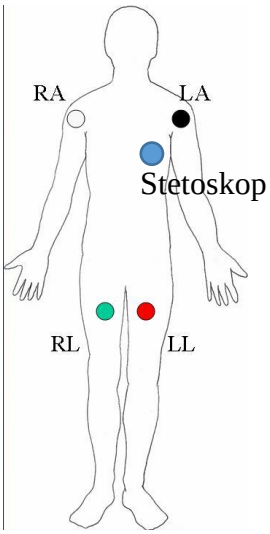
LAMPIRAN

1. Standar Operasional Prosedur

STANDAR OPERASIONAL PROSEDUR
ELECTROPHONOCARDIOGRAPH BERBASIS RASPBERRY PI

PERINGATAN : Jangan mengoperasikan Electrophonocardiograph berbasis raspberry pi sebelum membaca seluruh prosedur pemakaian ini.

<table>
<tr><td>Pengertian</td><td>Electrophonocardiograph berbasis raspberry pi merupakan gabungan dari 2 alat yaitu Electrocardiograph (ECG) dan Phonocardiograph (PCG)</td></tr>
<tr><td>Tujuan</td><td>1. Untuk mengetahui aktivitas listrik jantung
2. Untuk mengetahui suara detak jantung</td></tr>
<tr><td>Kelengkapan</td><td>1. Unit Electrophonocardiograph berbasis raspberry pi
2. Kabel input ECG
3. Stetoskop yang dilengkapi Mic Condensor</td></tr>
<tr><td>Persiapan</td><td>1. Siapkan elektroda disposable.
2. Atur posisi pasien dengan posisi terlentang datar.
3. Buka dan longgarkan pakaian pasien bagian atas, bila pasien memakai jam tangah, gelang ataupun logam lain agar dilepas.
4. Bersihkan kotoran dengan menggunakan kapas pada daerah dada.
5. Pasangkan elektroda pada permukaan kulit pasien sesuai dengan metode segitiga einthoven serta pasangkan stetoskop pada bagian pulmonary arteri atau antara rusuk ke 4 dan ke 6 dekat dibawah puting susu. atau lihat pada gambar.
 

Right Arm = RA
Left Armg = LA
Right Leg = RL
Left Leg = LL
6. Pasangkan kabel input ECG dengan elektroda.</td></tr>
</table>

	7. Pasangkan kabel input PCG dan ECG pada alat.
Prosedur	1. Hidupkan mesin Elektrophonocardiograph berbasis raspberry pi dengan menekan tombol power. 2. Tunggu sampai proses booting selesai, ditandai dengan tampilan user interface dari ECG dan PCG. 3. Putar volume sesuai kebutuhan untuk mendengarkan bunyi detak jantung pasien. 4. Tekan tombol REC untuk menyimpan hasil perekaman sinyal ECG dan PCG ke dalam bentuk file. 5. Masukkan nomor pasien. 6. Tekan Save untuk memulai menyimpan. 7. Tombol OPEN berfungsi membuka form untuk menampilkan hasil penyimpanan sinyal ECG dan PCG. 8. Tekan Browse untuk mencari file yang akan dibuka, dicopy, atau dihapus yaitu dengan alamat directory /home/pi/DataPasien. 9. Tekan file yang akan dibuka kemudian tekan tombol OK. 10. Tekan tombol Play untuk memulai membuka file. 11. Scrolbar digunakan untuk menggeser tampilan grafik. 12. Tekan tombol Save Graphic untuk menyimpan gambar per parameter. 13. Tekan tombol Screenshot untuk melakukan screenshot layar. 14. Alamat directory file graphic /home/pi/Datagambar. 15. Alamat directory file screenshot /home/pi/Screenshot 16. Untuk mematikan, atau merestart tekan tombol Shutdown.
Penyimpanan	1. Setelah selesai matikan alat Electrophonocardiograph berbasis raspberry pi dengan cara shutdown. 2. Tunggu beberapa detik setelah layar mati sebelum mematikan power. 3. Putar volume ke titik terendah atau putar sampai penuh ke kanan. 4. Lepas kabel input ECG yang terhubung ke pasien. 5. Lepas elektroda yang menempel pada pasien. 6. Lepas kabel input ECG dan PCG dari alat. 7. Rapihkan kabel input ECG dan PCG. 8. Simpan unit serta kabel ditempat yang kering.

2. Program Arduino

```
int command;
void setup() {
  // put your setup code here, to run once:
  Serial.begin (115200);
}
void loop() {
  // put your main code here, to run repeatedly:
  double lead_1 = analogRead(A0);
  double lead_2 = analogRead(A1);
  double lead_3 = analogRead(A2);
  double PCG = analogRead(A3);
  if(Serial.available() > 0)
  {
    if(Serial.peek() == 'c')
    {
      Serial.read();
      command = Serial.parseInt();
    }
    while(Serial.available() > 0)
    {
      Serial.read();
    }
  }
  if(command == 1)
  {
    Serial.print(lead_1);
    Serial.println("A");
    Serial.print(lead_2);
    Serial.println("B");
    Serial.print(lead_3);
    Serial.println("C");
    Serial.print(PCG);
    Serial.println("D");
  }
}
```

3. Program QT

1. Arduino.h

```
1. #ifndef ARDUINO_H
2. #define ARDUINO_H
3. #include <QMainWindow>
4. #include <QDebug>
5. #include <QByteArray> //untuk menerima data serial
6. #include <QStringList> //untuk menampung data berupa type
   data string
7. #include <QList>
8. #include <QSerialPortInfo>
9. #include "qextserialport.h"
10.     #include <QTextStream>
```

```
11.     #include <QFile>
12.     class Arduino : public QMainWindow
13.     {
14.         Q_OBJECT
15.     public:
16.         explicit Arduino(QWidget *parent = 0);
17.         QextSerialPort *arduinoserial;
18.     signals:
19.         //signal dengan overloading QString (menampung data
    berupa string)
20.         void Serial_lead1(double);
21.         void Serial_lead2(double);
22.         void Serial_lead3(double);
23.         void Serial_PCG(double);
24.         void alldata(QString);
25.         void data_realtime(QString);
26.     private slots:
27.         //slots yang digunakan untuk menerima data serial
28.         void serialReceiver();
29.     private:
30.         static const quint16 arduino_uno_vendor_id =6790;
    //nano //Asli 1027  //kloning 6790
31.         static const quint16 arduino_uno_product_id =
    29987; //nano //Asli 24577 //kloning 29987
32.     };
33.     #endif // ARDUINO_H
```

2. dialog_warning.h

```
1. #ifndef DIALOG_WARNING_H
2. #define DIALOG_WARNING_H
3. #include <QDialog>
4. #include <QProcess>
5. namespace Ui {
6. class Dialog_Warning;
7. }
8. class Dialog_Warning : public QDialog
9. {
10.         Q_OBJECT
11.     public:
12.         explicit Dialog_Warning(QWidget *parent = 0);
13.         ~Dialog_Warning();
14.     private slots:
15.         //Slot untuk pushbutton
16.         void on_pushButton_clicked();
17.         void on_pushButton_3_clicked();
18.         void on_pushButton_2_clicked();
19.     private:
20.         Ui::Dialog_Warning *ui;
21.     };
22.     #endif // DIALOG_WARNING_H
```

3. formbrowserfolder.h

```
1. #ifndef FORMBROWSERFOLDER_H
2. #define FORMBROWSERFOLDER_H
3. #include <QDialog>
```

```
4. #include <QFileSystemModel>
5. #include <QFile>
6. #include <QDebug>
7. namespace Ui {
8. class FormBrowserFolder;
9. }
10.     class FormBrowserFolder : public QDialog
11.     {
12.         Q_OBJECT
13.     public:
14.         explicit FormBrowserFolder(QWidget *parent = 0);
15.         ~FormBrowserFolder();
16.         //Inisialisasi class
17.         QFile *qfile;
18.         QFileSystemModel *dirmodel;
19.         QFileSystemModel *filemodel;
20.     signals:
21.         //Inisialisasi Signal
22.         void sendspath(QString);
23.         void sendfilename(QString);
24.     private slots:
25.         void on_pushButton_clicked();
26.         void on_treeView_clicked(const QModelIndex &index);
27.         void on_listView_clicked(const QModelIndex &index);
28.         void on_pushButton_2_clicked();
29.         void on_pushButton_3_clicked();
30.         void on_pushButton_4_clicked();
31.     private:
32.         QString sPathfile;
33.         Ui::FormBrowserFolder *ui;
34.     };
35.     #endif // FORMBROWSERFOLDER_H
```

4. Formopenfile.h

```
1. #ifndef FORMOPENFILE_H
2. #define FORMOPENFILE_H
3. #include <QDialog>
4. #include "formbrowserfolder.h"
5. #include <QTimer>
6. #include <QFile>
7. #include <QDebug>
8. #include <QMessageBox>
9. #include <QDataStream>
10.     #include <QVector>
11.     #include <QPixmap>
12.     namespace Ui {
13.     class FormOpenFile;
14.     }
15.     class FormOpenFile : public QDialog
16.     {
17.         Q_OBJECT
18.     public:
19.         explicit FormOpenFile(QWidget *parent = 0);
20.         ~FormOpenFile();
21.         FormBrowserFolder *browserfolder;
22.         QTimer *timerplot;
23.         QFile *file;
```

```
24.     private slots:
25.         void receivefilename(QString);
26.         void makeplot_openfile_1();
27.         void makeplot_openfile_2();
28.         void makeplot_openfile_3();
29.         void makeplot_openfile_4();
30.         void timerplotting();
31.         void receivespathfile(QString);
32.         void on_pushButton_clicked();
33.         void on_pushButton_2_clicked();
34.         void on_pushButton_3_clicked();
35.         void horzScrollBarChanged(int);
36.         void on_pushButton_4_clicked();
37.
38.         void on_pushButton_5_clicked();
39.     private:
40.         QString sPath="";
41.         QString readfile;
42.         QString serialBuffer="";
43.         QStringList list;
44.         QString readData;
45.         int i=0;
46.         Ui::FormOpenFile *ui;
47.     };
48.     #endif // FORMOPENFILE_H
```

5. Formpatientdatasave.h

```
1. #ifndef FORMPATIENTDATASAVE_H
2. #define FORMPATIENTDATASAVE_H
3. #include <QDialog>
4. #include <QMessageBox>
5. namespace Ui {
6. class FormPatientDataSave;
7. }
8. class FormPatientDataSave : public QDialog
9. {
10.         Q_OBJECT
11.     public:
12.         explicit FormPatientDataSave(QWidget *parent = 0);
13.         ~FormPatientDataSave();
14.     signals:
15.         void sendnumberpatient(QString);
16.     private slots:
17.         void on_pushButton_clicked();
18.         void on_pushButton_12_clicked();
19.         void on_pushButton_3_clicked();
20.         void on_pushButton_4_clicked();
21.         void on_pushButton_5_clicked();
22.         void on_pushButton_6_clicked();
23.         void on_pushButton_7_clicked();
24.         void on_pushButton_8_clicked();
25.         void on_pushButton_9_clicked();
26.         void on_pushButton_10_clicked();
27.         void on_pushButton_11_clicked();
28.         void on_pushButton_13_clicked();
29.         void on_pushButton_2_clicked();
30.     private:
```

```
31.         Ui::FormPatientDataSave *ui;
32.     };
33.     #endif // FORMPATIENTDATASAVE_H
```

