

psyphibio-proc

Generated by Doxygen 1.8.2

Mon Dec 10 2012 13:05:51

Contents

| | | |
|----------|--|----------|
| 1 | Hierarchical Index | 1 |
| 1.1 | Class Hierarchy | 1 |
| 2 | Class Index | 3 |
| 2.1 | Class List | 3 |
| 3 | File Index | 5 |
| 3.1 | File List | 5 |
| 4 | Class Documentation | 7 |
| 4.1 | Adder Class Reference | 7 |
| 4.1.1 | Constructor & Destructor Documentation | 7 |
| 4.1.1.1 | Adder | 7 |
| 4.1.2 | Member Function Documentation | 7 |
| 4.1.2.1 | Add | 7 |
| 4.1.2.2 | Add | 7 |
| 4.1.2.3 | SetDefAddNo | 7 |
| 4.1.3 | Member Data Documentation | 7 |
| 4.1.3.1 | _defAddNo | 7 |
| 4.2 | BDFReader Class Reference | 7 |
| 4.2.1 | Constructor & Destructor Documentation | 9 |
| 4.2.1.1 | BDFReader | 9 |
| 4.2.2 | Member Function Documentation | 9 |
| 4.2.2.1 | Close | 9 |
| 4.2.2.2 | getNextDataPoint | 9 |
| 4.2.2.3 | getSignal | 9 |
| 4.2.2.4 | Open | 9 |
| 4.2.2.5 | PrintHeader | 9 |
| 4.2.2.6 | ReadData | 9 |
| 4.2.3 | Member Data Documentation | 9 |
| 4.2.3.1 | buf | 9 |
| 4.2.3.2 | channelNo | 9 |

| | | |
|----------|--|----|
| 4.2.3.3 | datetime | 9 |
| 4.2.3.4 | hdr | 9 |
| 4.2.3.5 | sampleCounter | 9 |
| 4.2.3.6 | samplingFreq | 9 |
| 4.3 | BDFWriter Class Reference | 9 |
| 4.3.1 | Constructor & Destructor Documentation | 10 |
| 4.3.1.1 | BDFWriter | 10 |
| 4.3.2 | Member Function Documentation | 11 |
| 4.3.2.1 | DisplayHistogram | 11 |
| 4.3.2.2 | DisplayMessage | 11 |
| 4.3.2.3 | saveNextDataPoint | 11 |
| 4.3.2.4 | saveSignal | 11 |
| 4.3.3 | Member Data Documentation | 11 |
| 4.3.3.1 | nosSamples | 11 |
| 4.3.3.2 | sigDim | 11 |
| 4.3.3.3 | sigLabels | 11 |
| 4.3.3.4 | signals | 11 |
| 4.3.3.5 | smpFreqs | 11 |
| 4.4 | Display Class Reference | 11 |
| 4.4.1 | Member Function Documentation | 12 |
| 4.4.1.1 | DisplayHistogram | 12 |
| 4.4.1.2 | DisplayMessage | 12 |
| 4.4.1.3 | saveNextDataPoint | 12 |
| 4.4.1.4 | saveSignal | 13 |
| 4.5 | DSPStatistic Class Reference | 13 |
| 4.5.1 | Constructor & Destructor Documentation | 14 |
| 4.5.1.1 | DSPStatistic | 14 |
| 4.5.1.2 | DSPStatistic | 14 |
| 4.5.2 | Member Function Documentation | 14 |
| 4.5.2.1 | CalculateDescriptivesRunning | 14 |
| 4.5.2.2 | CalculateDescriptivesStatic | 14 |
| 4.5.2.3 | CalculateDescriptivesStaticHistogram | 14 |
| 4.5.2.4 | ConvolutionInputSide | 15 |
| 4.5.2.5 | ConvolutionOutputSide | 15 |
| 4.5.2.6 | DiscreteFourierTransformationFreqSide | 15 |
| 4.5.2.7 | DiscreteFourierTransformationTimeSide | 15 |
| 4.5.2.8 | FirstDifference | 16 |
| 4.5.2.9 | InverseDiscreteFourierTransformationFreqSide | 16 |
| 4.5.2.10 | InverseDiscreteFourierTransformationTimeSide | 16 |
| 4.5.2.11 | PhaseUnwrapping | 17 |

| | | |
|----------|--|----|
| 4.5.2.12 | PolarToRectangularConversion | 18 |
| 4.5.2.13 | ProcesInput | 18 |
| 4.5.2.14 | Process | 18 |
| 4.5.2.15 | ProcesSignal | 18 |
| 4.5.2.16 | RectangularToPolarConversion | 18 |
| 4.5.2.17 | RunningSumIntegration | 19 |
| 4.5.2.18 | setInput | 19 |
| 4.5.2.19 | setOutput | 19 |
| 4.5.3 | Member Data Documentation | 19 |
| 4.5.3.1 | _input | 19 |
| 4.5.3.2 | _output | 19 |
| 4.5.3.3 | noSamples | 20 |
| 4.5.3.4 | signal | 20 |
| 4.5.3.5 | smpFreq | 20 |
| 4.6 | ECG Class Reference | 20 |
| 4.6.1 | Constructor & Destructor Documentation | 21 |
| 4.6.1.1 | ECG | 21 |
| 4.6.1.2 | ECG | 21 |
| 4.6.2 | Member Function Documentation | 21 |
| 4.6.2.1 | ProcesInput | 21 |
| 4.6.2.2 | Process | 21 |
| 4.6.2.3 | ProcesSignal | 21 |
| 4.6.2.4 | setInput | 21 |
| 4.6.2.5 | setOutput | 21 |
| 4.6.3 | Member Data Documentation | 21 |
| 4.6.3.1 | _input | 21 |
| 4.6.3.2 | _output | 21 |
| 4.7 | IAnalysis Class Reference | 22 |
| 4.7.1 | Member Function Documentation | 22 |
| 4.7.1.1 | Process | 22 |
| 4.7.1.2 | setInput | 22 |
| 4.7.1.3 | setOutput | 22 |
| 4.8 | IInput Class Reference | 22 |
| 4.8.1 | Member Function Documentation | 23 |
| 4.8.1.1 | Close | 23 |
| 4.8.1.2 | getNextDataPoint | 23 |
| 4.8.1.3 | getSignal | 23 |
| 4.8.1.4 | Open | 23 |
| 4.9 | IOutput Class Reference | 23 |
| 4.9.1 | Member Function Documentation | 24 |

| | | |
|----------|--|-----------|
| 4.9.1.1 | DisplayHistogram | 24 |
| 4.9.1.2 | DisplayMessage | 24 |
| 4.9.1.3 | saveNextDataPoint | 24 |
| 4.9.1.4 | saveSignal | 24 |
| 4.10 | NoiseGen Class Reference | 24 |
| 4.10.1 | Constructor & Destructor Documentation | 25 |
| 4.10.1.1 | NoiseGen | 25 |
| 4.10.2 | Member Function Documentation | 25 |
| 4.10.2.1 | Close | 26 |
| 4.10.2.2 | getNextDataPoint | 26 |
| 4.10.2.3 | getSignal | 26 |
| 4.10.2.4 | Open | 26 |
| 4.10.3 | Member Data Documentation | 26 |
| 4.10.3.1 | noRndSamples | 26 |
| 4.10.3.2 | rndSignalBuf | 26 |
| 4.10.3.3 | sampleCounter | 26 |
| 5 | File Documentation | 27 |
| 5.1 | include/BDFReader.h File Reference | 27 |
| 5.2 | include/BDFWriter.h File Reference | 28 |
| 5.3 | include/Display.h File Reference | 29 |
| 5.4 | include/DSPStatistic.h File Reference | 30 |
| 5.4.1 | Detailed Description | 31 |
| 5.5 | include/ECG.h File Reference | 31 |
| 5.6 | include/IAnalysis.h File Reference | 32 |
| 5.7 | include/IInput.h File Reference | 33 |
| 5.8 | include/IOutput.h File Reference | 34 |
| 5.9 | include/NoiseGen.h File Reference | 35 |
| 5.10 | source/BDFReader.cpp File Reference | 36 |
| 5.11 | source/BDFWriter.cpp File Reference | 36 |
| 5.12 | source/Display.cpp File Reference | 37 |
| 5.13 | source/DSPStatistic.cpp File Reference | 38 |
| 5.13.1 | Detailed Description | 38 |
| 5.14 | source/ECG.cpp File Reference | 39 |
| 5.15 | source/main.cpp File Reference | 39 |
| 5.15.1 | Function Documentation | 40 |
| 5.15.1.1 | main | 40 |
| 5.16 | source/main_test.cpp File Reference | 40 |
| 5.16.1 | Function Documentation | 41 |
| 5.16.1.1 | TEST | 41 |

| | | |
|------------------|---|---------------|
| 5.17 | source/NoiseGen.cpp File Reference | 41 |
| 5.18 | source/test_main.cpp File Reference | 41 |
| 5.18.1 | Function Documentation | 42 |
| 5.18.1.1 | Add | 42 |
| 5.18.1.2 | TEST | 42 |
| 5.18.1.3 | TEST | 42 |
| Index | | 42 |

