi

Untuk dapat mengakses dengan mudah, maka perlu dirancang syntax program antar muka dalam bahasa pemrograman tingkat tinggi (atau menengah) yang dirancang dengan baik sehingga mudah dipakai oleh pengguna. Kemudian dirancang skrip antarmuka program aplikasi untuk mengakses data yang telah tertampung didalam buffer. Flowchart dari Aplikasi ditampilakan seperti pada Gambar 3.1 dan konsep dasar solusi ditampilkan seperti pada Gambar 3.2 berikut.



Gambar 3.1 Flowchart Aplikasi untuk menganalisa sinyal EKG

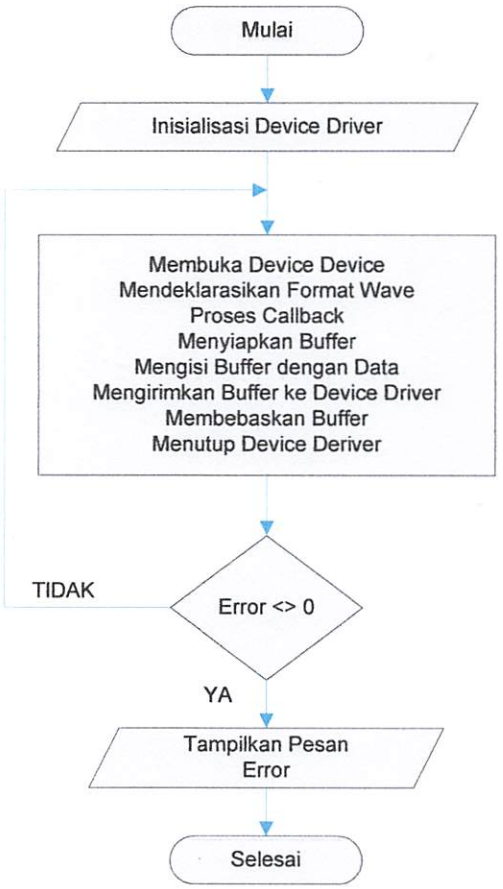
Berbasis akuisisi data isyarat elektris jantung



Gambar 3.2 Konsep Dasar Solusi Rancangan Aplikasi

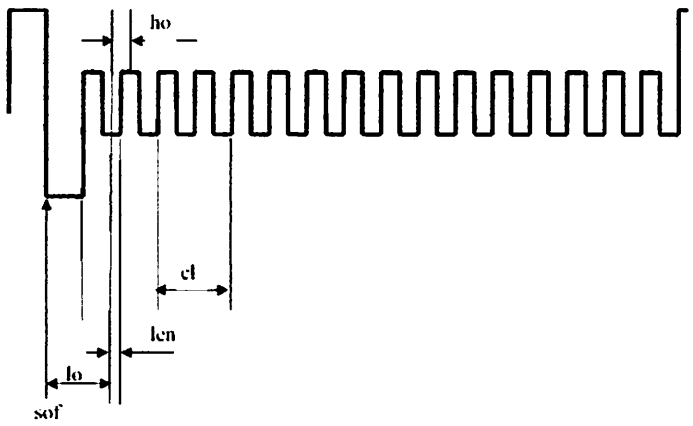
3.3. Perancangan Aplikasi Pendekodean

Pada perancangan aplikasi pendekodean yang dilakukan adalah memeriksa apakah device driver telah terbuka, jika driver telah terbuka maka langkah selanjutnya melakukan proses penyimpanan data ke dalam buffer, untuk lebih jelasnya perancangan aplikasi pendekodean ditampilkan seperti pada Gambar 3.3 berikut.



Gambar 3.3 Flowchart perancangan aplikasi pendekodean

Algoritma decoding didalam aplikasi pendekodean dilakukan oleh thread pemroses sinyal. Bentuk sinyal beserta parameter yang diperlukan dalam pendekodean ditampilkan pada Gambar 3.4.



Gambar 3.4 Bentuk sinyal dan parameter decoding[11]

jika sinyal yang direpresentasikan pada Gambar 3.4, sehingga menjadi data diskrit yang dapat disimpan dalam suatu variable array dengan nama buffer, maka besarnya sinyal pada kanal adalah

$$offset + Gain \times \left(\frac{\sum_{i=0}^{len} buffer[sof+lo+ho+(cl\times n)+i] - \sum_{i=0}^{len} buffer[sof+lo+(cl\times n)+i]}{len} \right)$$

Penjelasan

sof = indeks awal frame

lo = offset(banyaknya data yang harus dilewati) untuk mengambil data fase "rendah" dihitung dari sof

ho = offset(banyaknya data yang harus dilewati) untuk mengambil data fase "tinggi" dihitung dari sof+lo

len = banyaknya titik data yang akan diambil nilai rata-ratanya

cl = banyaknya titik data tiap kanal

banyaknya kanal yang digunakan pada perancangan pembuatan aplikasi = 1 (cl=1).

3.3.1. Membuka Device

Untuk dapat menggunakan sound card didalam melakukan pembacaan, yang perlu dilakukan adalah membuka device untuk proses pembacaan. Fungsi yang diperlukan adalah WaveInOpen().

Fungsi Mendeklarasikan WaveInOpen;

```
function waveInOpen (lphWaveOut : PHWaveOut; uDeviceID: UINT; lpFormat
: PWaveFormatEx; dwCallBack, dwInstance, dwFlags : DWORD) :
MMRESULT; stdcall;
```

Penjelasan Mengenai Fungsi WaveInOpen

a) ***lphWaveOut*** adalah parameter lphWaveOut berfungsi sebagai variable yang akan menerima handle HWAVEOUT yang akan dipergunakan untuk mengakses waveform audio output device.

b) ***UDeviceID*** adalah ID waveform audio output device. Jika tidak mengetahui ID device dapat menggunakan WAVE_MAPPER, bertujuan agar waveInOpen sendiri yang mengisi ID device secara otomatis.

c) ***lpFormat*** adalah berisi alamat ke struktur data WaveFormatEx yang menentukan format wave yang akan di jalankan.

d) ***dwCallback*** adalah berisi alamt fungsi callback, event handle atau handle window yang akan diberi notifikasi ketika terjadi event saat playback.

e) ***dwInstance*** adalah user data yang akan dikirimkan ke callback.

f) ***dwFlags*** adalah memberitahukan tipe callback dan informasi mengenai wave. Dapat diisi salah satu Flag berikut;

- ***CALLBACK_NULL*** Mekanisme callback tidak digunakan
- ***CALLBACK_FUNCTION***

Callback menggunakan fungsi. Jika flag ini digunakan maka, dwCallback harus berisi alamat fungsi callback yang akan dikirimi pesan. Pesannya adalah:

- ✓ ***MM_WIM_OPEN*** Device ditutup dengan waveinClose().
- ✓ ***MM_WIM_CLOSE*** Device ditutup dengan waveinOpen().
- ✓ ***MM_WIM_DATA*** Device selesai mengisi buffer dengan data.

- ***CALLBACK_WINDOW***

Callback menggunakan handle window. Jika flag ini digunakan maka, dwCallback harus berisi handle window.

- ***CALLBACK_THREAD***

Callback menggunakan thread. jika flag ini digunakan maka, dwCallback harus berisi thread ID.

- ***CALLBACK_EVENT***

Callback menggunakan handle event. Jika flag ini digunakan maka, dwCallback harus berisi handle event.

- ***WAVE_FORMAT_QUERY***

Jika flag ini digunakan maka, waveInOpen akan melakukan pengecekan apakah output device sanggup memainkan format wave yang ditentukan.

Pada implementasi Aplikasi menganlisa sinyal EKG yang dibuat dengan menerapkan fungsi yang telah dideklarasikan maka fungsi membuka device ditulis dengan nama fungsi *TWavein.SetupInput:Boolean*;

```
function TWavein.SetupInput:Boolean;
var
        iErr, i : integer;
begin
...
...
iErr := waveInOpen(@Wavein, WAVE_MAPPER,
ptrWaveFmtEx,
formhandle,0,CALLBACK_WINDOW));

if (iErr<>0) then begin
        "Menampilkan Peringatan"
        Result := FALSE;
        exit;
end;
...
...
```

3.3.2. Format Wave

Format wave didefinisikan menggunakan struktur dari TWaveFormatEx yang dideklarasikan sebagai berikut;

Fungsi menentukan format wave:

```
...
PWaveFormatEx  = ^TWaveFormatEx;
TWaveFormatEx = packed record
        wFormatTag     :       word;
        nChannels      :       word;
        nSamplesPerSec :       DWORD;
        nAvgBytesPerSec :      DWORD;
        nBlockAlign    :       Word;
        wBitsPerSample :       Word;
        cbSize         :       Word;
end;
```

Penjelasan dari fungsi format wave berikut;

wFormatTag

Adalah berisi format wave standar yang dimiliki secara default oleh windows. Didalam penggunaan tipe format PCM(Pulse Code Modulation), format wave di setting menjadi WAVE_FORMAT_PCM.

nChannels

Adalah berisi jumlah channel. Channel 1 sebagai Mono dan Channel 2 sebagai Stereo.

nSamplesPerSec

Adalah berisi jumlah sample yang dimainkan per detik dalam satuan Hertz(Hz). Untuk WAVE_FORMAT_PCM, nilai yang paling sering digunakan adalah 8000 Hz, 11025 Hz, 22050 Hz dan 44100 Hz.

nAvgBytesPerSec

Adalah berisi rata-rata laju data transfer yang diperlukan. Untuk WAVE_FORMAT_PCM, nilainya adalah hasil perkalian antara ***nSamplesPerSec*** *dan* ***nBlockAlign***.

nBlockAlign

Adalah satuan terkecil data untuk suatu format wave. Untuk PCM, nBlockAlign adalah ***nChannels * wBitsPerSample*** dibagi ***8.*** Dimana untuk 8 bit = 1 byte.

wBitsPerSample

Adalah jumlah bit untuk tiap sample data. Untuk PCM nilainya harus 8 atau 16. Untuk Mono digunakan nilai 8 dan untuk stereo digunakan nilai 16.

Pada implementasi Aplikasi menganalisa sinyal EKG yang dibuat maka fungsi format wave ditulis dengan nama fungsi *TWavein.AllocWaveFormatEx:Boolean*;

```
function TWavein.AllocWaveForamtEx : Boolean;
begin
new (ptrWavefmtEx)
...
with ptrwavefmtEx^ do
begin
  wFormatTag :=       WAVE_FORMAT_PCM;
  nChannels  :=       1; //untuk Mono
  nSamplesPerSec :=   44100;  //frekuensi bisa dganti
  nBlockAlign :=      1;
  wBitsPerSample :=   8;
  nAvgBytesPerSec :=  nSamplesPerSec
              *(wBitsPerSample div 8) * nChannels;
        cbSize :=     0;
...
```

3.3.3. CallBack

Callback terdiri atas beberapa, yang paling sering digunakan adalah callback berupa fungsi. Callback fungsi menggunakan fungsi dengan format sebagai berikut;

```
procedure BufferDoneCallBack (Handle : HWAVEOUT; uMsg : UINT;
dwInstance : DWORD; dwParam1,dwParam2 : DWORD; stdcall);
```

Penjelasan mengenai fungsi CallBack;

Handle

Adalah handle wave out yang diperoleh melalui pemanggilan waveInOpen().

uMsg

Adalah tipe kejadian yang terjadi yakni WOM_OPEN, WOM_CLOSE dan WOM_DONE. Yang paling sering diperlukan didalam menggunakan uMsg adalah WOM_DONE, pesan ini memberitahukan bahwa device driver telah selesai memproses data.

dwInstance

Adalah user data yang kita kirim pada saat pemanggilan waveInOpen().

dwParam1 dan ***dwParam2***

Adalah sebuah parameter.

Pada implementasi Aplikasi menganalisa sinyal EKG yang dibuat, fungsi callback ditulis kedalam syntax berikut, dengan nama *procedure BufferDoneCallBack()*;

```
procedure BufferDoneCallBack(
        hW              :       HWAVE;
        uMsg            :       DWORD;
        dwInstance      :       DWORD;
        dwParam1        :       DWORD;
        dwParam2        :       DWORD;
        ); stdcall;
var
        BaseRecorder    ;       TWavein;
begin
        BaseRecorder    :=      TWavein(DwInstance);
        try
        ...
        ...
        except ...
end;
```

3.3.4. Menyiapkan Buffer

Pada proses pembacaan terhadap data yang masuk kedalam device perlu disiapkan sebuah buffer yang akan menampung data wave. Berffer yang dibuat harus di daftarkan ke device driver menggunakan *Twavein.initWaveHeaders* sebelum menggunakannya untuk mengirimkan data wave ke device driver.

Fungsi menyiapkan buffer

```
function Twavein.initWaveHeaders (hWaveOut) : HWAVEOUT; pWaveOutHdr
: PWaveHdr; uSize : UINT); MMRESULT; stdCall;
```

Penjelasan dari Fungsi diatas;

hWaveOut adalah handle wave out.

lpWave OutHdr adalah menyimpan informasi mengenai buffer yang dibuat.

uSize adalah ukuran wave header.

Berikut ini adalah deklarasi PWaveHdr

```
Type
        PWaveHdr = ^TWaveHdr
        Wavehdr_tag = record
        lpData          :   PChar;
        dwBufferLength      :   DWORD;
        dwUser              :   DWORD;
        dwFlags             :   DWORD;
        dwLoops             :   DWORD;
        lpNext              :   PWaveHdr;
        reserved        :   DWORD;
end;
TWaveHdr    =       wavehdr tag;
```

Penjelasan dari deklarasi diatas adalah sebagai berikut;

LpData adalah berisi alamat memori buffer.

dwBufferLength adalah menyimpan berapa banyak panjang buffer yang tersedia.

dwBytesRecorded adalah hanya dipergunakan untuk proses recording.

dwUser adalah dapat digunakan untuk menyimpan user data.

dwFlags adalah berisi falg mengenai buffer.

Pada implementasi Aplikasi menganalisa sinyal EKG yang dibuat, proses menyiapkan buffer ditulis kedalam syntax berikut, dengan nama *function TWavein.InitWaveHeaders:Boolean*;

```
function Twavein.initWaveHeaders : Boolean;
var
        i:integer;
begin
        WaveBufSize := fBufferSize – (fBufferSize mod
ptrwavefmtex.nBlockAlign);
For I := 0 to TotalBuffers-1 do
with pWaveHeader[i]^ do
Begin
        ...
        lpData  :=      pWaveBuffer[i];
        dwBufferLength := WaveBufSize;
        dwBytesRecorded := 0;
        dwUser:=        0;
        dwFlags:=       0;
        dwLoops:=       0;
        lpNext  :=      Nil;
        reserved:=      0;
end;
        result  :=      TRUE;
...
```

3.3.5. Mengisi Buffer dengan Data

Didalam pengisian data kedalam buffer adalah jika jumlah data yang di copy kedalam buffer lebih kecil dari ukuran buffer, *wBufferLength* perlu dilakukan pengaturan sama dengan panjang data yang berada didalam buffer.

Pada implementasi Aplikasi menganalisa sinyal EKG yang dibuat, proses pengisian data ke buffer ditulis kedalam syntax berikut, dengan nama *function TWavein.StartInput:Boolean*;

```
function Twavein.StartInput : Boolean;
begin
        result :=false;
        waveInStart(Wavein);
end;
```

3.3.6. Mengirimkan Buffer ke Device Driver

Setelah buffer terisi dengan data, maka proses selanjutnya yang harus dilakukan adalah mengubah data yang berada kedalam buffer menjadi sebuah tampilan visual grafis yang berbentuk sinyal gelombang sesuai dengan data yang berada pada buffer. Pemanggilan data dapat dilakukan dengan menggunakan fungsi *TWavein.AddNextBuffer : Boolean*

Fungsi mengirimkan buffer ke device driver

```
function TWavein.AddNextBuffer (hWaveOut : HWAVEOUT; lpWaveOutHdr
: PWaveHdr; uSize : UINT) : MMRESULT; stdcall;
```

Penjelasan untuk fungsi diatas adalah sebagi berikut;

lpWaveOutHdr adalah wave header yang sebelumnya telah di lakukan prepare.

uSize adalah ukuran data wave header.

Pada implementasi Aplikasi menganalisa sinyal EKG yang dibuat, maka proses pengiriman buffer ke device driver ditulis kedalam syntax berikut, dengan nama *function TWavein.AddNextBuffer:Boolean*;

```
function Twavein.AddNextBuffer: Boolean;
var
        iErr:integer;
begin
result := True;
        ...
 iErr := waveinAddBUffer(Wavein, pwaveheader[bufindex],
sizeof(TWAVEHDR));
 if (iErr <>0) then begin
    Stop Input
  end;
  Bufindex := (bufindex + 1)mod TotalBuffers;
  QueuedBuffers := QueuedBuffers + 1;
Result   := True;
end;
```

3.3.7. Membebaskan Buffer

Setelah proses pengambilan data wave terselesaikan, sebelum membebaskan buffer pastikan proses pemanggilan *FreePCMBuffers* dilakukan untuk memberitahukan bahwa buffer akan segera dibebaskan. Dengan pemanggilan *FreeWaveHeader*,device driver akan diinformasikan agar tidak menggunakan buffer yang mengalami proses pembebasan. Setelah proses unprepared, buffer dapat dibebaskan dengan aman.

Pada implementasi Aplikasi menganalisa sinyal EKG yang dibuat proses pembebasan Buffer dapat dilihat pada syntax berikut;

Syntax untuk membebaskan buffer

```
procedure Twavein.FreePCMBuffers;
var     i : integer;
begin
        for i := 0 to TotalBuffers-1 do
                Freemem(pwavebuffer[i]);
end;
```

Syntax konfirmasi device driver

```
procedure TWavein.FreeWaveHeader;
var     i : integer;
begin
        for i := 0 to TotalBuffers-1 do begin
        dispose (pwaveheader[i]);
                pwaveheader[i] := nil;
        end;
end;
```

3.3.8. Menutup Device

Untuk Menutup Device setelah melakukan proses pengambilan data dari device driver maka diperlukan sebuah fungsi untuk mengembalikan proses ke awal dengan menutup waveform audio output device.

Fungsi Menutup Device

```
function waveinUnprepareHeader(hWaveOut : HWAVEOUT) : MMRESULT;
stdcall;
```

Pada implementasi Aplikasi menganalisa sinyal EKG yang dibuat, maka proses menutup device ditulis dengan nama *procedure TWavein.CloseWaveDevice;*

```
procedure TWavein.CloseWaveDevice;
Var
        iErr : integer;
begin
        ...
        ...
        iErr :=waveinUnprepareHeader (Wavein,
pWaveHeader[i],sizeof(TWAVEHDR));
        if (iErr <> 0) then begin
        case ierr of
                MMSYSERR_INVALHANDLE;
                MMSYSERR_NODRIVER;
                MMSYSERR_NOMEM;
        end;
...
```

Penjelasan dari pesan kesalahan diatas;

MMSYSERR_INVALHANDLE	Pesan Penanganan kesalahan saat pemilihan device.
MMSYSERR_NODRIVER	Pesan Jika tidak ada device driver.
MMSYSERR_NOMEM	Pesan Penanganan Alokasi Memori.

3.3.9. Mendapatkan Status Pesan Kesalahan

Untuk mendapatkan pesan kesalahan dari kode kesalahan yang dikembalikan fungsi waveinOpen, kita menggunakan *function GetErrorText()*. Fungsi Mendapatkan status pesan kesalahan.

```
function GetErrorText(mmrError : MMRESULT; lpText : PChar; uSize : UINT)
: MMRESULT ; stdcall;
```

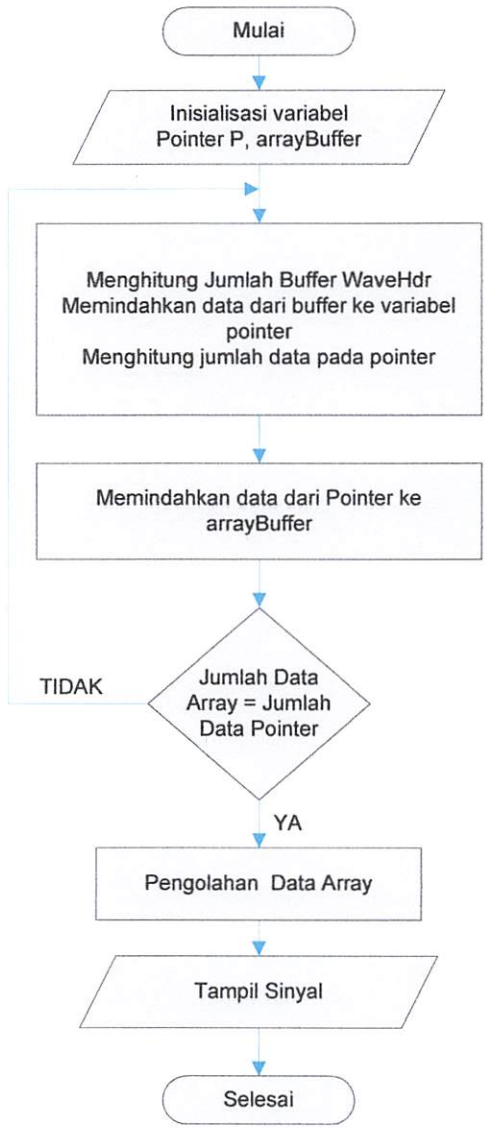
Pada implementasi Aplikasi menganalisa sinyal EKG yang dibuat, maka proses mendapatkan status pesan kesalahan ditulis kedalam syntax berikut, dengan nama *function GetErrorText()*.

```
function GetErrorText (iErr : integer):string;
var
        PlayInErroMsg : Array [0..255] of Char;
begin
        waveinGetErrorText(iErr, PlayInErrorMsg,255)
        ...
end;
```

3.4. Perancangan Kode Antarmuka Dalam Delphi

3.4.1. Menampilkan Sinyal Berdasarkan Data pada Buffer

Dalam Pembuatan tampilan visualisasi sinyal yang nilainya didapatkan dari buffer, yang terlebih dahulu dilakukan yaitu membuat visualisasi kertas elektrokardiogram. Didalam membuat visualisasi kertas elektrokardiogram menggunakan fungsi *Graphics* dimana fungsi ini ada pada **Windows API.** Flowchart dari perancangan menampilkan sinyal berdasarkan data pada buffer ditampilkan seperti pada Gambar 3.5.