6. Mainwindow.h

```
1. #ifndef MAINWINDOW_H
2. #define MAINWINDOW_H
3. #include <QMainWindow>
4. #include "qcustomplot.h"
5. #include <QTimer>
6. #include "arduino.h"
7. #include "formpatientdatasave.h"
8. #include <QFile>
9. #include <QMessageBox>
10.     #include <QTextStream>
11.     #include "formopenfile.h"
12.     #include <QBuffer>
13.     #include <QDataStream>
14.     #include <QByteArray>
15.     #include <dialog_warning.h>
16.     #include <QImage>
17.     #include <QPixmap>
18.     namespace Ui {
19.     class MainWindow;
20.     }
21.     class MainWindow : public QMainWindow
22.     {
23.         Q_OBJECT
24.     public:
25.         explicit MainWindow(QWidget *parent = 0);
26.         ~MainWindow();
27.         Dialog_Warning *dw;
28.         FormOpenFile *openfile;
29.         FormPatientDataSave *patientdatasave;
30.         QTimer *qtimer; //untuk pemanggilan fungsi QTimer
31.         QTimer *qtimer_savedata;
32.         Arduino *ar; //penamaan class untuk proses
   pemanggilan class tersebut
33.     private slots:
34.         /*Slots untuk fungsi setting grafik*/
35.         void makePlot();
36.         void makePlot_2();
37.         void makePlot_3();
38.         void makePlot_4();
39.         /*Slots untuk mengatur plotter*/
40.         void myTimer();
41.         /*SLots untuk menerima data lead*/
42.         void receivealldata(QString);
43.         void receivenumberpatient(QString);
44.         void timersavedata();
45.         void receive_data_realtime(QString);
46.         void on_pushButton_2_clicked();
47.         void on_pushButton_clicked();
48.         void on_pushButton_3_clicked();
49.         void on_pushButton_4_clicked();
50.         void on_pushButton_5_clicked();
51.     private:
```

```
52.         int detik=0;
53.         int menit=0;
54.         QString patientnumber="";
55.         Ui::MainWindow *ui;
56.     };
57.     #endif // MAINWINDOW_H
```

7. Arduino.cpp

```
1. #include "arduino.h"
2. QString saving;
3. QString serialBuffer;
4. QStringList list;
5. Arduino::Arduino(QWidget *parent) : QMainWindow(parent)
6. {
7.     qDebug()<<"Number of ports: "
   <<QSerialPortInfo::availablePorts().length() <<"\n";
   //menampilkan port yang terpakai
8.
9.     foreach (const QSerialPortInfo &serialPortInfo,
   QSerialPortInfo::availablePorts())
10.         {
11.             qDebug()<<"Description: "
   <<serialPortInfo.description() << "\n"; //menampilkan
   deskripsi port
12.             qDebug()<<"Has vendor id?: "
   <<serialPortInfo.hasVendorIdentifier()<<"\n"; //menampilkan
   vendor
13.             qDebug()<<"Vendor ID: "
   <<serialPortInfo.vendorIdentifier() <<"\n"; //menampilkan
   nomor vendor
14.             qDebug()<<"Has product id?: "
   <<serialPortInfo.hasProductIdentifier() <<"\n";
15.             qDebug()<<"Product ID: "
   <<serialPortInfo.productIdentifier()<<"\n"; //menampilkan
   nomor product
16.         }
17.         bool arduino_is_available = true; //available
   arduino kondisi awal "false"
18.         QString arduino_uno_port_name; //variabel dengan
   type data string
19.
20.         foreach (const QSerialPortInfo &serialPortInfo,
   QSerialPortInfo::availablePorts())
21.         {
22.             if(serialPortInfo.hasProductIdentifier() &&
   serialPortInfo.hasVendorIdentifier()) //apabila arduino sudah
   diketahui nomor produk dan nomor vendor
23.             {
24.                 if((serialPortInfo.productIdentifier() ==
   arduino_uno_product_id) //apabila nomor produk yang diketahui
   dan yang telah di setting sesuai
25.                         &&(serialPortInfo.vendorIdentifier()
   == arduino_uno_vendor_id)) //apabila nomor vendor yang
   diketahui dan yang telah disetting sesuai
26.                 {
27.                     arduino_is_available = true; //apabila
   kondisinya sesuai maka available arduino menjadi "true"
```

```
28.                     arduino_uno_port_name =
   serialPortInfo.portName(); //variabel arduino_uno_port_name
   akan diganti dengan yang telah diketahui
29.                 }
30.             }
31.         }
32.         arduinoserial = new
   QextSerialPort(QextSerialPort::EventDriven);
33.         if(arduino_is_available) //apabila available arduino
   == true?
34.         {
35.             qDebug()<<"Found the arduino port...\n";
36.             arduinoserial->setPortName("ttyS0");//ttyS0
37.             arduinoserial->open(QextSerialPort::ReadWrite);
38.             arduinoserial-
   >setBaudRate((BaudRateType(BAUD115200)));
39.             arduinoserial->setDataBits(DATA_8);
40.             arduinoserial->setFlowControl(FLOW_OFF);
41.             arduinoserial->setParity(PAR_NONE);
42.             arduinoserial->setStopBits(STOP_1);
43.          //   arduinoserial->waitForReadyRead(5000);
44.
   QObject::connect(arduinoserial,SIGNAL(readyRead()),this,SLOT(
   serialReceiver()));
45.             arduinoserial->write("c1");
46.           //  arduinoserial->setRts(true);
47.         }
48.         else
49.         {
50.             qDebug()<< "Couldn't find correct port for the
   arduino.\n"; //apabila arduino beravaliable "false" maka akan
   ditampilkan
51.         }
52.     }
53.     void Arduino::serialReceiver() //fungsi
54.     {
55.         QString readData;
56.         QByteArray serialData;
57.         if(arduinoserial->bytesAvailable())
58.         {
59.             serialData = arduinoserial->readAll();
60.             QString dataserial ;
61.             dataserial
   =QString::fromStdString(serialData.toStdString());
62.             emit alldata(dataserial);
63.             serialBuffer =
   QString::fromStdString(serialData.toStdString());
64.             emit data_realtime(serialBuffer);
65.                //serialBuffer="";
66.                 //arduinoserial->waitForReadyRead(1000);
67.         }
```

8. Dialog_warning.cpp

```
1. #include "dialog_warning.h"
2. #include "ui_dialog_warning.h"
3. #include <QDebug>
4. Dialog_Warning::Dialog_Warning(QWidget *parent) :
```

```
5.      QDialog(parent),
6.      ui(new Ui::Dialog_Warning)
7. {
8.      ui->setupUi(this);
9. }
10.     Dialog_Warning::~Dialog_Warning()
11.     {
12.         delete ui;
13.     }
14.     void Dialog_Warning::on_pushButton_clicked()
15.     {
16.          QProcess process;
17.          process.startDetached("shutdown -P now");
18.     }
19.
20.     void Dialog_Warning::on_pushButton_3_clicked()
21.     {
22.         this->close();
23.     }
24.     void Dialog_Warning::on_pushButton_2_clicked()
25.     {
26.         QProcess process2;
27.         process2.startDetached("reboot");
28.     }
```

9. Formbrowserfolder.cpp

```
1. #include "formbrowserfolder.h"
2. #include "ui_formbrowserfolder.h"
3. QString sPath_copy,sPath_paste,filename,sPath_to_del;
4. QString name_file_search;
5. FormBrowserFolder::FormBrowserFolder(QWidget *parent) :
6.      QDialog(parent),
7.      ui(new Ui::FormBrowserFolder)
8. {
9.      ui->setupUi(this);
10.         qfile = new QFile(this);
11.         QString sPath = "/home/pi/DataPasien/";
12.         dirmodel = new QFileSystemModel(this);
13.         dirmodel->setFilter(QDir::NoDotAndDotDot |
   QDir::AllDirs);
14.         dirmodel->setRootPath(sPath);
15.         ui->treeView->setModel(dirmodel);
16.         filemodel = new QFileSystemModel(this);
17.         filemodel->setFilter(QDir::NoDotAndDotDot |
   QDir::Files);
18.         filemodel->setRootPath(sPath);
19.         ui->listView->setModel(filemodel);
20.         ui->pushButton_3->setEnabled(false);
21.     }
22.     FormBrowserFolder::~FormBrowserFolder()
23.     {
24.         delete ui;
25.     }
26.
27.     void FormBrowserFolder::on_pushButton_clicked()
28.     {
29.         this->close();
```

```
30.     }
31.
32.     void FormBrowserFolder::on_treeView_clicked(const
   QModelIndex &index)
33.     {
34.         QString sPath = dirmodel-
   >fileInfo(index).absoluteFilePath();
35.         ui->listView->setRootIndex(filemodel-
   >setRootPath(sPath));
36.         if(sPath_copy!="")
37.         {
38.             sPath_paste = sPath+"/"+filename;
39.         }
40.     }
41.
42.     void FormBrowserFolder::on_listView_clicked(const
   QModelIndex &index)
43.     {
44.         sPathfile = filemodel-
   >fileInfo(index).absoluteFilePath();
45.         sPath_to_del=sPathfile;
46.         sPath_copy = sPathfile;
47.         QStringList search_name_file = sPathfile.split("/");
48.         for(int i=0;i<search_name_file.count();i++)
49.         {
50.             name_file_search = search_name_file.value(i);
51.             if(name_file_search.contains("txt"))
52.             {
53.                 filename=search_name_file.value(i);
54.             }
55.         }
56.         ui->label->setText(sPathfile);
57.         ui->pushButton_3->setEnabled(true);
58.     }
59.
60.     void FormBrowserFolder::on_pushButton_2_clicked()
61.     {
62.         emit sendspath(sPathfile);
63.         emit sendfilename(filename);
64.         this->close();
65.     }
66.
67.     void FormBrowserFolder::on_pushButton_3_clicked()
68.     {
69.         if(sPath_copy!="")
70.         {
71.             ui->listView->setEnabled(false);
72.             ui->pushButton_3->setText("Paste");
73.         }
74.         if(sPath_paste!="")
75.         {
76.             qfile->copy(sPath_copy,sPath_paste);
77.
   qDebug()<<"copy"<<sPath_copy<<"paste"<<sPath_paste;
78.             ui->listView->setEnabled(true);
79.             ui->pushButton_3->setEnabled(false);
80.             sPath_copy="";
81.             sPath_paste="";
82.             ui->pushButton_3->setText("Copy");
```

```
83.         }
84.     }
85.
86.     void FormBrowserFolder::on_pushButton_4_clicked()
87.     {
88.         QFile delete_file(sPath_to_del);
89.         delete_file.remove();
90.     }
```