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

| | |
|------------------------|----|
| Adder | 7 |
| IAnalysis | 22 |
| DSPStatistic | 13 |
| ECG | 20 |
| IInput | 22 |
| BDFReader | 7 |
| NoiseGen | 24 |
| IOutput | 23 |
| BDFWriter | 9 |
| Display | 11 |

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

| | |
|------------------------------|----|
| Adder | 7 |
| BDFReader | 7 |
| BDFWriter | 9 |
| Display | 11 |
| DSPStatistic | 13 |
| ECG | 20 |
| IAnalysis | 22 |
| IInput | 22 |
| IOutput | 23 |
| NoiseGen | 24 |

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

| | |
|--|----|
| include/BDFReader.h | 27 |
| include/BDFWriter.h | 28 |
| include/Display.h | 29 |
| include/DSPStatistic.h | |
| This header file defines an Analysis component using methods of DSP on signals | 30 |
| include/ECG.h | 31 |
| include/IAnalysis.h | 32 |
| include/IInput.h | 33 |
| include/IOutput.h | 34 |
| include/NoiseGen.h | 35 |
| source/BDFReader.cpp | 36 |
| source/BDFWriter.cpp | 36 |
| source/Display.cpp | 37 |
| source/DSPStatistic.cpp | |
| This source file defines an Analysis component using methods of DSP on signals | 38 |
| source/ECG.cpp | 39 |
| source/main.cpp | 39 |
| source/main_test.cpp | 40 |
| source/NoiseGen.cpp | 41 |
| source/test_main.cpp | 41 |

Chapter 4

Class Documentation

4.1 Adder Class Reference

Public Member Functions

- [Adder](#) ()
- int [Add](#) (int a, int b)
- int [Add](#) (int a)
- void [SetDefAddNo](#) (int defAddNo)

Private Attributes

- int [_defAddNo](#)

4.1.1 Constructor & Destructor Documentation

4.1.1.1 `Adder::Adder ()` `[inline]`

4.1.2 Member Function Documentation

4.1.2.1 `int Adder::Add (int a, int b)` `[inline]`

4.1.2.2 `int Adder::Add (int a)` `[inline]`

4.1.2.3 `void Adder::SetDefAddNo (int defAddNo)` `[inline]`

4.1.3 Member Data Documentation

4.1.3.1 `int Adder::_defAddNo` `[private]`

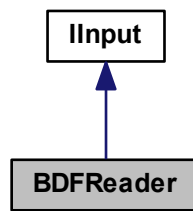
The documentation for this class was generated from the following file:

- source/[test_main.cpp](#)

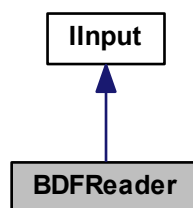
4.2 BDFReader Class Reference

```
#include <BDFReader.h>
```

Inheritance diagram for BDFReader:



Collaboration diagram for BDFReader:



Public Member Functions

- [BDFReader](#) ()
- double * [getSignal](#) (int &noSamples, int &smpFreq)
- double [getNextDataPoint](#) ()
- void [Open](#) (string fileName, int channel)
- void [Close](#) ()

Private Member Functions

- void [PrintHeader](#) (int nChannel)
- void [ReadData](#) (int nChannel)

Private Attributes

- struct edf_hdr_struct [hdr](#)
- struct tm * [datetime](#)
- float [samplingFreq](#)
- double * [buf](#)
- int [channelNo](#)
- int [sampleCounter](#)

4.2.1 Constructor & Destructor Documentation

4.2.1.1 BDFReader::BDFReader ()

4.2.2 Member Function Documentation

4.2.2.1 void BDFReader::Close () [virtual]

Implements [IInput](#).

4.2.2.2 double BDFReader::getNextDataPoint () [virtual]

Implements [IInput](#).

4.2.2.3 double * BDFReader::getSignal (int & *noSamples*, int & *smpFreq*) [virtual]

Implements [IInput](#).

4.2.2.4 void BDFReader::Open (string *fileName*, int *channel*) [virtual]

Implements [IInput](#).

4.2.2.5 void BDFReader::PrintHeader (int *nChannel*) [private]

4.2.2.6 void BDFReader::ReadData (int *nChannel*) [private]

4.2.3 Member Data Documentation

4.2.3.1 double* BDFReader::buf [private]

4.2.3.2 int BDFReader::channelNo [private]

4.2.3.3 struct tm* BDFReader::datetime [private]

4.2.3.4 struct edf_hdr_struct BDFReader::hdr [private]

4.2.3.5 int BDFReader::sampleCounter [private]

4.2.3.6 float BDFReader::samplingFreq [private]

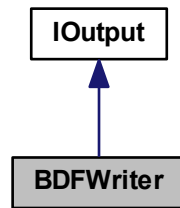
The documentation for this class was generated from the following files:

- [include/BDFReader.h](#)
- [source/BDFReader.cpp](#)

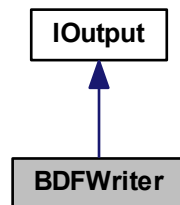
4.3 BDFWriter Class Reference

```
#include <BDFWriter.h>
```

Inheritance diagram for BDFWriter:



Collaboration diagram for BDFWriter:



Public Member Functions

- [BDFWriter](#) ()
- void [saveSignal](#) (double *sig, int noSamp, int smp_freq, string sigLabel, string [sigDim](#))
- void [saveNextDataPoint](#) ()
- void [DisplayMessage](#) (string message)
- void [DisplayHistogram](#) (deque< int > histogram)

Private Attributes

- deque< double * > [signals](#)
- deque< int > [smpFreqs](#)
- deque< int > [nosSamples](#)
- deque< string > [sigLabels](#)
- deque< string > [sigDim](#)

4.3.1 Constructor & Destructor Documentation

4.3.1.1 BDFWriter::BDFWriter ()

4.3.2 Member Function Documentation

4.3.2.1 void BDFWriter::DisplayHistogram (deque< int > *histogram*) [virtual]

Implements [IOutput](#).

4.3.2.2 void BDFWriter::DisplayMessage (string *message*) [virtual]

Implements [IOutput](#).

4.3.2.3 void BDFWriter::saveNextDataPoint () [virtual]

Implements [IOutput](#).

4.3.2.4 void BDFWriter::saveSignal (double * *sig*, int *noSamp*, int *smp_freq*, string *sigLabel*, string *sigDim*) [virtual]

Implements [IOutput](#).

4.3.3 Member Data Documentation

4.3.3.1 deque<int> BDFWriter::nosSamples [private]

4.3.3.2 deque<string> BDFWriter::sigDim [private]

4.3.3.3 deque<string> BDFWriter::sigLabels [private]

4.3.3.4 deque<double*> BDFWriter::signals [private]

4.3.3.5 deque<int> BDFWriter::smpFreqs [private]

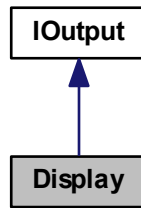
The documentation for this class was generated from the following files:

- include/[BDFWriter.h](#)
- source/[BDFWriter.cpp](#)

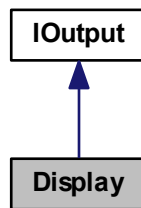
4.4 Display Class Reference

```
#include <Display.h>
```

Inheritance diagram for Display:



Collaboration diagram for Display:



Public Member Functions

- void [saveSignal](#) (double *sig, int noSamp, int smp_freq, string sigLabel, string sigDim)
- void [saveNextDataPoint](#) ()
- void [DisplayMessage](#) (string message)
- void [DisplayHistogram](#) (deque< int > histogram)

4.4.1 Member Function Documentation

4.4.1.1 void **Display::DisplayHistogram** (deque< int > *histogram*) [virtual]

Implements [IOutput](#).

4.4.1.2 void **Display::DisplayMessage** (string *message*) [virtual]

Implements [IOutput](#).

4.4.1.3 void **Display::saveNextDataPoint** () [virtual]

Implements [IOutput](#).

4.4.1.4 void Display::saveSignal (double * sig, int noSamp, int smp_freq, string sigLabel, string sigDim) [virtual]

Implements [IOutput](#).