Gambar 3.5 Flowchart Perancangan Menampilkan Sinyal berdasarkan Data pada Buffer

Pada implementasi Aplikasi menganalisa sinyal EKG yang dilakukan, membutuhkan sebuah procedure untuk membuat garis Vertikal dan Horizontal. Procedure tersebut diberi nama *DrawScale*. Berikut syntaxnya;

```
procedure Tfrm_EKGMonitor.DrawScale;
...
...
   with imagebackground do
   begin

   //Membuat garis Vertikal
   For loop :=1 to 8 do begin
   Canvas.MoveTo(0,Loop*tinggiGaris);
   Canvas.LineTo(LebarImage,Loop*tinggiGaris);
   end;
   ...
   ...
   //Membuat garis Horizontal
   For loop :=1 to 8 do begin
   Canvas.MoveTo(round(Loop*lebarGaris),0);
   Canvas.LineTo(round(Loop*LebarGaris),LebarImage);
   end;
   ...
   end;
....
```

Untuk melakukan pembuatan line pada sinyal EKG dari data yang tersimpan didalam buffer diperlukan suatu procedure yang melakukan penggambaran berupa line pada image. Berikut syntax nya;

```
procedure Tfrm_EKGMonitor.DataBeam(const DataBuffer:array
of TPoint);
Begin
        ...
        imgscreenLine.Canvas.Pen.Color := clime;
        imgscreenLine.Canvas.Polyline(DataBuffer);
        ...
        ...
end;
```

Pengambilan data pada buffer terlihat pada syntax berikut ini. Pada implementasinya dibuat sebuah procedure yang diberi nama procedure *Tfrm_UtamaEKG.Bufferfull(var Message:TMessage)*;

```
procedure Tfrm_UtamaEKG.Bufferfull(var
Message:TMessage);
var
 i:Integer;
 p:PBufArray;
 pstart:Integer;
 frametoScan : Integer;

  function ChScale(x:integer):integer;
  var
   N:Integer;
  begin
   N := Offsety-x;

   if ChGain < 3 then
    N := N shr (3-ChGain)
   else if ChGain > 3 then
    N := N shl (ChGain-3);

   Result := cy + N;
  end;

var
  BeamI : Integer;
begin {Memulai Procedure Buffer}

 //=======================================
 //     Membuat Panel LED BErkedip
 time2:=now;
 if (time2-time1) > 1/secsperday then
 begin
  Application.ProcessMessages;
  time1 := time2;
if not btnRun.Down or (P_LED.Color = clLime) then
   P_LED.Color := clNavy
  else
   P_LED.Color := clLime ;
 end;
 //=======================================
 frametoScan := nbrframes - 1;
 if Assigned(WaveIN) and (WaveIN.RecordActive) then
  with WaveIN do
  begin
  {Lparam pada isi pesan seubuat ponter ke sebuah structur
   WaveHDR. yang pertama dimasukan ke ponter buffer}

   p := pointer(pwavehdr(Message.LParam)^.lpData);
   x:=0;
```

```
if calibrate then {reset nilai NOL}
   begin
    Offsety := 0;
    for i:= 0 to bufSize-1 do
     inc(offsety,p^[i]);

    Offsety := Offsety div bufSize;
    calibrate := False;
   end;

   trigsign := 0;
   //=================================

   buffer1found :=True;
   pstart       := Width;

   x    := 0;//biar enak harusnya pake posisi trackbar
   BeamI := -1;
   i    := 0;

   triggered := trigsign=0;
   //singel Trace

   SetLength(BeamEKG,round(PlotPtsPerFrame)-3);
   ValyR := BeamEKG[0].Y; ValyS  :=  BeamEKG[0].Y;

while i <= PlotPtsPerFrame-4 do
    begin
     inc(i);
     inc(BeamI);
     BeamEKG[BeamI].X:=x;
     BeamEKG[BeamI].Y:=ChScale(p^[i]) +
trOfsCh.Position;
...
...
...
{selalu memasukan statemen selanjutnya - diperlukan
 Twavein untuk menambah buffer}
   inc(ProcessedBuffers);
  AddNextBuffer;
//Jika Button CapFrame di Tekan
  //Twavein akan dihentikan dan MengCapture Frame
  If singleFrame and triggered and (frametoScan >= 0) then
  begin

SetLength(savedFrameData, PlotPtsPerFrame);

for i:=pstart to pstart + PlotPtsPerFrame-1 do
     savedFrameData[i-pstart]:=p^[i]-128;
```

```
  Stop;
   statusText.Caption := 'Status Dihentikan';
   btnCapFrame.Down:=False;
      ...
 end;
end;
```

Berikut syntax yang digunakan pada implementasi untuk mendeklarasikan format wave yang digunakan.

```
procedure Tfrm_UtamaEKG.setup;
begin
 errcount  := 0;
 x         := 0;

 if not Assigned(WaveIN) then
  WaveIN :=
TWaveIn.Create(Handle,{org}bufSize,numbuffers);

  with WaveIN, ptrWaveFmtEx^ do
  begin
   wFormatTag := WAVE_FORMAT_PCM;

   //mengaktifkan chanel 1
   nChannels := 1; //1 = Mono ; 2 = Stereo
   nSamplesPerSec := StrToInt(txt_SP_rate.Text);

ShowScaleValue;
...
  wBitsPerSample := 8;

  //membandingkan data yang ada
  nAvgBytesPerSec := nSamplesPerSec*(wBitsPerSample
div 8)* nChannels;
  if SetupInput then
   statusText.Caption:='BERHENTI'
  else
   statusText.Caption:='ERROR - COBA LAGI';

SetLength(frameSaveBuf,2*bufSize);
cy := frmEKGMonitoring.ImgLine.Height div 2;

end;
 SetLength(savedFrameData,0);
end:
```

Syntax untuk mengaktifkan pengambilan data

```
procedure Tfrm_UtamaEKG.Start;
begin
...
 If WaveIN.RecordActive then
 begin
  WaveIN.StopInput;
  Application.ProcessMessages;
 end;
 WaveIN.StartInput;
 statusText.Caption:='Status Dijalankan';
end;
```

Syntax untuk menonaktifkan pengambilan data

```
procedure Tfrm_UtamaEKG.Stop;
begin
 If Assigned(WaveIN) then
  WaveIN.StopInput;

 statusText.Caption:='Status Dihentikan';
end;
```

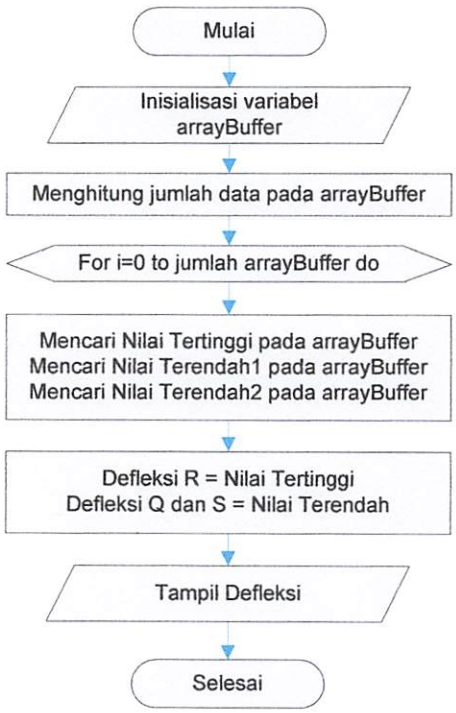
3.4.2. Perancangan Aplikasi Menentukan Defleksi

Pada perancangan aplikasi untuk menentukan Defleksi gelombang yang harus dilakukan adalah memeriksa data yang tersimpan didalam buffer kemudian data tersebut dipindahkan ke dalam sebuah variable sementara kemudian dari data tersebut di cari nilai tertinggi untuk menyatakan Defleksi R dan nilai terendah untuk menyatakan Defleksi S. Defleksi juga bisa dinyatakan secara manual dengan menentukan posisi pada gelombang, yang mana dari posisi tersebut dapat diberikan keterangan bahwa Defleksi pada posisi tersebut merupakan salah satu dari Defleksi P, Kompleks QRS dan T.

Langkah Menentukan Defleksi gelombang ada 2 cara, yaitu;

1. **Menentukan secara Otomatis**

Defleksi yang mudah dicari adalah Q, R dan S, dikarenakan defleksi R adalah nilai yang tertinggi, defleksi Q adalah nilai terendah pertama dan defleksi S adalah nilai terendah kedua. Flowchart ditampilkan seperti pada Gambar 3.6.



Gambar 3.6 Flowchart Menentukan Defleksi Otomatis

Dari data yang ada didalam buffer (bertipekan array) untuk mencari defleksi R dengan cara membandingkan nilai terbesar dari nilai array ke 1 sampai nilai array ke- n, dan mencari defleksi Q dan S dengan cara membandingkan nilai terkecil dari nilai array ke 1 sampai ke- n yang ada pada buffer. Pada implementasinya pencarian pada data dituliskan ke dalam syntax berikut;

```
...
...

 SetLength(BeamEKG,round(PlotPtsPerFrame)-3);
 ValyR := BeamEKG[0].Y; ValyS  :=  BeamEKG[0].Y;

 while i <= PlotPtsPerFrame-4 do
    begin
      inc(i);
      inc(BeamI);
      BeamEKG[BeamI].X:=x;
      BeamEKG[BeamI].Y:=ChScale(p^[i]) +
trOfsCh.Position;
      if ValyR > BeamEKG[BeamI].Y then begin
        ValyR :=  BeamEKG[BeamI].Y;
        ValxR :=  BeamEKG[BeamI].X;
        ValyQ :=  BeamEKG[BeamI - 10].Y;
        ValxQ :=  BeamEKG[BeamI - 10].X;
        ValyS :=  BeamEKG[BeamI + 7].Y;
        ValxS :=  BeamEKG[BeamI + 7].X;
      end;

      {If ValyS < BeamEKG[BeamI].Y then begin
        ValyS :=  BeamEKG[BeamI].Y;
        ValxS :=  BeamEKG[BeamI].X;
      end;}

      Inc(x,xinc);
    end;

    ...
...
    // menggambar
    //   unit frm EKG_Monitoring
    //=============================================
      frmEKGMonitoring.DataBeam(BeamEKG);
      frmEKGMonitoring.FindR(ValxR,ValyR);
      frmEKGMonitoring.FindS(ValxS,ValyS);
      frmEKGMonitoring.FindQ(ValxQ,ValyQ);
    //=============================================
```

Didalam menentukan defleksi Q, R dan S, pada Aplikasi diperlukan beberapa fungsi untuk menampilkan defleksi dari data yang diperoleh. Beberapa fungsi tersebut dapat terlihat pada syntax berikut;

Fungsi untuk menampilkan defleksi R

```
function Tfrm_EKGMonitor.FindR(X, Y :integer): integer;
begin
  ...
  ImgLine.Canvas.TextOut(X,Y,'R');
end;
```

Fungsi untuk menampilkan deflkesi S

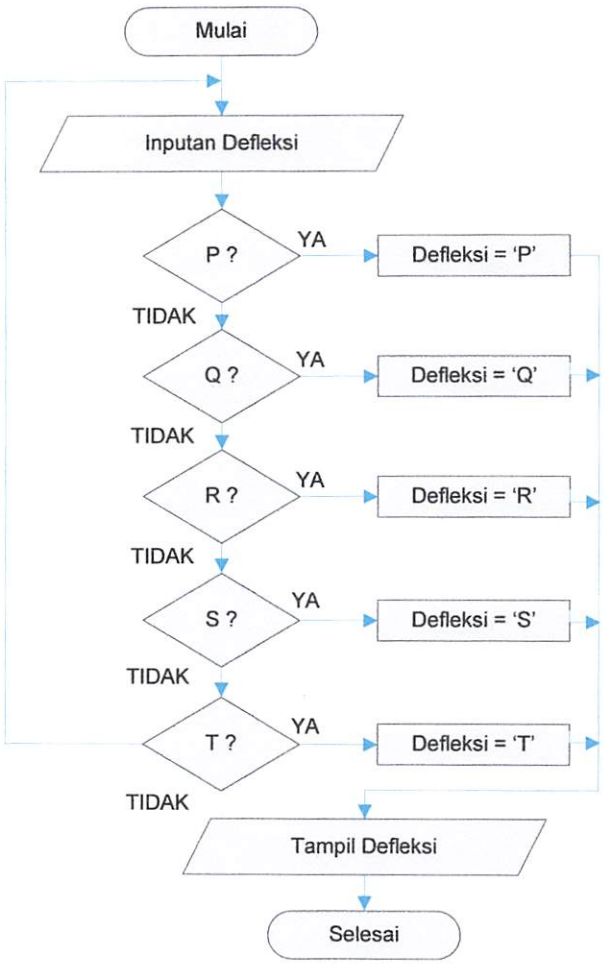
```
function Tfrm_EKGMonitor.FindS(X, Y: integer): integer;
begin
  ...
  ImgLine.Canvas.TextOut(X,Y,'S');
end;
```

Fungsi untuk menampilkan defleksi Q

```
function Tfrm_EKGMonitor.FindQ(X, Y: integer): integer;
begin
  ...
  ImgLine.Canvas.TextOut(X,Y,'Q');
end;
```

2. Menentukan secara Manual

Defleksi yang ditentukan secara manual hanya dengan menentukan posisi yang dijadikan sebagai salah satu defleksi pada visualisasi elektrokardiogram. Untuk menentukan secara manual flowchart ditampilkan seperti pada Gambar 3.7.



Gambar 3.7 Flowchart Menentukan Defleksi Manual

Berikut beberapa procedure yang digunakan untuk menentukan masing masing defleksi.

Syntax untuk menentukan defleksi P.

```
...
If <kondisi 1> then
  ImgLine.Canvas.TextOut(ValX1,ValY1,'P');
  Glmb1 :='P';
end  else if <kondisi 2> then begin
  ImgLine.Canvas.TextOut(ValX2,ValY2,'P');
  Glmb2 :='P';
...
```

Syntax untuk menentukan defleksi Q.

```
...
If <kondisi 1> then
  ImgLine.Canvas.TextOut(ValX1,ValY1,'Q');
  Glmb1 :='Q';
end  else if <kondisi 2> then begin
  ImgLine.Canvas.TextOut(ValX2,ValY2,'Q');
  Glmb2 :='Q';
...
```

Syntax untuk menentukan defleksi R.

```
...
If <kondisi 1> then
  ImgLine.Canvas.TextOut(ValX1,ValY1,'R');
  Glmb1 :='R';
end  else if <kondisi 2> then begin
  ImgLine.Canvas.TextOut(ValX2,ValY2,'R');
  Glmb2 :='R';
...
```

Syntax untuk menentukan defleksi S.

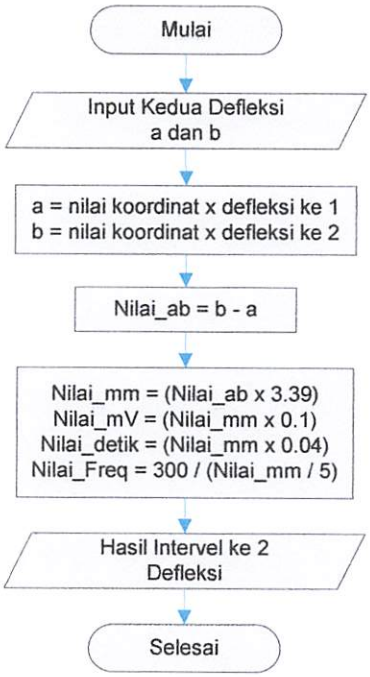
```
...
If <kondisi 1> then
  ImgLine.Canvas.TextOut(ValX1,ValY1,'S');
  Glmb1 :='S';
end  else if <kondisi 2> then begin
  ImgLine.Canvas.TextOut(ValX2,ValY2,'S');
  Glmb2 :='S';
```

Syntax untuk menentukan defleksi T.

```
...
If <kondisi 1> then
  ImgLine.Canvas.TextOut(ValX1,ValY1,'T');
  Glmb1 :='T';
end  else if <kondisi 2> then begin
  ImgLine.Canvas.TextOut(ValX2,ValY2,'T');
   Glmb2 :='T';
...
```

3.4.3. Perancangan Aplikasi Menentukan Interval antar Defleksi

Pada perancangan aplikasi menentukan interval antar defleksi gelombang P kompleks QRS dan T, yang diperlukan adalah interval(jarak) dari satu Defleksi ke Defleksi lainnya. Proses awal yang harus dilakukan adalah menentukan ukuran pixel pada gambar (disini gelombang ditampilkan dalam bentuk gambar), setelah itu menentukan berapa nilai koordinat(x,y) untuk defleksi yang pertama dan defleksi yang kedua. Nilai koordinat yang didapat berukuran pixel yang harus diubah kedalam bentuk millimeter (mm). Flowchart Menentukan Interval antar defleksi ditampilkan seperti pada Gambar 3.8 berikut.



Gambar 3.8 Flowchart Menentukan Interval antar Defleksi

Rumus:

1 Pixel = 3,39 Milimeter

Untuk mengetahui jumlah kotak pada visualisasi kertas EKG, berdasarkan jarak defleksi R ke 1 dan defleksi R ke 2 menggunakan *rumus*;

JmlKotak = (Nilai koordinat X R ke 1) – (Nilai koordinat X R ke 2)

Dikarenakan hasil yang didapat pada variable *JmlKotak* masih dalam bentuk pixel maka hasil perlu di rubah kedalam satuan millimeter.

Ukuran-Ukuran pada kertas EKG Pada perekaman EKG standar telah ditetapkan yaitu :

1. Kecepatan rekaman 25 mm/detik (***25 kotak kecil***).
2. Kekuatan voltage 10 mm = 1 millivolt (***10 kotak kecil***).

Jadi ini berarti ukuran dikertas EKG adalah

1. Pada garis horisontal
   - ✓ Tiap satu kotak kecil = **1 mm** = 1/25 detik = **0,04 detik.**
   - ✓ Tiap satu kotak sedang = **5 mm** = 5/25 detik = **0,20 detik.**
   - ✓ Tiap satu kotak besar = **25 mm** = 25/25 detik = **1 ,00 detik.**
2. Pada garis vertikal
   - ✓ 1 kotak kecil = **1 mm =0.1 mv.**
   - ✓ 1 kotak sedang = **5 mm = 0,5 mv.**
   - ✓ 2 kotak sedang = **10 mm= 1 milivolt (mv).**

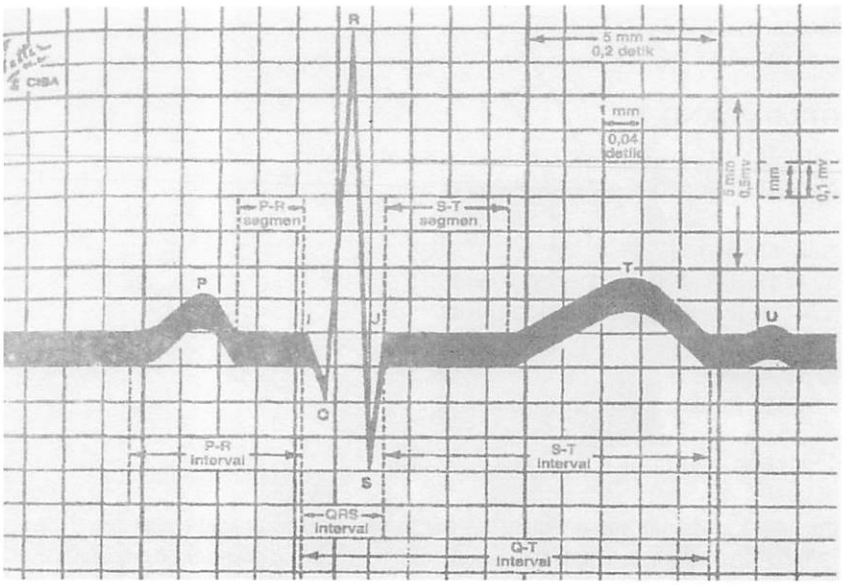
Cara Menghitung EKG

Kecepatan EKG adalah 25 mm / detik. Satu menit = 60 detik, maka kecepatan EKG dalam 1 menit yaitu **60 x 25 = 1500 mm.**

Satu (1) kotak kecil panjangnya = **1 mm.**

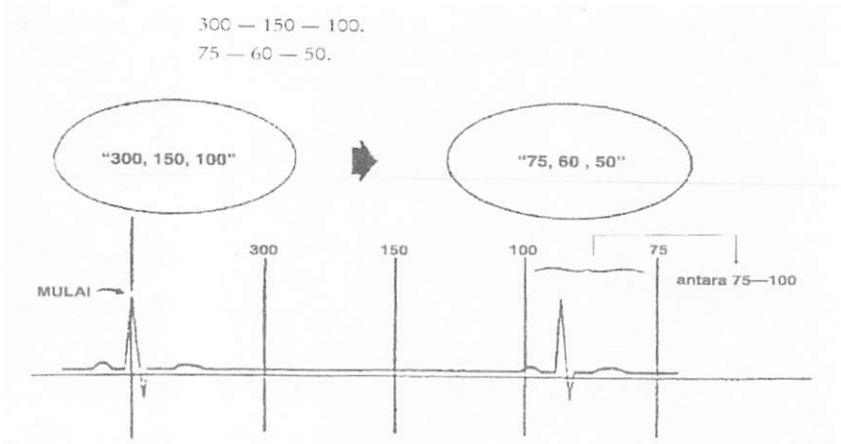
Satu (1) kotak sedang (5 kotak kecil) : **1500 / 5 = 300 mm.**

EKG normal ditampilan seperti pada Gambar 3.9 berikut.



Gambar 3.9 EKG Normal[1]

Jarak Frekuensi Jantung ditampilkan seperti pada Gambar 3.10 berikut.



Gambar 3.10 Jarak Frekuensi Jantung[1]

Dari jarak Frekuensi yang tertera pada gambar diatas, Frekuensi jantung kemudian ditentukan dengan rumus pada Tabel 3.1.

Tabel 3.1 Rumus Frekuensi Jantung

No	Jarak / Kotak	Ukuran x/Menit
1	1 Kotak Sedang	300
2	2 Kotak Sedang	150
3	3 Kotak Sedang	100
4	4 Kotak Sedang	75
5	5 Kotak Sedang	60
6	6 Kotak Sedang	50

Batas Normal yang digunakan didalam menentukan Interval dari sebuah ekg normal dijelaskan sebagai berikut;

a) **Interval PR** = 0,12 – 0,20 detik
b) **Interval QRS** = < 0,12 detik
c) **Interval QT**
Laki-Laki = 0,42 detik
Wanita = 0,43 detik
d) **Segment ST** = < 0,08 detik

Frekuensi jantung dapat ditentukan secara cepat dengan memperhatikan interval RR.

Frekuensi jantung **Normal** adalah 60 – 100 x/menit
Sinus Takikardia > 100 x/menit
Sinus Bradikardia < 60 x/menit
Takikardia Abnormal 140 – 250 x/menit
Flutter 250 – 350 x/menit
Fibrilasi > 350 x/menit

Pada aplikasi yang dibuat, proses menentukan interval antara defleksi dibuat suatu prosedur yang dapat menghitung interval setiap defleksi kedalam satuan milivolt(mV), millimeter (mm) dan detik. Berikut cuplikan syntaxnya;

```
//membuat variabel
 G_TmpilX, NilaiMM  : double;
  KonWaktu, KonVolt  : Double;
  NilaiFreq : Double;
  ketStts   : String; {keterangan Umur}
  ketJntung : String;
  SAnormal  : Double;
 begin
  { Rumus Pixel Ke Milimeter
    ==========================
    ||1 Pixel = 3,39 Milimeter ||
    ========================== }
//Perhitungan Antara Defleksi satu ke Defleksi berikutnya.
satuannya masih dalam pixel
  if ValX1 > ValX2 then
    GarisMM :=  (ValX1 - ValX2)
  else
    GarisMM :=  (ValX2 - ValX1);
  //G_TmpilX := round(GarisMM / 2) + ValX1;

// mengubah dari pixel ke milimeter
  NilaiMM   := GarisMM / 3.39;

//mengubah dari millimeter ke miliVolt
  KonVolt   := NilaiMM * 0.1;

 //mengubah dari millimeter ke detik
  KonWaktu  := NilaiMM * 0.04;
   NilaiFreq := 300 / ((NilaiMM / 5));
    SAnormal  :=  NilaiFreq;
      if ((SAnormal >= 70) and (SAnormal <= 80)) then
       NORMAL
      else
       TDK NORMAL;
...
...
```

Pada perancangan program yang ditampilkan masih berupa potongan-potongan progam secara umum, lebih detailnya pada lampiran yang terlampir.

BAB IV

IMPLEMENTASI DAN PENGUJIAN SISTEM

4.1. Implementasi Sistem

Implementasi dilakukan dengan menerapkan hasil desain yang telah dibuat kedalam bahasa pemrograman (*Coding*) Borland Delphi7, sehingga prosedur-prosedur yang telah dibuat dapat dimengerti oleh mesin sehingga menghasilkan keluaran seperti yang diharapkan. Tabel 4.1 adalah spesifikasi perlengkapan yang digunakan dalam implementasi system.

Tabel 4.1 Spesifikasi Perlengkapan implementasi.