10. Formopenfile.cpp

```
1. #include "formopenfile.h"
2. #include "ui_formopenfile.h"
3. QStringList listsavedata,listdataopen;
4. int i=0;
5. int plot1,plot2,plot3,plot4;
6. int hitung=0;
7. double plottertime;
8. int second_number_file;
9. int average_plot_open;
10.     bool pulse_high_open=false;
11.     bool pulse_low_open=false;
12.     int a_open;
13.     int search_open;
14.     int nomorgambar_open;
15.     volatile int rate[10];
16.     volatile unsigned long sampleCounter = 0;
17.     volatile unsigned long lastBeatTime = 0;
18.     volatile int R = 82;
19.     volatile int T = 82;
20.     volatile int thresh = 103;
21.     volatile int amp = 50;
22.     volatile bool firstBeat = true;
23.     volatile bool secondBeat = false;
24.     volatile int Signal;
25.     volatile int IBI = 600;
26.     volatile bool Pulse = false;
27.     volatile int BPM;
28.     QString readdatasave;
29.     QString number_file_second;
30.     FormOpenFile::FormOpenFile(QWidget *parent) :
31.         QDialog(parent),
32.         ui(new Ui::FormOpenFile)
33.     {
34.         ui->setupUi(this);
35.         makeplot_openfile_1();
36.         makeplot_openfile_2();
37.         makeplot_openfile_3();
38.         makeplot_openfile_4();
39.         browserfolder = new FormBrowserFolder(this);
40.
    connect(browserfolder,SIGNAL(sendspath(QString)),this,SLOT(re
    ceivespathfile(QString)));
41.
    connect(browserfolder,SIGNAL(sendfilename(QString)),this,SLOT
    (receivefilename(QString)));
42.
43.         timerplot = new QTimer(this);
```

```
44.
   connect(timerplot,SIGNAL(timeout()),this,SLOT(timerplotting()
   ));
45.         timerplot->start(0);
46.         ui->horizontalScrollBar->setRange(0,200);
47.         connect(ui->horizontalScrollBar,
   SIGNAL(valueChanged(int)), this,
   SLOT(horzScrollBarChanged(int)));
48.         ui->progressBar->setValue(0);
49.         ui->progressBar->setEnabled(false);
50.         ui->pushButton_3->setEnabled(false);
51.     }
52.     FormOpenFile::~FormOpenFile()
53.     {
54.         delete ui;
55.     }
56.     void FormOpenFile::receivefilename(QString y)
57.     {
58.         if(y.contains(".txt"))
59.         {
60.             y=y.replace(".txt","").trimmed();
61.         }
62.         ui->label_2->setAlignment(Qt::AlignCenter);
63.         ui->label_2->setText(y);
64.     }
65.     void FormOpenFile::makeplot_openfile_1()
66.     {
67.         //Untuk memberi warna pada plot
68.         ui->customPlot_openfile->addGraph();
69.         ui->customPlot_openfile->graph(0)-
   >setPen(QPen(QColor(0,0,0)));
70.         //Untuk memberi range X dan Y
71.         ui->customPlot_openfile->xAxis->setRange(0,60);
72.         ui->customPlot_openfile->yAxis->setRange(0,1100);
73.         ui->customPlot_openfile->xAxis2->setVisible(true);
74.         ui->customPlot_openfile->xAxis2-
   >setTickLabels(false);
75.         ui->customPlot_openfile->yAxis2->setVisible(true);
76.         ui->customPlot_openfile->yAxis2-
   >setTickLabels(false);
77.         connect(ui->customPlot_openfile->xAxis,
   SIGNAL(rangeChanged(QCPRange)),ui->customPlot_openfile-
   >xAxis2, SLOT(setRange(QCPRange)));
78.         connect(ui->customPlot_openfile->yAxis,
   SIGNAL(rangeChanged(QCPRange)),ui->customPlot_openfile-
   >yAxis2, SLOT(setRange(QCPRange)));
79.         ui->customPlot_openfile->xAxis2-
   >setBasePen(QPen(Qt::black,1));
80.         ui->customPlot_openfile->yAxis2-
   >setBasePen(QPen(Qt::black,1));
81.         ui->customPlot_openfile->xAxis2-
   >setTickPen(Qt::NoPen);
82.         ui->customPlot_openfile->yAxis2-
   >setTickPen(Qt::NoPen);
83.         ui->customPlot_openfile->xAxis2-
   >setSubTickPen(Qt::NoPen);
84.         ui->customPlot_openfile->yAxis2-
   >setSubTickPen(Qt::NoPen);
```

```
85.         ui->customPlot_openfile->xAxis-
   >setBasePen(QPen(Qt::black, 1));
86.         ui->customPlot_openfile->yAxis-
   >setBasePen(QPen(Qt::black, 1));
87.         ui->customPlot_openfile->xAxis-
   >setTickPen(Qt::NoPen);
88.         ui->customPlot_openfile->yAxis-
   >setTickPen(Qt::NoPen);
89.         ui->customPlot_openfile->xAxis-
   >setSubTickPen(Qt::NoPen);
90.         ui->customPlot_openfile->yAxis-
   >setSubTickPen(Qt::NoPen);
91.         ui->customPlot_openfile->xAxis-
   >setTickLabels(false);
92.         ui->customPlot_openfile->yAxis-
   >setTickLabels(false);
93.         ui->customPlot_openfile->xAxis->grid()-
   >setPen(QPen(Qt::NoPen));
94.         ui->customPlot_openfile->yAxis->grid()-
   >setPen(QPen(Qt::NoPen));
95.         ui->customPlot_openfile->xAxis->grid()-
   >setSubGridPen(Qt::NoPen);
96.         ui->customPlot_openfile->yAxis->grid()-
   >setSubGridPen(Qt::NoPen);
97.         ui->customPlot_openfile->xAxis->grid()-
   >setSubGridVisible(false);
98.         ui->customPlot_openfile->yAxis->grid()-
   >setSubGridVisible(false);
99.         ui->customPlot_openfile->xAxis->grid()-
   >setZeroLinePen(Qt::NoPen);
100.        ui->customPlot_openfile->yAxis->grid()-
   >setZeroLinePen(Qt::NoPen);
101.        QPixmap pixmap;
102.        pixmap.load("/home/pi/ECG_Layout_fix.png");
103.        ui->customPlot_openfile-
   >setBackground(pixmap.scaled(599,115),Qt::KeepAspectRatioByEx
   panding);
104.        QLinearGradient plotGradient;
105.        plotGradient.setStart(0, 0);
106.        plotGradient.setFinalStop(0, 350);
107.        plotGradient.setColorAt(0, QColor(255,255,255));
108.       // plotGradient.setColorAt(1, QColor(50, 50, 50));
109.        ui->customPlot_openfile-
   >setBackground(plotGradient);
110.    }
111.    void FormOpenFile::makeplot_openfile_2()
112.    {
113.        //Untuk memberi warna pada plot
114.        ui->customPlot_openfile_2->addGraph();
115.        ui->customPlot_openfile_2->graph(0)-
   >setPen(QPen(QColor(0,0,0)));
116.        //Untuk memberi range X dan Y
117.        ui->customPlot_openfile_2->xAxis->setRange(0,60);
118.        ui->customPlot_openfile_2->yAxis->setRange(0,1055);
119.        ui->customPlot_openfile_2->xAxis2->setVisible(true);
120.        ui->customPlot_openfile_2->xAxis2-
   >setTickLabels(false);
121.        ui->customPlot_openfile_2->yAxis2->setVisible(true);
```

```
122.        ui->customPlot_openfile_2->yAxis2-
   >setTickLabels(false);
123.        connect(ui->customPlot_openfile_2->xAxis,
   SIGNAL(rangeChanged(QCPRange)),ui->customPlot_openfile_2-
   >xAxis2, SLOT(setRange(QCPRange)));
124.        connect(ui->customPlot_openfile_2->yAxis,
   SIGNAL(rangeChanged(QCPRange)),ui->customPlot_openfile_2-
   >yAxis2, SLOT(setRange(QCPRange)));
125.        ui->customPlot_openfile_2->xAxis2-
   >setBasePen(QPen(Qt::black,1));
126.        ui->customPlot_openfile_2->yAxis2-
   >setBasePen(QPen(Qt::black,1));
127.        ui->customPlot_openfile_2->xAxis2-
   >setTickPen(Qt::NoPen);
128.        ui->customPlot_openfile_2->yAxis2-
   >setTickPen(Qt::NoPen);
129.        ui->customPlot_openfile_2->xAxis2-
   >setSubTickPen(Qt::NoPen);
130.        ui->customPlot_openfile_2->yAxis2-
   >setSubTickPen(Qt::NoPen);
131.        ui->customPlot_openfile_2->xAxis-
   >setBasePen(QPen(Qt::black, 1));
132.        ui->customPlot_openfile_2->yAxis-
   >setBasePen(QPen(Qt::black, 1));
133.        ui->customPlot_openfile_2->xAxis-
   >setTickPen(Qt::NoPen);
134.        ui->customPlot_openfile_2->yAxis-
   >setTickPen(Qt::NoPen);
135.        ui->customPlot_openfile_2->xAxis-
   >setSubTickPen(Qt::NoPen);
136.        ui->customPlot_openfile_2->yAxis-
   >setSubTickPen(Qt::NoPen);
137.        ui->customPlot_openfile_2->xAxis-
   >setTickLabels(false);
138.        ui->customPlot_openfile_2->yAxis-
   >setTickLabels(false);
139.        ui->customPlot_openfile_2->xAxis->grid()-
   >setPen(QPen(Qt::NoPen));
140.        ui->customPlot_openfile_2->yAxis->grid()-
   >setPen(QPen(Qt::NoPen));
141.        ui->customPlot_openfile_2->xAxis->grid()-
   >setSubGridPen(Qt::NoPen);
142.        ui->customPlot_openfile_2->yAxis->grid()-
   >setSubGridPen(Qt::NoPen);
143.        ui->customPlot_openfile_2->xAxis->grid()-
   >setSubGridVisible(false);
144.        ui->customPlot_openfile_2->yAxis->grid()-
   >setSubGridVisible(false);
145.        ui->customPlot_openfile_2->xAxis->grid()-
   >setZeroLinePen(Qt::NoPen);
146.        ui->customPlot_openfile_2->yAxis->grid()-
   >setZeroLinePen(Qt::NoPen);
147.        QPixmap pixmap;
148.        pixmap.load("/home/pi/ECG_Layout_fix.png");
149.        ui->customPlot_openfile_2-
   >setBackground(pixmap.scaled(599,115),Qt::KeepAspectRatioByEx
   panding);
150.        QLinearGradient plotGradient;
151.        plotGradient.setStart(0, 0);
```

```
152.        plotGradient.setFinalStop(0, 350);
153.        plotGradient.setColorAt(0, QColor(255, 255, 255));
154.       // plotGradient.setColorAt(1, QColor(50, 50, 50));
155.        ui->customPlot_openfile_2-
   >setBackground(plotGradient);
156.    }
157.    void FormOpenFile::makeplot_openfile_3()
158.    {
159.        //Untuk memberi warna pada plot
160.        ui->customPlot_openfile_3->addGraph();
161.        ui->customPlot_openfile_3->graph(0)-
   >setPen(QPen(QColor(0,0,0)));
162.        //Untuk memberi range X dan Y
163.        ui->customPlot_openfile_3->xAxis->setRange(0,60);
164.        ui->customPlot_openfile_3->yAxis->setRange(0,850);
165.        ui->customPlot_openfile_3->xAxis2->setVisible(true);
166.        ui->customPlot_openfile_3->xAxis2-
   >setTickLabels(false);
167.        ui->customPlot_openfile_3->yAxis2->setVisible(true);
168.        ui->customPlot_openfile_3->yAxis2-
   >setTickLabels(false);
169.        connect(ui->customPlot_openfile_3->xAxis,
   SIGNAL(rangeChanged(QCPRange)),ui->customPlot_openfile_3-
   >xAxis2, SLOT(setRange(QCPRange)));
170.        connect(ui->customPlot_openfile_3->yAxis,
   SIGNAL(rangeChanged(QCPRange)),ui->customPlot_openfile_3-
   >yAxis2, SLOT(setRange(QCPRange)));
171.        ui->customPlot_openfile_3->xAxis2-
   >setBasePen(QPen(Qt::black,1));
172.        ui->customPlot_openfile_3->yAxis2-
   >setBasePen(QPen(Qt::black,1));
173.        ui->customPlot_openfile_3->xAxis2-
   >setTickPen(Qt::NoPen);
174.        ui->customPlot_openfile_3->yAxis2-
   >setTickPen(Qt::NoPen);
175.        ui->customPlot_openfile_3->xAxis2-
   >setSubTickPen(Qt::NoPen);
176.        ui->customPlot_openfile_3->yAxis2-
   >setSubTickPen(Qt::NoPen);
177.        ui->customPlot_openfile_3->xAxis-
   >setBasePen(QPen(Qt::black, 1));
178.        ui->customPlot_openfile_3->yAxis-
   >setBasePen(QPen(Qt::black, 1));
179.        ui->customPlot_openfile_3->xAxis-
   >setTickPen(Qt::NoPen);
180.        ui->customPlot_openfile_3->yAxis-
   >setTickPen(Qt::NoPen);
181.        ui->customPlot_openfile_3->xAxis-
   >setSubTickPen(Qt::NoPen);
182.        ui->customPlot_openfile_3->yAxis-
   >setSubTickPen(Qt::NoPen);
183.        ui->customPlot_openfile_3->xAxis-
   >setTickLabels(false);
184.        ui->customPlot_openfile_3->yAxis-
   >setTickLabels(false);
185.        ui->customPlot_openfile_3->xAxis->grid()-
   >setPen(QPen(Qt::NoPen));
186.        ui->customPlot_openfile_3->yAxis->grid()-
   >setPen(QPen(Qt::NoPen));
```

```
187.        ui->customPlot_openfile_3->xAxis->grid()-
   >setSubGridPen(Qt::NoPen);
188.        ui->customPlot_openfile_3->yAxis->grid()-
   >setSubGridPen(Qt::NoPen);
189.        ui->customPlot_openfile_3->xAxis->grid()-
   >setSubGridVisible(false);
190.        ui->customPlot_openfile_3->yAxis->grid()-
   >setSubGridVisible(false);
191.        ui->customPlot_openfile_3->xAxis->grid()-
   >setZeroLinePen(Qt::NoPen);
192.        ui->customPlot_openfile_3->yAxis->grid()-
   >setZeroLinePen(Qt::NoPen);
193.
194.        QPixmap pixmap;
195.        pixmap.load("/home/pi/ECG_Layout_fix.png");
196.        ui->customPlot_openfile_3-
   >setBackground(pixmap.scaled(599,115),Qt::KeepAspectRatioByEx
   panding);
197.        QLinearGradient plotGradient;
198.        plotGradient.setStart(0, 0);
199.        plotGradient.setFinalStop(0, 350);
200.        plotGradient.setColorAt(0, QColor(255, 255, 255));
201.      //  plotGradient.setColorAt(1, QColor(50, 50, 50));
202.        ui->customPlot_openfile_3-
   >setBackground(plotGradient);
203.    }
204.    void FormOpenFile::makeplot_openfile_4()
205.    {
206.        //Untuk memberi warna pada plot
207.        ui->customPlot_openfile_4->addGraph();
208.        ui->customPlot_openfile_4->graph(0)-
   >setPen(QPen(QColor(88, 255, 85)));
209.        //Untuk memberi range X dan Y
210.        ui->customPlot_openfile_4->xAxis->setRange(0,60);
211.        ui->customPlot_openfile_4->yAxis->setRange(0,900);
212.        ui->customPlot_openfile_4->xAxis2->setVisible(true);
213.        ui->customPlot_openfile_4->xAxis2-
   >setTickLabels(false);
214.        ui->customPlot_openfile_4->yAxis2->setVisible(true);
215.        ui->customPlot_openfile_4->yAxis2-
   >setTickLabels(false);
216.        connect(ui->customPlot_openfile_4->xAxis,
   SIGNAL(rangeChanged(QCPRange)),ui->customPlot_openfile_4-
   >xAxis2, SLOT(setRange(QCPRange)));
217.        connect(ui->customPlot_openfile_4->yAxis,
   SIGNAL(rangeChanged(QCPRange)),ui->customPlot_openfile_4-