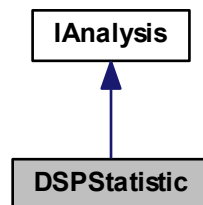
The documentation for this class was generated from the following files:

- include/[Display.h](#)
- source/[Display.cpp](#)

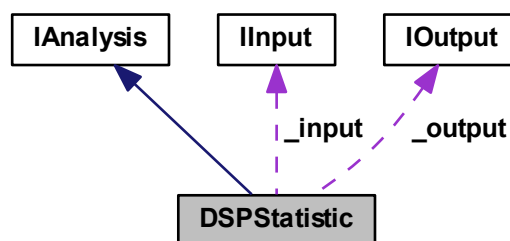
4.5 DSPStatistic Class Reference

```
#include <DSPStatistic.h>
```

Inheritance diagram for DSPStatistic:



Collaboration diagram for DSPStatistic:



Public Member Functions

- void [setInput](#) (IInput *input)
- void [setOutput](#) (IOutput *output)
- void [Process](#) ()
- [DSPStatistic](#) ()
- [DSPStatistic](#) (IInput *input, IOutput *output)

Private Member Functions

- void [ProcesInput](#) ()
- void [ProcesSignal](#) ()
- void [CalculateDescriptivesStatic](#) (double *sig, int noSamp, double &avg, double &stdev, double &var, double &snr, double &cv)
- void [CalculateDescriptivesRunning](#) (double &avg, double &stdev, double &var, double &snr, double &cv)
- deque< int > [CalculateDescriptivesStaticHistogram](#) (double *sig, int noSamp, double &avg, double &stdev, double &var, double &snr, double &cv)
- void [ConvolutionInputSide](#) (double *sig, int noSamp, double *impulseResponse, int irSize, double **outputSignal, int &osSize)
- void [ConvolutionOutputSide](#) (double *sig, int noSamp, double *impulseResponse, int irSize, double **outputSignal, int &osSize)
- void [FirstDifference](#) (double *sig, int noSamp, double **outputSignal, int &osSize)
- void [RunningSumIntegration](#) (double *sig, int noSamp, double **outputSignal, int &osSize)
- void [DiscreteFourierTransformationTimeSide](#) (double *timeSignal, int timeSigSize, double **realX, int &reXSize, double **imaginaryX, int &imXSize)
- void [DiscreteFourierTransformationFreqSide](#) (double *timeSignal, int timeSigSize, double **realX, int &reXSize, double **imaginaryX, int &imXSize)
- void [InverseDiscreteFourierTransformationFreqSide](#) (double *realX, int reXSize, double *imaginaryX, int imXSize, double **timeSignal, int &timeSigSize)
- void [InverseDiscreteFourierTransformationTimeSide](#) (double *realX, int reXSize, double *imaginaryX, int imXSize, double **timeSignal, int &timeSigSize)
- void [RectangularToPolarConversion](#) (double *realX, int reXSize, double *imaginaryX, int imXSize, double **mag, int &magSize, double **phase, int &phaseSize)
- void [PolarToRectangularConversion](#) (double *mag, int magSize, double *phase, int phaseSize, double **realX, int &reXSize, double **imaginaryX, int &imXSize)
- void [PhaseUnwrapping](#) (double *phase, int phaseSize, double **uwPhase, int &uwPhaseSize)

Private Attributes

- `Input *` [_input](#)
- `IOutput *` [_output](#)
- `double *` [signal](#)
- `int` [noSamples](#)
- `int` [smpFreq](#)

4.5.1 Constructor & Destructor Documentation

4.5.1.1 `DSPStatistic::DSPStatistic ()`

4.5.1.2 `DSPStatistic::DSPStatistic (Input * input, IOutput * output)`

4.5.2 Member Function Documentation

4.5.2.1 `void DSPStatistic::CalculateDescriptivesRunning (double & avg, double & stdev, double & var, double & snr, double & cv)` [private]

4.5.2.2 `void DSPStatistic::CalculateDescriptivesStatic (double * sig, int noSamp, double & avg, double & stdev, double & var, double & snr, double & cv)` [private]

4.5.2.3 `deque< int > DSPStatistic::CalculateDescriptivesStaticHistogram (double * sig, int noSamp, double & avg, double & stdev, double & var, double & snr, double & cv)` [private]

4.5.2.4 void DSPStatistic::ConvolutionInputSide (double * *sig*, int *noSamp*, double * *impulseResponse*, int *irSize*, double ** *outputSignal*, int & *osSize*) [private]

4.5.2.5 void DSPStatistic::ConvolutionOutputSide (double * *sig*, int *noSamp*, double * *impulseResponse*, int *irSize*, double ** *outputSignal*, int & *osSize*) [private]

4.5.2.6 void DSPStatistic::DiscreteFourierTransformationFreqSide (double * *timeSignal*, int *timeSigSize*, double ** *realX*, int & *reXSize*, double ** *imaginaryX*, int & *imXSize*) [private]

Decompose a time domain signal into sinusoids in frequency domain signals from frequency side. Frequency domain signals, held in *realX* and *imaginaryX*, are calculated from the time domain signal, held in *timeSignal*.

Parameters

| | | |
|-----|--------------------|---|
| in | <i>timeSignal</i> | Will hold the time domain signal |
| in | <i>timeSigSize</i> | Will hold the time domain signal size |
| out | <i>realX</i> | Holds the real part of the frequency domain |
| out | <i>reXSize</i> | Holds the real part of the frequency domain size |
| out | <i>imaginaryX</i> | Holds the imaginary part of the frequency domain |
| out | <i>imXSize</i> | Holds the imaginary part of the frequency domain size |

Decompose a time domain signal into sinusoids in frequency domain signals from time side. The program loops through each sample in the time domain, calculating the contribution of that point to the frequency domain. The overall frequency domain is found by adding the contributions from the individual time domain points.

Author

Petar Jerčić

Date

09/12/2012 Check correct sizes of the pek anh imk arrays.

Allocate array for real and imaginary parts of frequency domain

Zero out the *realX* and *imaginaryX* so it can be used as accumulators

Decomposition method loops through each sample in the time domain *timeSignal*, calculating the contribution of that point to the frequency domain.

4.5.2.7 void DSPStatistic::DiscreteFourierTransformationTimeSide (double * *timeSignal*, int *timeSigSize*, double ** *realX*, int & *reXSize*, double ** *imaginaryX*, int & *imXSize*) [private]

Decompose a time domain signal into sinusoids in frequency domain signals from time side. Frequency domain signals, held in *realX* and *imaginaryX*, are calculated from the time domain signal, held in *timeSignal*.

Parameters

| | | |
|-----|--------------------|---|
| in | <i>timeSignal</i> | Will hold the time domain signal |
| in | <i>timeSigSize</i> | Will hold the time domain signal size |
| out | <i>realX</i> | Holds the real part of the frequency domain |
| out | <i>reXSize</i> | Holds the real part of the frequency domain size |
| out | <i>imaginaryX</i> | Holds the imaginary part of the frequency domain |
| out | <i>imXSize</i> | Holds the imaginary part of the frequency domain size |

Decompose a time domain signal into sinusoids in frequency domain signals from time side. Describes how an individual sample in the frequency domain is affected by all of the samples in the time domain. That is, the program calculates each of the values in the frequency domain in succession, not as a group.

Author

Petar Jerčić

Date

09/12/2012 Check correct sizes of the pek anh imk arrays.

Alocate array for real and imaginary parts of frequency domain

Zero out the *realX* and *imaginaryX* so it can be used as accumulators

Decomposition method Correlate *timeSignal* with cosine and sine waves

4.5.2.8 void DSPStatistic::FirstDifference (double * sig, int noSamp, double ** outputSignal, int & osSize) [private]

4.5.2.9 void DSPStatistic::InverseDiscreteFourierTransformationFreqSide (double * realX, int reXSize, double * imaginaryX, int imXSize, double ** timeSignal, int & timeSigSize) [private]

Synthesis of time domain signal from frequency domain signals from frequency side. Time domain signal, held in *timeSignal*, is calculated from the frequency domain signal, held in *realX* and *imaginaryX*

Parameters

| | | |
|-----|--------------------|---|
| in | <i>realX</i> | Holds the real part of the frequency domain |
| in | <i>reXSize</i> | Holds the real part of the frequency domain size |
| in | <i>imaginaryX</i> | Holds the imaginary part of the frequency domain |
| in | <i>imXSize</i> | Holds the imaginary part of the frequency domain size |
| out | <i>timeSignal</i> | Will hold the time domain signal |
| out | <i>timeSigSize</i> | Will hold the time domain signal size |

Synthesis of time domain signal from frequency domain signals from frequency side. Each of the scaled sinusoids are generated one at a time and added to an accumulation array, which ends up becoming the time domain signal.

Author

Petar Jerčić

Date

09/12/2012 Check correct sizes of the pek anh imk arrays.

Alocate array for time domain signal (minus two extra data points)

Alocate array for cosine and sine wave amplitudes

Find the cosine and sine wave amplitudes

Cover two special cases

Zero out the *timeSignal* so it can be used as accumulator

Synthesis method Loop through each frequency generating the entire length of the sine and cosine waves, and add them to the accumulator signal *timeSignal*

4.5.2.10 void DSPStatistic::InverseDiscreteFourierTransformationTimeSide (double * realX, int reXSize, double * imaginaryX, int imXSize, double ** timeSignal, int & timeSigSize) [private]

Synthesis of time domain signal from frequency domain signals from time side. Time domain signal, held in *timeSignal*, is calculated from the frequency domain signal, held in *realX* and *imaginaryX*

Parameters

| | | |
|-----|--------------------|---|
| in | <i>realX</i> | Holds the real part of the frequency domain |
| in | <i>reXSize</i> | Holds the real part of the frequency domain size |
| in | <i>imaginaryX</i> | Holds the imaginary part of the frequency domain |
| in | <i>imXSize</i> | Holds the imaginary part of the frequency domain size |
| out | <i>timeSignal</i> | Will hold the time domain signal |
| out | <i>timeSigSize</i> | Will hold the time domain signal size |

Synthesis of time domain signal from frequency domain signals from time side. Each sample in the time domain signal is calculated one at a time, as the sum of all the corresponding samples in the cosine and sine waves.