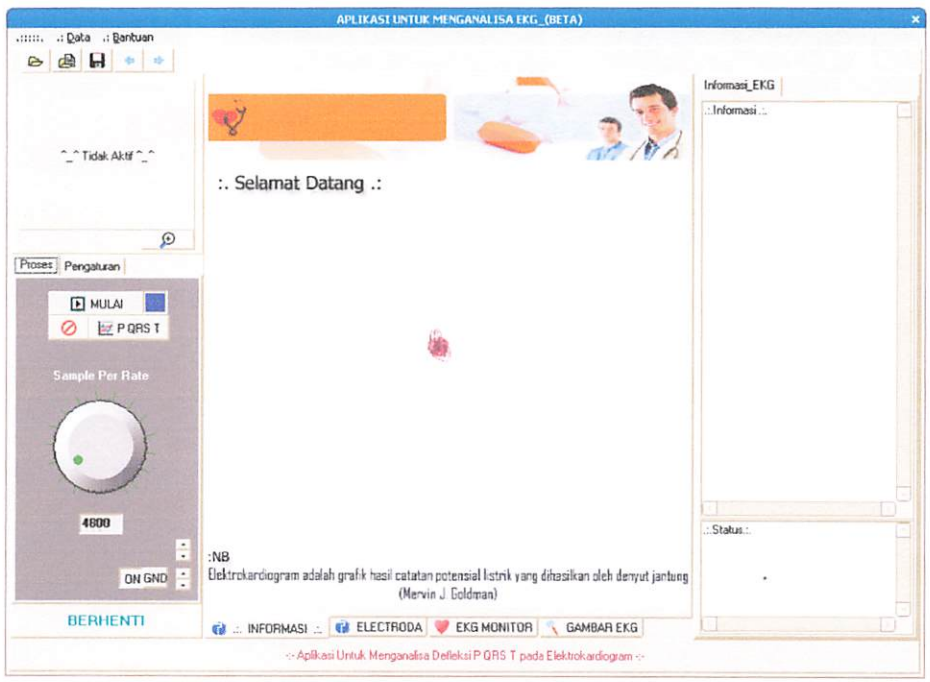
No	Perlengkapan	Keterangan
1	Software	
	Siste Operasi	Windows XP Service Pack 2
	Bahasa Pemrograman	Borland Delphi 7
2	Hardware	
	Elektrokardiogram	Electrocardiogram Heart Monitor Kit, Ramsey Electronics
3	Notebook	
	Processor	Intel Core i5-450 M
	Memori	2 GB DDR3
	Harddisk	500 GB

Aplikasi menganalisa sinyal ekg berbasis akuisisi data isyarat elektris jantung merupakan hasil implementasi yang telah dilakukan, berikut ini penjelasan bagian-bagian dari aplikasi menganalisa sinyal ekg berbasis akuisisi data isyarat elektris jantung tersebut :

4.1.1. Tampilan Aplikasi Menganalisa Sinyal EKG

4.1.1.1. Hasil Implementasi Form Awal

Berdasarkan perancangan pada pembuatan aplikasi yang sebelumnya telah direncanakan, maka aplikasi untuk menganalisa sinyal EKG berbasis akuisisi data isyarat elektris jantung diimplementasikan seperti pada Gambar 4.1 berikut ini;

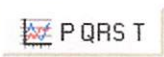
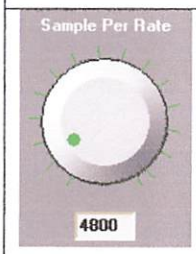


Gambar 4.1 Hasil implementasi Aplikasi Menganalisa sinyal EKG berbasis akuisisi data isyarat elektris jantung

Penjelasan Menu yang terdapat pada Tampilan Awal dijelasakan pada Tabel 4.2 berikut ini.

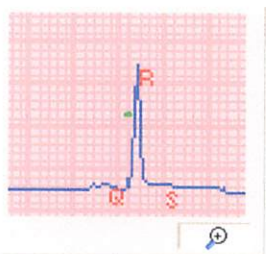
Tabel 4.2 Tools pada Aplikasi

Tools	Keterangan
	Berfungsi menampilkan jendela open dialog gambar yang telah tersimpan yang kemudian

	ditampilkan pada "*tab GAMBAR EKG*".
	Tools yang berfungsi menampilkan jendela open dialog gambar untuk memilih gambar rekaman yang kemudian melakukan proses diagnosa.
	Tools yang berfungsi menyimpan hasil rekaman ke dalam bentuk format (.bmp).
	Tools yang berfungsi menampilkan tab sebelumnya.
	Tools yang berfungsi menampilkan tab berikutnya.
	Tools yang berfungsi untuk memulai pembacaan data waveform.
	Tools yang berfungsi untuk menghentikan pembacaan data waveform.
	Tools yang berfungsi untuk menentukan defleksi terhadap ekg secara otomatis.
	Tools yang berfungsi untuk menentukan jumlah sample per rate yang dibutuhkan didalam melakukan penyimpanan banyaknya sample data dalam byte.

Untuk mempermudah didalam menganalisa terdapat tools Zoom Defleksi yang memudahkan didalam menentukan titik defleksi pertama dengan defleksi kedua.

(a)



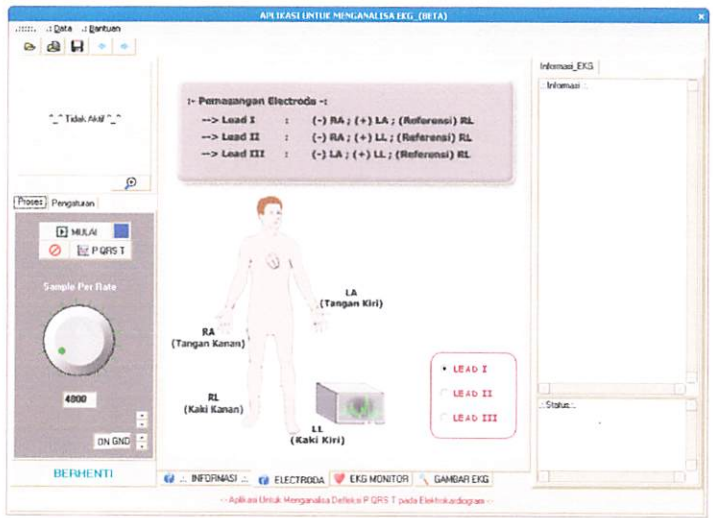
(b)

Gambar 4.2 (a) Hasil Implementasi Zoom Defleksi tidak aktif,
(b) Zoom Defleksi Aktif

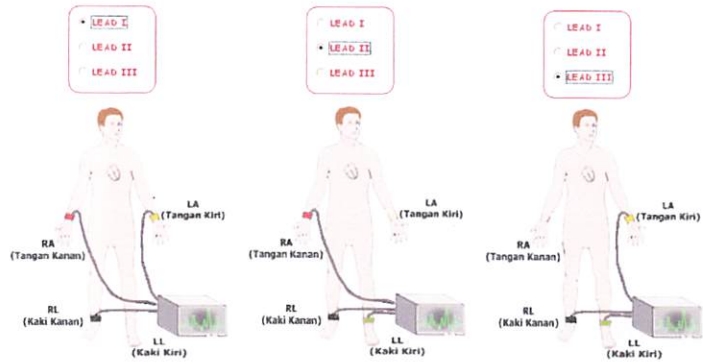
Tekan Key pada keyboard (F5) untuk mengaktifkan tools dan (F6) untuk menonaktifkan tools.

4.1.1.2. Hasil Implementasi Form Pemasangan Elektroda

Pada tab electroda, menerangkan bagaimana cara melakukan pemasangan electroda untuk lead I, II dan III pada permukaan tubuh. Pada aplikasi untuk menganalisa sinyal EKG berbasis akuisisi data isyarat elektris jantung, pemasangan elektroda ditampilkan dalam bentuk visualisasi pasien dengan penempatan elektroda Gambar 4.4, berdasarkan sandapan bipolar.



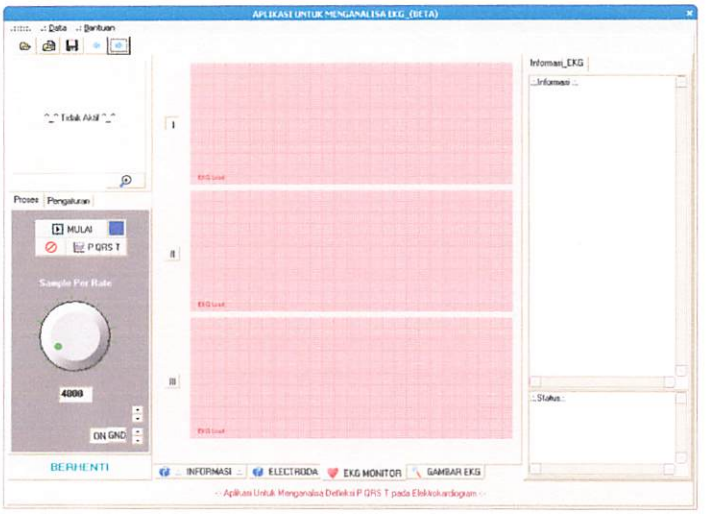
Gambar 4.3 Hasil Implementasi pada tab Elektroda



Gambar 4.4 Pemasangan Electroda pada permukaan tubuh

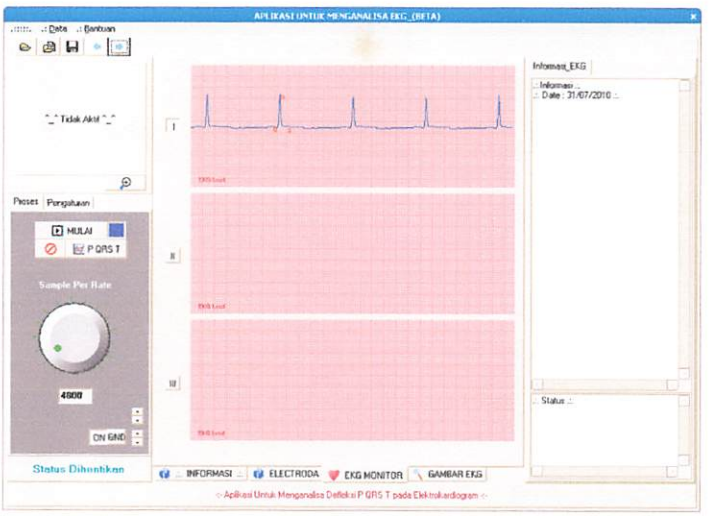
4.1.1.3. Hasil Implementasi Form Pengambilan Data

Berdasarkan perancangan pada aplikasi yang sebelumnya di rencanakan, maka hasil implementasi dari form pengambilan data dapat terlihat pada Gambar 4.5 berikut.



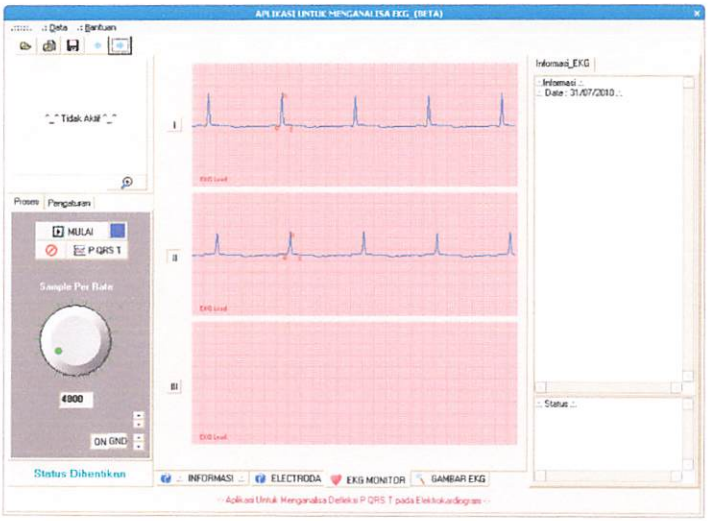
Gambar 4.5 Hasil Implementasi Pengambilan data

Pengambilan data rekaman ekg pasien untuk lead I meletakkan posisi elektroda (-) RA, (+) LA dan (Referensi) RL, Gambar 4.6 hasil yang diperoleh dari pengambilan data Lead I.



Gambar 4.6 Pengambilan data pada Lead I

Pengambilan data rekaman ekg pasien untuk lead II meletakkan posisi elektroda (-) RA, (+) LL dan (Referensi) RL, Gambar 4.7 hasil yang diperoleh dari pengambilan data Lead II.



Gambar 4.7 Pengambilan data pada Lead II

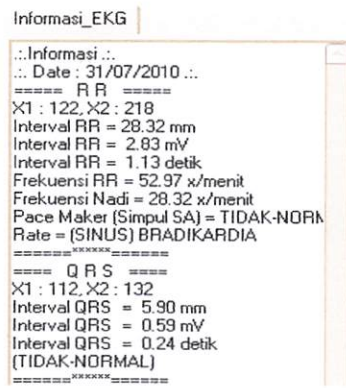
Pengambilan data rekaman ekg pasien untuk lead III meletakkan posisi elektroda (-) LA, (+) LL dan (Referensi) RL, Gambar 4.8 hasil yang diperoleh dari pengambilan data Lead III.

Gambar 4.8 Pengambilan data pada Lead III

Untuk menganalisa sinyal EKG secara otomatis dapat dilakukan dengan menekan tools P QRS T . kemudian akan tampil garis yang menghubungkan antar defleksi pertama (R sebagai defleksi ke 1) dengan defleksi kedua (R sebagai defleksi berikutnya), jadi, menampilkan interval R-R dan Kompleks QRS, Gambar 4.9 tampilan ketika tools "*PQRST*" di klik;

Gambar 4.9 Hasil Implementasi analisa interval antar defleksi

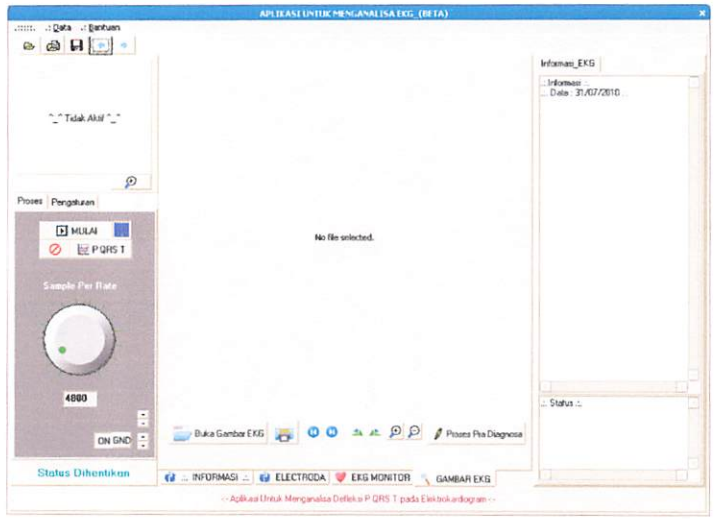
Hasil dari analisa terhadap defleksi yang ada dapat terlihat pada Gambar 4.10 "*informasi EKG*". Informasi yang ada menerangkan interval R-R, Frekuensi R-R, Pace Maker (Simpul SA), Rate dan Kompleks QRS.



Gambar 4.10 Hasil Implementasi Informasi EKG

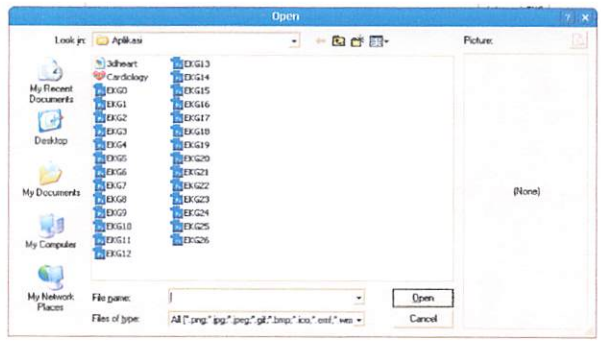
4.1.1.4. Hasil Implementasi Form Preview Rekam EKG

Berdasarkan perancangan pada aplikasi yang sebelumnya di rencanakan, maka hasil implementasi dari form preview (form yang digunakan untuk melihat hasil rekaman EKG yang telah tersimpan kedalam directori harddisk) dapat terlihat pada Gambar 4.11 berikut;



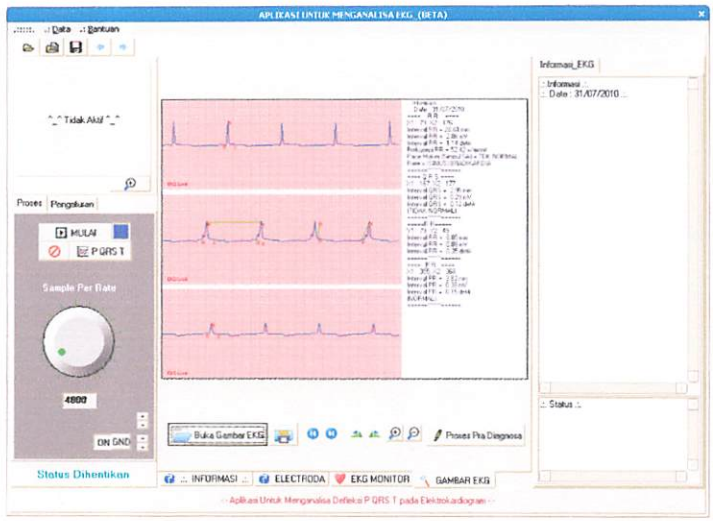
Gambar 4.11 Hasil Implementasi pada Tab Gambar EKG

Gambar yang telah direkam, dapat dilihat dengan menekan tombol "Buka Gambar EKG" maka akan tampil kotak dialog open sebagai berikut, kemudian dapat memilih hasil rekaman yang diinginkan, dapat terlihat pada Gambar 4.12.



Gambar 4.12 Kotak Dialog Open Picture

Setelah pemilihan hasil rekaman ekg pada kotak dialog open picture telah terpilih, hasil rekaman di perlihatkan pada Gambar 4.13.



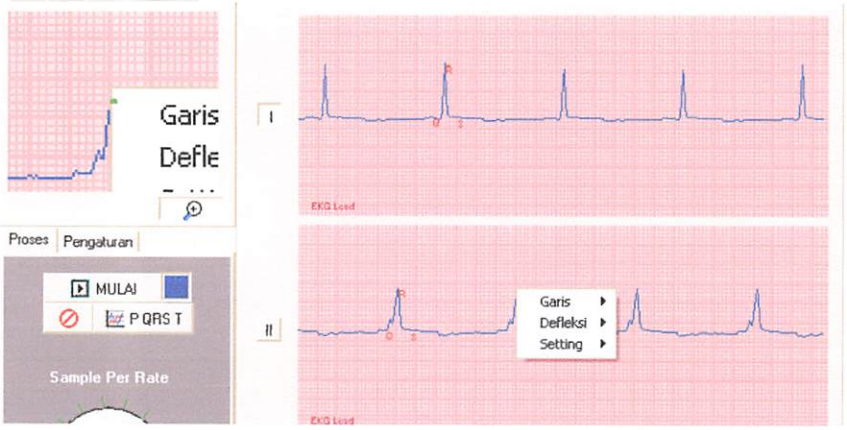
Gambar 4.13 Hasil Implementasi Data Rekam EKG

Hasil Rekam EKG dapat diproses ulang dalam penganalisaan sinyal dengan menekan tombol Proses Pra Diagnosa maka hasil rekam ekg akan berpindah ke dalam tab EKG Monitor seperti pada Gambar 4.14.

Gambar 4.14 Hasil Implementasi Data Rekam EKG

pada tab Monitoring

Proses penganalisaan terhadap hasil rekaman ekg yang telah tersimpan pada direktori harddisk sebelumnya agak sedikit berbeda dengan proses penganalisaan sebelumnya. Perbedaannya, saat menentukan defleksi pertama dan kedua dilakukan secara manual. Gambar 4.15 sampai Gambar 4.20 adalah proses menentukan defleksi secara manual.

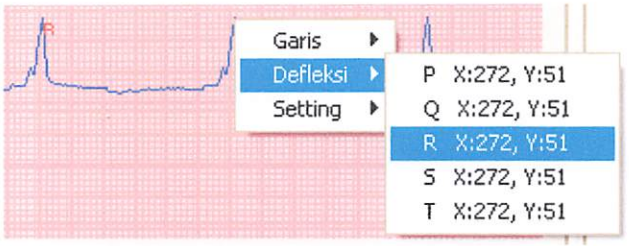


Gambar 4.15 Tampilan Menu pada Rekaman EKG

Menu pada rekaman EKG dapat terlihat dengan cara melakukan klik kanan pada mouse, pada tempat yang telah ditentukan sebelumnya. Seperti pada Gambar 4.16 berikut.

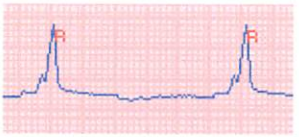
Gambar 4.16 Menentukan Defleksi R Pertama

Kemudian menentukan defleksi R kedua seperti pada Gambar 4.17 berikut.



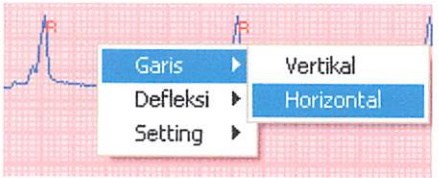
Gambar 4.17 Menentukan Defleksi R Kedua

Setelah menentukan titik dari kedua defleksi maka defleksi akan terlihat seperti pada Gambar 4.18 berikut.



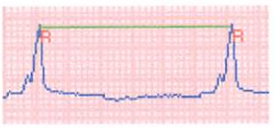
Gambar 4.18 Hasil Dari kedua Defleksi R

Proses selanjutnya mengetahui interval R-R, dengan cara menghubungkan kedua defleksi dengan garis seperti pada Gambar 4.19 berikut.



Gambar 4.19 Menghubungkan Kedua Difleksi dengan garis

Setelah kedua defleksi terhubung maka garis penghubung akan terlihat seperti pada Gambar 4.20 berikut.



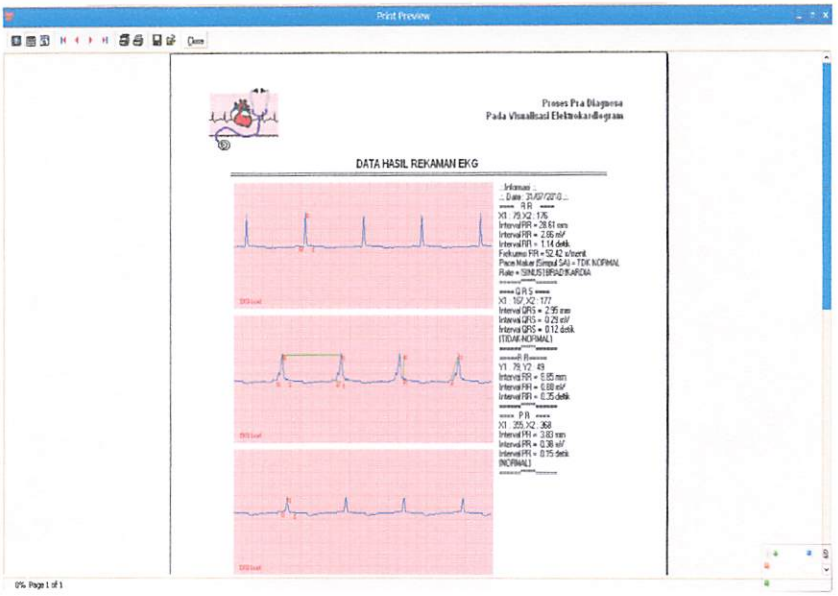
Gambar 4.20 Hasil Garis yang menghubungkan Difleksi R-R

Hasil dari perhitungan interval R-R akan tampil saat kedua defleksi telah dihubungkan dengan garis seperti yang terlihat pada Gambar 4.21.

Gambar 4.21 Hasil perhitungan interval R-R

Proses menentukan defleksi secara manual yang dilakukan seperti yang dilakukan dari Gambar 4.16 sampai Gambar 4.21, tidak hanya dapat dilakukan untuk interval R-R saja. Untuk defleksi P, Q, R, S dan T dapat ditentukan dengan mengikuti langkah-langkah yang telah ditampilkan sebelumnya.

Hasil rekam ekg dapat dicetak dengan menekan tombol  (**print**), dan akan menampilkan hasil rekaman ekg yang siap di cetak, seperti pada Gambar 4.22 berikut.