   >yAxis2, SLOT(setRange(QCPRange)));
218.        ui->customPlot_openfile_4->xAxis2-
   >setBasePen(QPen(Qt::white,1));
219.        ui->customPlot_openfile_4->yAxis2-
   >setBasePen(QPen(Qt::white,1));
220.        ui->customPlot_openfile_4->xAxis2-
   >setTickPen(Qt::NoPen);
221.        ui->customPlot_openfile_4->yAxis2-
   >setTickPen(Qt::NoPen);
222.        ui->customPlot_openfile_4->xAxis2-
   >setSubTickPen(Qt::NoPen);
223.        ui->customPlot_openfile_4->yAxis2-
   >setSubTickPen(Qt::NoPen);
```

```
224.        ui->customPlot_openfile_4->xAxis-
   >setBasePen(QPen(Qt::white, 1));
225.        ui->customPlot_openfile_4->yAxis-
   >setBasePen(QPen(Qt::white, 1));
226.        ui->customPlot_openfile_4->xAxis-
   >setTickPen(Qt::NoPen);
227.        ui->customPlot_openfile_4->yAxis-
   >setTickPen(Qt::NoPen);
228.        ui->customPlot_openfile_4->xAxis-
   >setSubTickPen(Qt::NoPen);
229.        ui->customPlot_openfile_4->yAxis-
   >setSubTickPen(Qt::NoPen);
230.        ui->customPlot_openfile_4->xAxis-
   >setTickLabels(false);
231.        ui->customPlot_openfile_4->yAxis-
   >setTickLabels(false);
232.        ui->customPlot_openfile_4->xAxis->grid()-
   >setPen(QPen(Qt::NoPen));
233.        ui->customPlot_openfile_4->yAxis->grid()-
   >setPen(QPen(Qt::NoPen));
234.        ui->customPlot_openfile_4->xAxis->grid()-
   >setSubGridPen(Qt::NoPen);
235.        ui->customPlot_openfile_4->yAxis->grid()-
   >setSubGridPen(Qt::NoPen);
236.        ui->customPlot_openfile_4->xAxis->grid()-
   >setSubGridVisible(false);
237.        ui->customPlot_openfile_4->yAxis->grid()-
   >setSubGridVisible(false);
238.        ui->customPlot_openfile_4->xAxis->grid()-
   >setZeroLinePen(Qt::NoPen);
239.        ui->customPlot_openfile_4->yAxis->grid()-
   >setZeroLinePen(Qt::NoPen);
240.        QLinearGradient plotGradient;
241.        plotGradient.setStart(0, 0);
242.        plotGradient.setFinalStop(0, 350);
243.        plotGradient.setColorAt(0, QColor(80, 80, 80));
244.        plotGradient.setColorAt(1, QColor(50, 50, 50));
245.        ui->customPlot_openfile_4-
   >setBackground(plotGradient);
246.    }
247.    void FormOpenFile::timerplotting()
248.    {
249.        QString readData;
250.       // qDebug()<<listsavedata.count();
251.        if(hitung==0)
252.        {
253.            hitung = listsavedata.count();
254.        }
255.        if(i<hitung&&hitung>0)
256.        {
257.            ui->progressBar->setEnabled(true);
258.            ui->progressBar->setRange(0,hitung);
259.        for(int a =0;a<hitung;a++)
260.        {
261.            ui->pushButton_2->setEnabled(false);
262.            ui->pushButton_3->setEnabled(false);
263.            i++;
264.            ui->progressBar->setValue(i);
265.            plottertime++;
```

```
266.            readData = listsavedata.value(i);
267.            if(readData.contains("A"))
268.            {
269.                readData =
   readData.replace("A","").trimmed();
270.                double in = readData.toDouble();
271.               // qDebug()<< in;
272.                if(in>9&&in<1024)
273.                {
274.                   //qDebug()<<"nilai
   A"<<in<<i<<"/"<<hitung;
275.                    plot1=in;
276.                }
277.            }
278.            else if (readData.contains("B"))
279.            {
280.                readData =
   readData.replace("B","").trimmed();
281.                double in = readData.toDouble();
282.              //  qDebug()<< in;
283.                if(in>9)
284.                {
285.                    //qDebug()<<"nilai B"<<in;
286.                    plot2=in;
287.                   // qDebug()<<plot2;
288.                    Signal = plot2;
289.                    for(int i=0;i<100;i++)
290.                    {
291.                        average_plot_open +=Signal;
292.                    }
293.                    average_plot_open =
   average_plot_open/100;
294.                  // qDebug()<<average_lead;
295.
   if(average_plot_open>630&&pulse_high_open==false)
296.                    {
297.                        pulse_high_open=true;
298.                        pulse_low_open=true;
299.                       // qDebug()<<"pulse high";
300.                    }
301.
   if(average_plot_open<450&&pulse_low_open==true)
302.                    {
303.                        pulse_low_open=false;
304.                        pulse_high_open=false;
305.                        for(int i=0;i<1;i++)
306.                        {
307.                            a_open++;
308.                            search_open++;
309.                            qDebug()<<search_open;
310.                        }
311.                    }
312.                }
313.            }
314.            else if (readData.contains("C"))
315.            {
316.                readData =
   readData.replace("C","").trimmed();
317.                double in = readData.toDouble();
```

```
318.                // qDebug()<<in;
319.
320.                 if(in>9)
321.                 {
322.                     //qDebug()<<"nilai C"<<in;
323.                     plot3=in;
324.                 }
325.
326.             }
327.             else if (readData.contains("D"))
328.             {
329.                 readData =
    readData.replace("D","").trimmed();
330.                 double in = readData.toDouble();
331.                // qDebug()<<in;
332.                 if(in>9)
333.                 {
334.                     //qDebug()<<"nilai D"<<in;
335.                     plot4=in;
336.                 }
337.             }
338.             double key = plottertime/1190.0; // time elapsed
    since start of demo, in seconds
339.             //double input = 1;
340.
341.             static double lastPointKey = 0;
342.             if (key-lastPointKey > 0.002) // at most add
    point every 2 ms
343.             {
344.                 ui->customPlot_openfile->graph(0)-
    >addData(key,plot1);
345.                 ui->customPlot_openfile->xAxis-
    >setRange(key/2.0, 8, Qt::AlignRight);
346.
347.                // qDebug()<<"nilai key"<<key;
348.
349.                 ui->customPlot_openfile_2->graph(0)-
    >addData(key,plot2);
350.                 ui->customPlot_openfile_2->xAxis-
    >setRange(key/2.0, 8, Qt::AlignRight);
351.
352.
353.                 ui->customPlot_openfile_3->graph(0)-
    >addData(key,plot3);
354.                 ui->customPlot_openfile_3->xAxis-
    >setRange(key/2.0, 8, Qt::AlignRight);
355.
356.
357.                 ui->customPlot_openfile_4->graph(0)-
    >addData(key,plot4);
358.                 ui->customPlot_openfile_4->xAxis-
    >setRange(key/2.0, 8, Qt::AlignRight);
359.
360.             }
361.         }
362.         double bpm=search_open;
363.         ui->label_5->setAlignment(Qt::AlignCenter);
364.         ui->label_5->setText(QString::number(bpm,'f',0));
365.         ui->pushButton_2->setEnabled(true);
```

```
366.        ui->pushButton_3->setEnabled(false);
367.        ui->customPlot_openfile->replot();
368.        ui->customPlot_openfile_2->replot();
369.        ui->customPlot_openfile_3->replot();
370.        ui->customPlot_openfile_4->replot();
371.
372.        }
373.    }
374.    void FormOpenFile::receivespathfile(QString x)
375.    {
376.       sPath = x;
377.    }
378.
379.    void FormOpenFile::on_pushButton_clicked()
380.    {
381.        this->close();
382.    }
383.    void FormOpenFile::on_pushButton_2_clicked()
384.    {
385.        browserfolder->show();
386.        readdatasave="";
387.        ui->customPlot_openfile->graph(0)->clearData();
388.        ui->customPlot_openfile_2->graph(0)->clearData();
389.        ui->customPlot_openfile_3->graph(0)->clearData();
390.        ui->customPlot_openfile_4->graph(0)->clearData();
391.        ui->customPlot_openfile->replot();
392.        ui->customPlot_openfile_2->replot();
393.        ui->customPlot_openfile_3->replot();
394.        ui->customPlot_openfile_4->replot();
395.        ui->progressBar->setValue(0);
396.        ui->horizontalScrollBar->setValue(0);
397.        i=0;
398.        plottertime=0;
399.        listsavedata.clear();
400.        hitung=0;
401.        sampleCounter = 0;
402.        lastBeatTime = 0;
403.        R = 82;
404.        T = 82;
405.        thresh = 103;
406.        amp = 0;
407.        firstBeat = true;
408.        secondBeat = false;
409.        Signal=0;
410.        IBI = 600;
411.        Pulse = false;
412.        BPM=0;
413.        ui->label_5->setText(QString::number(BPM));
414.        ui->pushButton_3->setEnabled(true);
415.        a_open=0;
416.        search_open=0;
417.    }
418.
419.    void FormOpenFile::on_pushButton_3_clicked()
420.    {
421.        i=0;
422.        QFile file (sPath);
423.        if(!file.open(QIODevice::ReadOnly|QIODevice::Text))
424.        {
```

```
425.            QMessageBox::warning(this,"title","file not
   open");
426.        }
427.        QTextStream datain(&file);
428.        readdatasave=datain.readAll();
429.        listsavedata = readdatasave.split("|");
430.        //qDebug()<<list;
431.        file.close();
432.    }
433.    void FormOpenFile::horzScrollBarChanged(int value)
434.    {
435.        if (qAbs(ui->customPlot_openfile->xAxis-
   >range().center()-value/100.0) > 0.01) // if user is dragging
   customPlot, we don't want to re customPlot twice
436.        {
437.          ui-> customPlot_openfile->xAxis-
   >setRange(value/10.0, ui->customPlot_openfile->xAxis-
   >range().size(), Qt::AlignCenter);
438.          ui-> customPlot_openfile->replot();
439.        }
440.        if (qAbs(ui->customPlot_openfile_2->xAxis-
   >range().center()-value/100.0) > 0.01) // if user is dragging
   customPlot, we don't want to re customPlot twice
441.        {
442.          ui-> customPlot_openfile_2->xAxis-
   >setRange(value/10.0, ui->customPlot_openfile_2->xAxis-
   >range().size(), Qt::AlignCenter);
443.          ui-> customPlot_openfile_2->replot();
444.        }
445.        if (qAbs(ui->customPlot_openfile_3->xAxis-
   >range().center()-value/100.0) > 0.01) // if user is dragging
   customPlot, we don't want to re customPlot twice
446.        {
447.          ui-> customPlot_openfile_3->xAxis-
   >setRange(value/10.0, ui->customPlot_openfile_3->xAxis-
   >range().size(), Qt::AlignCenter);
448.          ui-> customPlot_openfile_3->replot();
449.        }
450.        if (qAbs(ui->customPlot_openfile_4->xAxis-
   >range().center()-value/100.0) > 0.01) // if user is dragging
   customPlot, we don't want to re customPlot twice
451.        {
452.          ui-> customPlot_openfile_4->xAxis-
   >setRange(value/10.0, ui->customPlot_openfile_4->xAxis-
   >range().size(), Qt::AlignCenter);
453.          ui-> customPlot_openfile_4->replot();
454.        }
455.    }
456.    void FormOpenFile::on_pushButton_4_clicked()
457.    {
458.        QString file_Name;
459.        QString istring;
460.        for(int i=0;i<1;i++)
461.        {
462.            second_number_file++;
463.            number_file_second =
   QString::number(second_number_file);
464.        }
465.        for(int i=0;i<4;i++)
```

```
466.         {
467.             istring = QString::number(i+1);
468.             if(i<3)
469.             {
470.
    file_Name="/home/pi/Datagambar/lead"+istring+"("+number_file_
    second+").png";
471.             }
472.             else
473.             {
474.
    file_Name="/home/pi/Datagambar/Pcg("+number_file_second+").pn
    g";
475.             }
476.             QFile file_png(file_Name);
477.
478.             if (!file_png.open(QIODevice::WriteOnly))
479.             {
480.                qDebug() << file_png.errorString();
481.             }
482.             else if(istring=="1")
483.             {
484.                ui->customPlot_openfile->savePng(file_Name);
485.             }
486.             else if(istring=="2")
487.             {
488.                 ui->customPlot_openfile_2-
    >savePng(file_Name);
489.             }
490.             else if(istring=="3")
491.             {
492.                 ui->customPlot_openfile_3-
    >savePng(file_Name);
493.             }
494.             else
495.             {
496.                 ui->customPlot_openfile_4-
    >savePng(file_Name);
497.             }
498.         }
499.     }
500.
501.     void FormOpenFile::on_pushButton_5_clicked()
502.     {
503.         for(int i =0;i<1;i++)
504.         {
505.             nomorgambar_open++;
506.             QFile
    file_ss("/home/pi/ScreenshotDataSave/ScreenshotDataSave"+QStr
    ing::number(nomorgambar_open)+".jpeg");
507.
508.             if (!file_ss.open(QIODevice::WriteOnly))
509.             {
510.                qDebug() << file_ss.errorString();
511.             }
512.             else
513.             {
514.                 QPixmap pixmap = QPixmap::grabWidget(this);
515.                 QImage image = pixmap.toImage();
```

```
516.
   image.save("/home/pi/ScreenshotDataSave/ScreenshotDataSave"+Q
   String::number(nomorgambar_open)+".jpeg");
517.            }
518.        }
519.    }
```