Author

Petar Jerčić

Date

09/12/2012 N/A.

Alocate array for time domain signal (minus two extra data points)

Alocate array for cosine and sine wave amplitudes

Find the cosine and sine wave amplitudes

Cover two special cases

Zero out the *timeSignal* so it can be used as accumulator

Synthesis method Loop through each sample in the time domain and sum the corresponding samples from each cosine and sine waves.

4.5.2.11 void DSPStatistic::PhaseUnwrapping (double * *phase*, int *phaseSize*, double ** *uwPhase*, int & *uwPhaseSize*)
[private]

Unwrapping the phase It is often easier to understand the phase if it does not have these discontinuities, even if it means that the phase extends above , or below -.

Parameters

| | | |
|-----|--------------------|--------------------------------|
| in | <i>phase</i> | Holds the phase |
| in | <i>phaseSize</i> | Holds the phase size |
| out | <i>uwPhase</i> | Holds the unwrapped phase |
| out | <i>uwPhaseSize</i> | Holds the unwrapped phase size |

Unwrapping the phase. It is often easier to understand the phase if it does not have these discontinuities, even if it means that the phase extends above , or below -. a multiple of 2 is added or subtracted from each value of the phase.

Author

Petar Jerčić

Date

09/12/2012 N/A.

Alocate array for magnitude and phase parts of frequency domain

First point of all phase signals is 0

Go through the unwrappnig algorithm

4.5.2.12 void DSPStatistic::PolarToRectangularConversion (double * *mag*, int *magSize*, double * *phase*, int *phaseSize*, double ** *realX*, int & *reXSize*, double ** *imaginaryX*, int & *imXSize*) [private]

Polar To Rectangular Conversion From magnitude and phase to real and imaginary

Parameters

| | | |
|-----|-------------------|---|
| in | <i>mag</i> | Holds the magnitude |
| in | <i>magSize</i> | Holds the magnitude size |
| in | <i>phase</i> | Holds the phase |
| in | <i>phaseSize</i> | Holds the phase size |
| out | <i>realX</i> | Holds the real part of the frequency domain |
| out | <i>reXSize</i> | Holds the real part of the frequency domain size |
| out | <i>imaginaryX</i> | Holds the imaginary part of the frequency domain |
| out | <i>imXSize</i> | Holds the imaginary part of the frequency domain size |

Polar To Rectangular Conversion From magnitude and phase to real and imaginary

Author

Petar Jerčić

Date

09/12/2012 N/A.

Allocate array for magnitude and phase parts of frequency domain

Polar To Rectangular Conversion

4.5.2.13 void DSPStatistic::ProcesInput () [private]

4.5.2.14 void DSPStatistic::Process () [virtual]

Implements [IAnalysis](#).

4.5.2.15 void DSPStatistic::ProcesSignal () [private]

4.5.2.16 void DSPStatistic::RectangularToPolarConversion (double * *realX*, int *reXSize*, double * *imaginaryX*, int *imXSize*, double ** *mag*, int & *magSize*, double ** *phase*, int & *phaseSize*) [private]

Rectangular to polar Conversion From real and imaginary to magnitude and phase

Parameters

| | | |
|-----|-------------------|---|
| in | <i>realX</i> | Holds the real part of the frequency domain |
| in | <i>reXSize</i> | Holds the real part of the frequency domain size |
| in | <i>imaginaryX</i> | Holds the imaginary part of the frequency domain |
| in | <i>imXSize</i> | Holds the imaginary part of the frequency domain size |
| out | <i>mag</i> | Holds the magnitude |
| out | <i>magSize</i> | Holds the magnitude size |
| out | <i>phase</i> | Holds the phase |
| out | <i>phaseSize</i> | Holds the phase size |

Rectangular to polar Conversion. From real and imaginary to magnitude and phase.

Author

Petar Jerčić

Date

09/12/2012 N/A.

Alocate array for magnitude and phase parts of frequency domain

Rectangular to polar conversion

Prevent divide by zero

Correct arcus tangens ambiguity

4.5.2.17 void DSPStatistic::RunningSumIntegration (double * *sig*, int *noSamp*, double ** *outputSignal*, int & *osSize*)
[private]

4.5.2.18 void DSPStatistic::setInput (IInput * *input*) [virtual]

This method will be used to create random noise signal for other components to use.

Author

Petar Jerčić

Parameters

| | |
|-----------------|-----|
| <i>fileName</i> | N/A |
| <i>channel</i> | N/A |

Returns

N/A

See Also

something

Date

08/12/2012

Implements [IAnalysis](#).

4.5.2.19 void DSPStatistic::setOutput (IOutput * *output*) [virtual]

Implements [IAnalysis](#).

4.5.3 Member Data Documentation

4.5.3.1 IInput* DSPStatistic::_input [private]

4.5.3.2 IOutput* DSPStatistic::_output [private]

4.5.3.3 `int DSPStatistic::noSamples` `[private]`

4.5.3.4 `double* DSPStatistic::signal` `[private]`

4.5.3.5 `int DSPStatistic::smpFreq` `[private]`

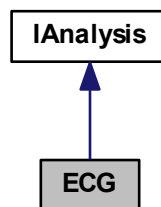
The documentation for this class was generated from the following files:

- [include/DSPStatistic.h](#)
- [source/DSPStatistic.cpp](#)

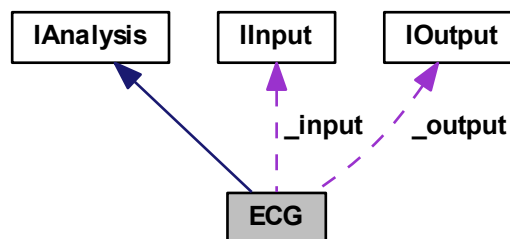
4.6 ECG Class Reference

```
#include <ECG.h>
```

Inheritance diagram for ECG:



Collaboration diagram for ECG:



Public Member Functions

- `void setInput (IInput *input)`
- `void setOutput (IOutput *output)`
- `void Process ()`

- [ECG \(\)](#)
- [ECG \(IInput *input, IOutput *output\)](#)

Private Member Functions

- void [ProcesInput \(\)](#)
- void [ProcesSignal \(\)](#)

Private Attributes

- [IInput * _input](#)
- [IOOutput * _output](#)

4.6.1 Constructor & Destructor Documentation

4.6.1.1 [ECG::ECG \(\)](#)

4.6.1.2 [ECG::ECG \(IInput * *input*, IOutput * *output* \)](#)

4.6.2 Member Function Documentation

4.6.2.1 [void ECG::ProcesInput \(\)](#) [private]

4.6.2.2 [void ECG::Process \(\)](#) [virtual]

Implements [IAnalysis](#).

4.6.2.3 [void ECG::ProcesSignal \(\)](#) [private]

4.6.2.4 [void ECG::setInput \(IInput * *input* \)](#) [virtual]

Implements [IAnalysis](#).

4.6.2.5 [void ECG::setOutput \(IOOutput * *output* \)](#) [virtual]

Implements [IAnalysis](#).

4.6.3 Member Data Documentation

4.6.3.1 [IInput* ECG::_input](#) [private]

4.6.3.2 [IOOutput* ECG::_output](#) [private]

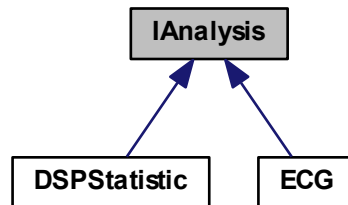
The documentation for this class was generated from the following files:

- include/[ECG.h](#)
- source/[ECG.cpp](#)

4.7 IAnalysis Class Reference

```
#include <IAnalysis.h>
```

Inheritance diagram for IAnalysis:



Public Member Functions

- virtual void [setInput](#) ([IInput](#) *input)=0
- virtual void [setOutput](#) ([IOutput](#) *input)=0
- virtual void [Process](#) ()=0

4.7.1 Member Function Documentation

4.7.1.1 virtual void [IAnalysis::Process](#) () [pure virtual]

Implemented in [DSPStatistic](#), and [ECG](#).

4.7.1.2 virtual void [IAnalysis::setInput](#) ([IInput](#) * *input*) [pure virtual]

Implemented in [DSPStatistic](#), and [ECG](#).

4.7.1.3 virtual void [IAnalysis::setOutput](#) ([IOutput](#) * *input*) [pure virtual]

Implemented in [DSPStatistic](#), and [ECG](#).

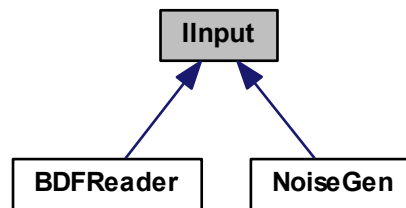
The documentation for this class was generated from the following file:

- include/[IAnalysis.h](#)

4.8 Input Class Reference

```
#include <IInput.h>
```

Inheritance diagram for IInput:



Public Member Functions

- virtual void [Open](#) (string fileName, int channel)=0
- virtual void [Close](#) ()=0
- virtual double * [getSignal](#) (int &noSamples, int &smpFreq)=0
- virtual double [getNextDataPoint](#) ()=0

4.8.1 Member Function Documentation

4.8.1.1 virtual void IInput::Close () [pure virtual]

Implemented in [BDFReader](#), and [NoiseGen](#).