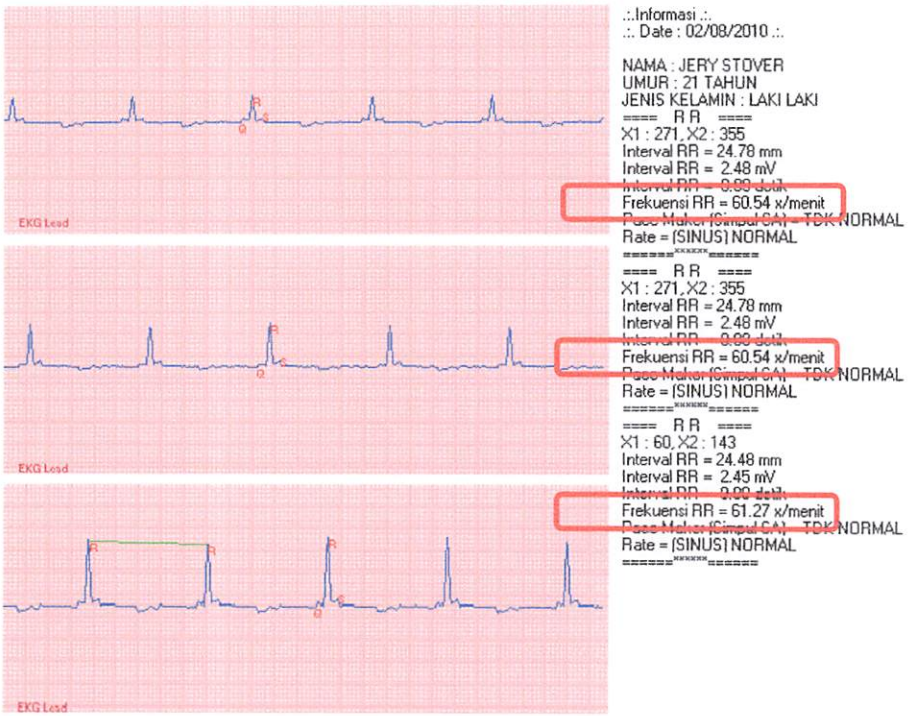
Gambar 4.22 Hasil Implementasi Laporan Rekaman EKG

4.2. Pengujian Sistem

Pengujian ini dilakukan untuk mengetahui kinerja aplikasi menganalisa sinyal EKG berbasis akuisisi data isyarat jantung. Pada pengujian membandingkan hasil perhitungan frekuensi jantung dari mesin EKG yang ada di Rumah Sakit Islam Malang dengan Aplikasi yang dibuat. Hasil Rekaman EKG yang dilakukan di RS Islam Malang seperti pada Gambar 4.23 berikut.

Gambar 4.23 Hasil Rekaman EKG pada Rumah Sakit

Hasil Rekaman EKG yang diperoleh dari Aplikasi yang dibuat terlihat pada Gambar 4.24 berikut.



Gambar 4.24 Hasil Rekaman EKG pada aplikasi

Hasil rekaman EKG dan rekaman aplikasi untuk setiap pasien secara detailnya terlampir pada laporan. Berikut presentase kesalahan yang diperoleh dari hasil kedua rekaman.

Presentase Kesalahan:

$$\%Err = \frac{\text{Hasil Standard} - \text{Hasil Aplikasi}}{\text{Hasil Standard}} \times 100\ \%$$

Keterangan :

Hasil Standard = Hasil Perhitungan yang di dapatkan dari Rumah Sakit Islam Malang

Hasil Aplikasi = Hasil Perhitungan yang di dapatkan dari aplikasi yang dibuat.

Dari pengujian yang dilakukan, hasil dari perekaman pada Lead I, II dan III ditampilkan pada Tabel 4.3 berikut.

Tabel 4.3 Pengujian Sistem

No	Nama Pasien	EKG R.S ISLAM MALANG			Aplikasi EKG yang dibuat			%Err
		Heart Rate						
		I	II	III	I	II	III	
1	**Dwi Cahya Pratowo** *Umur = 22 Tahun*	79	79	83	71	70	69	0,11
2	**M. Ali Jainuri** *Umur = 22 Tahun*	84	84	80	84	84	84	0
3	**M. Farrid** *Umur = 23 Tahun*	58	58	64	70	69	69	0,18
4	**Jery Stover** *Umur = 22 Tahun*	61	61	60	60	60	61	0,01
	∑Err							0,3

Total Presentase Kesalahan

%Err rata-rata = **∑Err / 4**

= **0,3 / 4**

= **0,075 %**

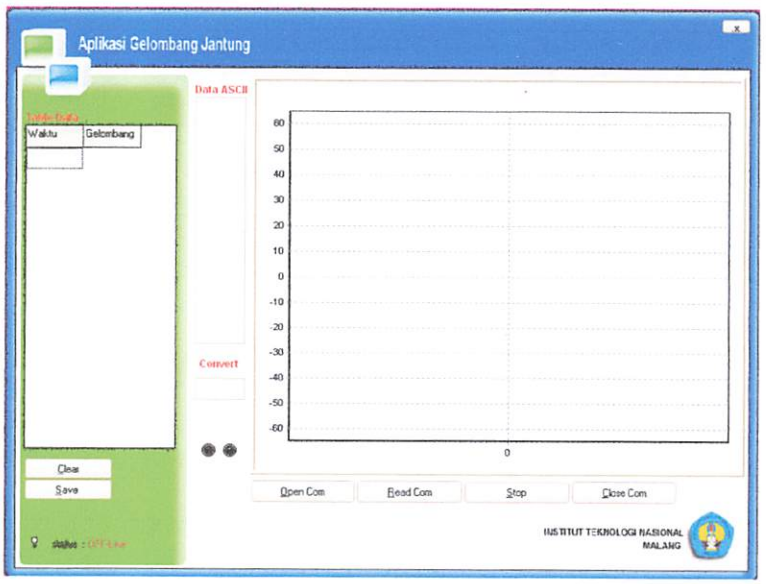
Dari data yang diperoleh dengan membandingkan hasil pengukuran Heart Rate yang dilakukan di Rumah Sakit Islam Malang dengan Aplikasi untuk menganalisa sinyal EKG berbasis akuisisi data isyarat elektris jantung dapat diambil kesimpulan, bahwa selisih antara perhitungan yang dilakukan tidak terlalu besar. Ini dikarenakan alat yang digunakan untuk menghasilkan sinyal

berdasarkan akuisisi data isyarat jantung yang berasal dari permukaan tubuh dengan menghubungkan elektroda masih kurang baik, sehingga mengakibatkan tampilan gelombang pada visualisasi EKG yang dibuat, tampak berbeda dengan tampilan gelombang yang dihasilkan melalui perekaman yang dilakukan di Rumah Sakit Islam Malang.

Perbedaan dari hasil perhitungan dapat terjadi dikarenakan saat melakukan pengukuran, pasien tidak melakukan relaksasi penuh. Apabila pasien mengalami relaksasi penuh, besar kemungkinan terjadinya perubahan pada saat pengambilan data melalui EKG.

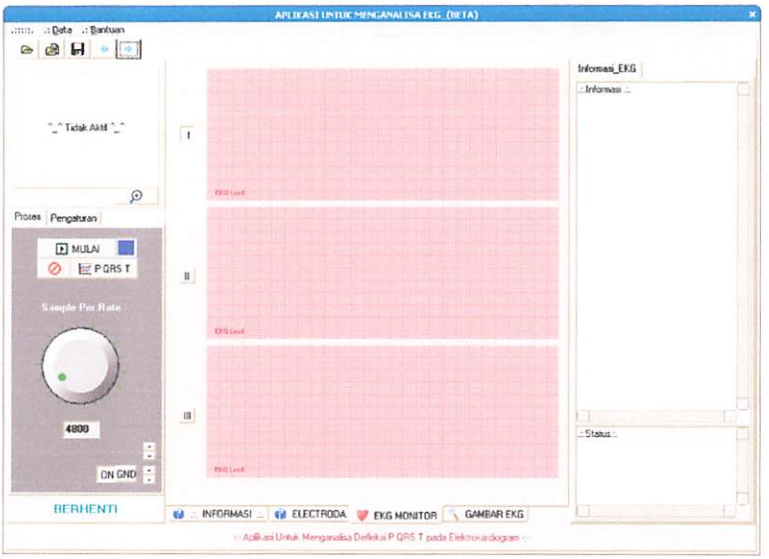
4.3. Membandingkan Aplikasi yang dibuat dengan Aplikasi sebelumnya yang telah ada.

Aplikasi yang sebelumnya ada, merupakan aplikasi yang berfungsi hanya untuk membaca data EKG melalui port USB. Aplikasi sebelumnya ditampilkan seperti pada Gambar 4.25.



Gambar 4.25 Tampilan Aplikasi Gelombang Jantung

Aplikasi yang saat ini dirancang merupakan aplikasi yang berfungsi untuk melakukan pembacaan data EKG melalui Port Audio (Soundcard), kemudian dari data yang didapat dilakukan proses analisa terhadap data yang diperoleh dengan menentukan Defleksi P, Kompleks QRS dan T serta menentukan batasan normal dari interval antar defleksi yang didapat. Aplikasi yang saat ini dirancangn ditampilkan seperti pada Gambar 4.26 berikut.



Gambar 4.26 Tampilan Aplikasi untuk menganalisa sinyal EKG berbasis akuisisi data isyarat elektris jantung

Pengembangan dari aplikasi gelombang jantung menjadi aplikasi untuk menganalisa sinyal EKG berbasis akuisisi data isyarat elektris jantung dimaksudkan untuk memudahkan tenaga ahli / non medis didalam melakukan pendiagnosaan terhadap hasil rekaman EKG yang dilakukan. Pada aplikasi untuk menganalisa sinyal EKG berbasis akuisisi data isyarat elektris jantung memiliki tampilan visualisasi layaknya kertas elektrokardiogram yang digunakan dan memiliki informasi mengenai batasan normal dari interval pada setiap defleksi.

BAB V

PENUTUP

5.1. Kesimpulan

1. Dari pengujian yang dilakukan, proses pengambilan data menggunakan masukan Primer dan Sekunder;

   a) Primer

   Dilakukan dengan menggunakan EKG dan meletakan elektroda sesuai dengan sandapan bipolar (Lead I, Lead II dan Lead III). Pengambilan data Real-Time.

   b) Sekunder

   Menggunakan Image, dimana image diperoleh dari hasil penyimpanan melalui masukan primer.

2. Dari pengujian yang dilakukan, perbedaan yang mendasar terjadinya kesalahan dikarenakan;

   a) Alat yang digunakan, sebagai pembaca kontraksi otot melalui elektroda masih jauh dari sempurna.

   b) Pasien, kurang relaksasinya saat pengambilan data.

3. Dari pengujian yang dilakukan, presentasi kesalahan rata-rata (**%Err Rata-rata**) = **0,075 %**

4. Dari pengujian yang dilakukan, kecepatan didalam menentukan batasan normal interval antar defleksi = **2 detik**. Lebih cepat, bila dibandingkan dengan menentukan batasan normal interval antar defleksi, dengan cara membaca elektrokardiogram yang membutuhkan waktu lebih lama

dikarenakan setelah pasien melakukan pengambilan data dengan EKG, hasil rekaman pasien masih mengalami proses menunggu keterangan dari dokter, setelah dokter melakukan pembacaan terhadap hasil EKG, baru terlihat jelas nilai dari interval antar defleksi yang menyatakan batasan normal dari masing masing defleksi P, kompleks QRS dan T.

5.2 Saran

1. Untuk pengembangan aplikasi selanjutnya, diperlukannya modul EKG yang dapat melakukan perekaman data sebanyak 12 sandapan. Sehingga didalam visualisasi pada aplikasi akan terlihat 12 bentuk sinyal yang berbeda serta perlu adanya proses pendiagnosaan secara detail terhadap hasil yang didapat dari perekaman EKG.
2. Fitur fitur tambahan dari aplikasi yang perlu dikembangkan adalah dengan memberikan fasilitas SMS Gateway atau MMS, sehingga hasil dari menganalisa batasan normal pada setiap defleksi dapat terkirim berupa pesan singkat atau pesan multimedia kepada dokter spesialis.

DAFTAR PUSTAKA

[1]. Sutopo Widjaja, ECG Praktis, Binarupa Aksara, Jakarta, 1990, hal 8-37

[2]. John R Hampton, Dasar Dasar EKG Ed 6, Penerbit Buku Kedokteran EGC, Jakarta, 2004, hal 1-7

[3]. Budiyanto, Alex, 2003, *Pengantar Algoritma dan Pemrograman,* www.ilmukomputer.com

[4]. Rahadian Hadi, Pemrograman Windows API dengan Microsoft Visual Basic, Media Elextindo Jakarta, 2001

[5]. Fakultas Kedokteran Universitas Indonesia, EKG dan penanggulangan beberapa penyakit jantung untuk dokter umum, Balai Penerbit FKUI, Jakarta, 1996, hal 25-37

[6] Madcoms, 2003. *Pemrograman Borland Delphi 7,* Penerbit Andi Yogyakarta.

[7]. Zamrony P Juhara, Memainkan Wav dengan Wave API, http://delphindo.wordpress.com/2006/09/19/memainkan-wav-dengan-wave-api/

[8]. ABD, Sumsel Masih Minim Dokter Spesialis Jantung, http://kompas.com

[9]. Perhimpunan Dokter Spesialis Kardiovaskular Indonesia (PERKI), Penyakit Jantung Bawaan, angka tinggi dengan tenaga terbatas, http://www.inaheart.org/index.php/public/information/news-detail/12

[10]. Amos, S.W.,1988, *Kamus Elektronika,* terj., PT Elex Media Komputindo (Kelompok Gramedia), Jakarta.

[11]. Hasan Murod, Perancangan Sistem Akuisisi Data Menggunakan Masukan Soundcard, http://murod.web.ugm.ac.id/scdaq.pdf .

L A M P I R A N

PERKUMPULAN PENGELOLA PENDIDIKAN UMUM DAN TEKNOLOGI NASIONAL MALANG
INSTITUT TEKNOLOGI NASIONAL MALANG
FAKULTAS TEKNOLOGI INDUSTRI
FAKULTAS TEKNIK SIPIL DAN PERENCANAAN
PROGRAM PASCASARJANA MAGISTER TEKNIK
BNI (PERSERO) MALANG
BANK NIAGA MALANG
Kampus I : Jl. Bendungan Sigura-gura No.2 Telp. (0341) 551431 (Hunting), Fax. (0341) 553015 Malang 65145
Kampus II : Jl. Raya Karanglo, Km 2 Telp. (0341) 417636 Fax. (0341) 417634 Malang

BERITA ACARA UJIAN SKRIPSI
FAKULTAS TEKNOLOGI INDUSTRI

NAMA	:	Dwi Cahya Pratowo
NIM	:	06.12.597
JURUSAN	:	Teknik Elektro S-1
KONSENTRASI	:	Teknik Komputer dan Informatika
JUDUL	:	Pengembangan Aplikasi Untuk Menganalisa Sinyal Elektrokardiograp (EKG) Berbasis Akuisisi Data Isyarat Elektris Jantung

Dipertahankan di hadapan Tim Penguji Ujian Skripsi Jenjang Program Strata Satu (S-1)

Pada Hari : Rabu
Tanggal : 18 Agustus 2010
Dengan Nilai : 86.1 (A)

PANITIA UJIAN SKRIPSI

KETUA

Ir. Yusuf Ismail Nakhoda, MT.
NIP.Y. 1018800189

ANGGOTA PENGUJI

PENGUJI I

M. Ibrahim Ashari, ST. MT.
NIP. P. 1030100358

PENGUJI II

Sotyohadi, ST.
NIP. Y. 1039700309

PERKUMPULAN PENGELOLA PENDIDIKAN UMUM DAN TEKNOLOGI NASIONAL MALANG
INSTITUT TEKNOLOGI NASIONAL MALANG
FAKULTAS TEKNOLOGI INDUSTRI
FAKULTAS TEKNIK SIPIL DAN PERENCANAAN
PROGRAM PASCASARJANA MAGISTER TEKNIK
BNI (PERSERO) MALANG
BANK NIAGA MALANG
Kampus I : Jl. Bendungan Sigura-gura No.2 Telp. (0341) 551431 (Hunting), Fax. (0341) 553015 Malang 65145
Kampus II : Jl. Raya Karanglo, Km 2 Telp. (0341) 417636 Fax. (0341) 417634 Malang

FORMULIR PERBAIKAN SKRIPSI

Dalam pelaksanaan ujian skripsi jenjang Strata satu (S-1) Jurusan Teknik Elektro konsentrasi Teknik Komputer dan Informatika, maka perlu adanya perbaikan skripsi untuk Mahasiswa :

Nama	:	Dwi Cahya Pratowo
NIM	:	06.12.597
Jurusan	:	Teknik Elektro S-1
Konsentrasi	:	Teknik Komputer dan Informatika
Masa Bimbingan	:	21-April-2010 s/d 21-Oktober-2010
Judul Skripsi	:	Pengembangan Aplikasi Untuk Menganalisa Sinyal Elektrokardiograp (EKG) Berbasis Akuisisi Data Isyarat Elektris Jantung

Tanggal	Uraian	Paraf
18 Agustus 2010	1. Tambahkan kalimat pengantar pada gambar dan tabel.	

Mengetahui,

Dosen Pembimbing

I Komang Somawirata, ST.MT.
NIP. Y. 1030100361

Dosen Penguji,

M. Ibrahim Ashari, ST. MT.
NIP. P. 1030100358

INSTITUT TEKNOLOGI NASIONAL
Jl. Bendungan Sigura – gura No.2
MALANG

FORMULIR BIMBINGAN SKRIPSI

Nama : Dwi Cahya Pratowo
Nim : 06.12.597
Masa Bimbingan : 21-April-2010 s/d 21-Oktober-2010
Judul Skripsi : Pengembangan Aplikasi untuk Menganalisa Sinyal Elektrokardiograp (EKG) Berbasis Akuisisi Data Isyarat Elektris Jantung

No	Tanggal	Uraian	Paraf Pembimbing
1	20/07/2010	Demo Program	
2	02/08/2010	Bab I	
3	02/08/2010	Bab II (Revisi Teori Dasar)	
4	02/08/2010	Bab III (Revisi Flowchart Aplikasi)	
5	02/08/2010	Bab IV (Revisi Presentase Kesalahan Pengujian)	
6	02/08/2010	Bab V (Revisi Kesimpulan)	
7	06/08/2010	Makalah Seminar Hasil	
8			
9			
10			

Malang,

Dosen pembimbing

I Komang Somawirata, ST. MT

NIP. 1030100361

Form S-4b

YAYASAN UNIVERSITAS ISLAM MALANG

RUMAH SAKIT ISLAM MALANG

بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ

19 Sya'ban 1431 H
Malang ----------
31 Juli 2010 M

SURAT KETERANGAN
Nomor : 143/Ket.A/RSI-U/VII/2010

Assalamu'alaikum Warahmatullahi Wabarakatuh

Direktur Rumah Sakit Islam Malang menerangkan bahwa :

Nama	:	**Dwi Cahya Pratowo**
NIM	:	06.12.597
Jurusan / Fakultas	:	Teknik Elektro / Teknik Industri Institut Teknologi Nasional Malang
Judul Karya Tulis Ilmiah	:	Pengembangan Aplikasi untuk Menganalisa Sinyal Elektrokardiograf (EKG) Berbasis Akuisisi Data Isyarat Elektris Jantung ;

telah selesai melakukan pengumpulan data / bahan di Rumah Sakit Islam Malang dalam rangka penyusunan Karya Tulis Ilmiah / Skripsi, pada tanggal 13 Juli s/d 02 Agustus 2010.

Demikian Surat Keterangan dibuat untuk dipergunakan sebagaimana mestinya.

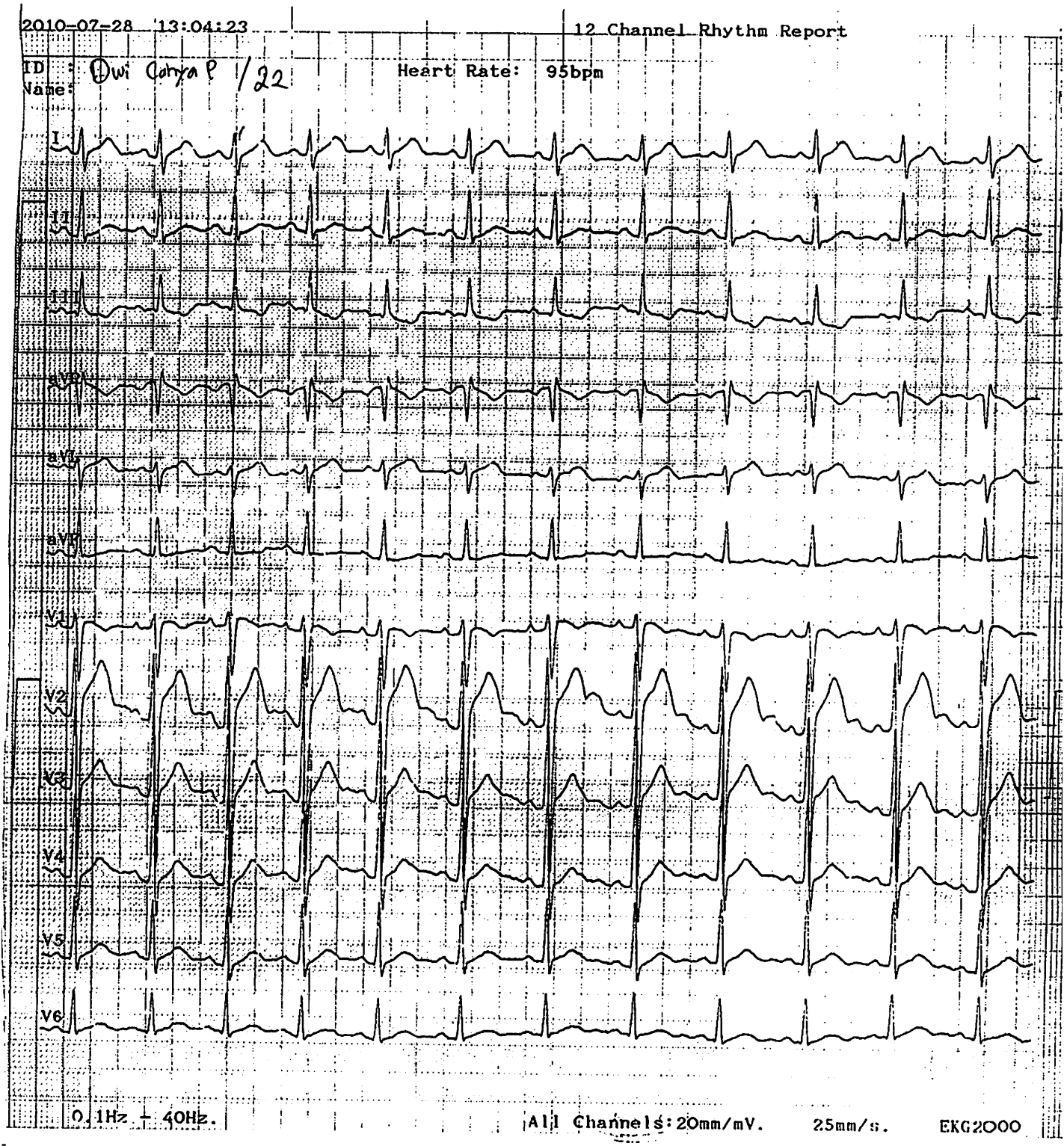
Wallahul muwaffiq ilaa aqwamith thorieq
Wassalamu'alaikum Warahmatullahi Wabarakatuh

RUMAH SAKIT ISLAM MALANG

Direktur,

dr. V. H. Pratomo
NPP 928894196

Jl. M.T. Haryono No. 139 Malang, Jawa Timur - 65144
Telp. (0341) 551356, 580798, 565448, Fax. (0341) 551257, 577205, IGD : (0341) 570400
Ruang Klas VIP : (0341) 569956, ROI : (0341) 581823, Inst. Farmasi : (0341) 577321, Inst. Bedah : (0341) 552330
Care Line : 081333280001
E-mail : rsi_malang@telkom.net

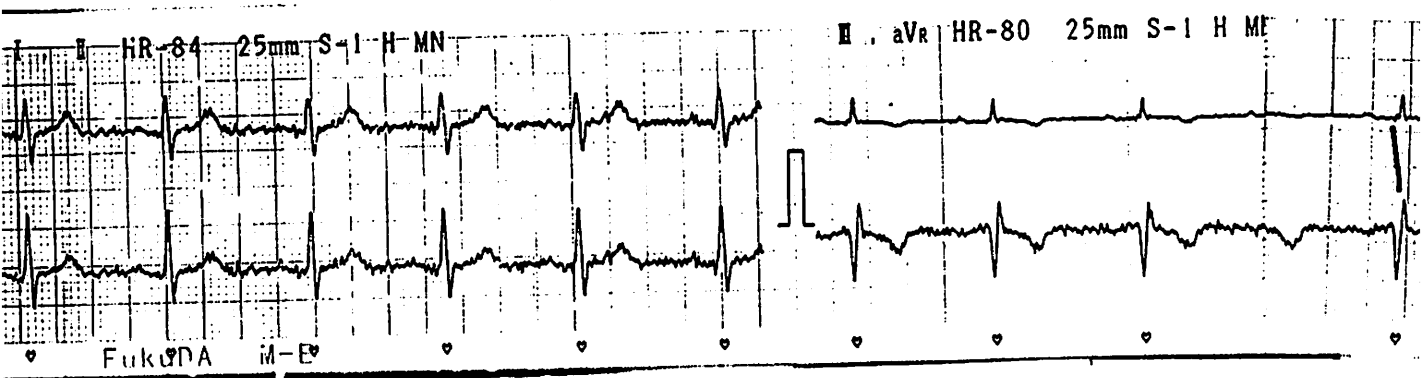

2010-07-28 13:04:23
12 Channel Rhythm Report
ID :
Name:
Heart Rate: 95bpm
I
II
III
aVR
aVL
aVF
V1
V2
V3
V4
V5
V6
0.1Hz - 40Hz.
All Channels:20mm/mV.
25mm/s.
EKG2000