11. Formpatientdatasave.cpp

```
1. #include "formpatientdatasave.h"
2. #include "ui_formpatientdatasave.h"
3. QString value="";
4. FormPatientDataSave::FormPatientDataSave(QWidget *parent) :
5.     QDialog(parent),
6.     ui(new Ui::FormPatientDataSave)
7. {
8.     ui->setupUi(this);
9. }
10.     FormPatientDataSave::~FormPatientDataSave()
11.     {
12.         delete ui;
13.     }
14.     void FormPatientDataSave::on_pushButton_clicked()
15.     {
16.         this->close();
17.     }
18.     void FormPatientDataSave::on_pushButton_12_clicked()
19.     {
20.         value=value+"0";
21.         ui->lineEdit->setText(value);
22.     }
23.     void FormPatientDataSave::on_pushButton_3_clicked()
24.     {
25.         value=value+"1";
26.         ui->lineEdit->setText(value);
27.     }
28.     void FormPatientDataSave::on_pushButton_4_clicked()
29.     {
30.         value=value+"2";
31.         ui->lineEdit->setText(value);
32.     }
33.
34.     void FormPatientDataSave::on_pushButton_5_clicked()
35.     {
36.         value=value+"3";
37.         ui->lineEdit->setText(value);
38.     }
39.
40.     void FormPatientDataSave::on_pushButton_6_clicked()
41.     {
42.         value=value+"4";
43.         ui->lineEdit->setText(value);
44.     }
45.
46.     void FormPatientDataSave::on_pushButton_7_clicked()
47.     {
48.         value=value+"5";
49.         ui->lineEdit->setText(value);
50.     }
51.
```

```
52.     void FormPatientDataSave::on_pushButton_8_clicked()
53.     {
54.         value=value+"6";
55.         ui->lineEdit->setText(value);
56.     }
57.
58.     void FormPatientDataSave::on_pushButton_9_clicked()
59.     {
60.         value=value+"7";
61.         ui->lineEdit->setText(value);
62.     }
63.
64.     void FormPatientDataSave::on_pushButton_10_clicked()
65.     {
66.         value=value+"8";
67.         ui->lineEdit->setText(value);
68.     }
69.
70.     void FormPatientDataSave::on_pushButton_11_clicked()
71.     {
72.         value=value+"9";
73.         ui->lineEdit->setText(value);
74.     }
75.
76.     void FormPatientDataSave::on_pushButton_13_clicked()
77.     {
78.         ui->lineEdit->clear();
79.         value="";
80.     }
81.
82.     void FormPatientDataSave::on_pushButton_2_clicked()
83.     {
84.         if(value=="")
85.         {
86.             QMessageBox::warning(this,"title","file not
   open");
87.         }
88.         else
89.         {
90.             emit sendnumberpatient(value);
91.             this->close();
92.         }
93.     }
```