4.8.1.2 virtual double IInput::getNextDataPoint () [pure virtual]

Implemented in [BDFReader](#), and [NoiseGen](#).

4.8.1.3 virtual double* IInput::getSignal (int & *noSamples*, int & *smpFreq*) [pure virtual]

Implemented in [BDFReader](#), and [NoiseGen](#).

4.8.1.4 virtual void IInput::Open (string *fileName*, int *channel*) [pure virtual]

Implemented in [BDFReader](#), and [NoiseGen](#).

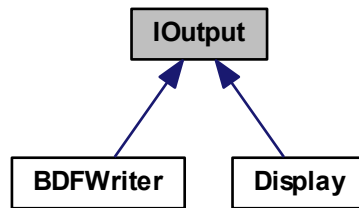
The documentation for this class was generated from the following file:

- [include/IInput.h](#)

4.9 IOutput Class Reference

```
#include <IOutput.h>
```

Inheritance diagram for IOutput:



Public Member Functions

- virtual void [saveSignal](#) (double *sig, int noSamp, int smp_freq, string sigLabel, string sigDim)=0
- virtual void [saveNextDataPoint](#) ()=0
- virtual void [DisplayMessage](#) (string message)=0
- virtual void [DisplayHistogram](#) (deque< int > histogram)=0

4.9.1 Member Function Documentation

4.9.1.1 virtual void IOutput::DisplayHistogram (deque< int > *histogram*) [pure virtual]

Implemented in [BDFWriter](#), and [Display](#).

4.9.1.2 virtual void IOutput::DisplayMessage (string *message*) [pure virtual]

Implemented in [BDFWriter](#), and [Display](#).

4.9.1.3 virtual void IOutput::saveNextDataPoint () [pure virtual]

Implemented in [BDFWriter](#), and [Display](#).

4.9.1.4 virtual void IOutput::saveSignal (double * *sig*, int *noSamp*, int *smp_freq*, string *sigLabel*, string *sigDim*) [pure virtual]

Implemented in [BDFWriter](#), and [Display](#).

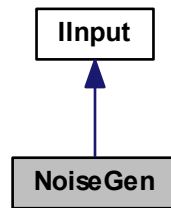
The documentation for this class was generated from the following file:

- include/[IOutput.h](#)

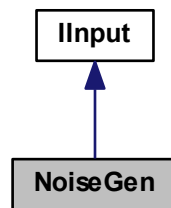
4.10 NoiseGen Class Reference

```
#include <NoiseGen.h>
```


Inheritance diagram for NoiseGen:



Collaboration diagram for NoiseGen:



Public Member Functions

- [NoiseGen](#) ()
- double * [getSignal](#) (int &noSamples, int &smpFreq)
- double [getNextDataPoint](#) ()
- void [Open](#) (string fileName, int channel)
- void [Close](#) ()

Private Attributes

- double * [rndSignalBuf](#)
- int [sampleCounter](#)
- int [noRndSamples](#)

4.10.1 Constructor & Destructor Documentation

4.10.1.1 [NoiseGen::NoiseGen](#) ()

4.10.2 Member Function Documentation

4.10.2.1 `void NoiseGen::Close () [virtual]`

Implements [IInput](#).

4.10.2.2 `double NoiseGen::getNextDataPoint () [virtual]`

Implements [IInput](#).

4.10.2.3 `double * NoiseGen::getSignal (int & noSamples, int & smpFreq) [virtual]`

Implements [IInput](#).

4.10.2.4 `void NoiseGen::Open (string fileName, int channel) [virtual]`

Implements [IInput](#).

4.10.3 Member Data Documentation

4.10.3.1 `int NoiseGen::noRndSamples [private]`

4.10.3.2 `double* NoiseGen::rndSignalBuf [private]`

4.10.3.3 `int NoiseGen::sampleCounter [private]`

The documentation for this class was generated from the following files:

- [include/NoiseGen.h](#)
- [source/NoiseGen.cpp](#)

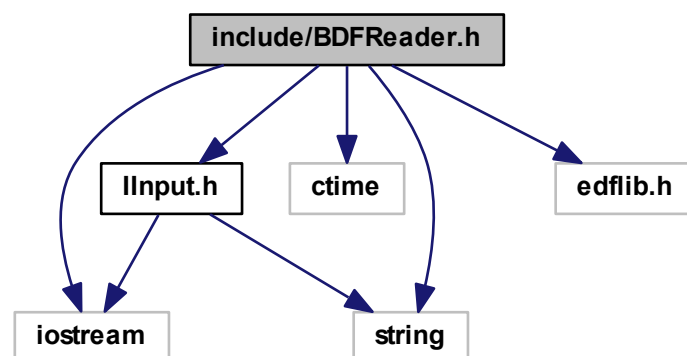
Chapter 5

File Documentation

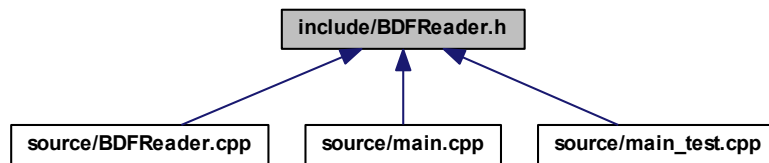
5.1 include/BDFReader.h File Reference

```
#include <iostream>
#include <string>
#include <ctime>
#include "IInput.h"
#include "edflib.h"
```

Include dependency graph for BDFReader.h:



This graph shows which files directly or indirectly include this file:



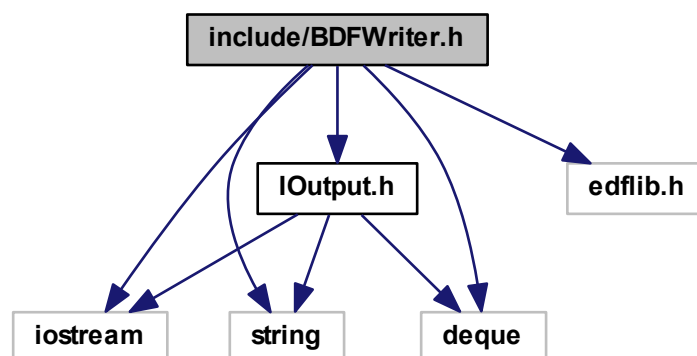
Classes

- class [BDFReader](#)

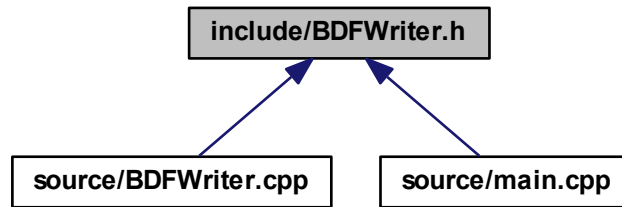
5.2 include/BDFWriter.h File Reference

```
#include <iostream>
#include <string>
#include <deque>
#include "IOutput.h"
#include <edflib.h>
```

Include dependency graph for BDFWriter.h:



This graph shows which files directly or indirectly include this file:



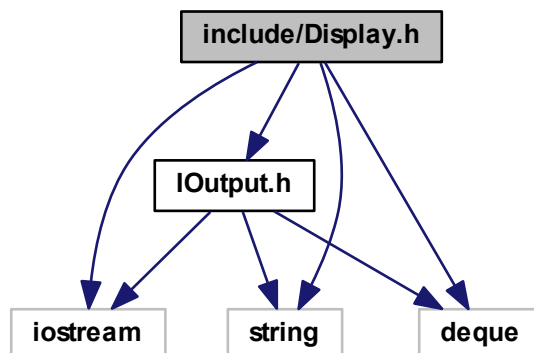
Classes

- class [BDFWriter](#)

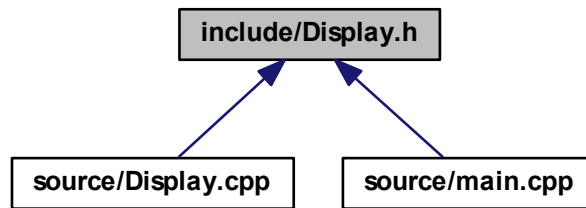
5.3 include/Display.h File Reference

```
#include <iostream>
#include <string>
#include <deque>
#include "IOutput.h"
```

Include dependency graph for Display.h:



This graph shows which files directly or indirectly include this file:



Classes

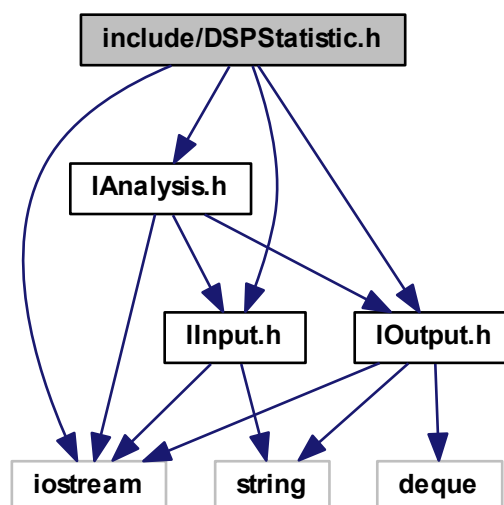
- class [Display](#)

5.4 include/DSPStatistic.h File Reference

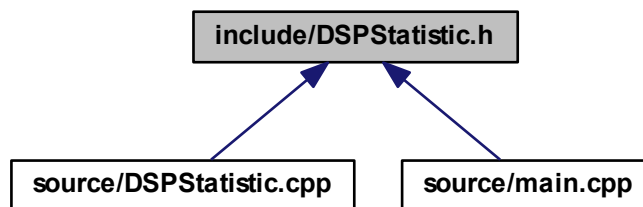
This header file defines an Analysis component using methods of DSP on signals.

```
#include <iostream>
#include "IInput.h"
#include "IOutput.h"
#include "IAnalysis.h"
```

Include dependency graph for DSPStatistic.h:



This graph shows which files directly or indirectly include this file:



Classes

- class [DSPStatistic](#)

5.4.1 Detailed Description

This header file defines an Analysis component using methods of DSP on signals.

Author

Petar Jerčić

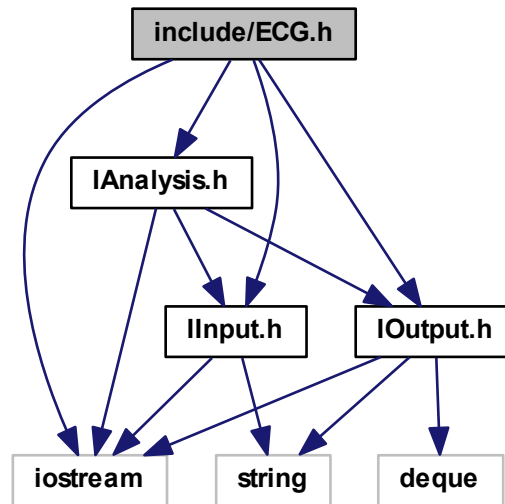
Date

09/12/2012

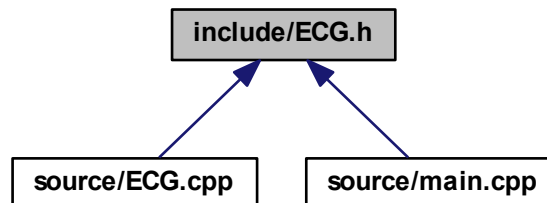
5.5 include/ECG.h File Reference

```
#include <iostream>
#include "IInput.h"
#include "IOutput.h"
#include "IAnalysis.h"
```

Include dependency graph for ECG.h:



This graph shows which files directly or indirectly include this file:



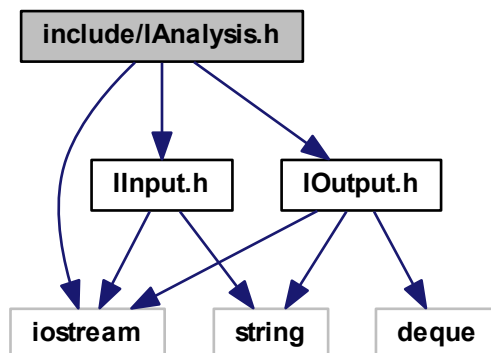
Classes

- class [ECG](#)

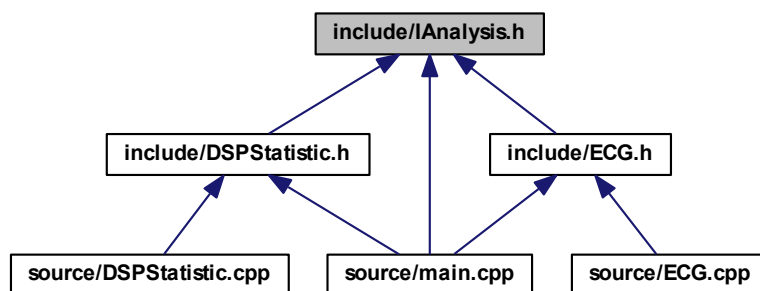
5.6 include/IAnalysis.h File Reference

```
#include <iostream>
#include "IInput.h"
#include "IOutput.h"
```


Include dependency graph for IAnalysis.h:



This graph shows which files directly or indirectly include this file:



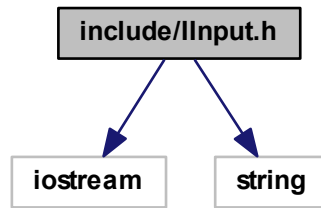
Classes

- class [IAnalysis](#)

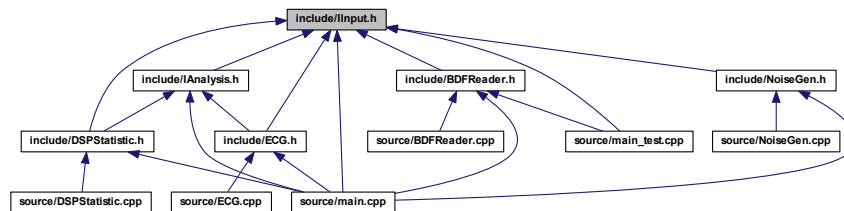
5.7 include/IInput.h File Reference

```
#include <iostream>
#include <string>
```

Include dependency graph for `lInput.h`:



This graph shows which files directly or indirectly include this file:



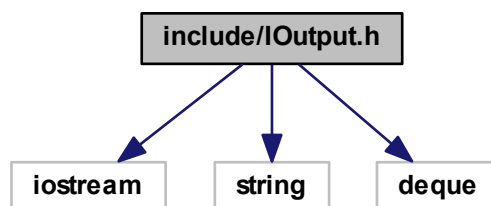
Classes

- class `lInput`

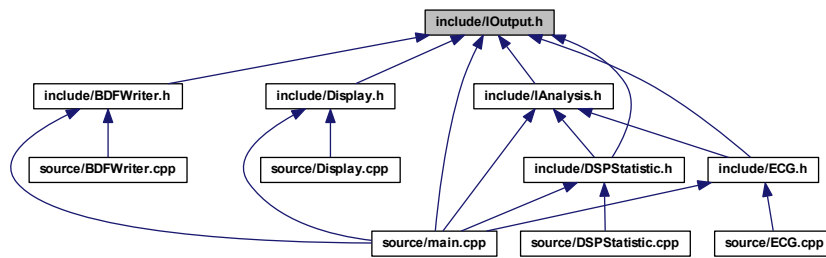
5.8 include/IOOutput.h File Reference

```
#include <iostream>
#include <string>
#include <deque>
```

Include dependency graph for `IOOutput.h`:



This graph shows which files directly or indirectly include this file:

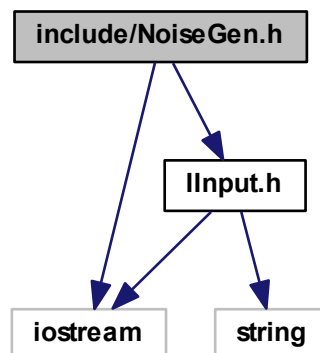


Classes

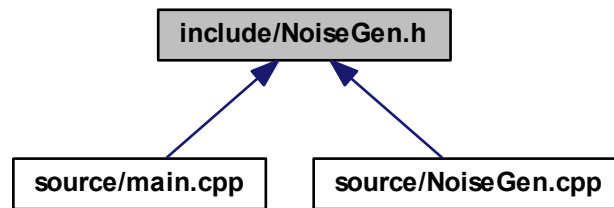
- class [IOutput](#)

5.9 include/NoiseGen.h File Reference

```
#include <iostream>
#include <IInput.h>
Include dependency graph for NoiseGen.h:
```



This graph shows which files directly or indirectly include this file:



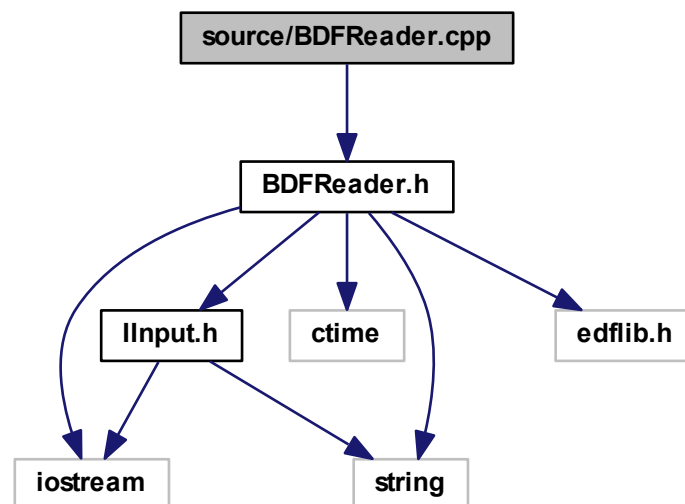
Classes

- class [NoiseGen](#)