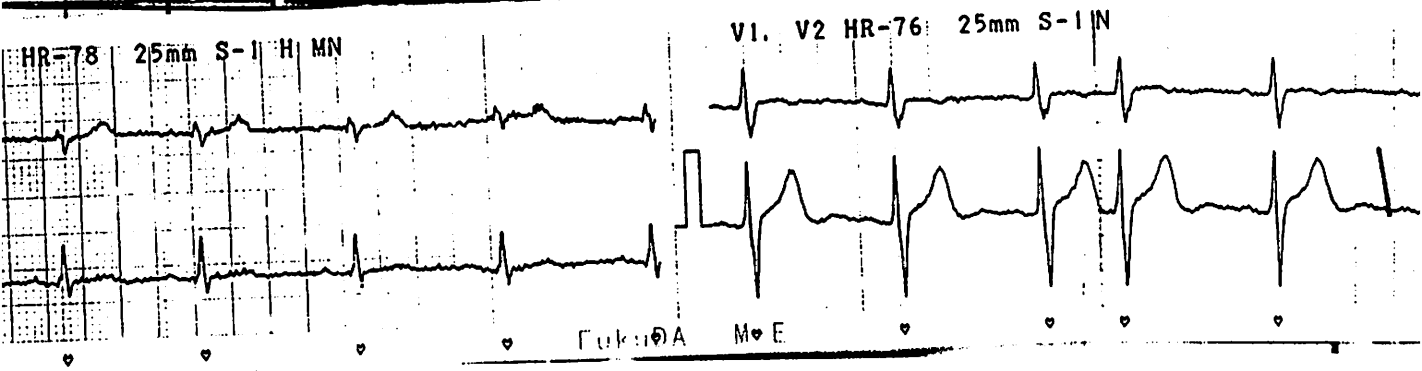
RM 11

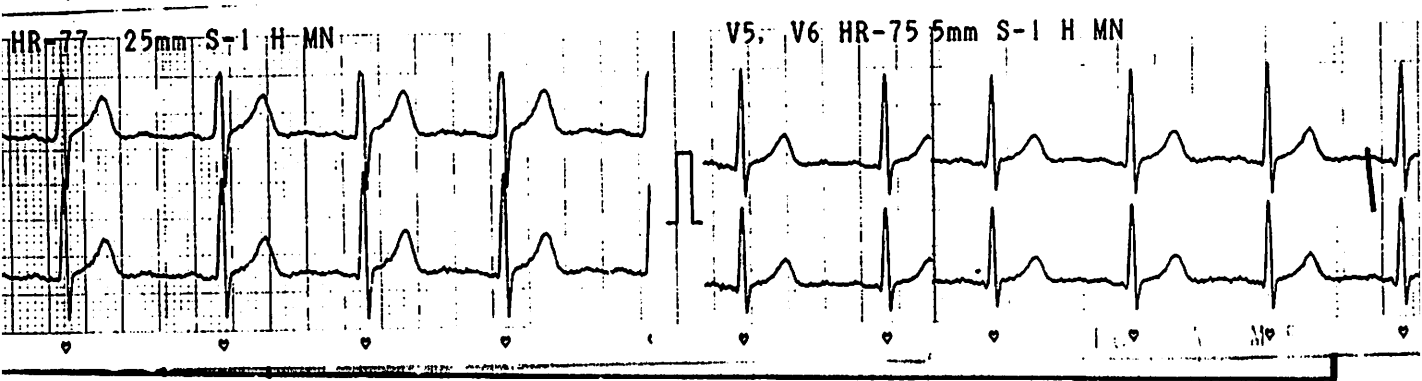
YAYASAN UNIVERSITAS ISLAM MALANG
RUMAH SAKIT ISLAM MALANG
Jl. MT. Haryono No. 139, Telp. (0341) 551356,
580798 Fax. 565443 Malang 65144

PEMERIKSAAN EKG

NAMA PASIEN : .....
NO. RM : .....
RUANG / LANTAI : .....
KELAS : .....







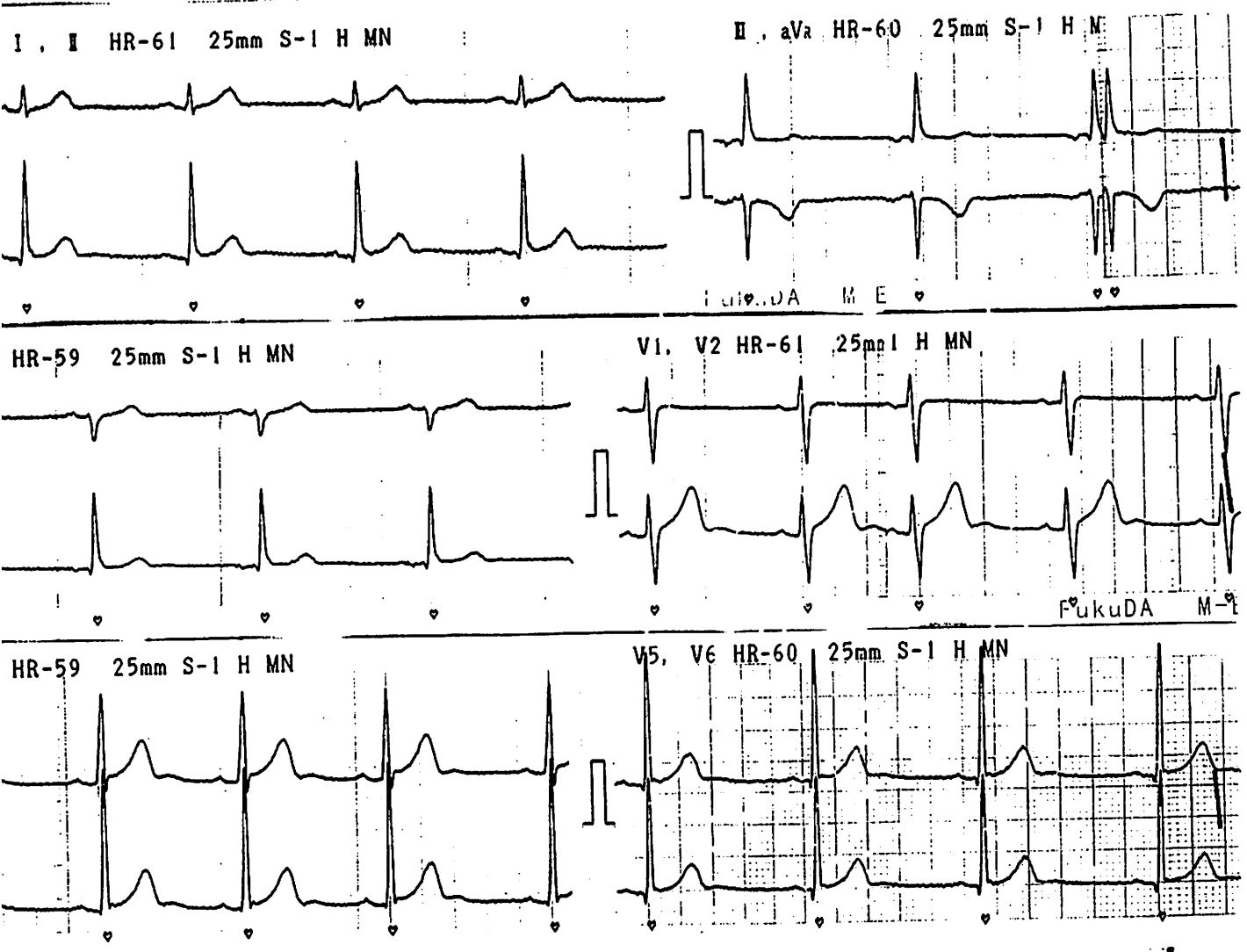
JNY C110
2-/8-/16 : HR V:2.0
BP 78bpm
mmHg
Muhamad Ali J.
62 / 22 Hgt. Wt. cm kg
********** [MEMO] **********

RM 11

YAYASAN UNIVERSITAS ISLAM MALANG
RUMAH SAKIT ISLAM MALANG
Jl. MT. Haryono No. 139, Telp. (0341) 551356,
580798 Fax. 565448 Malang 65144

PEMERIKSAAN EKG

NAMA PASIEN : .....
NO. RM : .....
RUANG / LANTAI : .....
KELAS : .....



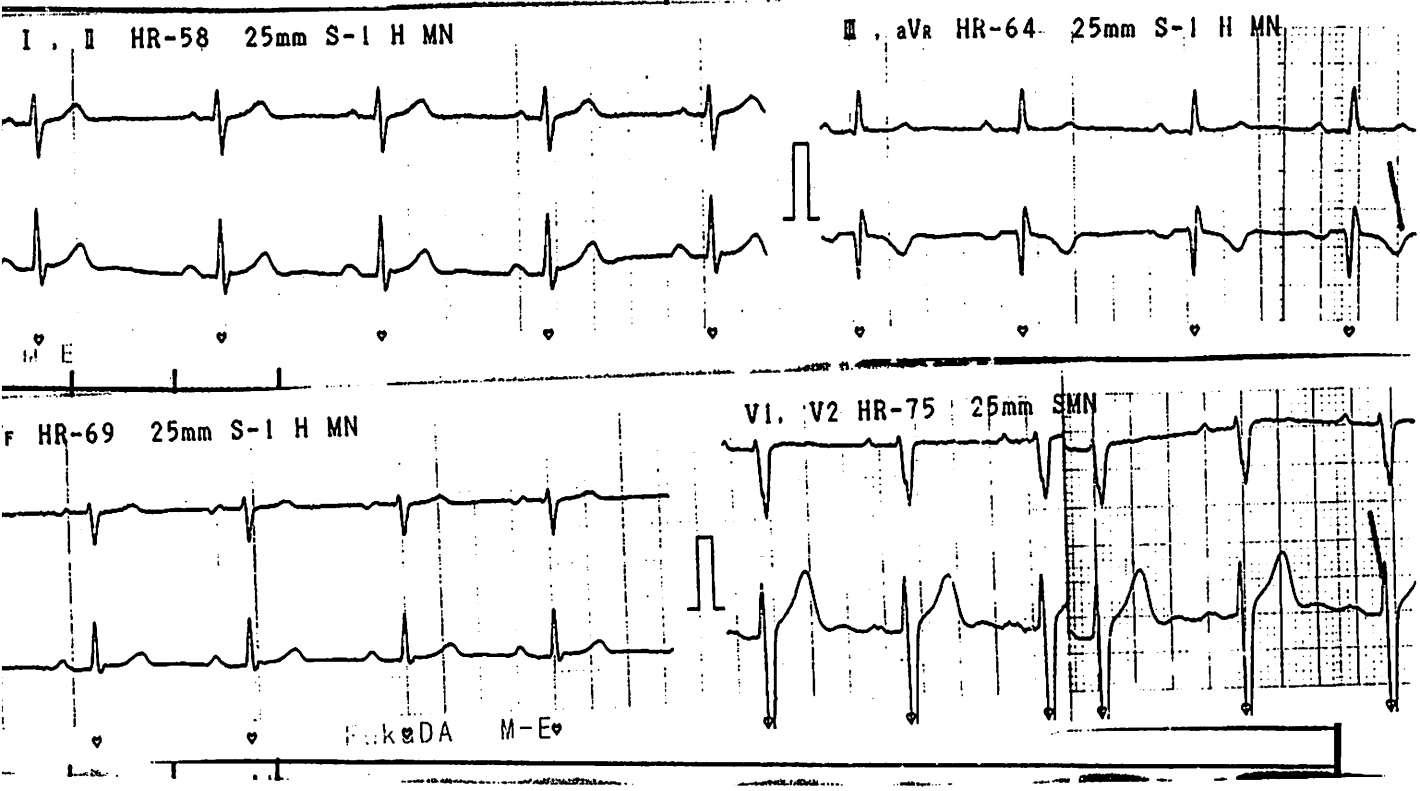
SUNY CLIO
2/8/10 : HR: V:2.0
BP: 60bpm
Jerry Stouls mmHg
Age 21 / 02 Hgt. Wt. cm kg
********** [MEMO] **********

RM 11

YAYASAN UNIVERSITAS ISLAM MALANG
RUMAH SAKIT ISLAM MALANG
Jl. MT. Haryono No. 139, Telp. (0341) 551356, 580798 Fax. 565448 Malang 65144

PEMERIKSAAN EKG

NAMA PASIEN : .....
NO. RM : .....
RUANG / LANTAI : .....
KELAS : .....