12. Main.cpp

```
1. #include "mainwindow.h"
2. #include <QApplication>
3. int main(int argc, char *argv[])
4. {
5.     QApplication a(argc, argv);
6.     MainWindow w;
7.     w.showMaximized();
8.     return a.exec();
9. }
```

13. Mainwindow.cpp

```
1. #include "mainwindow.h"
2. #include "ui_mainwindow.h"
3. int search,average_lead,a=0;
4. bool pulse_high=false;
5. bool pulse_low=false;
6. /*Mendeklarasikan variabel*/
7. int command = 0;
8. double inputlead1, inputlead2, inputlead3, inputPCG;
   //variabel dengan type data double, diletakkan diluar fungsi
   untuk bisa digunakan diberbagai fungsi
9. double settingms = 200.0;
10.     double key1=0;
11.     double key2=0;
12.     int changegraph=0;
13.     double keyswitch=60;
14.     int saving_enable = 0;
15.     int second_number_file_realtime;
16.     int detik_bpm;
17.     int menit_bpm;
18.     int nomorgambar=0;
19.     int R_value=40;
20.     int T_value=40;
21.     int thresh_value=72;
22.     double time_plot;
23.     int detik_realtime;
24.     //Mendeklarasikan variabel
25.     QString serial,readData;
26.     QStringList list_data;
27.     QByteArray byteArray;
28.     QBuffer buffer;
29.     QTextStream out(&buffer);
30.     QTextStream in(&buffer);
31.     QString number_file_second_realtime;
32.     QString filename_screenshot;
33.     QString text,test;
34.     int gel_search_1,gel_search_2,gel_high,gel_reference;
35.     int count,count_result;
36.     MainWindow::MainWindow(QWidget *parent) :
37.         QMainWindow(parent),
38.         ui(new Ui::MainWindow)
39.     {
40.         ui->setupUi(this);
41.         /*Memanggil fungasi setting grafik*/
42.         MainWindow::makePlot();
43.         MainWindow::makePlot_2();
44.         MainWindow::makePlot_3();
45.         MainWindow::makePlot_4();
46.         /*Pemanggilan ulang QTimer*/
47.         qtimer=new QTimer(this); //pembaruan class
48.         connect(qtimer, SIGNAL(timeout()), this,
   SLOT(myTimer())); //untuk menghubungkan class qtimer dengan
   slots
49.         qtimer->start(0); // interval 0 atau melakukan
   refresh secara cepat
50.         qtimer_savedata = new QTimer(this);
```

```
51.
   connect(qtimer_savedata,SIGNAL(timeout()),this,SLOT(timersave
   data()));
52.         qtimer_savedata->start(1000);
53.         ar=new Arduino(this); //pembaruan class
54.         patientdatasave =new FormPatientDataSave(this);
55.         openfile = new FormOpenFile(this);
56.         /*Menghubungkan Arduino class dengan mainwindow
   class*/
57.
   connect(ar,SIGNAL(alldata(QString)),this,SLOT(receivealldata(
   QString)));
58.
   connect(patientdatasave,SIGNAL(sendnumberpatient(QString)),th
   is,SLOT(receivenumberpatient(QString)));
59.
60.
   connect(ar,SIGNAL(data_realtime(QString)),this,SLOT(receive_d
   ata_realtime(QString)));
61.         dw = new Dialog_Warning(this);
62.         ui->progressBar->setRange(0,59);
63.         ui->progressBar->setValue(0);
64.     }
65.     MainWindow::~MainWindow()
66.     {
67.         delete ui;
68.     }
69.     /*Menerima data signal dari Arduino class*/
70.     void MainWindow::receivealldata(QString i)
71.     {
72.         if(patientnumber=="")
73.         {
74.             detik=0;
75.             menit=0;
76.         }
77.         else if(patientnumber!="" &&menit<1)
78.         {
79.             ui->progressBar->setValue(detik);
80.             text =i;
81.             text = text.replace("\n","|").trimmed();
82.             test +=text;
83.             qDebug()<<"detik"<<detik;
84.
85.         }
86.         else if (patientnumber!=""&&menit==1)
87.         {
88.             QString coba;
89.             coba=test;
90.             qDebug()<<coba;
91.             QFile file
   ("/home/pi/DataPasien/"+patientnumber+".txt");
92.
   if(!file.open(QIODevice::WriteOnly|QIODevice::Text))
93.             {
94.                 QMessageBox::warning(this,"title","file not
   open");
95.             }
96.             QTextStream dataout(&file);
97.             dataout << coba;
```

```
98.             file.flush();
99.             file.close();
100.            patientnumber="";
101.            text ="";
102.            test = "";
103.        }
104.        else
105.        {
106.            patientnumber="";
107.            detik=0;
108.            menit=0;
109.        }
110.    }
111.    void MainWindow::receivenumberpatient(QString x)
112.    {
113.        patientnumber=x;
114.    }
115.    void MainWindow::timersavedata()
116.    {
117.        detik_realtime++;
118.        detik++;
119.        if(detik>59)
120.        {
121.            menit++;
122.            detik=0;
123.        }
124.        //qDebug()<<"detik ="<<detik<<"menit ="<<menit;
125.        detik_bpm++;
126.        //qDebug()<<detik_bpm;
127.        if (detik_bpm>59)
128.        {
129.            double bpm=search;
130.            qDebug()<<"bpm"<<bpm;
131.            ui->label_7-
   >setText(QString::number(bpm,'f',0));
132.            search=0;
133.            detik_bpm=0;
134.        }
135.    }
136.    void MainWindow::receive_data_realtime(QString s)
137.    {
138.        if(detik_realtime<1)
139.        {
140.            serial+=s;
141.        }
142.        else
143.        {
144.            list_data = serial.split("\n");
145.            for(int i=0;i<list_data.count();i++)
146.            {
147.                time_plot++;
148.                readData = list_data.value(i);
149.                //qDebug()<<serialBuffer.count();
150.                //qDebug()<<serialBuffer;
151.                if(readData.contains("A"))
152.                {
153.                    readData =
   readData.replace("A","").trimmed();
154.                    double in = readData.toDouble();
```

```
155.                    // qDebug()<< in;
156.                     if(in>9)
157.                     {
158.                         //qDebug()<<"nilai A"<<in;
159.                         inputlead1=in;
160.                         //qDebug()<<inputlead1;
161.                     }
162.                 }
163.                 else if (readData.contains("B"))
164.                 {
165.                     readData =
    readData.replace("B","").trimmed();
166.                     double in = readData.toDouble();
167.                   //  qDebug()<< in;
168.                     if(in>9)
169.                     {
170.                         //qDebug()<<"nilai B"<<in;
171.                         inputlead2=in;
172.                        // qDebug()<<inputlead2;
173.                         for(int i=0;i<100;i++)
174.                         {
175.                             average_lead +=inputlead2;
176.                             //qDebug()<<L2;
177.                         }
178.                         average_lead = average_lead/100;
179.                        // qDebug()<<average_lead;
180.
    if(average_lead>630&&pulse_high==false)
181.                         {
182.                             pulse_high=true;
183.                             pulse_low=true;
184.                            // qDebug()<<"pulse high";
185.                         }
186.
    if(average_lead<450&&pulse_low==true)
187.                         {
188.                             pulse_low=false;
189.                             pulse_high=false;
190.                             for(int i=0;i<1;i++)
191.                             {
192.                                 a++;
193.                                 search++;
194.                                // qDebug()<<search;
195.                             }
196.                         }
197.                     }
198.                 }
199.                 else if (readData.contains("C"))
200.                 {
201.                     readData =
    readData.replace("C","").trimmed();
202.                     double in = readData.toDouble();
203.                    // qDebug()<<in;
204.                     if(in>9)
205.                     {
206.                         //qDebug()<<"nilai C"<<in;
207.                         inputlead3=in;
208.                     }
209.                 }
```

```
210.                else if (readData.contains("D"))
211.                {
212.                    readData =
   readData.replace("D","").trimmed();
213.                    double in = readData.toDouble();
214.                   // qDebug()<<in;
215.                    if(in>9)
216.                    {
217.                        //qDebug()<<"nilai D"<<in;
218.                        inputPCG=in;
219.                    }
220.                }
221.                double key = time_plot/160.0; // time
   elapsed since start of demo, in seconds
222.                //double input = 1;
223.                static double lastPointKey = 0;
224.                if (key-lastPointKey > 0.002) // at most add
   point every 2 ms
225.                {
226.                    // plot new data -> remove old one
227.                    //line
228.                    if(key>keyswitch)
229.                    {
230.                        for(int i=0;i<1;i++)
231.                        {
232.                         changegraph++;
233.                         keyswitch +=60 ;
234.                         key1 +=60;
235.                         key2 +=60;
236.                         if(key2>=180&&changegraph==1)
237.                         {
238.                             ui->customPlot->graph(1)-
   >clearData();
239.                             ui->customPlot_2->graph(1)-
   >clearData();
240.                             ui->customPlot_3->graph(1)-
   >clearData();
241.                             ui->customPlot_4->graph(1)-
   >clearData();
242.                         }
243.                        }
244.                        if(changegraph>1)
245.                        {
246.                            changegraph=0;