5.10 source/BDFReader.cpp File Reference

```
#include "BDFReader.h"
```

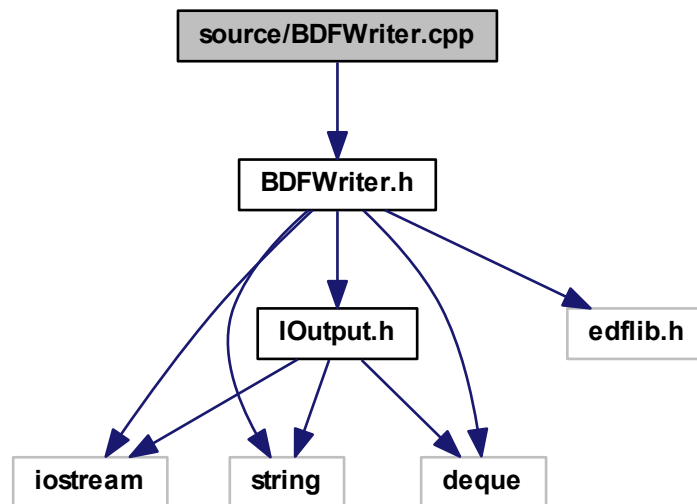
Include dependency graph for BDFReader.cpp:



5.11 source/BDFWriter.cpp File Reference

```
#include <BDFWriter.h>
```

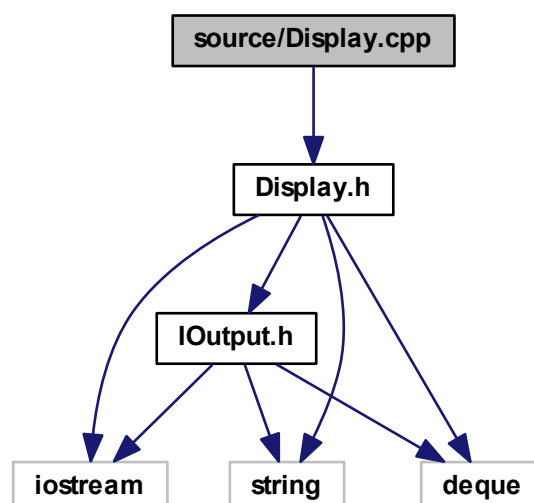
Include dependency graph for BDFWriter.cpp:



5.12 source/Display.cpp File Reference

```
#include "Display.h"
```

Include dependency graph for `Display.cpp`:

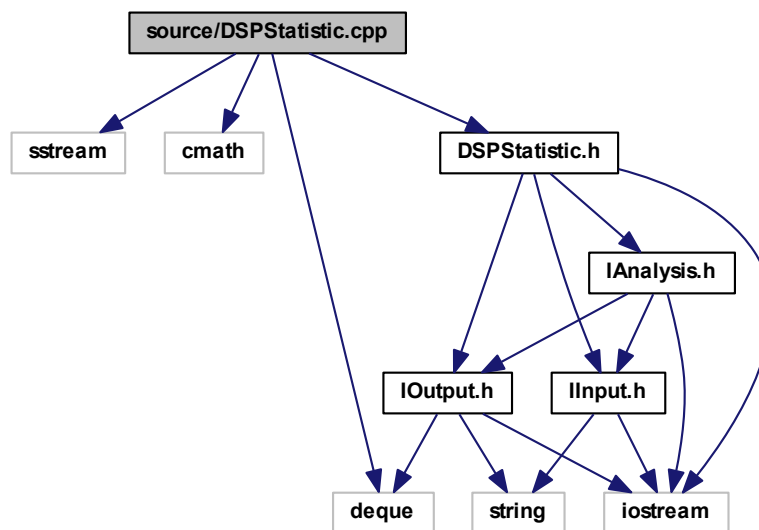


5.13 source/DSPStatistic.cpp File Reference

This source file defines an Analysis component using methods of DSP on signals.

```
#include <sstream>
#include <cmath>
#include <deque>
#include "DSPStatistic.h"
```

Include dependency graph for DSPStatistic.cpp:



5.13.1 Detailed Description

This source file defines an Analysis component using methods of DSP on signals.

Author

Petar Jerčić

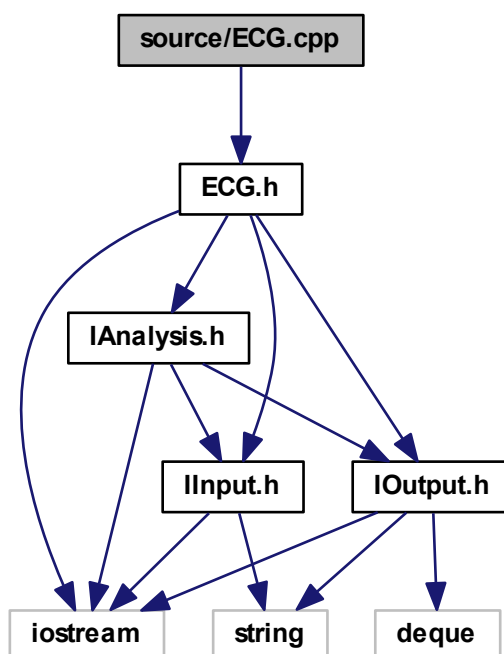
Date

09/12/2012

5.14 source/ECG.cpp File Reference

```
#include "ECG.h"
```

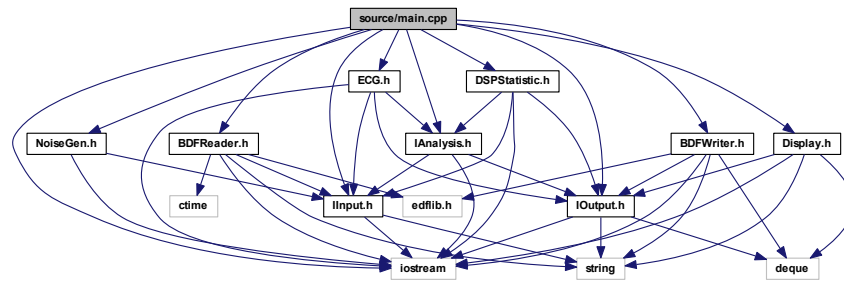
Include dependency graph for ECG.cpp:



5.15 source/main.cpp File Reference

```
#include <iostream>
#include "IInput.h"
#include "IOutput.h"
#include "IAnalysis.h"
#include "BDFReader.h"
#include "NoiseGen.h"
#include "ECG.h"
#include "DSPStatistic.h"
#include "Display.h"
#include <BDFWriter.h>
```

Include dependency graph for main.cpp:



Functions

- int [main](#) (int argc, char *argv[])

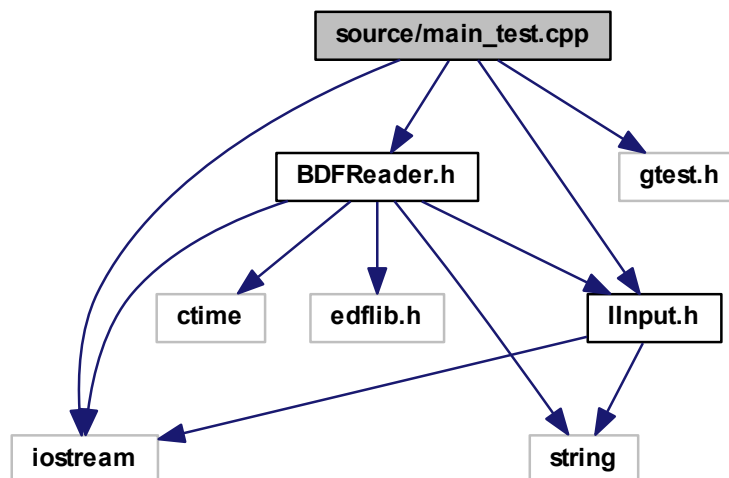
5.15.1 Function Documentation

5.15.1.1 int main (int argc, char * argv[])

5.16 source/main_test.cpp File Reference

```
#include <iostream>
#include <gtest.h>
#include <IInput.h>
#include <BDFReader.h>
```

Include dependency graph for main_test.cpp:



Functions

- [TEST](#) ([BDFReader](#), ComapareSignalData)