DISUNY C110
e: 02/08/20 :
e: Muhamad Farid
Age 23 / ♂
HR:
BP:
V: 2.0
72bpm
mmHg
Hgt. cm
Wt. kg
********** [MEMO] **********

```
{
 => Form Awal Aplikasi
//=============================================================//
 Aplikasi Untuk Menganalisa Sinyal EKG Berbasis
 Akuisisi Data Isyarat Elektris Jantung di buat
 untuk Menyelesaikan Skripsi.
 dibuat pada : Agustus 2010

 Oleh : dwy_cah@yahoo.co.id
//=============================================================//
}
unit U_EKG;
interface
uses
 Windows, Messages, mmsystem, shellAPi, SysUtils, Variants, Classes,
 Graphics, Controls, Forms, Dialogs, U_formEKG, ComCtrls, ExtCtrls,
 U_Wave, Buttons, StdCtrls, Menus, pngimage, mKnob, RadioDials, ExtDlgs,
 OleCtrls, PREVIEWLib_TLB, U_ZoomPic,GIFImage, ImgList, jpeg, U_PrintEKG;
type
 PBufArray=PByteArray;
 Tfrm_UtamaEKG = class(TForm)
  P_Utama: TPanel;
  P_Kiri: TPanel;
  P_Bawah: TPanel;
  P_isi: TPanel;
  PageControl1: TPageControl;
  Tab_EKGmonitoring: TTabSheet;
  Tab_Info: TTabSheet;
  SpLead1: TSpeedButton;
  SpLead2: TSpeedButton;
  SpLead3: TSpeedButton;
  frm_EKGMonitor1: Tfrm_EKGMonitor;
  frm_EKGMonitor2: Tfrm_EKGMonitor;
  frm_EKGMonitor3: Tfrm_EKGMonitor;
  P_Kanan1: TPanel;
  P_isi_kiri: TPanel;
  btnRun: TSpeedButton;
  btnCapFrame: TSpeedButton;
  P_isi_kanan: TPanel;
  btnExpand1: TSpeedButton;
  btnGain1: TSpeedButton;
  UpBeamLight: TUpDown;
  UpFocus: TUpDown;
  P_isi_bawah: TPanel;
  Label2: TLabel;
  Ed_SwepUD: TEdit;
  SwepUD: TUpDown;
  Label1: TLabel;
  Ed_UpGain: TEdit;
  UpGain: TUpDown;
  trOfsCh: TTrackBar;
  Lbl_Scale: TLabel;
  Label3: TLabel;
  Btn_SimpanImage: TBitBtn;
  Btn_PQRST: TBitBtn;
  img_hasilEKG: TImage;
  statusText: TPanel;
  P_atas: TPanel;
  btnCHGnd: TSpeedButton;
  btnCHOn: TSpeedButton;
  P_LED: TPanel;
  img_isi: TImage;
  img_GifSHow: TImage;
  Tab_LihatEKG: TTabSheet;
  Preview_EKG: TPreview;
  OPDialog_Img: TOpenPictureDialog;
  btn_OpenImgEkg: TBitBtn;
  btn_ZoomIn: TBitBtn;
  btn_ZoomOut: TBitBtn;
  frm_ZoomPic1: Tfrm_ZoomPic;
  btn_ZoomPic: TSpeedButton;
  PageControl2: TPageControl;
  Tab_Info1: TTabSheet;
  Img_Back_ShowGif: TImage;
  MainMenu1: TMainMenu;
  N1: TMenuItem;
  btn_Close: TMenuItem;
  Bantuan: TMenuItem;
  btn_Help: TMenuItem;
  ImageList1: TImageList;
  P_kiriAtas: TPanel;
  P_Control_kiri: TPageControl;
  Tab_Proses: TTabSheet;
  Tab_Setting: TTabSheet;
  btn_SP_Rate: TRadioDial;
  Label4: TLabel;
  txt_SP_rate: TEdit;
  Image1: TImage;
  Data1: TMenuItem;
  btn_Simpan_EKG2: TMenuItem;
  Tab_Posisi_Electroda: TTabSheet;
  btn_FowardPage: TBitBtn;
  btn_NextPage: TBitBtn;
  btn_Print: TBitBtn;
  N2: TMenuItem;
  btn_Menu_Perbesar: TMenuItem;
  btn_Menu_Perbesar2: TMenuItem;
  N3: TMenuItem;
```

```
btn_Menu_BukaEKG: TMenuItem;
N4: TMenuItem;
btn_Menu_Sebelumnya: TMenuItem;
btn_Menu_Selanjutnya: TMenuItem;
P_MemoInfo: TPanel;
P_memoStatus: TPanel;
M_info: TMemo;
M_Status: TMemo;
img_pasien_electroda: TImage;
btn_PraDiagnosa: TBitBtn;
btn_imgRun: TBitBtn;
btn_imgSTOP: TBitBtn;
btn_OpenImage: TBitBtn;
btn_OpenImgDiagnosa: TBitBtn;
RB_LeadI: TRadioButton;
RB_LeadII: TRadioButton;
RB_LeadIII: TRadioButton;
img_RA: TImage;
img_LA: TImage;
img_RL: TImage;
img_LL: TImage;
Shape1: TShape;

procedure Bufferfull(Var Message: TMessage); message MM_WIM_DATA;
procedure FormCreate(Sender: TObject);
procedure FormActivate(Sender: TObject);
procedure FormCloseQuery(Sender: TObject; var CanClose: Boolean);
procedure btnRunClick(Sender: TObject);
procedure btnCapFrameClick(Sender: TObject);
procedure Ed_UpGainChange(Sender: TObject);
procedure Ed_SwepUDChange(Sender: TObject);
procedure trOfsChChange(Sender: TObject);
procedure UpFocusClick(Sender: TObject; Button: TUDBtnType);
procedure UpBeamLightClick(Sender: TObject; Button: TUDBtnType);
procedure btnCHOnClick(Sender: TObject);
procedure btnExpand1Click(Sender: TObject);
procedure Btn_SimpanImageClick(Sender: TObject);
procedure SpLead1Click(Sender: TObject);
procedure SpLead2Click(Sender: TObject);
procedure SpLead3Click(Sender: TObject);
procedure Btn_PQRSTClick(Sender: TObject);
procedure frm_EKGMonitor3btn_GarisClick(Sender: TObject);
procedure frm_EKGMonitor1btn_Garis_HorizontalClick(Sender: TObject);
procedure frm_EKGMonitor2btn_Garis_HorizontalClick(Sender: TObject);
procedure frm_EKGMonitor3btn_Garis_HorizontalClick(Sender: TObject);
procedure frm_EKGMonitor1btn_Garis_VertikalClick(Sender: TObject);
procedure frm_EKGMonitor2btn_Garis_VertikalClick(Sender: TObject);
procedure frm_EKGMonitor3btn_Garis_VertikalClick(Sender: TObject);
procedure btn_OpenImgEkgClick(Sender: TObject);
procedure btn_ZoomInClick(Sender: TObject);
procedure btn_ZoomOutClick(Sender: TObject);
procedure btn_ZoomPicClick(Sender: TObject);
procedure btn_SP_RateChange(Sender: TObject);
procedure btn_CloseClick(Sender: TObject);
procedure btn_FowardPageClick(Sender: TObject);
procedure btn_NextPageClick(Sender: TObject);
procedure btn_Simpan_EKG2Click(Sender: TObject);
procedure btn_PrintClick(Sender: TObject);
procedure btn_Menu_PerbesarClick(Sender: TObject);
procedure btn_Menu_Perbesar2Click(Sender: TObject);
procedure btn_Menu_BukaEKGClick(Sender: TObject);
procedure btn_Menu_SebelumnyaClick(Sender: TObject);
procedure btn_Menu_SelanjutnyaClick(Sender: TObject);
procedure btn_PraDiagnosaClick(Sender: TObject);
procedure btn_OpenImageClick(Sender: TObject);
procedure btn_OpenImgDiagnosaClick(Sender: TObject);
procedure RB_LeadIClick(Sender: TObject);
procedure RB_LeadIIClick(Sender: TObject);
procedure RB_LeadIIIClick(Sender: TObject);


private
  { Private declarations }
  BeamEKG      : array of TPoint;
  ValxR,ValyR  : Integer;
  ValxS,ValyS  : Integer;
  ValxQ,ValyQ  : Integer;
  StoredExpand : Integer;
  StoredCHOff  : Integer;

  procedure DoDrawBeamText(Sender:Tobject);
  procedure saveImage;
  procedure ChangeSampleRate;
  procedure Start;
  procedure Stop;
  procedure SetButtonState;
  procedure ShowStored;
  procedure CenterAdjust;
  procedure Recalc;
  procedure ShowScaleValue;
public
  { Public declarations }
  WaveIN:TWaveIn;
  frmEKGMonitoring:Tfrm_EKGMonitor;
  cy:Integer;
  hInc:Single;
  errcount:Integer;
  ChGain:Integer;
```

```
    origbufSize,bufSize:Integer;
    frameSize:Integer;
    xinc:Integer;
    X,Y:Integer;
    PlotPtsPerFrame:Integer;
    nbrframes:Integer;
    time1,time2:TTime;
    singleFrame:Boolean;
    trigsign:Integer;
    triglevel:Integer;
    triggered:Boolean;
    triggerindex:Integer;
    trigbarchanging, trigGrpchanging:Boolean;
    frameSaveBuf:array of Byte;
    calibrate:Boolean;
    Offsety:Integer;
    savedFrameData:array of Integer;
    buffer1found:Boolean;

    procedure Posterror(S:String);
    procedure setup;
    procedure setMaxPtsToAVG;
    procedure SetOscState;
    procedure CopyImageEKG;
    procedure TextToImage;
    procedure ShowGif;
    procedure BackCopyImageEKG;
   end;

var
  frm_UtamaEKG: Tfrm_UtamaEKG;
implementation
uses Math, DateUtils, Types;
{$R *.dfm}
var
  framesPerBuffer:Integer=2;
  numbuffers:Integer=4;
  const
    nbrHLines=10;
procedure Tfrm_UtamaEKG.FormCreate(Sender: TObject);
begin
  frmEKGMonitoring := frm_EKGMonitor1; //deklarasi awal frame EKG Monitor
  DoubleBuffered   := True;
  origbufSize      := framesPerBuffer*frmEKGMonitoring.ImgLine.Width; // ukuran buffer
  bufSize          := origbufSize;
  setlength(frameSaveBuf,2*bufSize);
  xinc   := SwepUD.Position;
  Offsety := 128; {tengah screen di asumsikan 8-bit data}
  StoredExpand := 1;
  setup;
  btnCHOn.Down:=True;
  frmEKGMonitoring.OnAfterBeamDraw := DoDrawBeamText;
  ShowGif; //Menampilkan Gif
end;
procedure Tfrm_UtamaEKG.FormActivate(Sender: TObject);
begin
  SetButtonState; //Mengatur Button
end;
procedure Tfrm_UtamaEKG.FormCloseQuery(Sender: TObject;
  var CanClose: Boolean);
begin
  {menghentikan perekaman}
  if Assigned(WaveIN) then
  begin
    WaveIN.StopInput;
    FreeAndNil(Wavein);
  end;
  CanClose := True;
end;

// inisialisasi awal=============================
//==============================================
procedure Tfrm_UtamaEKG.setup;
begin
  errcount  := 0;
  x         := 0;
  if not Assigned(WaveIN) then
    WaveIN := TWaveIn.Create(Handle,{org}bufSize,numbuffers);
    with WaveIN, ptrWaveFmtEx^ do
    begin
      wFormatTag := WAVE_FORMAT_PCM;
      //mengaktifkan chanel 1
      nChannels := 1; //1 = Mono ; 2 = Stereo
    nSamplesPerSec := StrToInt(txt_SP_rate.Text);
    ShowScaleValue;
    SetOscState;      //procedure yang mengakses Frame EKG Monitor
    wBitsPerSample := 8;
    //membandingkan data yang ada
    nAvgBytesPerSec := nSamplesPerSec*(wBitsPerSample div 8)* nChannels;
    OnError := Posterror;

    if SetupInput then
      statusText.Caption:='BERHENTI'
    else
      statusText.Caption:='ERROR - COBA LAGI';

    SetLength(frameSaveBuf,2*bufSize);
    cy := frmEKGMonitoring.ImgLine.Height div 2;
```

```
  end;
  SetLength(savedFrameData,0);
end;

procedure Tfrm_UtamaEKG.btnRunClick(Sender: TObject);
begin
  Recalc;
  if btnRun.Down then begin
    Start;
    btnRun.Glyph   := btn_imgSTOP.Glyph;
    btnRun.Caption  := 'STOP';
  end else
  begin
    Stop;
    btnRun.Glyph   := btn_imgRun.Glyph;
    btnRun.Caption  := 'MULAI';
  end;
  SetButtonState;
end;

procedure Tfrm_UtamaEKG.Recalc;
begin
  PlotPtsPerFrame := (frmEKGMonitoring.ImgLine.Width);
  setMaxPtsToAVG;
end;

procedure Tfrm_UtamaEKG.Start;
begin
  M_info.Clear;
  M_info.Lines.Add('.:.Informasi .:.');
  M_info.Lines.Add('.:. Date : '+FormatDateTime('dd/mm/yyyy',Now)+' .:.');

  M_Status.Clear;
  M_Status.Lines.Add('.:. Status .:.');
  setup;
  singleFrame := False;
  triggered  := False;

  If WaveIN.RecordActive then
  begin
    WaveIN.StopInput;
    Application.ProcessMessages;
  end;

  WaveIN.StartInput;
  statusText.Caption:='Status Dijalankan';
end;

procedure Tfrm_UtamaEKG.Stop;
begin
  If Assigned(WaveIN) then
    WaveIN.StopInput;
  statusText.Caption:='Status Dihentikan';
end;

procedure Tfrm_UtamaEKG.btnCapFrameClick(Sender: TObject);
begin
  if btnCapFrame.Down then
  begin
    StoredCHOff := trOfsCh.Position;
    SetButtonState;
    Recalc;
    btnExpand1.Down := True;
    btnGain1.Down  := True;
    M_info.Clear;
    M_info.Lines.Add('.:.Informasi .:.');
    M_info.Lines.Add('.:. Date : '+FormatDateTime('dd/mm/yyyy',Now)+' .:.');
    setup;
    singleFrame := True;
    triggered := False;
    triggerindex :=0;
    if Assigned(WaveIN) and WaveIN.RecordActive then
    begin
      WaveIN.StopInput;
      Application.ProcessMessages;
    end;
    //debug=============================
    buffer1found  := False;

  if Assigned(WaveIN) then
      WaveIN.StartInput;
      btnRun.Glyph   := btn_imgRun.Glyph;
      btnRun.Caption  := 'MULAI';
      statusText.Caption:='Status Dihentikan';
    end
    else
    begin
      singleFrame := False;
      if Assigned(WaveIN) and WaveIN.RecordActive then
      begin
        WaveIN.StopInput;
        Application.ProcessMessages;
      end;
       SetButtonState;
  end;
end;

//=================================
```

```
//Buffer=============================
//===================================
procedure Tfrm_UtamaEKG.Bufferfull(var Message:TMessage);
var
 i:Integer;
 p:PBufArray;
 pstart:Integer;
 frametoScan : Integer;
  function scanBufferfortrigger(p:PBufArray):Integer;
  var
   i:Integer;
  begin
   Result := -1;
   for i:=1 to bufsize-1 do
   begin
    if (P^[i-1]<triglevel)and(P^[i]>=triglevel)then
    begin
     Result:=i;
     Break;
    end;
   end;
  end;

  function ChScale(x:integer):integer;
  var
   N:Integer;
  begin
   N := Offsety-x;
   if ChGain < 3 then
    N := N shr (3-ChGain)
   else if ChGain > 3 then
    N := N shl (ChGain-3);
   Result := cy + N;
  end;
var
  BeamI : Integer;
begin {Memulai Procedure Buffer}

 //===================================
 //    Membuat Panel LED BErkedip
 time2:=now;
 if (time2-time1) > 1/secsperday then
 begin
  Application.ProcessMessages;
  time1 := time2;
  if not btnRun.Down or (P_LED.Color = clLime) then
   P_LED.Color := clNavy
  else
   P_LED.Color := clLime ;
end;
//===================================
frametoScan := nbrframes - 1;
if Assigned(WaveIN) and (WaveIN.RecordActive) then
 with WaveIN do
 begin
  {Lparam pada isi pesan seubuat ponter ke sebuah structur
  WaveHDR yang pertama dimasukan ke ponter buffer}
  p := pointer(pwavehdr(Message.LParam)^.lpData);
  x:=0;
  if calibrate then {reset nilai NOL}
  begin
   Offsety := 0;
   for i:= 0 to bufSize-1 do
    inc(offsety,p^[i]);

   Offsety := Offsety div bufSize;
   calibrate := False;
  end;

  trigsign := 0;
  //===================================
  buffer1found :=True;
  pstart      := Width;
  x    := 0;//biar enak harusnya pake posisi trackbar
  BeamI := -1;
  i    := 0;

  triggered := trigsign=0;
  //singel Trace
  SetLength(BeamEKG,round(PlotPtsPerFrame)-3);
  ValyR := BeamEKG[0].Y; ValyS := BeamEKG[0].Y;
  while i <= PlotPtsPerFrame-4 do
  begin
   inc(i);
   inc(BeamI);
   //triggered dihilangkan
   BeamEKG[BeamI].X:=x;
   BeamEKG[BeamI].Y:=ChScale(p^[i]) + trOfsCh.Position;
   if ValyR > BeamEKG[BeamI].Y then begin
    ValyR := BeamEKG[BeamI].Y;
    ValxR := BeamEKG[BeamI].X;
    ValyQ := BeamEKG[BeamI - 10].Y;
    ValxQ := BeamEKG[BeamI - 10].X;
    ValyS := BeamEKG[BeamI + 7].Y;
    ValxS := BeamEKG[BeamI + 7].X;
   end;
   Inc(x,xinc);
  end;
```

```
    //=============================================
    // menggambar Trace pada
    //   unit frm EKG_Monitoring
    //=============================================
      frmEKGMonitoring.DataBeam(BeamEKG);
      frmEKGMonitoring.FindR(ValxR,ValyR);
      frmEKGMonitoring.FindS(ValxS,ValyS);
      frmEKGMonitoring.FindQ(ValxQ,ValyQ);
    //=============================================
    {selalu memasukan statemen selanjutnya - diperlukan
    Twavein untuk menambah buffer}
   inc(ProcessedBuffers);
   AddNextBuffer;
   //Jika Button CapFrame di Tekan
   //Twavein akan dihentikan dan MengCapture Frame
   If singleFrame and triggered and (frametoScan >= 0) then
   begin
    SetLength(savedFrameData, PlotPtsPerFrame);
    for i:=pstart to pstart + PlotPtsPerFrame-1 do
     savedFrameData[i-pstart]:=p^[i]-128;
     Stop;
     statusText.Caption  :=  'Status Dihentikan';
     btnCapFrame.Down:=False;
     SetButtonState;
     ShowStored;
   end;
  end;
end;

procedure Tfrm_UtamaEKG.Posterror(s:String);
begin
  Inc(errcount);
  if errcount < 100 then
  begin
    M_Status.Lines.Add(s);
    M_Status.Refresh;
  end
  else begin
    ShowMessage('Kesalahan Terhadap Loop, Proses Dihentikan');
    if  Assigned(WaveIN) then
      WaveIN.StopInput;
    end;
  statusText.Caption:='ERRORR - Coba Lagi';
end;

procedure Tfrm_UtamaEKG.setMaxPtsToAVG;
begin
  frameSize :=  frmEKGMonitoring.ImgLine.Width;
  nbrframes :=  framesPerBuffer{bufsize div framesize};
end;

//=============================================
//MEngubah TracKBar UpGain berdasarkan Edit Up Gain
//=============================================
procedure Tfrm_UtamaEKG.Ed_UpGainChange(Sender: TObject);
begin
  ChGain  :=  UpGain.Position;
end;

procedure Tfrm_UtamaEKG.Ed_SwepUDChange(Sender: TObject);
begin
  xinc  :=  SwepUD.Position;
  setMaxPtsToAVG;
  ShowScaleValue;
  SetOscState;
end;
//=============================================
//=============================================
// Menyimpan Image pada Img_hasilEKG
//=============================================
procedure Tfrm_UtamaEKG.saveImage;
var
  i:Integer;
  s:String;
  path:String;
begin
  //=====++++++++++=====//
  TextToImage;     //Copy Text Memo ke Image
  CopyImageEKG;    //Copy 3 Image menjadi 1 Image
  //=====++++++++++=====//
  i:=0;
  path  :=  ExtractFilePath(Application.ExeName);
  while (i<50) and FileExists(path+'EKG'+IntToStr(i)+'.bmp') do
    inc(i);
  s:=path+'EKG'+IntToStr(i)+'.bmp';
  if not FileExists(s) then
    With (img_hasilEKG.Picture.Bitmap) do
    begin
      PixelFormat :=  pf24bit;
      SaveToFile(s);
      Posterror('Gambar EKG '+s +'Tersimpan');
    end
    else
      Posterror('Gambar gagal disimpan - maksimal 50 gambar');
end;
procedure Tfrm_UtamaEKG.ChangeSampleRate;
begin
```

```
  if Assigned(WaveIN) then
   if (WaveIN.RecordActive) then
   begin
    WaveIN.StopInput;
    Application.ProcessMessages;
    Start;
   end
   else if not singleFrame then
    setup
   else
    FreeAndNil(wavein);
end;
procedure Tfrm_UtamaEKG.trOfsChChange(Sender: TObject);
begin
 SetOscState;
end;

procedure Tfrm_UtamaEKG.UpFocusClick(Sender: TObject; Button: TUDBtnType);
begin
 SetOscState;
end;

procedure Tfrm_UtamaEKG.UpBeamLightClick(Sender: TObject;
 Button: TUDBtnType);
begin
 SetOscState;
end;

procedure Tfrm_UtamaEKG.btn2205Click(Sender: TObject);
begin
 ChangeSampleRate;
end;

procedure Tfrm_UtamaEKG.btn11025Click(Sender: TObject);
begin
 ChangeSampleRate;
end;

procedure Tfrm_UtamaEKG.btn22050Click(Sender: TObject);
begin
 ChangeSampleRate;
end;

procedure Tfrm_UtamaEKG.btn44100Click(Sender: TObject);
begin
 ChangeSampleRate;
end;

procedure Tfrm_UtamaEKG.btnCHOnClick(Sender: TObject);
begin
 SetButtonState;
 SetOscState;
end;

procedure Tfrm_UtamaEKG.btnExpand1Click(Sender: TObject);
begin
 CenterAdjust;
end;

//SetButton STate=============================
procedure Tfrm_UtamaEKG.SetButtonState;
begin
 btnRun.Enabled := not btnCapFrame.Down;

 if not btnRun.Down then
  P_LED.Color := clNavy;

 if not btnCapFrame.Enabled and not btnRun.Enabled and not assigned(WaveIN) then
  btnRun.Enabled:=True;

// frm_EKGMonitor1.ChOn := btnCHOn.Down;
end;
//============================================
//SetOscState=================================
procedure Tfrm_UtamaEKG.SetOscState;
begin
 if SpLead1.Down then
  frmEKGMonitoring := frm_EKGMonitor1
 else if SpLead2.Down then
  frmEKGMonitoring := frm_EKGMonitor2
 else if SpLead3.Down then
  frmEKGMonitoring := frm_EKGMonitor3;
 frmEKGMonitoring.BeamLight := UpBeamLight.Position;  //beamlight
 frmEKGMonitoring.Focus     := UpFocus.Position;      //beamLebar
 frmEKGMonitoring.ChOn      := btnCHOn.Down;
end;
//============================================
//============================================
//  PRoses Gambar Data
//============================================
procedure Tfrm_UtamaEKG.DoDrawBeamText(Sender:TObject);
var
 s:String;
 expand:Integer;
begin
 if singleFrame and triggered then
 begin
  expand := 1;
```

```
  if expand > 1 then
    s := s + 'Expand : X'+IntToStr(expand);
 end;
 frmEKGMonitoring.ImgLine.Canvas.TextOut(10,10,s);
end;
//=========================================
//  PRoses StoredData
//=========================================
procedure Tfrm_UtamaEKG.ShowStored;
var
 myBeamEKG : array of TPoint;  {beam disini Size nya sesuai lebar dari Image.}
 myX,myY   : Integer;
 myX1,myY1   : Integer;
 myX2,myY2   : Integer;
 myX3,myY3   : Integer;
 myX4,myY4   : Integer;
 mxQ1,myQ1   : Integer;
 Loop     : Integer;
 Loop2    : Integer;
 Gain     : Double;
 ofs      : Integer;
begin
 if singleFrame then
 begin
  SetLength(myBeamEKG,high(BeamEKG));
myX:=BeamEKG[0].X;myX1:=BeamEKG[0].X;myX2:=BeamEKG[0].X;{myX3:=BeamEKG
[0].X;myX4:=BeamEKG[0].X};
myY:=BeamEKG[0].Y;myY1:=BeamEKG[0].Y;myY2:=BeamEKG[0].Y;{myY3:=BeamEKG
[0].Y;myY4:=BeamEKG[0].Y};
 end;
 Gain := 1;
 StoredExpand := 1;
 ofs :=0;
 //Panggil Ulang Beam
 for Loop:=0 to high(BeamEKG)-1 do
 begin
  myBeamEKG[Loop].X := (BeamEKG[Loop].X)*StoredExpand;
  myBeamEKG[Loop].Y := Trunc(BeamEKG[Loop].Y*Gain)-
StoredCHOff+trOfsCh.Position+ofs;
 end;
 frmEKGMonitoring.DataBeam(BeamEKG);
 frmEKGMonitoring.FindR(ValxR,ValyR);
 frmEKGMonitoring.FindS(ValxS,ValyS);
 frmEKGMonitoring.FindQ(ValxQ,ValyQ);
  {myQ1:=ValyR;
   for Loop2:=ValxR downto 0 do begin
    if myQ1 > myBeamEKG[Loop2].X then begin
     mxQ1:=myBeamEKG[Loop2].X;
     myQ1:=myBeamEKG[Loop2].Y;
    end;
   end;}
 //frmEKGMonitoring.FindS(ValxS,ValyS);
 //frmEKGMonitoring.FindQ(ValxQ,ValyQ);
 {frmEKGMonitoring.FindQ(mxQ1,myQ1); }
//=========================Menentukan Gelombang R=============================
 for Loop:=0 to Trunc(high(myBeamEKG) div 4) do
 begin
  if myY > myBeamEKG[Loop].Y then begin
   myX:=myBeamEKG[Loop].X;
   myY:=myBeamEKG[Loop].Y;
  end;
 end;
 frmEKGMonitoring.FindR(myX,myY);

 for Loop:= Trunc(high(myBeamEKG) div 4) to Trunc(high(myBeamEKG) div 2) do
 begin
  if myY1 > myBeamEKG[Loop].Y then begin
   myX1:=myBeamEKG[Loop].X;
   myY1:=myBeamEKG[Loop].Y;
  end;
 end;
 frmEKGMonitoring.FindR(myX1,myY1);

 for Loop:= Trunc(high(myBeamEKG) div 2) to Trunc(high(myBeamEKG) div 1) do
 begin
  if myY2 > myBeamEKG[Loop].Y then begin
   myX2:=myBeamEKG[Loop].X;
   myY2:=myBeamEKG[Loop].Y;
  end;
 end;
 frmEKGMonitoring.FindR(myX2,myY2);

{ for Loop:= Trunc(high(myBeamEKG) div 4) to Trunc(high(myBeamEKG) div 2) do
 begin
  if myY3 > myBeamEKG[Loop].Y then begin
   myX3:=myBeamEKG[Loop].X;
   myY3:=myBeamEKG[Loop].Y;
  end;
 end;
 frmEKGMonitoring.FindR(myX3,myY3);