247.                            ui->customPlot->graph(0)-
   >clearData();
248.                            ui->customPlot_2->graph(0)-
   >clearData();
249.                            ui->customPlot_3->graph(0)-
   >clearData();
250.                            ui->customPlot_4->graph(0)-
   >clearData();
251.                        }
252.                    }
253.                    if(changegraph==0)
254.                    {
255.                        ui->customPlot->graph(0)-
   >addData(key-key1, inputlead1);
```

```
256.                     //   ui->customPlot->graph(0)-
   >rescaleValueAxis();
257.                         ui->customPlot->graph(1)-
   >removeDataBefore(key-key2);
258.
259.                         ui->customPlot_2->graph(0)-
   >addData(key-key1, inputlead2);
260.                       //  ui->customPlot_2->graph(0)-
   >rescaleValueAxis();
261.                         ui->customPlot_2->graph(1)-
   >removeDataBefore(key-key2);
262.
263.
264.                         ui->customPlot_3->graph(0)-
   >addData(key-key1, inputlead3);
265.                      //    ui->customPlot_3->graph(0)-
   >rescaleValueAxis();
266.                         ui->customPlot_3->graph(1)-
   >removeDataBefore(key-key2);
267.
268.                         ui->customPlot_4->graph(0)-
   >addData(key-key1, inputPCG);
269.                      //    ui->customPlot_4->graph(0)-
   >rescaleValueAxis();
270.                         ui->customPlot_4->graph(1)-
   >removeDataBefore(key-key2);
271.                      }
272.
273.                      if(changegraph==1)
274.                      {
275.
276.                         ui->customPlot->graph(0)-
   >removeDataBefore(key-key1);
277.                         ui->customPlot->graph(1)-
   >addData(key-key2, inputlead1);
278.                      //    ui->customPlot->graph(1)-
   >rescaleValueAxis();
279.
280.                         ui->customPlot_2->graph(0)-
   >removeDataBefore(key-key1);
281.                         ui->customPlot_2->graph(1)-
   >addData(key-key2, inputlead2);
282.                      //    ui->customPlot_2->graph(1)-
   >rescaleValueAxis();
283.
284.                         ui->customPlot_3->graph(0)-
   >removeDataBefore(key-key1);
285.                         ui->customPlot_3->graph(1)-
   >addData(key-key2, inputlead3);
286.                       //  ui->customPlot_3->graph(1)-
   >rescaleValueAxis();
287.
288.                         ui->customPlot_4->graph(0)-
   >removeDataBefore(key-key1);
289.                         ui->customPlot_4->graph(1)-
   >addData(key-key2, inputPCG);
290.                      //    ui->customPlot_4->graph(1)-
   >rescaleValueAxis();
291.                      }
```

```
292.                     //dot
293.
294.                    // ui->customPlot->graph(1)->addData(key,
   inputlead1);
295.                   //lastPointKey = key;
296.
297.                }
298.            }
299.            ui->customPlot->xAxis-
   >setRange(0,0,Qt::AlignRight);
300.            ui->customPlot->replot();
301.            //customplot_2:
302.            //ui->customPlot_2->xAxis-
   >setRange(key,8,Qt::AlignRight);
303.            ui->customPlot_2->replot();
304.            //customplot_3:
305.            //ui->customPlot_3->xAxis-
   >setRange(key,8,Qt::AlignRight);
306.            ui->customPlot_3->replot();
307.            //customplot_4:
308.            //ui->customPlot_4->xAxis-
   >setRange(key,8,Qt::AlignRight);
309.            ui->customPlot_4->replot();
310.            //
   qDebug()<<inputlead1<<inputlead2<<inputlead3<<inputPCG<<endl;
311.            //qDebug()<<data_read.count();
312.            detik_realtime=0;
313.            serial="";
314.        }
315.    }
316.
317.    /*fungsi untuk mengatur plotter*/
318.    void MainWindow::myTimer()
319.    {
320.        static QTime time(QTime::currentTime());
321.        // calculate two new data points:
322.
323.        // make key axis range scroll with the data (at a
   constant range size of 8):
324.        //customplot_1:
325.    }
326.
327.
328.    /*Setting tampilan grafik*/
329.    void MainWindow::makePlot()
330.    {
331.        //Untuk memberi warna pada plot
332.        ui->customPlot->addGraph();
333.        ui->customPlot->graph(0)-
   >setPen(QPen(QColor(0,0,0)));
334.        ui->customPlot->addGraph();
335.        ui->customPlot->graph(1)-
   >setPen(QPen(QColor(0,0,0)));
336.        //Untuk memberi range X dan Y
337.        ui->customPlot->xAxis->setRange(0,60);
338.        ui->customPlot->yAxis->setRange(0,1200);
339.        ui->customPlot->xAxis2->setVisible(true);
340.        ui->customPlot->xAxis2->setTickLabels(false);
341.        ui->customPlot->yAxis2->setVisible(true);
```

```
342.        ui->customPlot->yAxis2->setTickLabels(false);
343.        connect(ui->customPlot->xAxis,
   SIGNAL(rangeChanged(QCPRange)),ui->customPlot->xAxis2,
   SLOT(setRange(QCPRange)));
344.        connect(ui->customPlot->yAxis,
   SIGNAL(rangeChanged(QCPRange)),ui->customPlot->yAxis2,
   SLOT(setRange(QCPRange)));
345.        ui->customPlot->xAxis2-
   >setBasePen(QPen(Qt::black,1));
346.        ui->customPlot->yAxis2-
   >setBasePen(QPen(Qt::black,1));
347.        ui->customPlot->xAxis2->setTickPen(Qt::NoPen);
348.        ui->customPlot->yAxis2->setTickPen(Qt::NoPen);
349.        ui->customPlot->xAxis2->setSubTickPen(Qt::NoPen);
350.        ui->customPlot->yAxis2->setSubTickPen(Qt::NoPen);
351.        ui->customPlot->xAxis->setBasePen(QPen(Qt::black,
   1));
352.        ui->customPlot->yAxis->setBasePen(QPen(Qt::black,
   1));
353.        ui->customPlot->xAxis->setTickPen(Qt::NoPen);
354.        ui->customPlot->yAxis->setTickPen(Qt::NoPen);
355.        ui->customPlot->xAxis->setSubTickPen(Qt::NoPen);
356.        ui->customPlot->yAxis->setSubTickPen(Qt::NoPen);
357.        ui->customPlot->xAxis->setTickLabels(false);
358.        ui->customPlot->yAxis->setTickLabels(false);
359.        ui->customPlot->xAxis->grid()-
   >setPen(QPen(Qt::NoPen));
360.        ui->customPlot->yAxis->grid()-
   >setPen(QPen(Qt::NoPen));
361.        ui->customPlot->xAxis->grid()-
   >setSubGridPen(Qt::NoPen);
362.        ui->customPlot->yAxis->grid()-
   >setSubGridPen(Qt::NoPen);
363.        ui->customPlot->xAxis->grid()-
   >setSubGridVisible(false);
364.        ui->customPlot->yAxis->grid()-
   >setSubGridVisible(false);
365.        ui->customPlot->xAxis->grid()-
   >setZeroLinePen(Qt::NoPen);
366.        ui->customPlot->yAxis->grid()-
   >setZeroLinePen(Qt::NoPen);
367.        QPixmap pixmap;
368.        pixmap.load("/home/pi/ECG_Layout_fix.png");
369.        ui->customPlot-
   >setBackground(pixmap.scaled(599,115),Qt::KeepAspectRatioByEx
   panding);
370.        QLinearGradient plotGradient;
371.        plotGradient.setStart(0, 0);
372.        plotGradient.setFinalStop(0, 350);
373.        plotGradient.setColorAt(0, QColor(255,255,255));
374.       // plotGradient.setColorAt(1, QColor(50, 50, 50));
375.        ui->customPlot->setBackground(plotGradient);
376.    }
377.    void MainWindow::makePlot_2()
378.    {
379.        //Untuk memberi warna pada plot
380.        ui->customPlot_2->addGraph();
381.        ui->customPlot_2->graph(0)-
   >setPen(QPen(QColor(0,0,0)));
```

```
382.        ui->customPlot_2->addGraph();
383.        ui->customPlot_2->graph(1)-
   >setPen(QPen(QColor(0,0,0)));
384.        //Untuk memberi range X dan Y
385.        ui->customPlot_2->xAxis->setRange(0,60);
386.        ui->customPlot_2->yAxis->setRange(0,1159);
387.        ui->customPlot_2->xAxis2->setVisible(true);
388.        ui->customPlot_2->xAxis2->setTickLabels(false);
389.        ui->customPlot_2->yAxis2->setVisible(true);
390.        ui->customPlot_2->yAxis2->setTickLabels(false);
391.        connect(ui->customPlot_2->xAxis,
   SIGNAL(rangeChanged(QCPRange)),ui->customPlot_2->xAxis2,
   SLOT(setRange(QCPRange)));
392.        connect(ui->customPlot_2->yAxis,
   SIGNAL(rangeChanged(QCPRange)),ui->customPlot_2->yAxis2,
   SLOT(setRange(QCPRange)));
393.        ui->customPlot_2->xAxis2-
   >setBasePen(QPen(Qt::black,1));
394.        ui->customPlot_2->yAxis2-
   >setBasePen(QPen(Qt::black,1));
395.        ui->customPlot_2->xAxis2->setTickPen(Qt::NoPen);
396.        ui->customPlot_2->yAxis2->setTickPen(Qt::NoPen);
397.        ui->customPlot_2->xAxis2->setSubTickPen(Qt::NoPen);
398.        ui->customPlot_2->yAxis2->setSubTickPen(Qt::NoPen);
399.        ui->customPlot_2->xAxis->setBasePen(QPen(Qt::black,
   1));
400.        ui->customPlot_2->yAxis->setBasePen(QPen(Qt::black,
   1));
401.        ui->customPlot_2->xAxis->setTickPen(Qt::NoPen);
402.        ui->customPlot_2->yAxis->setTickPen(Qt::NoPen);
403.        ui->customPlot_2->xAxis->setSubTickPen(Qt::NoPen);
404.        ui->customPlot_2->yAxis->setSubTickPen(Qt::NoPen);
405.        ui->customPlot_2->xAxis->setTickLabels(false);
406.        ui->customPlot_2->yAxis->setTickLabels(false);
407.        ui->customPlot_2->xAxis->grid()-
   >setPen(QPen(Qt::NoPen));
408.        ui->customPlot_2->yAxis->grid()-
   >setPen(QPen(Qt::NoPen));
409.        ui->customPlot_2->xAxis->grid()-
   >setSubGridPen(Qt::NoPen);
410.        ui->customPlot_2->yAxis->grid()-
   >setSubGridPen(Qt::NoPen);
411.        ui->customPlot_2->xAxis->grid()-
   >setSubGridVisible(false);
412.        ui->customPlot_2->yAxis->grid()-
   >setSubGridVisible(false);
413.        ui->customPlot_2->xAxis->grid()-
   >setZeroLinePen(Qt::NoPen);
414.        ui->customPlot_2->yAxis->grid()-
   >setZeroLinePen(Qt::NoPen);
415.        QPixmap pixmap;
416.        pixmap.load("/home/pi/ECG_Layout_fix.png");
417.        ui->customPlot_2-
   >setBackground(pixmap.scaled(599,115),Qt::KeepAspectRatioByEx
   panding);
418.        QLinearGradient plotGradient;
419.        plotGradient.setStart(0, 0);
420.        plotGradient.setFinalStop(0, 350);
421.        plotGradient.setColorAt(0, QColor(255,255,255));
```