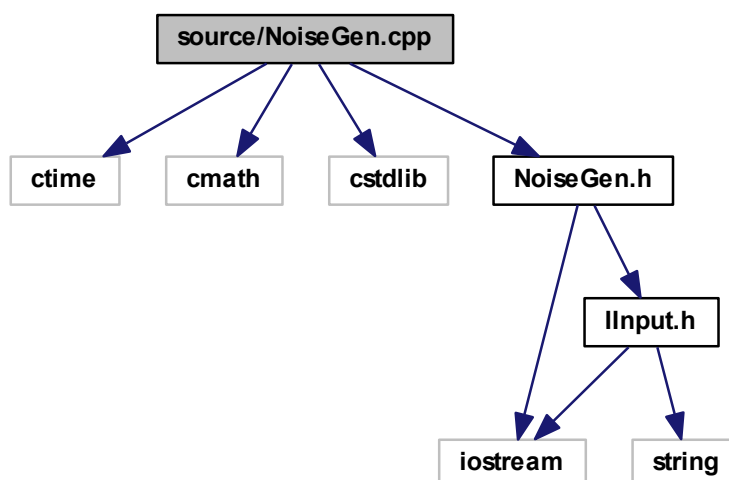
5.16.1 Function Documentation

5.16.1.1 TEST (BDFReader , ComapareSignalData)

5.17 source/NoiseGen.cpp File Reference

```
#include <ctime>
#include <cmath>
#include <cstdlib>
#include <NoiseGen.h>
```

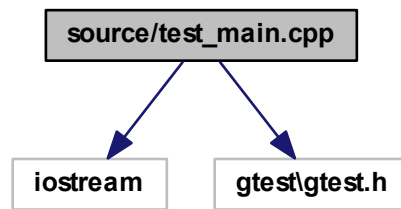
Include dependency graph for NoiseGen.cpp:



5.18 source/test_main.cpp File Reference

```
#include <iostream>
#include <gtest/gtest.h>
```

Include dependency graph for test_main.cpp:



Classes

- class [Adder](#)

Functions

- int [Add](#) (int *a*, int *b*)
- [TEST](#) (AddTest, ReturnValueEq)
- [TEST](#) (AddTest, ReturnValueDiff)

5.18.1 Function Documentation

5.18.1.1 int [Add](#) (int *a*, int *b*)

5.18.1.2 [TEST](#) (AddTest , ReturnValueEq)

5.18.1.3 [TEST](#) (AddTest , ReturnValueDiff)

Index

- `_defAddNo`
 - Adder, 7
 - `_input`
 - DSPStatistic, 19
 - ECG, 21
 - `_output`
 - DSPStatistic, 19
 - ECG, 21
- Add
 - Adder, 7
 - test_main.cpp, 42
- Adder, 7
 - `_defAddNo`, 7
 - Add, 7
 - Adder, 7
 - SetDefAddNo, 7
- BDFReader, 7
 - BDFReader, 9
 - BDFReader, 9
 - buf, 9
 - channelNo, 9
 - Close, 9
 - datetime, 9
 - getNextDataPoint, 9
 - getSignal, 9
 - hdr, 9
 - Open, 9
 - PrintHeader, 9
 - ReadData, 9
 - sampleCounter, 9
 - samplingFreq, 9
- BDFWriter, 9
 - BDFWriter, 10
 - BDFWriter, 10
 - DisplayHistogram, 11
 - DisplayMessage, 11
 - nosSamples, 11
 - saveNextDataPoint, 11
 - saveSignal, 11
 - sigDim, 11
 - sigLabels, 11
 - signals, 11
 - smpFreqs, 11
- buf
 - BDFReader, 9
- CalculateDescriptivesRunning
 - DSPStatistic, 14
- CalculateDescriptivesStatic
 - DSPStatistic, 14
- CalculateDescriptivesStaticHistogram
 - DSPStatistic, 14
- channelNo
 - BDFReader, 9
- Close
 - BDFReader, 9
 - Input, 23
 - NoiseGen, 25
- ConvolutionInputSide
 - DSPStatistic, 14
- ConvolutionOutputSide
 - DSPStatistic, 15
- DSPStatistic, 13
 - `_input`, 19
 - `_output`, 19
 - CalculateDescriptivesRunning, 14
 - CalculateDescriptivesStatic, 14
 - CalculateDescriptivesStaticHistogram, 14
 - ConvolutionInputSide, 14
 - ConvolutionOutputSide, 15
 - DSPStatistic, 14
 - DiscreteFourierTransformationFreqSide, 15
 - DiscreteFourierTransformationTimeSide, 15
 - DSPStatistic, 14
 - FirstDifference, 16
 - InverseDiscreteFourierTransformationFreqSide, 16
 - InverseDiscreteFourierTransformationTimeSide, 16
 - noSamples, 19
 - PhaseUnwrapping, 17
 - PolarToRectangularConversion, 17
 - ProcesInput, 18
 - ProcessSignal, 18
 - Process, 18
 - RectangularToPolarConversion, 18
 - RunningSumIntegration, 19
 - setInput, 19
 - setOutput, 19
 - signal, 20
 - smpFreq, 20
- datetime
 - BDFReader, 9
- DiscreteFourierTransformationFreqSide
 - DSPStatistic, 15
- DiscreteFourierTransformationTimeSide
 - DSPStatistic, 15
- Display, 11
 - DisplayHistogram, 12

- DisplayMessage, 12
 - saveNextDataPoint, 12
 - saveSignal, 12
- DisplayHistogram
 - BDFWriter, 11
 - Display, 12
 - IOutput, 24
- DisplayMessage
 - BDFWriter, 11
 - Display, 12
 - IOutput, 24
- ECG, 20
 - _input, 21
 - _output, 21
 - ECG, 21
 - ECG, 21
 - ProcesInput, 21
 - ProcesSignal, 21
 - Process, 21
 - setInput, 21
 - setOutput, 21
- FirstDifference
 - DSPStatistic, 16
- getNextDataPoint
 - BDFReader, 9
 - IInput, 23
 - NoiseGen, 26
- getSignal
 - BDFReader, 9
 - IInput, 23
 - NoiseGen, 26
- hdr
 - BDFReader, 9
- IAnalysis, 22
 - Process, 22
 - setInput, 22
 - setOutput, 22
- IInput, 22
 - Close, 23
 - getNextDataPoint, 23
 - getSignal, 23
 - Open, 23
- IOutput, 23
 - DisplayHistogram, 24
 - DisplayMessage, 24
 - saveNextDataPoint, 24
 - saveSignal, 24
- include/BDFReader.h, 27
- include/BDFWriter.h, 28
- include/DSPStatistic.h, 30
- include/Display.h, 29
- include/ECG.h, 31
- include/IAnalysis.h, 32
- include/IInput.h, 33
- include/IOutput.h, 34
- include/NoiseGen.h, 35
- InverseDiscreteFourierTransformationFreqSide
 - DSPStatistic, 16
- InverseDiscreteFourierTransformationTimeSide
 - DSPStatistic, 16
- main
 - main.cpp, 40
- main.cpp
 - main, 40
- main_test.cpp
 - TEST, 41
- noRndSamples
 - NoiseGen, 26
- noSamples
 - DSPStatistic, 19
- NoiseGen, 24
 - Close, 25
 - getNextDataPoint, 26
 - getSignal, 26
 - noRndSamples, 26
 - NoiseGen, 25
 - NoiseGen, 25
 - Open, 26
 - rndSignalBuf, 26
 - sampleCounter, 26
- nosSamples
 - BDFWriter, 11
- Open
 - BDFReader, 9
 - IInput, 23
 - NoiseGen, 26
- PhaseUnwrapping
 - DSPStatistic, 17
- PolarToRectangularConversion
 - DSPStatistic, 17
- PrintHeader
 - BDFReader, 9
- ProcesInput
 - DSPStatistic, 18
 - ECG, 21
- ProcesSignal
 - DSPStatistic, 18
 - ECG, 21
- Process
 - DSPStatistic, 18
 - ECG, 21
 - IAnalysis, 22
- ReadData
 - BDFReader, 9
- RectangularToPolarConversion
 - DSPStatistic, 18
- rndSignalBuf
 - NoiseGen, 26

- RunningSumIntegration
 - DSPStatistic, [19](#)
- sampleCounter
 - BDFReader, [9](#)
 - NoiseGen, [26](#)
- samplingFreq
 - BDFReader, [9](#)
- saveNextDataPoint
 - BDFWriter, [11](#)
 - Display, [12](#)
 - IOutput, [24](#)
- saveSignal
 - BDFWriter, [11](#)
 - Display, [12](#)
 - IOutput, [24](#)
- SetDefAddNo
 - Adder, [7](#)
- setInput
 - DSPStatistic, [19](#)
 - ECG, [21](#)
 - IAnalysis, [22](#)
- setOutput
 - DSPStatistic, [19](#)
 - ECG, [21](#)
 - IAnalysis, [22](#)
- sigDim
 - BDFWriter, [11](#)
- sigLabels
 - BDFWriter, [11](#)
- signal
 - DSPStatistic, [20](#)
- signals
 - BDFWriter, [11](#)
- smpFreq
 - DSPStatistic, [20](#)
- smpFreqs
 - BDFWriter, [11](#)
- source/BDFReader.cpp, [36](#)
- source/BDFWriter.cpp, [36](#)
- source/DSPStatistic.cpp, [38](#)
- source/Display.cpp, [37](#)
- source/ECG.cpp, [39](#)
- source/NoiseGen.cpp, [41](#)
- source/main.cpp, [39](#)
- source/main_test.cpp, [40](#)
- source/test_main.cpp, [41](#)
- TEST
 - main_test.cpp, [41](#)
 - test_main.cpp, [42](#)
- test_main.cpp
 - Add, [42](#)
 - TEST, [42](#)