 for Loop:= Trunc(high(myBeamEKG) div 2) to Trunc(high(myBeamEKG) div 1) do
 begin
  if myY4 > myBeamEKG[Loop].Y then begin
   myX4:=myBeamEKG[Loop].X;
   myY4:=myBeamEKG[Loop].Y;
  end;
 end;
```

```
 frmEKGMonitoring.FindR(myX4,myY4);  }
//=============================================================
 //Menampilkan Interval R-R Secara Automatis
 frmEKGMonitoring.FindValueAuto(myX,myY,myX1,myY1,M_info);
 //Menampilkan QRS secara Automatis
 frmEKGMonitoring.FindQRSAuto(ValxQ,ValyQ,ValxS,ValyS,M_info);
end;

procedure Tfrm_UtamaEKG.CenterAdjust;
var
 myCenter : Integer;
begin
 myCenter := trunc(frm_EKGMonitor1.ImgLine.Width/2);
end;

procedure Tfrm_UtamaEKG.Btn_SimpanImageClick(Sender: TObject);
begin
 saveImage;
end;

procedure Tfrm_UtamaEKG.SpLead1Click(Sender: TObject);
begin
 SetOscState;
end;

procedure Tfrm_UtamaEKG.SpLead2Click(Sender: TObject);
begin
 SetOscState;
end;

procedure Tfrm_UtamaEKG.SpLead3Click(Sender: TObject);
begin
 SetOscState;
end;

procedure Tfrm_UtamaEKG.Btn_PQRSTClick(Sender: TObject);
begin
 singleFrame:=True;
 ShowStored;
end;

//Copy Text Memo To Image
//=============================================
procedure Tfrm_UtamaEKG.TextToImage;
var
 Row:Integer;
begin
 img_hasilEKG.Picture:=nil;
  for Row:=0 to M_info.Lines.Count-1 do
   begin
    img_hasilEKG.Canvas.TextOut(435,(Row*12),M_info.Lines[Row]);
   end;
end;
//=============================================
//Copy ke 3 Image pada Form Monitoring EKG
//kedalam 1 Image=============================================
procedure Tfrm_UtamaEKG.CopyImageEKG;
var
 DstRect,DstRect2,DstRect3 : TRect;
 SrcRect,SrcRect2,SrcRect3 : TRect;
begin
 with img_hasilEKG.Canvas do
 begin
  CopyMode := cmSrcCopy;

  DstRect  := Rect(0{kiri}, 0{atas}, img_hasilEKG.Width-224 {kanan}, 161{bawah});
  DstRect2 := Rect(0{kiri}, 169{atas},img_hasilEKG.Width-224 {kanan}, 330{bawah});
  DstRect3 := Rect(0{kiri}, 337{atas},img_hasilEKG.Width-224 {kanan}, 499{bawah});

  SrcRect  := Rect(0, 0, frm_EKGMonitor1.ImgLine.Width,
frm_EKGMonitor1.ImgLine.Height);
  SrcRect2 := Rect(0, 0, frm_EKGMonitor2.ImgLine.Width,
frm_EKGMonitor2.ImgLine.Height);
  SrcRect3 := Rect(0, 0, frm_EKGMonitor3.ImgLine.Width,
frm_EKGMonitor3.ImgLine.Height);

  CopyRect(DstRect, frm_EKGMonitor1.ImgLine.Canvas, SrcRect);
  CopyRect(DstRect2, frm_EKGMonitor2.ImgLine.Canvas, SrcRect2);
  CopyRect(DstRect3, frm_EKGMonitor3.ImgLine.Canvas, SrcRect3);
 end;
end;
//=============================================================
//Menampilkan Gif Pada Form
//=============================================
procedure Tfrm_UtamaEKG.ShowGif;
var
 vFile: string;
 FGIFImage : TGIFImage;//deklarasikan GIf

begin
 vFile := ExtractFilePath(ParamStr(0))+'3dheart.gif';
 FGIFImage := TGIFImage.Create;
 FGIFImage.LoadFromFile(vFile);
 img_GifSHow.Width := FGIFImage.Width;
 img_GifSHow.Height := FGIFImage.Height;
 FGIFImage.Paint(img_GifSHow.Canvas, Rect(0, 0, FGIFImage.Width,
 FGIFImage.Height), FGIFImage.DrawOptions);
```

```
end;
//=============================================================
procedure Tfrm_UtamaEKG.frm_EKGMonitor1btn_Garis_HorizontalClick(
  Sender: TObject);
begin
  frm_EKGMonitor1.FindValue(M_info);
end;

procedure Tfrm_UtamaEKG.frm_EKGMonitor2btn_Garis_HorizontalClick(
  Sender: TObject);
begin
  frm_EKGMonitor2.FindValue(M_info);
end;

procedure Tfrm_UtamaEKG.frm_EKGMonitor3btn_Garis_HorizontalClick(
  Sender: TObject);
begin
  frm_EKGMonitor3.FindValue(M_info);
end;

procedure Tfrm_UtamaEKG.frm_EKGMonitor1btn_Garis_VertikalClick(
  Sender: TObject);
begin
  frm_EKGMonitor1.FindValueY(M_info);
end;

procedure Tfrm_UtamaEKG.frm_EKGMonitor2btn_Garis_VertikalClick(
  Sender: TObject);
begin
  frm_EKGMonitor2.FindValueY(M_info);
end;

procedure Tfrm_UtamaEKG.frm_EKGMonitor3btn_Garis_VertikalClick(
  Sender: TObject);
begin
  frm_EKGMonitor3.FindValueY(M_info);
end;

procedure Tfrm_UtamaEKG.btn_OpenImgEkgClick(Sender: TObject);
begin
  if OPDialog_Img.Execute then
    Preview_EKG.ShowFile(OPDialog_Img.FileName,1);
    img_hasilEKG.Picture.LoadFromFile(OPDialog_Img.FileName);
end;

procedure Tfrm_UtamaEKG.btn_ZoomInClick(Sender: TObject);
begin
  Preview_EKG.Zoom(1);
end;

procedure Tfrm_UtamaEKG.btn_ZoomOutClick(Sender: TObject);
begin
  Preview_EKG.Zoom(-1);
end;

procedure Tfrm_UtamaEKG.btn_ZoomPicClick(Sender: TObject);
begin
  if btn_ZoomPic.Down=False then begin
    frm_ZoomPic1.Timer_Zoom.Enabled := False;
    frm_ZoomPic1.Pnel_Zoom.Visible  := True;
  end
  else begin
    frm_ZoomPic1.Timer_Zoom.Enabled := True;
    frm_ZoomPic1.Pnel_Zoom.Visible  := False;
  end;
end;

{//==========cara mengambil data pada Memo
  //=======dengan cara Seleksi per Linenya==
var
  line,S1  :String;
begin
  line  :=  M_info.Lines[0];
  Delete(line,1,4);
  Edit1.Text:=line;
}
procedure Tfrm_UtamaEKG.btn_SP_RateChange(Sender: TObject);
begin
  txt_SP_rate.Text:= IntToStr(btn_SP_Rate.Position);
  ChangeSampleRate;
end;

procedure Tfrm_UtamaEKG.btn_CloseClick(Sender: TObject);
begin
  MessageDlg('.: Applikasi Berakhir :.',mtInformation,[mbOK],0);
  Close;
end;

procedure Tfrm_UtamaEKG.btn_FowardPageClick(Sender: TObject);
begin
  PageControl1.SelectNextPage(False);
end;

procedure Tfrm_UtamaEKG.btn_NextPageClick(Sender: TObject);
begin
  PageControl1.SelectNextPage(True);
end;
```

```
procedure Tfrm_UtamaEKG.btn_Simpan_EKG2Click(Sender: TObject);
begin
 saveImage;
end;

procedure Tfrm_UtamaEKG.btn_PrintClick(Sender: TObject);
begin
 frm_CetakEKG.QRImg_EKG.Picture.LoadFromFile(OPDialog_Img.FileName);
 frm_CetakEKG.QR_CetakEKG.Preview;
end;

procedure Tfrm_UtamaEKG.btn_Menu_PerbesarClick(Sender: TObject);
begin
 btn_ZoomPic.Down:=True;
 frm_ZoomPic1.Timer_Zoom.Enabled := True;
 frm_ZoomPic1.Pnel_Zoom.Visible := False;
end;

procedure Tfrm_UtamaEKG.btn_Menu_Perbesar2Click(Sender: TObject);
begin
 btn_ZoomPic.Down:=False;
 frm_ZoomPic1.Timer_Zoom.Enabled := False;
 frm_ZoomPic1.Pnel_Zoom.Visible := True;
end;

procedure Tfrm_UtamaEKG.btn_Menu_BukaEKGClick(Sender: TObject);
begin
 Tab_LihatEKG.Show;
 btn_OpenImgEkg.Click;
end;

procedure Tfrm_UtamaEKG.btn_Menu_SebelumnyaClick(Sender: TObject);
begin
 PageControl1.SelectNextPage(False);
end;

procedure Tfrm_UtamaEKG.btn_Menu_SelanjutnyaClick(Sender: TObject);
begin
 PageControl1.SelectNextPage(True);
end;

//Memindahkan Image kedalam Image PageControl
//=============================================
procedure Tfrm_UtamaEKG.BackCopyImageEKG;
var
 DstRect,DstRect2,DstRect3 : TRect;
 SrcRect,SrcRect2,SrcRect3 : TRect;
begin
 with frm_EKGMonitor1.ImgLine.Canvas do
 begin
  CopyMode := cmSrcCopy;

  DstRect  := Rect(0{kiri}, 0{atas}, frm_EKGMonitor1.ImgLine.Width {kanan},
frm_EKGMonitor1.Height {bawah});
  SrcRect  := Rect(0, 0, img_hasilEKG.Width - 224, 161{tinggi});
  CopyRect(DstRect, img_hasilEKG.Canvas , SrcRect);

 end;

 with frm_EKGMonitor2.ImgLine.Canvas do
 begin
  CopyMode := cmSrcCopy;

  DstRect2  := Rect(0{kiri}, 0{atas}, frm_EKGMonitor2.ImgLine.Width {kanan},
frm_EKGMonitor2.Height {bawah});
  SrcRect2  := Rect(0, 169, img_hasilEKG.Width - 224, 330{tinggi});
  CopyRect(DstRect2, img_hasilEKG.Canvas , SrcRect2);

 end;

 with frm_EKGMonitor3.ImgLine.Canvas do
 begin
  CopyMode := cmSrcCopy;

  DstRect3  := Rect(0{kiri}, 0{atas}, frm_EKGMonitor3.ImgLine.Width {kanan},
frm_EKGMonitor3.Height {bawah});
  SrcRect3  := Rect(0, 337, img_hasilEKG.Width - 224, 499{tinggi});
  CopyRect(DstRect3, img_hasilEKG.Canvas , SrcRect3);
 end;
end;

procedure Tfrm_UtamaEKG.btn_PraDiagnosaClick(Sender: TObject);
begin
 BackCopyImageEKG;
 Tab_EKGmonitoring.Show;
end;

procedure Tfrm_UtamaEKG.btn_OpenImageClick(Sender: TObject);
begin
  if OPDialog_Img.Execute then
  Preview_EKG.ShowFile(OPDialog_Img.FileName,1);
  img_hasilEKG.Picture.LoadFromFile(OPDialog_Img.FileName);
  Tab_LihatEKG.Show;
end;

procedure Tfrm_UtamaEKG.btn_OpenImgDiagnosaClick(Sender: TObject);
begin
 if OPDialog_Img.Execute then
```

```
  Preview_EKG.ShowFile(OPDialog_Img.FileName,1);
  img_hasilEKG.Picture.LoadFromFile(OPDialog_Img.FileName);
 btn_PraDiagnosa.Click;
end;

procedure Tfrm_UtamaEKG.RB_LeadIClick(Sender: TObject);
begin
 img_RA.Visible  := True;
 img_LA.Visible  := True;
 img_RL.Visible  := True;
 img_LL.Visible  := False;
end;

procedure Tfrm_UtamaEKG.RB_LeadIIClick(Sender: TObject);
begin
 img_RA.Visible  := True;
 img_LA.Visible  := False;
 img_RL.Visible  := True;
 img_LL.Visible  := True;
end;

procedure Tfrm_UtamaEKG.RB_LeadIIIClick(Sender: TObject);
begin
 img_RA.Visible  := False;
 img_LA.Visible  := True;
 img_RL.Visible  := True;
 img_LL.Visible  := True;
end;
end.


{
 => Form Visual Elektrokardiogram
//=============================================================//
 Aplikasi Untuk Menganalisa Sinyal EKG Berbasis
 Akuisisi Data Isyarat Elektris Jantung di buat
 untuk Menyelesaikan Skripsi.
 dibuat pada : Agustus 2010

 Oleh  : dwy_cah@yahoo.co.id
//=============================================================//
}
unit U_formEKG;
interface
uses
 Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
 Dialogs, ExtCtrls, Menus,StdCtrls;
type
 Tfrm_EKGMonitor = class(TFrame)
 Img_BackGround: TImage;
 ImgLine: TImage;
 PopM_EKG: TPopupMenu;
 btn_Garis: TMenuItem;
 btn_Gelombang: TMenuItem;
 btn_P: TMenuItem;
 btn_Q: TMenuItem;
 btn_R: TMenuItem;
 btn_S: TMenuItem;
 btn_T: TMenuItem;
 btn_Setting: TMenuItem;
 btn_SettingGaris: TMenuItem;
 btn_SettingHuruf: TMenuItem;
 btn_Garis1: TMenuItem;
 btn_Garis2: TMenuItem;
 btn_Garis3: TMenuItem;
 btn_Huruf1: TMenuItem;
 btn_Huruf2: TMenuItem;
 btn_Huruf3: TMenuItem;
 btn_Garis_Vertikal: TMenuItem;
 btn_Garis_Horizontal: TMenuItem;
 procedure ImgLineMouseMove(Sender: TObject; Shift: TShiftState; X,
  Y: Integer);
 procedure btn_PClick(Sender: TObject);
 procedure btn_QClick(Sender: TObject);
 procedure btn_RClick(Sender: TObject);
 procedure btn_SClick(Sender: TObject);
 procedure btn_TClick(Sender: TObject);
private
 { Private declarations }
 FScaleWarna:TColor;
 FPenLebar:Integer;
 FChOfsett:Integer;
 FChon:Boolean;
 FChGnd:Boolean;
 FCekLead:Integer;
 FWarnaScreen:TColor;
 FScreenTengah:Integer;
 FBeamWarna:TColor;
 FBeamLight:Integer;
 FAfterBeamDraw:TNotifyEvent;
 ValX1,ValX2  : Integer;
 ValY1,ValY2  : Integer;
 a        : Integer;
 Glmb1, Glmb2  : String;
 procedure DrawScale;
 procedure InitGraph;
 procedure DrawBeam;
 procedure setBeamLight(const Value:Integer);
```

```
    procedure setChOfsett(const Value:Integer);
    procedure setChOn(const Value:Boolean);
    procedure setChGnd(const Value:Boolean);
    procedure setWarnaScreen(const Value:TColor);
    procedure setBeamLebar(const Value:Integer);
    procedure setCheckLead(const Value:Integer);
  public
    { Public declarations }
    constructor Create(AOwner:TComponent);override;
    procedure Clear;
    property BeamLight:Integer read FBeamLight write setBeamLight;
    property Focus:Integer read FPenLebar write setBeamLebar;
    property ChOfSett:Integer read FChOfsett write setChOfsett;
    property ChOn:Boolean read FChOn write setChOn;
    property ChGnd:Boolean read FChGnd write setChGnd;
    procedure DataBeam(const ChPoints:array of TPoint);
    property WarnaScreen:TColor read FWarnaScreen write setWarnaScreen;
    property OnAfterBeamDraw:TNotifyEvent read FAfterBeamDraw write FAfterBeamDraw;
    property CekLead:Integer read FCekLead write setCheckLead;
    function FindR(X: integer; Y:integer): integer;
    procedure LineColor;
    function FindS(X, Y: integer): integer;
    procedure FindValue(var memo: Tmemo);
    procedure FindValueY(var memo: Tmemo);
    procedure FindValueAuto(var Xa1, Ya1, Xa2, Ya2: integer;
      var memo: Tmemo);
    function FindQ(X, Y: integer): integer;
    procedure FindQRSAuto(var Xa1, Ya1, Xa2, Ya2: integer; var memo:Tmemo);
  end;
implementation
uses Math;
{$R *.dfm}
procedure divmod(const d:Integer; const q:word; var result, remainder:Word);
begin
  result:=d div q;
  remainder:= d mod q;
end;

constructor Tfrm_EKGMonitor.Create(AOwner:TComponent);
begin
  inherited;
  a          := 0;
  FScaleWarna  := $9fa7e8;
  FChon        := True;
  FChOfsett    := 0;
  FWarnaScreen := $CDD1F5;
  FBeamWarna   := $00FF00;
  InitGraph;
end;

procedure Tfrm_EKGMonitor.Clear;
begin
  ImgLine.Canvas.Draw(0,0,Img_BackGround.Picture.Bitmap);
end;

procedure Tfrm_EKGMonitor.InitGraph;
begin
  FScreenTengah  := round(ImgLine.Height/2);
  Img_BackGround.Canvas.Brush.Color := FWarnaScreen;
  Img_BackGround.Canvas.FloodFill(60,60,FWarnaScreen,fsBorder);
  DrawScale;
  Clear;
end;
procedure Tfrm_EKGMonitor.setBeamLight(const Value:Integer);
begin
  FBeamWarna  := Value;
  DrawBeam;
end;

procedure Tfrm_EKGMonitor.setBeamLebar(const Value:Integer);
begin
  FPenLebar := Value;
  DrawBeam;
end;

procedure Tfrm_EKGMonitor.setWarnaScreen(const Value:TColor);
begin
  FWarnaScreen  :=  Value;
  Self.Color  := Value;
  InitGraph;
  Clear;
end;

procedure Tfrm_EKGMonitor.setChOfsett(const Value:Integer);
begin
  FChOfsett :=  Value;
  DrawBeam;
end;

procedure Tfrm_EKGMonitor.setChOn(const Value:Boolean);
begin
  FChon := Value;
  DrawBeam;
end;

procedure Tfrm_EKGMonitor.setChGnd(const Value:Boolean);
begin
  FChGnd  := Value;
```

```
  DrawBeam;
end;

procedure Tfrm_EKGMonitor.setCheckLead(const Value:Integer);
begin
  FCekLead := Value;
  DrawBeam;
end;

procedure Tfrm_EKGMonitor.DrawBeam;
begin
  Clear;
    if ChOn then
    begin
      ImgLine.Canvas.Pen.Color:=FBeamWarna;
      ImgLine.Canvas.Pen.Width:=FPenLebar;
     //=============================================
      if ChGnd then
      begin
        ImgLine.Canvas.MoveTo(0,FScreenTengah+ChOfSett);
        ImgLine.Canvas.LineTo(ImgLine.Width,FScreenTengah+ChOfSett);
      end;
    end;
end;

procedure  Tfrm_EKGMonitor.DrawScale;
var
  loop:Integer;
  h:Integer;
  a,b:Word;
  HTengah,VTengah:Integer;
  w:Double;
begin
  h:=3;
  w:=Img_BackGround.Width/134;
  with Img_BackGround do
  begin
    HTengah :=  FScreenTengah;
    VTengah :=  round(Img_BackGround.Width/2);
    Canvas.Pen.Width  := 1;
    Canvas.Pen.Color  :=FScaleWarna;
    //Buat Garis Horizontal==============
    for loop:=1 to 8 do
    begin
      Canvas.MoveTo(0,loop*h);
      Canvas.LineTo(Width,loop*h);
    end;
    for loop:=1 to 99 do
    begin
      DivMod(loop,5,a,b);
      if b = 0 then
      begin
        Canvas.Pen.Width:=2;
        Canvas.MoveTo(VTengah - width div 2,loop*3);
        Canvas.LineTo(VTengah + Width div 2,loop*3);
      end
      else
      begin
        Canvas.Pen.Width:=1;
        Canvas.MoveTo(VTengah - Width div 2,loop*3);
        Canvas.LineTo(VTengah + Width div 2,loop*3);
      end;
    end;
    //=============================================
    //Membuat Garis Vertikal==========================
    for loop:=1 to 8 do
    begin
      Canvas.MoveTo(round(loop*w*10),0);
      Canvas.LineTo(round(loop*w*10),Width);
    end;
    for loop:=1 to 140 do
    begin
      DivMod(loop,5,a,b);
      if b = 0 then
      begin
        Canvas.Pen.Width:=2;
        Canvas.MoveTo(round(loop*w),HTengah - Height div 2);
        Canvas.LineTo(round(loop*w),HTengah + Height div 2);
      end
      else
      begin
        Canvas.Pen.Width:=1;
        Canvas.MoveTo(round(loop*w),HTengah - Height div 2);
        Canvas.LineTo(round(loop*w),HTengah + Height div 2);
      end
    end;
    //=============================================
    if FWarnaScreen = $CDD1F5 then
      Canvas.Font.Color := clMaroon
    else
      Canvas.Font.Color := clSilver;

    Canvas.Font.Name  := 'Small Fonts';
    Canvas.Font.Size  := 6;
    Canvas.Brush.Style  := bsClear;
    Canvas.TextOut(10,Height-15,'EKG Lead');
  end;
end;
```

```
procedure Tfrm_EKGMonitor.DataBeam(const ChPoints:array of TPoint);
begin
 Clear;
 if ChOn then
 begin
  if FChGnd then
  begin
   ImgLine.Canvas.Pen.Color:=FBeamWarna;
   ImgLine.Canvas.MoveTo(0,FScreenTengah+ChOfSett);
   ImgLine.Canvas.LineTo(ImgLine.Width,FScreenTengah+ChOfSett);
  end
  else
  begin
   ImgLine.Canvas.Pen.Color := clNavy;
   ImgLine.Canvas.Polyline(ChPoints);
  end;
 if Assigned(FAfterBeamDraw) then
  FAfterBeamDraw(Self);
 end;
end;

procedure Tfrm_EKGMonitor.LineColor;
var
 LineWidTh,siZeFont : Integer;
begin
 if btn_Huruf2.Checked then
  siZeFont := 7
 else if btn_Huruf3.Checked then
  siZeFont := 8
 else
  siZeFont := 6;

 if btn_Garis2.Checked then
  LineWidTh := 2
 else if btn_Garis3.Checked then
  LineWidTh := 3
 else
  LineWidTh := 1;

 with ImgLine do begin
  Canvas.Font.Color  := clRed;
  Canvas.Font.Name   := 'Small Fonts';
  Canvas.Font.Size   := siZeFont;
  Canvas.Brush.Style := bsClear;
  Canvas.Pen.Color   := clGreen;
  Canvas.Pen.Width   := LineWidTh;
 end;
end;
```

```
function Tfrm_EKGMonitor.FindR(X: integer; Y :integer): integer;
begin
 LineColor;
 ImgLine.Canvas.TextOut(X,Y,'R');
end;

function Tfrm_EKGMonitor.FindS(X, Y: integer): integer;
begin
 LineColor;
 ImgLine.Canvas.TextOut(X,Y,'S');
end;

function Tfrm_EKGMonitor.FindQ(X, Y: integer): integer;
begin
 LineColor;
 ImgLine.Canvas.TextOut(X,Y,'Q');
end;

procedure Tfrm_EKGMonitor.ImgLineMouseMove(Sender: TObject;
 Shift: TShiftState; X, Y: Integer);
begin
 ImgLine.Hint:='X:'+IntToStr(X)+', Y:'+IntToStr(Y);
 btn_P.Caption:='P  X:'+IntToStr(X)+', Y:'+IntToStr(Y);
 btn_Q.Caption:='Q  X:'+IntToStr(X)+', Y:'+IntToStr(Y);
 btn_R.Caption:='R  X:'+IntToStr(X)+', Y:'+IntToStr(Y);
 btn_S.Caption:='S  X:'+IntToStr(X)+', Y:'+IntToStr(Y);
 btn_T.Caption:='T  X:'+IntToStr(X)+', Y:'+IntToStr(Y);
 if a = 0 then begin
  ValX1:=X;
  ValY1:=Y;
 end
 else if a = 1 then begin
  ValX2:=X;
  ValY2:=Y;
 end;
end;

procedure Tfrm_EKGMonitor.btn_PClick(Sender: TObject);
begin
 LineColor;
 if a = 0 then begin
  ImgLine.Canvas.TextOut(ValX1,ValY1,'P');
  Inc(a);
  Glmb1 :='P';
 end
 else if a = 1 then begin
  ImgLine.Canvas.TextOut(ValX2,ValY2,'P');
  inc(a);
```

```
    Glmb2 :='P';
  end;
end;

procedure Tfrm_EKGMonitor.btn_QClick(Sender: TObject);
begin
  LineColor;
  if a = 0 then begin
    ImgLine.Canvas.TextOut(ValX1,ValY1,'Q');
    Inc(a);
    Glmb1 :='Q';
  end
  else if a = 1 then begin
    ImgLine.Canvas.TextOut(ValX2,ValY2,'Q');
    inc(a);
    Glmb2 :='Q';
  end;
end;

procedure Tfrm_EKGMonitor.btn_RClick(Sender: TObject);
begin
  LineColor;
    if a = 0 then begin
    ImgLine.Canvas.TextOut(ValX1,ValY1,'R');
    Inc(a);
    Glmb1 :='R';
  end
  else if a = 1 then begin
    ImgLine.Canvas.TextOut(ValX2,ValY2,'R');
    inc(a);
    Glmb2 :='R';
  end;
end;

procedure Tfrm_EKGMonitor.btn_SClick(Sender: TObject);
begin
  LineColor;
  if a = 0 then begin
    ImgLine.Canvas.TextOut(ValX1,ValY1,'S');
    Inc(a);
    Glmb1 :='S';
  end
  else if a = 1 then begin
    ImgLine.Canvas.TextOut(ValX2,ValY2,'S');
    inc(a);
    Glmb2 :='S';
  end;
end;
procedure Tfrm_EKGMonitor.btn_TClick(Sender: TObject);
begin
  LineColor;
    if a = 0 then begin
    ImgLine.Canvas.TextOut(ValX1,ValY1,'T');
    Inc(a);
    Glmb1 :='T';
  end
  else if a = 1 then begin
    ImgLine.Canvas.TextOut(ValX2,ValY2,'T');
    inc(a);
    Glmb2 :='T';
  end;
end;

procedure Tfrm_EKGMonitor.FindValue(var memo: Tmemo);
var
  GarisMM   : double;
  G_TmpilX  : double;
  NilaiMM   : double;
  KonWaktu  : Double;
  KonVolt   : Double;
  NilaiFreq : Double;
  ketStts   : String; {keterangan Umur}
  ketJntung : String;
  ketRate   : String;
  SAnormal  : Double;

begin
  { Rumus Pixel Ke Milimeter
    ==========================
    ||1 Pixel = 3,39 Milimeter ||
    ==========================
    428 Pixel = 126 Milimeter
  }
  LineColor;
  ImgLine.Canvas.MoveTo(ValX1,ValY1);
  ImgLine.Canvas.LineTo(ValX2,ValY2);

  if ValX1 > ValX2 then
    GarisMM :=  (ValX1 - ValX2)
  else
    GarisMM :=  (ValX2 - ValX1);
  //G_TmpilX := round(GarisMM / 2) + ValX1;

  NilaiMM   := GarisMM / 3.39;
  KonVolt   := NilaiMM * 0.1;
  KonWaktu  := NilaiMM * 0.04;
  //ImgLine.Canvas.TextOut(G_TmpilX,ValY1,'=
+IntToStr(G_TmpilX)+'Pxl'+IntToStr(NilaiMM)+'mm');
```

```
//Menambahkan ke dalam memo Frm Utama
if ((Glmb1='Q') and (Glmb2='S')) then begin
  memo.Lines.Add('==== Q R S ====');
  memo.Lines.Add('X1 : '+IntToStr(ValX1)+', X2 : '+IntToStr(ValX2));
  memo.Lines.Add('Interval '+Glmb1+'R'+Glmb2+ Format(' = %5.2f mm',[NilaiMM]));
  memo.Lines.Add('Interval '+Glmb1+'R'+Glmb2+ Format(' = %5.2f mV',[KonVolt]));
  memo.Lines.Add('Interval '+Glmb1+'R'+Glmb2+ Format(' = %5.2f detik',[KonWaktu]));
  if ((KonWaktu >= 0.07) and (KonWaktu <= 0.10)) then
    ketJntung := 'NORMAL'
  else
    ketJntung := 'TIDAK-NORMAL';
  memo.Lines.Add('('+ ketJntung +')');
  memo.Lines.Add('=========******=========');
end
else
if ((Glmb1='S') and (Glmb2='T')) then begin
  memo.Lines.Add('==== S T ====');
  memo.Lines.Add('X1 : '+IntToStr(ValX1)+', X2 : '+IntToStr(ValX2));
  memo.Lines.Add('Interval '+Glmb1+Glmb2+ Format(' = %5.2f mm',[NilaiMM]));
  memo.Lines.Add('Interval '+Glmb1+Glmb2+ Format(' = %5.2f mV',[KonVolt]));