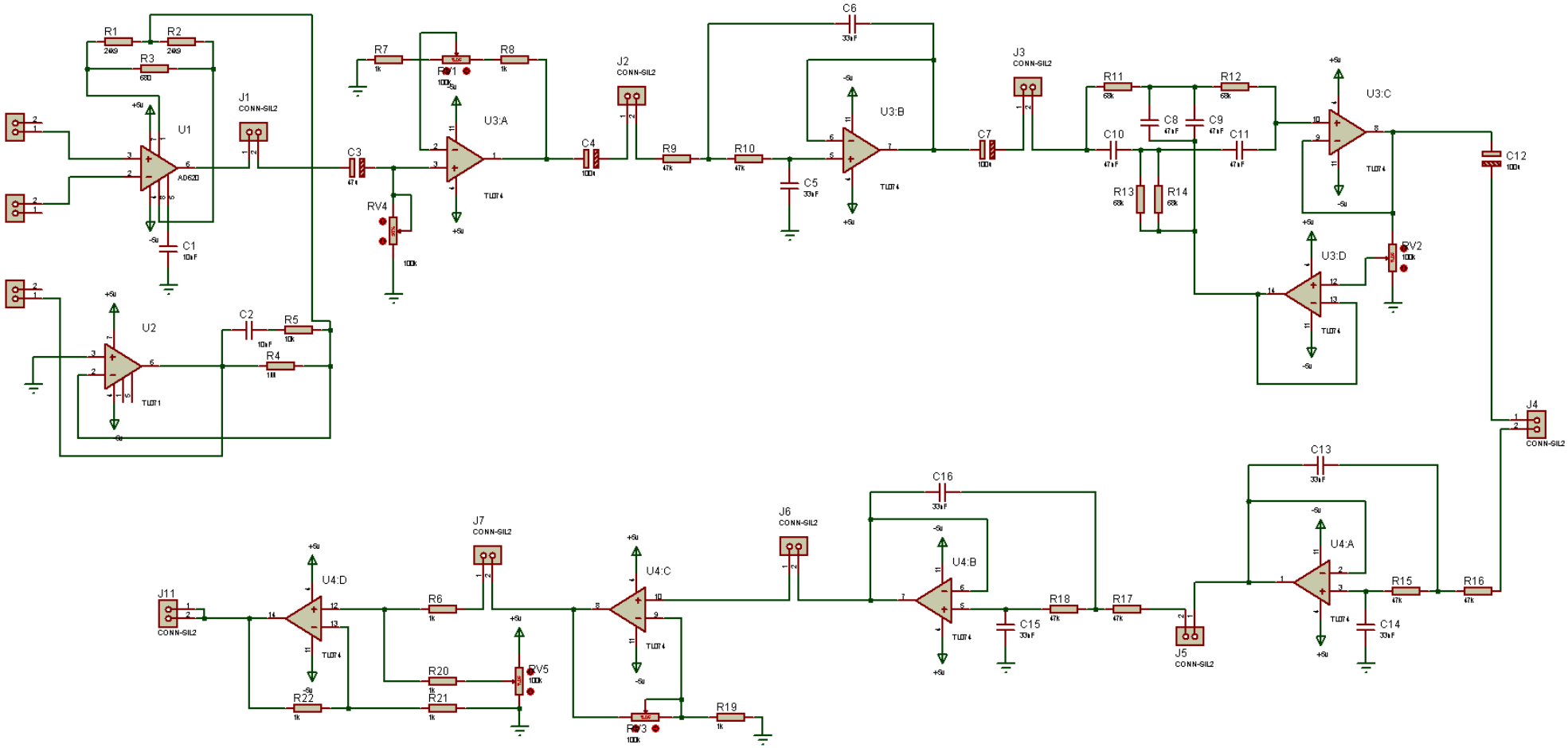
```
422.        //plotGradient.setColorAt(1, QColor(50, 50, 50));
423.        ui->customPlot_2->setBackground(plotGradient);
424.    }
425.    void MainWindow::makePlot_3()
426.    {
427.        //Untuk memberi warna pada plot
428.        ui->customPlot_3->addGraph();
429.        ui->customPlot_3->graph(0)-
   >setPen(QPen(QColor(0,0,0)));
430.        ui->customPlot_3->addGraph();
431.        ui->customPlot_3->graph(1)-
   >setPen(QPen(QColor(0,0,0)));
432.        //Untuk memberi range X dan Y
433.        ui->customPlot_3->xAxis->setRange(0,60);
434.        ui->customPlot_3->yAxis->setRange(0,1090);
435.        ui->customPlot_3->xAxis2->setVisible(true);
436.        ui->customPlot_3->xAxis2->setTickLabels(false);
437.        ui->customPlot_3->yAxis2->setVisible(true);
438.        ui->customPlot_3->yAxis2->setTickLabels(false);
439.
440.        connect(ui->customPlot_3->xAxis,
   SIGNAL(rangeChanged(QCPRange)),ui->customPlot_3->xAxis2,
   SLOT(setRange(QCPRange)));
441.        connect(ui->customPlot_3->yAxis,
   SIGNAL(rangeChanged(QCPRange)),ui->customPlot_3->yAxis2,
   SLOT(setRange(QCPRange)));
442.        ui->customPlot_3->xAxis2-
   >setBasePen(QPen(Qt::black,1));
443.        ui->customPlot_3->yAxis2-
   >setBasePen(QPen(Qt::black,1));
444.        ui->customPlot_3->xAxis2->setTickPen(Qt::NoPen);
445.        ui->customPlot_3->yAxis2->setTickPen(Qt::NoPen);
446.        ui->customPlot_3->xAxis2->setSubTickPen(Qt::NoPen);
447.        ui->customPlot_3->yAxis2->setSubTickPen(Qt::NoPen);
448.        ui->customPlot_3->xAxis->setBasePen(QPen(Qt::black,
   1));
449.        ui->customPlot_3->yAxis->setBasePen(QPen(Qt::black,
   1));
450.        ui->customPlot_3->xAxis->setTickPen(Qt::NoPen);
451.        ui->customPlot_3->yAxis->setTickPen(Qt::NoPen);
452.        ui->customPlot_3->xAxis->setSubTickPen(Qt::NoPen);
453.        ui->customPlot_3->yAxis->setSubTickPen(Qt::NoPen);
454.        ui->customPlot_3->xAxis->setTickLabels(false);
455.        ui->customPlot_3->yAxis->setTickLabels(false);
456.        ui->customPlot_3->xAxis->grid()-
   >setPen(QPen(Qt::NoPen));
457.        ui->customPlot_3->yAxis->grid()-
   >setPen(QPen(Qt::NoPen));
458.        ui->customPlot_3->xAxis->grid()-
   >setSubGridPen(Qt::NoPen);
459.        ui->customPlot_3->yAxis->grid()-
   >setSubGridPen(Qt::NoPen);
460.        ui->customPlot_3->xAxis->grid()-
   >setSubGridVisible(false);
461.        ui->customPlot_3->yAxis->grid()-
   >setSubGridVisible(false);
462.        ui->customPlot_3->xAxis->grid()-
   >setZeroLinePen(Qt::NoPen);
```

```
463.        ui->customPlot_3->yAxis->grid()-
   >setZeroLinePen(Qt::NoPen);
464.        QPixmap pixmap;
465.        pixmap.load("/home/pi/ECG_Layout_fix.png");
466.        ui->customPlot_3-
   >setBackground(pixmap.scaled(599,115),Qt::KeepAspectRatioByEx
   panding);
467.        QLinearGradient plotGradient;
468.        plotGradient.setStart(0, 0);
469.        plotGradient.setFinalStop(0, 350);
470.        plotGradient.setColorAt(0, QColor(255,255,255));
471.        //plotGradient.setColorAt(1, QColor(50, 50, 50));
472.        ui->customPlot_3->setBackground(plotGradient);
473.    }
474.    void MainWindow::makePlot_4()
475.    {
476.        //Untuk memberi warna pada plot
477.        ui->customPlot_4->addGraph();
478.        ui->customPlot_4->graph(0)->setPen(QPen(QColor(88,
   255, 85)));
479.        ui->customPlot_4->addGraph();
480.        ui->customPlot_4->graph(1)->setPen(QPen(QColor(88,
   225, 85)));
481.        //Untuk memberi range X dan Y
482.        ui->customPlot_4->xAxis->setRange(0,60);
483.        ui->customPlot_4->yAxis->setRange(0,1200);
484.        ui->customPlot_4->xAxis2->setVisible(true);
485.        ui->customPlot_4->xAxis2->setTickLabels(false);
486.        ui->customPlot_4->yAxis2->setVisible(true);
487.        ui->customPlot_4->yAxis2->setTickLabels(false);
488.        connect(ui->customPlot_4->xAxis,
   SIGNAL(rangeChanged(QCPRange)),ui->customPlot_4->xAxis2,
   SLOT(setRange(QCPRange)));
489.        connect(ui->customPlot_4->yAxis,
   SIGNAL(rangeChanged(QCPRange)),ui->customPlot_4->yAxis2,
   SLOT(setRange(QCPRange)));
490.        ui->customPlot_4->xAxis2-
   >setBasePen(QPen(Qt::white,1));
491.        ui->customPlot_4->yAxis2-
   >setBasePen(QPen(Qt::white,1));
492.        ui->customPlot_4->xAxis2->setTickPen(Qt::NoPen);
493.        ui->customPlot_4->yAxis2->setTickPen(Qt::NoPen);
494.        ui->customPlot_4->xAxis2->setSubTickPen(Qt::NoPen);
495.        ui->customPlot_4->yAxis2->setSubTickPen(Qt::NoPen);
496.        ui->customPlot_4->xAxis->setBasePen(QPen(Qt::white,
   1));
497.        ui->customPlot_4->yAxis->setBasePen(QPen(Qt::white,
   1));
498.        ui->customPlot_4->xAxis->setTickPen(Qt::NoPen);
499.        ui->customPlot_4->yAxis->setTickPen(Qt::NoPen);
500.        ui->customPlot_4->xAxis->setSubTickPen(Qt::NoPen);
501.        ui->customPlot_4->yAxis->setSubTickPen(Qt::NoPen);
502.        ui->customPlot_4->xAxis->setTickLabels(false);
503.        ui->customPlot_4->yAxis->setTickLabels(false);
504.        ui->customPlot_4->xAxis->grid()-
   >setPen(QPen(Qt::NoPen));
505.        ui->customPlot_4->yAxis->grid()-
   >setPen(QPen(Qt::NoPen));
```

```
506.        ui->customPlot_4->xAxis->grid()-
   >setSubGridPen(Qt::NoPen);
507.        ui->customPlot_4->yAxis->grid()-
   >setSubGridPen(Qt::NoPen);
508.        ui->customPlot_4->xAxis->grid()-
   >setSubGridVisible(false);
509.        ui->customPlot_4->yAxis->grid()-
   >setSubGridVisible(false);
510.        ui->customPlot_4->xAxis->grid()-
   >setZeroLinePen(Qt::NoPen);
511.        ui->customPlot_4->yAxis->grid()-
   >setZeroLinePen(Qt::NoPen);
512.        QLinearGradient plotGradient;
513.        plotGradient.setStart(0, 0);
514.        plotGradient.setFinalStop(0, 350);
515.        plotGradient.setColorAt(0, QColor(80, 80, 80));
516.        plotGradient.setColorAt(1, QColor(50, 50, 50));
517.        ui->customPlot_4->setBackground(plotGradient);
518.    }
519.    void MainWindow::on_pushButton_2_clicked()
520.    {
521.        patientdatasave->show();
522.    }
523.    void MainWindow::on_pushButton_clicked()
524.    {
525.        openfile->show();
526.    }
527.    void MainWindow::on_pushButton_3_clicked()
528.    {
529.        dw->show();
530.    }
531.    void MainWindow::on_pushButton_4_clicked()
532.    {
533.        QString file_Name;
534.        QString istring;
535.        for(int i=0;i<1;i++)
536.        {
537.            second_number_file_realtime++;
538.            number_file_second_realtime =
   QString::number(second_number_file_realtime);
539.
540.        }
541.        for(int i=0;i<4;i++)
542.        {
543.            istring = QString::number(i+1);
544.            if(i<3)
545.            {
546.
   file_Name="/home/pi/Datagambar/lead"+istring+"("+number_file_
   second_realtime+").png";
547.            }
548.            else
549.            {
550.
   file_Name="/home/pi/Datagambar/Pcg("+number_file_second_realt
   ime+").png";
551.            }
552.            QFile file_png(file_Name);
553.
```

```
554.            if (!file_png.open(QIODevice::WriteOnly))
555.            {
556.               qDebug() << file_png.errorString();
557.            }
558.            else if(istring=="1")
559.            {
560.                ui->customPlot->savePng(file_Name);
561.            }
562.            else if(istring=="2")
563.            {
564.                ui->customPlot_2->savePng(file_Name);
565.            }
566.            else if(istring=="3")
567.            {
568.                ui->customPlot_3->savePng(file_Name);
569.            }
570.            else
571.            {
572.                ui->customPlot_4->savePng(file_Name);
573.            }
574.        }
575.    }
576.    void MainWindow::on_pushButton_5_clicked()
577.    {
578.        for(int i =0;i<1;i++)
579.        {
580.            nomorgambar++;
581.            QFile
   file_ss("/home/pi/Screenshot/Screenshot"+QString::number(nomo
   rgambar)+".jpeg");
582.
583.            if (!file_ss.open(QIODevice::WriteOnly))
584.            {
585.               qDebug() << file_ss.errorString();
586.            }
587.            else
588.            {
589.                QPixmap pixmap = QPixmap::grabWidget(this);
590.                QImage image = pixmap.toImage();
591.
   image.save("/home/pi/Screenshot/Screenshot"+QString::number(n
   omorgambar)+".jpeg");
592.            }
593.        }
594.
595.    }
```

4. Rangkaian Modul ECG



5. Rangkaian pembalik fasa