iir1

Generated by Doxygen 1.8.13

Contents

1 IIR1 -- Realtime C++ filter library

An infinite impulse response (IIR) filter library for Linux, Mac OSX and Windows which implements Bessel, Butterworth, RBJ and Chebychev filters.

The filter processes the data sample by sample for realtime processing.

It uses templates to allocate the required memory so that it can run without any malloc / new commands for example on embedded systems.

How to use the filter

First the filter is instantiated, then the parameters are set with the function `setup` and then it's ready to be used for sample by sample realtime filtering.

Setting the filter parameters

All filters are available as lowpass, highpass, bandpass and bandstop/notch filters. Butterworth / Chebyshev offer also low/high/band-shelves with specified passband gain and 0dB gain in the stopband.

See the header files in `\iir` or the documentation for the arguments of the `setup` commands.

The examples below are for lowpass filters:

1. Butterworth – `Butterworth.h` Standard filter suitable for most applications. Monotonic response.

```
const int order = 4; // 4th order (=2 biquads)
Iir::Butterworth::LowPass<order> f;
const float samplingrate = 1000; // Hz
const float cutoff_frequency = 5; // Hz
f.setup (samplingrate, cutoff_frequency);
```

2. Chebyshev Type I – `ChebyshevI.h` With permissible passband ripple in dB.

```
Iir::ChebyshevI::LowPass<order> f;
const float passband_ripple_in_db = 5;
f.setup (samplingrate,
         cutoff_frequency,
         passband_ripple_in_dB);
```

3. Chebyshev Type II – `ChebyshevII.h` With worst permissible stopband rejection in dB.

```
Iir::ChebyshevII::LowPass<order> f;
double stopband_ripple_in_dB = 20;
f.setup (samplingrate,
         cutoff_frequency,
         stopband_ripple_in_dB);
```

4. RBJ – `RBJ.h` 2nd order filters with cutoff and Q factor.

```
Iir::RBJ::LowPass f;
const float cutoff_frequency = 100;
const float Q_factor = 5;
f.setup (samplingrate, cutoff_frequency, Q_factor);
```

Realtime filtering sample by sample

Samples are processed one by one. In the example below a sample `x` is processed with the `filter` command and then saved in `y`. The type of `x` can either be float or double:

```
float y = f.filter(x);
```

This is then repeated for every incoming sample in a loop or event handler.

Packages for Ubuntu (xenial / bionic):

If you have Ubuntu xenial or bionic then install it as a pre-compiled package:

```
sudo add-apt-repository ppa:berndporr/usbdux
```

It's available for 32,64 bit PC and 32,64 bit ARM (Raspberry PI etc)

Compilation from source

The build tool is `cmake` which generates the make- or project files for the different platforms. `cmake` is available for Linux, Windows and Mac. It also compiles directly on a Raspberry PI.

Linux / Mac

Run

```
cmake .
```

which generates the Makefile. Then run:

```
make
sudo make install
```

which installs it under `/usr/local/lib` and `/usr/local/include`.

Both gcc and clang have been tested.

Windows

```
cmake -G "Visual Studio 15 2017 Win64" .
```

See `cmake` for the different build-options. Above is for a 64 bit build. Then start Visual C++ and open the solution. This will create the DLL and the LIB files. Under Windows it's highly recommended to use the static library and link it into the application program.

Unit tests

Run unit tests by typing `make test` or just `ctest`. These test if after a delta pulse all filters relax to zero and that their outputs never become NaN.

Documentation

Overview

For an overview of the class structure and general concepts have a look at Documentation.txt.

Learn from the demos

The easiest way to learn is from the examples which are in the `demo` directory. A delta pulse as a test signal is sent into the different filters and saved in a file. With the Python script `plot_impulse_fresponse.py` you can then plot the frequency responses.

Also the directory containing the unit tests provides examples for every filter type.

Detailed documentation

A PDF of all classes, methods and in particular `setup` functions is in the `doc/pdf` directory.

Run `doxygen` to generate the HTML documentation.

Example filter responses

These are the filter responses generated by `iirdemo.cpp` in the `/demo/` directory and then plotted with `plot↩ _impulse_fresponse.py`.