  memo.Lines.Add('Interval '+Glmb1+Glmb2+ Format(' = %5.2f detik',[KonWaktu]));
  memo.Lines.Add('=========******=========');
end
else
if ((Glmb1='P') and (Glmb2='R')) then begin
  memo.Lines.Add('==== P R ====');
  memo.Lines.Add('X1 : '+IntToStr(ValX1)+', X2 : '+IntToStr(ValX2));
  memo.Lines.Add('Interval '+Glmb1+Glmb2+ Format(' = %5.2f mm',[NilaiMM]));
  memo.Lines.Add('Interval '+Glmb1+Glmb2+ Format(' = %5.2f mV',[KonVolt]));
  memo.Lines.Add('Interval '+Glmb1+Glmb2+ Format(' = %5.2f detik',[KonWaktu]));
  if ((KonWaktu >= 0.15) and (KonWaktu <= 0.18)) then begin
    ketStts  := 'ANAK-ANAK';
    ketJntung := 'NORMAL';
  end
  else
  if ((KonWaktu > 0.18) and (KonWaktu < 0.21)) then begin
    ketStts  := 'DEWASA';
    ketJntung := 'NORMAL';
  end
  else begin
    ketStts  := '-';
    ketJntung := 'TIDAK-NORMAL';
  end;
  memo.Lines.Add('('+ ketJntung +')');
  memo.Lines.Add('=========******=========');
end
else begin
  NilaiFreq := 300 / (NilaiMM / 5);
  memo.Lines.Add('==== R R ====');
  memo.Lines.Add('X1 : '+IntToStr(ValX1)+', X2 : '+IntToStr(ValX2));
  memo.Lines.Add('Interval '+Glmb1+Glmb2+ Format(' = %5.2f mm',[NilaiMM]));
  memo.Lines.Add('Interval '+Glmb1+Glmb2+ Format(' = %5.2f mV',[KonVolt]));
  memo.Lines.Add('Interval '+Glmb1+Glmb2+ Format(' = %5.2f detik',[KonWaktu]));
  memo.Lines.Add('Frekuensi '+Glmb1+Glmb2+ Format(' = %5f x/menit',[NilaiFreq]));

  //Menentukan Pace Maker
  SAnormal := NilaiFreq;
  if ((SAnormal >= 70) and (SAnormal <= 80)) then
    memo.Lines.Add('Pace Maker (Simpul SA) = NORMAL')
  else
    memo.Lines.Add('Pace Maker (Simpul SA) = TDK NORMAL');

  //Menentukan RATE
  if NilaiFreq >= 350 then
    ketRate := 'FIBRILASI'
  else if ((NilaiFreq >= 250) and (NilaiFreq < 350)) then
    ketRate := 'FLUTTER'
  else if ((NilaiFreq >= 140) and (NilaiFreq < 250)) then
    ketRate := 'TAKIKARDIA ABNORMAL'
  else if ((NilaiFreq >= 100) and (NilaiFreq < 140)) then
    ketRate := '(SINUS) TAKIKARDIA'
  else if ((NilaiFreq >= 60) and (NilaiFreq < 100)) then
    ketRate := '(SINUS) NORMAL'
  else if (NilaiFreq < 60 )then
    ketRate := '(SINUS) BRADIKARDIA'
  else
    ketRate := 'TIDAK-TERDETEKSI!';

  memo.Lines.Add('Rate = '+ketRate);
  memo.Lines.Add('=========******=========');
 end;

 a:=0;
end;

procedure Tfrm_EKGMonitor.FindValueY(var memo: Tmemo);
var
 GarisMM  : double;
 NilaiMM  : double;
 KonWaktu  : Double;
 KonVolt  : Double;
begin
 { Rumus Pixel Ke Milimeter
   ==========================================
   ||1 Pixel = 3,39 Milimeter ||
   ==========================================
   428 Pixel = 126 Milimeter
```

```
 }
 LineColor;
 ImgLine.Canvas.MoveTo(ValX1,ValY1);
 ImgLine.Canvas.LineTo(ValX2,ValY2);

 if ValY1 > ValY2 then
  GarisMM := (ValY1 - ValY2)
 else
  GarisMM := (ValY2 - ValY1);

 NilaiMM  := GarisMM / 3.39;
 KonVolt  := NilaiMM * 0.1;
 KonWaktu  := NilaiMM * 0.04;

 //Menambahkan ke dalam memo Frm Utama
  memo.Lines.Add('======'+Glmb1+' '+Glmb2+'======');
  memo.Lines.Add('Y1 : '+IntToStr(ValY1)+', Y2 : '+IntToStr(ValY2));
  memo.Lines.Add('Interval '+Glmb1+Glmb2+ Format(' = %5.2f mm',[NilaiMM]));
  memo.Lines.Add('Interval '+Glmb1+Glmb2+ Format(' = %5.2f mV',[KonVolt]));
  memo.Lines.Add('Interval '+Glmb1+Glmb2+ Format(' = %5.2f detik',[KonWaktu]));
  memo.Lines.Add('=======******=======');
 a:=0;

end;

procedure Tfrm_EKGMonitor.FindValueAuto(var Xa1, Ya1, Xa2, Ya2: integer;
 var memo: Tmemo);
var
 GarisMM  : double;
 NilaiMM  : double;
 KonWaktu  : Double;
 KonVolt  : Double;
 NilaiFreq : Double;
 FreqNadi  : Double;
 SAnormal  : Double;
 ketRate  : String;
begin
 { Rumus Pixel Ke Milimeter
  ==========================
  ||1 Pixel = 3,39 Milimeter ||
  ==========================
  428 Pixel = 126 Milimeter
 }
 LineColor;
 ImgLine.Canvas.MoveTo(Xa1,Ya1);
 ImgLine.Canvas.LineTo(Xa2,Ya2);

 if Xa1 > Xa2 then
  GarisMM := (Xa1 - Xa2)
 else
  GarisMM := (Xa2 - Xa1);

 NilaiMM  := GarisMM / 3.39;
 KonVolt  := NilaiMM * 0.1;
 KonWaktu  := NilaiMM * 0.04;
 if NilaiMM > 0 then begin
  NilaiFreq := 300 / ((NilaiMM / 5));
  FreqNadi  := 1500 / NilaiMM;
 end
 else begin
  NilaiFreq := 0;
  FreqNadi  := 0;
 end;

 //Menambahkan ke dalam memo Frm Utama
  memo.Lines.Add('======  R R  ======');
  memo.Lines.Add('X1 : '+IntToStr(Xa1)+', X2 : '+IntToStr(Xa2));
  memo.Lines.Add('Interval RR'+ Format(' = %5.2f mm',[NilaiMM]));
  memo.Lines.Add('Interval RR'+ Format(' = %5.2f mV',[KonVolt]));
  memo.Lines.Add('Interval RR'+ Format(' = %5.2f detik',[KonWaktu]));
  memo.Lines.Add('Frekuensi RR'+ Format(' = %5f x/menit', [NilaiFreq]));
  memo.Lines.Add('Frekuensi Nadi'+ Format(' = %5f x/menit', [FreqNadi]));

   //menentukan Pace-Maker
   SAnormal  := NilaiFreq;
   if ((SAnormal >= 70) and (SAnormal <= 80)) then
    memo.Lines.Add('Pace Maker (Simpul SA) = NORMAL')
   else
    memo.Lines.Add('Pace Maker (Simpul SA) = TIDAK-NORMAL');

   //Menentukan RATE
   if NilaiFreq >= 350 then
    ketRate :=  'FIBRILASI'
   else if ((NilaiFreq >= 250) and (NilaiFreq < 350)) then
    ketRate :=  'FLUTTER'
   else if ((NilaiFreq >= 140) and (NilaiFreq < 250)) then
    ketRate := 'TAKIKARDIA ABNORMAL'
   else if ((NilaiFreq >= 100) and (NilaiFreq < 140)) then
    ketRate :=  '(SINUS) TAKIKARDIA'
   else if ((NilaiFreq >= 60) and (NilaiFreq < 100)) then
    ketRate :=  '(SINUS) NORMAL'
   else if (NilaiFreq < 60) then
    ketRate :=  '(SINUS) BRADIKARDIA'
   else
    ketRate :=  'Tidak Terdeteksi!';

  memo.Lines.Add('Rate = '+ketRate);
  memo.Lines.Add('=======******=======');
```

```
 a:=0;
end;

procedure Tfrm_EKGMonitor.FindQRSAuto(var Xa1, Ya1, Xa2, Ya2: integer;var
memo:Tmemo);
var
 GarisMM   : double;
 NilaiMM   : double;
 KonWaktu  : Double;
 KonVolt   : Double;
 NilaiFreq : Double;
 ketJntung : String;
begin
 { Rumus Pixel Ke Milimeter
  ==========================
  ||1 Pixel = 3,39 Milimeter ||
  ==========================
  428 Pixel = 126 Milimeter
 }
 LineColor;
 ImgLine.Canvas.MoveTo(Xa1,Ya1);
 ImgLine.Canvas.LineTo(Xa2,Ya2);

 if Xa1 > Xa2 then
  GarisMM :=  (Xa1 - Xa2)
 else
  GarisMM :=  (Xa2 - Xa1);

 NilaiMM   := GarisMM / 3.39;
 KonVolt   := NilaiMM * 0.1;
 KonWaktu  := NilaiMM * 0.04;
 if NilaiMM > 0 then
  NilaiFreq := 300 / ((NilaiMM / 5))
 else
  NilaiFreq := 0;

  memo.Lines.Add('====  Q R S  ====');
  memo.Lines.Add('X1 : '+IntToStr(Xa1)+', X2 : '+IntToStr(Xa2));
  memo.Lines.Add('Interval QRS '+ Format(' = %5.2f mm',[NilaiMM]));
  memo.Lines.Add('Interval QRS '+ Format(' = %5.2f mV',[KonVolt]));
  memo.Lines.Add('Interval QRS '+ Format(' = %5.2f detik',[KonWaktu]));
  if ((KonWaktu >= 0.07) and(KonWaktu <= 0.10)) then
   ketJntung := 'NORMAL'
  else
   ketJntung := 'TIDAK-NORMAL';
  memo.Lines.Add('('+ ketJntung +')');
  memo.Lines.Add('=========******=========');
 a:=0;
end;

end.

{
 => Form Tools Pembesaran Visual Elektrokardiogram
//=============================================================//
 Aplikasi Untuk Menganalisa Sinyal EKG Berbasis
 Akuisisi Data Isyarat Elektris Jantung di buat
 untuk Menyelesaikan Skripsi.
 dibuat pada : Agustus 2010

 Oleh  : dwy_cah@yahoo.co.id
//=============================================================//
}
unit U_ZoomPic;
interface
uses
 Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
 Dialogs, ExtCtrls;
type
 Tfrm_ZoomPic = class(TFrame)
  Timer_Zoom: TTimer;
  Img_Zoom: TImage;
  Pnel_Zoom: TPanel;
  procedure Timer_ZoomTimer(Sender: TObject);
 private
  { Private declarations }
 public
  { Public declarations }
 end;
implementation
uses DateUtils;
{$R *.dfm}
procedure Tfrm_ZoomPic.Timer_ZoomTimer(Sender: TObject);
var
 Srect, Drect, PosForme  : TRect;
 iLebar, iTinggi, DmX, DmY :  integer;
 iTmpX, iTmpY   : Real;
 C : TCanvas;
 hDekstop  : HWND;
 Kursor   : TPoint;
begin
 if not IsIconic(Application.Handle) then begin
  hDekstop  :=  GetDesktopWindow;
  GetCursorPos(Kursor);
   PosForme  :=  Rect(Left,Top,Left+Width,Top+Height);
   if not PtInRect(PosForme,Kursor) then begin
    if Img_Zoom.Visible=False then Img_Zoom.Visible:=True;
    iLebar  :=  Img_Zoom.Width;
    iTinggi :=  Img_Zoom.Height;
```

```
      Drect  := Rect(0,0,iLebar,iTinggi);
      iTmpX := iLebar / 4;
      iTmpY := iTinggi / 4;
      Srect := Rect(Kursor.X,Kursor.Y,Kursor.X,Kursor.Y);
      InflateRect(Srect,round(iTmpX),round(iTmpY));
      if Srect.Left<0 then OffsetRect(Srect,-Srect.Left,0);
      if Srect.Top<0  then OffsetRect(Srect,0,-Srect.Top);
      if Srect.Right>Screen.Width then
        OffsetRect(Srect,-(Srect.Right-Screen.Width),0);
      if Srect.Bottom>Screen.Height then
        OffsetRect(Srect,0,-(Srect.Bottom-Screen.Height));
    C := TCanvas.Create;
    try
      C.Handle := GetDC(GetDesktopWindow);
      Img_Zoom.Canvas.CopyRect(Drect,C,Srect);
    finally
      ReleaseDC(hDekstop, C.Handle);
      C.Free;
    end;
    //Membuat Cursor
    With Img_Zoom.Canvas do begin
      Pen.Color:=clGreen;
      Pen.Width:=4;
      MoveTo((iLebar div 2)-1,(iTinggi div 2));
      LineTo((iLebar div 2)+2,(iTinggi div 2));
    end;
    Application.ProcessMessages;
   end;
  end;
end;
end.

{
  => Form Laporan Hasil Rekaman Data
//=============================================================//
  Aplikasi Untuk Menganalisa Sinyal EKG Berbasis
  Akuisisi Data Isyarat Elektris Jantung di buat
  untuk Menyelesaikan Skripsi.
  dibuat pada : Agustus 2010

  Oleh  : dwy_cah@yahoo.co.id
//=============================================================//
}
unit U_PrintEKG;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, QRCtrls, QuickRpt, ExtCtrls, GIFImage;
type
  Tfrm_CetakEKG = class(TForm)
    QR_CetakEKG: TQuickRep;
    TitleBand1: TQRBand;
    PageHeaderBand1: TQRBand;
    PageFooterBand1: TQRBand;
    DetailBand1: TQRBand;
    QRImg_EKG: TQRImage;
    QRLabel1: TQRLabel;
    QRSysData1: TQRSysData;
    QRLabel2: TQRLabel;
    QRShape1: TQRShape;
    QRShape2: TQRShape;
    QRShape3: TQRShape;
    QRShape4: TQRShape;
    QRLabel3: TQRLabel;
    QRLabel4: TQRLabel;
    QRImage1: TQRImage;
  private
    { Private declarations }
  public
    { Public declarations }
  end;
var
  frm_CetakEKG: Tfrm_CetakEKG;
implementation
{$R *.dfm}
end.

{==============
  "Form Wave"
==============}
unit U_Wave;
Interface
Uses Windows, MMSystem, SysUtils;
Const MAX_BUFFERS = 8;
type
  {Event definitions}
  TBufferFullEvent = procedure (Sender : TObject; Data : Pointer; Size : longint) of object;
  TErrorEvent=procedure(s:string) of object; {OnError}
  TWaveIn = class(TObject)
     Constructor Create(newFormHandle:HWnd; BfSize, newTotalBuffers : Integer);
     Destructor  Destroy;     Override;
    private
    fBufferSize:Integer; // Requested size of buffer
    BufIndex:Integer;
    pWaveHeader:Array [0..MAX_BUFFERS-1] of PWAVEHDR;
    dwByteDataSize:DWORD;
    dwTotalWaveSize:DWORD;
    bDeviceOpen:Boolean;
```

```
    FormHandle:HWND;
    Function  InitWaveHeaders : Boolean;
    Function  AllocPCMBuffers : Boolean;
    Procedure FreePCMBuffers;
    Function  AllocWaveFormatEx : Boolean;
    Procedure FreeWaveFormatEx;
    Function  AllocWaveHeaders : Boolean;
    Procedure FreeWaveHeader;
    Procedure CloseWaveDevice;
    procedure fposterror(s:string);
    public { Public declarations }
      RecordActive:Boolean;  {we are recording}
      ptrWaveFmtEx:PWaveFormatEx;
      WaveBufSize:Integer;    // Size aligned to nBlockAlign Field
      IniTWavein:Boolean;
      RecErrorMessage:String;
      QueuedBuffers,ProcessedBuffers:Integer;
      pWaveBuffer:Array [0..MAX_BUFFERS-1] of pointer {lpstr}{gdd};
      WaveIn:HWAVEIN;  { Wavedevice handle }
      {OnBufferFull:TBufferFullEvent;}
      OnError:TErrorEvent;
      TotalBuffers:Integer;
      Function  AddNextBuffer : Boolean;
      Procedure StopInput;
      Function  StartInput:Boolean;
      Function  SetupInput:Boolean;
  end;
implementation
{*************** GetErrorText *********}
Function GetErrorText(iErr : Integer) : String;
{Put the WaveIn error messages in a string format. }
{ iErr is the error number }
Var
  PlayInErrorMsg:Array [0..255] of Char;
Begin
  waveInGetErrorText(iErr,PlayInErrorMsg,255);
  result := StrPas(PlayInErrorMsg);
End;
{*************** InitWaveHeaders **********************}
Function TWaveIn.AllocWaveFormatEx : Boolean;
{ Allocate the largest format size required from installed ACM's        }
begin
  new(ptrWavefmtEx);
  If (ptrWaveFmtEx = Nil) Then
  begin
    fposterror('Error locking WaveFormatEx memory');
    AllocWaveFormatEx := False;
    Exit;
  end;
  { initialize the format to standard PCM }
  {ZeroMemory( lptrwavefmtex, maxFmtSize );}
  with ptrwavefmtex^ do
  begin
    wFormatTag := WAVE_FORMAT_PCM;
    nChannels := 1;
    nSamplesPerSec := 44100;
    nBlockAlign := 1;
    wBitsPerSample := 8;
    nAvgBytesPerSec := nSamplesPerSec*(wBitsPerSample div 8)*nChannels;
    cbSize := 0;
  end;
  { Success, go home }
  AllocWaveFormatEx := True;
end;
{******************* InitWaveHeaders ******************}
function TWavein.InitWaveHeaders : Boolean;
{Allocate memory, zero out wave headers and initialize                }
var  i:Integer;
begin
  { make the wave buffer size a multiple of the block align... }
  WaveBufSize := fBufferSize - (fBufferSize mod ptrwavefmtex.nBlockAlign);
  {Set the wave headers }
  for i := 0 to TotalBuffers-1 Do
  with pWaveHeader[i]^ Do
  begin
    lpData := pWaveBuffer[i];      {address of the waveform buffer}
    dwBufferLength := WaveBufSize; {length, in bytes, of the buffer}
    dwBytesRecorded := 0;
    dwUser := 0;                   {32 bits of user data}
    dwFlags := 0;
    dwLoops := 0;
    lpNext := Nil;
    reserved := 0;
  end;
  result:=TRUE;
end;
{***************** AllocWaveHeader ******************}
Function TWavein.AllocWaveHeaders : Boolean;
{Allocate and lock header memory                                      }
var i : Integer;
begin
  for i := 0 to TotalBuffers-1 Do
  begin
    new(pwaveheader[i]);
    If (pwaveheader[i] = Nil ) Then
    begin
      { NOTE: This could lead to a memory leak, fix someday }
      fposterror('Could not lock header memory for recording');
```

```
    AllocWaveHeaders := FALSE;
    Exit;
   end;
  end;
  AllocWaveHeaders := TRUE;
end;
{************ FreeWaveHeader ******************}
Procedure TWavein.FreeWaveHeader;
{Free up the memory AllocWaveHeaders allocated}
var i:Integer;
begin
 For i := 0 to TotalBuffers-1 Do
 begin
  dispose(pwaveheader[i]);
  pwaveheader[i]:=nil;
 end;
end;
{*************** AllocPCMBuffers *************}
function TWavein.AllocPCMBuffers : Boolean;
{Allocate and lock the waveform memory}
var i:Integer;
begin
 for i:=0 to TotalBuffers-1 Do
 begin
  getmem(pWaveBuffer[i],fBufferSize);
  If (pWaveBuffer[i] = Nil) Then
  begin
   { Possible Memory Leak here }
   fposterror('Error Locking wave buffer memory');
   AllocPCMBuffers := False;
   Exit;
  end;
  pWaveHeader[i].lpData := pWaveBuffer[i];
 end;
 AllocPCMBuffers := TRUE;
end;
{***************** FreePCMBuffers *************}
Procedure TWavein.FreePCMBuffers;
{Free up the memory AllocPCMBuffers used.                    }
var  i : Integer;
begin
 for i := 0 to TotalBuffers-1 do   freemem(pwavebuffer[i]);
end;
{**************** FreeWaveFormatEx *************}
  Procedure TWavein.FreeWaveFormatEx;
{ Free up the ExFormat headers                    }
begin
 If (ptrWaveFmtEx = Nil) Then Exit;
 dispose(ptrWavefmtEx);
 ptrWaveFmtEx := Nil;
end;
{************* TWavein.Create ******************}
  Constructor TWavein.Create(NewFormHandle:HWnd;
                             BFSize, newTotalBuffers : Integer);
{ Set up the wave headers, initializes the data pointers and
  allocate the sampling buffers
  BFSize is the size of the buffer in BYTES
}
Var
 i : Integer;
begin
 Inherited Create;
 formhandle:=newformhandle;
 for i := 0 to newTotalBuffers-1 Do    {gdd}
 begin
  pWaveBuffer[i] := Nil;
  ptrWaveFmtEx := Nil;
 end;
 fBufferSize := BFSize;
 totalbuffers:=newtotalbuffers;
 {allocate memory for wave format structure }
 if(Not AllocWaveFormatEx) Then
 begin
  IniTWavein := FALSE;
  exit;
 end;
 {find a device compatible with the available wave characteristics }
 if (waveInGetNumDevs < 1 ) Then
 begin
  fposterror('No wave audio recording devices found');
  IniTWavein := FALSE;
  exit;
 end;
 {allocate the wave header memory }
 if (Not AllocWaveHeaders) Then
 begin
  IniTWavein := FALSE;
  exit;
 end;
 {allocate the wave data buffer memory }
 if (Not AllocPCMBuffers)  Then
 begin
  IniTWavein := FALSE;
  exit;
 end;
 InitWaveIn := TRUE;
end;
{***************** Destroy ***************}
```

```
destructor TWavein.Destroy;
{Free up memory allocated by IniTWavein }
begin
 FreeWaveFormatEx;
 FreePCMBuffers;
 FreeWaveHeader;
 inherited Destroy;
end;
{************ CloseWaveDevice *****************}
Procedure TWavein.CloseWaveDevice;
{ Close the waveform device}
var
 i, ierr : Integer;
 s:string;
begin
 { if the device is already closed, just return }
 if (Not bDeviceOpen) Then Exit;
 {unprepare the headers }
 for i := 0 to TotalBuffers-1 Do
 begin
  ierr:=waveInUnprepareHeader(WaveIn, pWaveHeader[i], sizeof(TWAVEHDR));
  if ierr<> 0 then
  begin
   case ierr of
    MMSYSERR_INVALHANDLE:s:=' Specified device handle is invalid ';
    MMSYSERR_NODRIVER:s:='    No device driver is present';
    MMSYSERR_NOMEM:s:=' Unable to allocate or lock memory';
    WAVERR_STILLPLAYING:s:='Buffer still being propcessed';
    else s:='Uknown Error';
   end;
   fposterror('Error in waveInUnprepareHeader ' + s);
  end;
 end;
 { save the total size recorded and update the display }
 dwTotalwavesize := dwBytedatasize;
 { close the wave input device }
 if (waveInClose(WaveIn) <> 0) Then
 fposterror('Error closing input device');
 {tell this function we are now closed }
 bDeviceOpen := FALSE;
end;
{************ StopInput ********************}
Procedure TWavein.StopInput;
{Stop recording and close the device}
var iErr : Integer;
begin
 If (Not bDeviceOpen) Then Exit;
 RecordActive := False;
 {ierr:=waveinstop(wavein);}
 ierr := waveinreset(wavein);{hangs if callback procedure is used}
 { stop recording and return queued buffers }
 if (iErr <> 0) then fposterror('Error in waveInReset');
 CloseWaveDevice;
end;
{******************** AddNextBuffer *************}
Function TWavein.AddNextBuffer : Boolean;
{ Add a buffer to the input queue and toggles buffer index.      }
Var iErr : Integer;
begin
 result:=true;
 if not recordactive then exit; {can't add if recording has been stopped}
 { queue the buffer for input }
 iErr := waveInAddBuffer(WaveIn, pwaveheader[bufindex], sizeof(TWAVEHDR));
 If (iErr <> 0) Then
 begin
  StopInput;
  fposterror('Error adding buffer, recording stopped: ' + GetErrorText(iErr));
  result:=FALSE;
  exit;
 end;
 { toggle for next buffer }
 bufindex := (bufindex+1) mod TotalBuffers;
 QueuedBuffers := QueuedBuffers + 1;
 result:=TRUE;
end;
{******************* StartInput *****************}
Function TWaveIn.StartInput : Boolean;
{Start recording}
var  iErr, i : Integer;
begin
 result:=false;
 { start recording to first buffer }
 iErr := WaveInStart(WaveIn);
 If (iErr <> 0) Then
 begin
  CloseWaveDevice;
  fposterror('Error starting wavein, close attempted: ' + GetErrorText(iErr));
 end
 else
 begin
  RecordActive := TRUE;
  { queue the next buffers }
  For i := 1 to TotalBuffers-1 Do
  If (Not AddNextBuffer) Then
  begin
   fposterror('AddNextBuffer failed from open');
   exit;
  end;
```

```
  result:=True;
 end;
end;
{*************** SetupInput ******************}
Function TWavein.SetupInput:Boolean;
{open the device}
var  iErr, i : Integer;
begin
 if recordactive then StopInput;
 if bdeviceopen then closewavedevice;  {already open - close it first}
 dwTotalwavesize := 0;
 dwBytedatasize := 0;
 bufindex := 0;
 ProcessedBuffers := 0;
 QueuedBuffers := 0;
 { open the device for recording }
 (*iErr := waveInOpen(@WaveIn, WAVE_MAPPER, ptrWaveFmtEx,
      Integer(@BufferDoneCallBack),
          Integer(self), CALLBACK_FUNCTION {+ WAVE_ALLOWSYNC} );
 *)
 iErr := waveInOpen(@WaveIn, WAVE_MAPPER, ptrWaveFmtEx,
              formhandle, 0, CALLBACK_WINDOW);
 if (iErr <> 0) then
 begin
  fposterror('Could not open the input device for recording: ' + ^M
           +' '+ GetErrorText(iErr));
  result:=FALSE;
  exit;
 end;
 { tell CloseWaveDeviceRecord() that the device is open }
 bDeviceOpen := TRUE;
 { prepare the headers }
 InitWaveHeaders;
 For i := 0 to TotalBuffers-1 Do
 begin
  iErr := waveInPrepareHeader( WaveIn, pWaveHeader[I], sizeof(TWAVEHDR));
  If (iErr <> 0) Then
  begin
   CloseWaveDevice;
   fposterror('Error preparing header for recording: ' + ^M +
              GetErrorText(iErr));
   result:=FALSE;
   Exit;
  end;
 end;
 {add the first buffer }
 If (Not AddNextBuffer) then result:=FALSE
 else result:=TRUE;
end;

{*************** fposterror ***********}
procedure TWavein.fposterror(s:string);
begin
 RecErrorMessage:=s;
 if assigned(OnError) then Onerror(s);
end;
end.
```