Credits

This library has been adapted form Vinnie Falco's original work which can be found here:

https://github.com/vinniefalco/DSPFilters

While his original library processes audio arrays this library has been adapted to do realtime processing sample by sample. Also, in contrast to the original library the `setup` command won't require the filter order and instead remembers it from the template argument.

Enjoy!

Bernd Porr – http://www.berndporr.me.uk

2 Namespace Index

2.1 Namespace List

Here is a list of all documented namespaces with brief descriptions:

3 Hierarchical Index

3.1 Class Hierarchy

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

4 Class Index

4.1 Class List

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

5 Namespace Documentation

5.1 Iir Namespace Reference

Namespaces

- Bessel
- Butterworth
- ChebyshevI
- ChebyshevII
- RBJ

Classes

- class BandPassTransform
- class BandStopTransform
- class Biquad
- class BiquadBase
- struct BiquadPoleState
- class Cascade
- class CascadeStages
- struct ComplexPair
- class DirectFormI
- class DirectFormII
- class EnvelopeFollower
- class HighPassTransform
- class Layout
- class LayoutBase
- class LowPassTransform
- struct PoleFilter
- class PoleFilterBase
- class PoleFilterBase2
- struct PoleZeroPair
- struct RootFinder
- class RootFinderBase
- class SlopeDetector
- class TransposedDirectFormII

Enumerations

- enum Kind

Functions

- template<class Td , class Ts >
  void add (int samples, Td ∗dest, Ts const ∗src, int destSkip=0, int srcSkip=0)
- template<typename Td , typename Ts >
  void add (int channels, int samples, Td ∗const ∗dest, Ts const ∗const ∗src)
- template<typename Td , typename Ts >
  void copy (int samples, Td ∗dest, Ts const ∗src, int destSkip=0, int srcSkip=0)

5.1.1 Detailed Description

"A Collection of Useful C++ Classes for Digital Signal Processing" By Vinnie Falco and Bernd Porr

Official project location: `https://github.com/berndporr/iir1`

See Documentation.cpp for contact information, notes, and bibliography.

"A Collection of Useful C++ Classes for Digital Signal Processing" By Vinnie Falco and Bernd Porr

Official project location: `https://github.com/berndporr/iir1`

See Documentation.cpp for contact information, notes, and bibliography.

Describes a filter as a collection of poles and zeros along with normalization information to achieve a specified gain at a specified frequency. The poles and zeros may lie either in the s or the z plane.

5.1.2 Enumeration Type Documentation

5.1.2.1 Kind

`enum Iir::Kind`

Identifies the general class of filter

5.1.3 Function Documentation

5.1.3.1 add() [1/2]

```
template<class Td , class Ts >
void Iir::add (
             int samples,
             Td * dest,
             Ts const * src,
             int destSkip = 0,
             int srcSkip = 0 )
```

Utilities

These routines are handy for manipulating buffers of samples. Add src samples to dest, without clip or overflow checking.

5.1.3.2 add() [2/2]

```
template<typename Td , typename Ts >
void Iir::add (
             int channels,
             int samples,
             Td *const * dest,
             Ts const *const * src )
```

Multichannel add

5.1.3.3 copy()

```
template<typename Td , typename Ts >
void Iir::copy (
             int samples,
             Td * dest,
             Ts const * src,
             int destSkip = 0,
             int srcSkip = 0 )
```

Copy samples from src to dest, which may not overlap. Performs an implicit type conversion if Ts and Td are different (for example, float to double).

5.2 Iir::Bessel Namespace Reference

Classes

- class AnalogLowPass
- class AnalogLowShelf
- struct BandPass
- struct BandPassBase
- struct BandStop
- struct BandStopBase
- struct HighPass
- struct HighPassBase
- struct LowPass
- struct LowPassBase
- struct LowShelfBase
- struct Workspace
- struct WorkspaceBase

5.2.1 Detailed Description

Filters with Bessel response characteristics

5.3 Iir::Butterworth Namespace Reference

Classes

- class AnalogLowPass
- class AnalogLowShelf
- struct BandPass
- struct BandPassBase
- struct BandShelf
- struct BandShelfBase
- struct BandStop
- struct BandStopBase
- struct HighPass
- struct HighPassBase
- struct HighShelf
- struct HighShelfBase
- struct LowPass
- struct LowPassBase
- struct LowShelf
- struct LowShelfBase

5.3.1 Detailed Description

Filters with Butterworth response characteristics

5.4 Iir::ChebyshevI Namespace Reference

Classes

- class AnalogLowPass
- class AnalogLowShelf
- struct BandPass
- struct BandPassBase
- struct BandShelf
- struct BandShelfBase
- struct BandStop
- struct BandStopBase
- struct HighPass
- struct HighPassBase
- struct HighShelf
- struct HighShelfBase
- struct LowPass
- struct LowPassBase
- struct LowShelf
- struct LowShelfBase

5.4.1 Detailed Description

Filters with Chebyshev response characteristics. The last parameter defines the passband ripple in decibel.

5.5 Iir::ChebyshevII Namespace Reference

Classes

- class AnalogLowPass
- class AnalogLowShelf
- struct BandPass
- struct BandPassBase
- struct BandShelf
- struct BandShelfBase
- struct BandStop
- struct BandStopBase
- struct HighPass
- struct HighPassBase
- struct HighShelf
- struct HighShelfBase
- struct LowPass
- struct LowPassBase
- struct LowShelf
- struct LowShelfBase

5.5.1 Detailed Description

Filters with ChebyshevII response characteristics. The last parameter defines the minimal stopband rejection requested. Generally there will be frequencies where the rejection is much better but this parameter guarantees that the rejection is at least as specified.

5.6 Iir::RBJ Namespace Reference

Classes

- struct AllPass
- struct BandPass1
- struct BandPass2
- struct BandShelf
- struct BandStop
- struct HighPass
- struct HighShelf
- struct IIRNotch
- struct LowPass
- struct LowShelf
- struct RBJbase

5.6.1 Detailed Description

Filter realizations based on Robert Bristol-Johnson formulae:

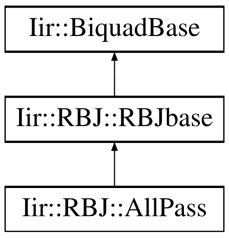
http://www.musicdsp.org/files/Audio-EQ-Cookbook.txt

These are all 2nd order filters which are tuned with the Q (or Quality factor). The Q factor causes a resonance at the cutoff frequency. The higher the Q factor the higher the responance. If 0.5 $<$ Q $<$ 1/sqrt(2) then there is no resonance peak. Above 1/sqrt(2) the peak becomes more and more pronounced. For bandpass and stopband the Q factor is replaced by the width of the filter. The higher Q the more narrow the bandwidth of the notch or bandpass.

6 Class Documentation

6.1 Iir::RBJ::AllPass Struct Reference

Inheritance diagram for Iir::RBJ::AllPass:



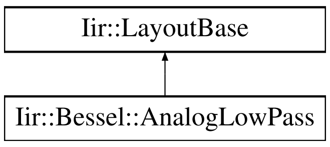
Additional Inherited Members

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

- iir/RBJ.h
- iir/RBJ.cpp

6.2 Iir::Bessel::AnalogLowPass Class Reference

Inheritance diagram for Iir::Bessel::AnalogLowPass:

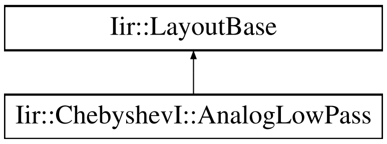


The documentation for this class was generated from the following files:

- iir/Bessel.h
- iir/Bessel.cpp

6.3 Iir::ChebyshevI::AnalogLowPass Class Reference

Inheritance diagram for Iir::ChebyshevI::AnalogLowPass:



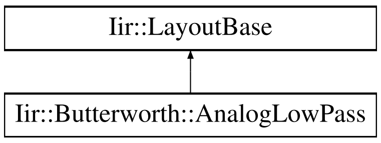
The documentation for this class was generated from the following files:

- iir/ChebyshevI.h
- iir/ChebyshevI.cpp

6.4 Iir::Butterworth::AnalogLowPass Class Reference

```
#include <Butterworth.h>
```

Inheritance diagram for Iir::Butterworth::AnalogLowPass:



6.4.1 Detailed Description

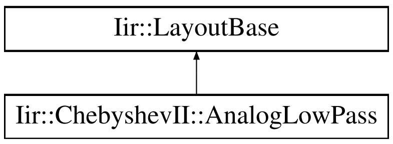
Half-band analog prototypes (s-plane)

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

- iir/Butterworth.h
- iir/Butterworth.cpp

6.5 Iir::ChebyshevII::AnalogLowPass Class Reference

Inheritance diagram for Iir::ChebyshevII::AnalogLowPass:

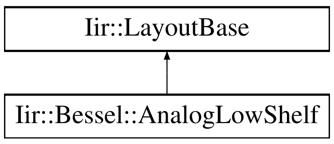


The documentation for this class was generated from the following files:

- iir/ChebyshevII.h
- iir/ChebyshevII.cpp

6.6 Iir::Bessel::AnalogLowShelf Class Reference

Inheritance diagram for Iir::Bessel::AnalogLowShelf:

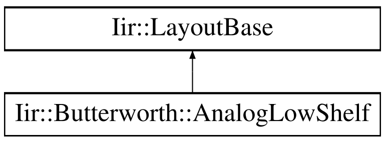


The documentation for this class was generated from the following files:

- iir/Bessel.h
- iir/Bessel.cpp

6.7 Iir::Butterworth::AnalogLowShelf Class Reference

Inheritance diagram for Iir::Butterworth::AnalogLowShelf:

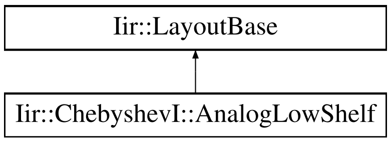


The documentation for this class was generated from the following files:

- iir/Butterworth.h
- iir/Butterworth.cpp

6.8 Iir::ChebyshevI::AnalogLowShelf Class Reference

Inheritance diagram for Iir::ChebyshevI::AnalogLowShelf:

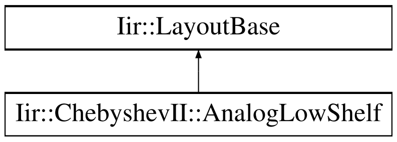


The documentation for this class was generated from the following files:

- iir/ChebyshevI.h
- iir/ChebyshevI.cpp

6.9 Iir::ChebyshevII::AnalogLowShelf Class Reference

Inheritance diagram for Iir::ChebyshevII::AnalogLowShelf:



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

- iir/ChebyshevII.h
- iir/ChebyshevII.cpp

6.10 Iir::RootFinderBase::Array Struct Reference

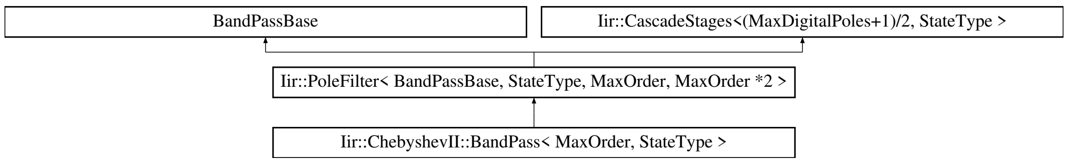
The documentation for this struct was generated from the following file:

- iir/RootFinder.h

6.11 Iir::ChebyshevII::BandPass< MaxOrder, StateType > Struct Template Reference

```
#include <ChebyshevII.h>
```

Inheritance diagram for Iir::ChebyshevII::BandPass< MaxOrder, StateType >:


BandPassBase
Iir::CascadeStages<(MaxDigitalPoles+1)/2, StateType >
Iir::PoleFilter< BandPassBase, StateType, MaxOrder, MaxOrder *2 >
Iir::ChebyshevII::BandPass< MaxOrder, StateType >

Public Member Functions

- void setup (double sampleRate, double centerFrequency, double widthFrequency, double stopBandDb)

6.11.1 Detailed Description

template<int MaxOrder, class StateType = DEFAULT_STATE>
struct Iir::ChebyshevII::BandPass< MaxOrder, StateType >

ChebyshevII bandpass filter

6.11.2 Member Function Documentation

6.11.2.1 setup()

```
template<int MaxOrder, class StateType = DEFAULT_STATE>
void Iir::ChebyshevII::BandPass< MaxOrder, StateType >::setup (
             double sampleRate,
             double centerFrequency,
             double widthFrequency,
             double stopBandDb )  [inline]
```

Calculates the coefficients of the filter

Parameters

sampleRate	Sampling rate
centerFrequency	Center frequency of the bandpass
widthFrequency	Width of the bandpass
stopBandDb	Permitted ripples in dB in the stopband

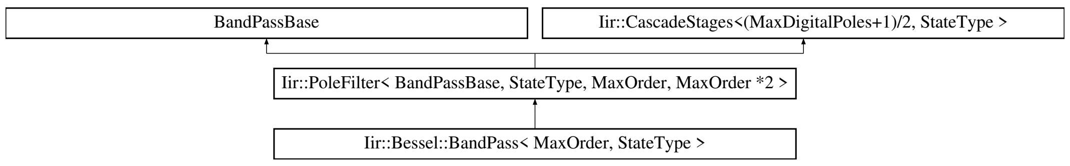
The documentation for this struct was generated from the following file:

- iir/ChebyshevII.h

6.12 Iir::Bessel::BandPass< MaxOrder, StateType > Struct Template Reference

```
#include <Bessel.h>
```

Inheritance diagram for Iir::Bessel::BandPass< MaxOrder, StateType >:


BandPassBase
Iir::CascadeStages<(MaxDigitalPoles+1)/2, StateType >
Iir::PoleFilter< BandPassBase, StateType, MaxOrder, MaxOrder *2 >
Iir::Bessel::BandPass< MaxOrder, StateType >

Public Member Functions

- void setup (double sampleRate, double centerFrequency, double widthFrequency)

6.12.1 Detailed Description

template<int MaxOrder, class StateType = DEFAULT_STATE>
struct Iir::Bessel::BandPass< MaxOrder, StateType >

Bessel bandpass.

6.12.2 Member Function Documentation

6.12.2.1 setup()

```
template<int MaxOrder, class StateType = DEFAULT_STATE>
void Iir::Bessel::BandPass< MaxOrder, StateType >::setup (
            double sampleRate,
            double centerFrequency,
            double widthFrequency )  [inline]
```

Calculate the coefficients

Parameters

sampleRate	Sampling rate
centerFrequency	Center frequency of the bandpass in Hz
widthFrequency	Width of the bandpass in Hz

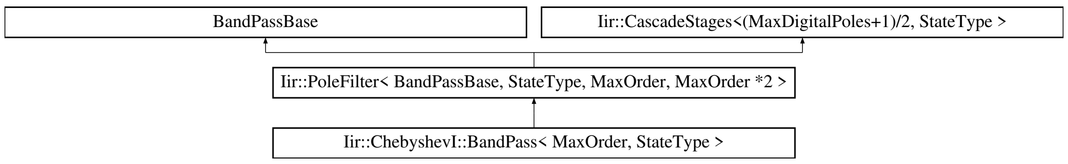
The documentation for this struct was generated from the following file:

- iir/Bessel.h

6.13 Iir::ChebyshevI::BandPass< MaxOrder, StateType > Struct Template Reference

```
#include <ChebyshevI.h>
```

Inheritance diagram for Iir::ChebyshevI::BandPass< MaxOrder, StateType >:


BandPassBase
Iir::CascadeStages<(MaxDigitalPoles+1)/2, StateType >
Iir::PoleFilter< BandPassBase, StateType, MaxOrder, MaxOrder *2 >
Iir::ChebyshevI::BandPass< MaxOrder, StateType >

Public Member Functions

- void setup (double sampleRate, double centerFrequency, double widthFrequency, double rippleDb)

6.13.1 Detailed Description

template<int MaxOrder, class StateType = DEFAULT_STATE>
struct Iir::ChebyshevI::BandPass< MaxOrder, StateType >

ChebyshevI bandpass filter

6.13.2 Member Function Documentation

6.13.2.1 setup()

```
template<int MaxOrder, class StateType = DEFAULT_STATE>
void Iir::ChebyshevI::BandPass< MaxOrder, StateType >::setup (
             double sampleRate,
             double centerFrequency,
             double widthFrequency,
             double rippleDb )  [inline]
```

Calculates the coefficients of the filter

Parameters

sampleRate	Sampling rate
centerFrequency	Center frequency of the bandpass
widthFrequency	Frequency with of the passband
rippleDb	Permitted ripples in dB in the passband

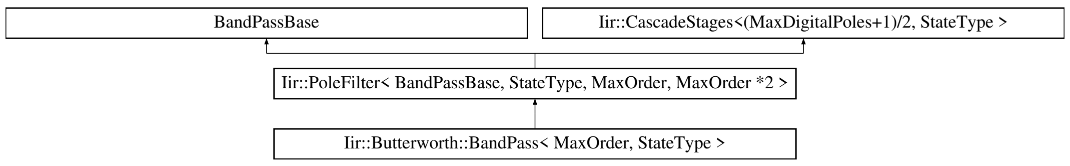
The documentation for this struct was generated from the following file:

- iir/ChebyshevI.h

6.14 Iir::Butterworth::BandPass< MaxOrder, StateType > Struct Template Reference

```
#include <Butterworth.h>
```

Inheritance diagram for Iir::Butterworth::BandPass< MaxOrder, StateType >:


BandPassBase
Iir::CascadeStages<(MaxDigitalPoles+1)/2, StateType >
Iir::PoleFilter< BandPassBase, StateType, MaxOrder, MaxOrder *2 >
Iir::Butterworth::BandPass< MaxOrder, StateType >

Public Member Functions

- void setup (double sampleRate, double centerFrequency, double widthFrequency)

6.14.1 Detailed Description

template<int MaxOrder, class StateType = DEFAULT_STATE>
struct Iir::Butterworth::BandPass< MaxOrder, StateType >

Butterworth Bandpass filter.

6.14.2 Member Function Documentation

6.14.2.1 setup()

```
template<int MaxOrder, class StateType = DEFAULT_STATE>
void Iir::Butterworth::BandPass< MaxOrder, StateType >::setup (
             double sampleRate,
             double centerFrequency,
             double widthFrequency )  [inline]
```

Calculates the coefficients

Parameters

sampleRate	Sampling rate
centerFrequency	Centre frequency of the bandpass
widthFrequency	Width of the bandpass

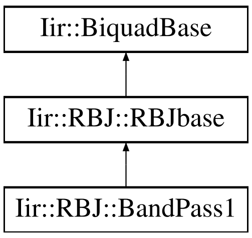
The documentation for this struct was generated from the following file:

- iir/Butterworth.h

6.15 Iir::RBJ::BandPass1 Struct Reference

```
#include <RBJ.h>
```

Inheritance diagram for Iir::RBJ::BandPass1:



Public Member Functions

- void setup (double sampleRate, double centerFrequency, double bandWidth)

6.15.1 Detailed Description

Bandpass with constant skirt gain

6.15.2 Member Function Documentation

6.15.2.1 setup()

```
void Iir::RBJ::BandPass1::setup (
            double sampleRate,
            double centerFrequency,
            double bandWidth )
```

Calculates the coefficients

Parameters

sampleRate	Sampling rate
centerFrequency	Center frequency of the bandpass
bandWidth	Bandwidth of the bandpass

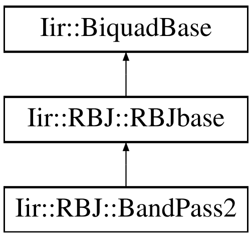
The documentation for this struct was generated from the following files:

- iir/RBJ.h
- iir/RBJ.cpp

6.16 Iir::RBJ::BandPass2 Struct Reference

```
#include <RBJ.h>
```

Inheritance diagram for Iir::RBJ::BandPass2:



Public Member Functions

- void setup (double sampleRate, double centerFrequency, double bandWidth)

6.16.1 Detailed Description

Bandpass with constant 0 dB peak gain

6.16.2 Member Function Documentation

6.16.2.1 setup()

```
void Iir::RBJ::BandPass2::setup (
            double sampleRate,
            double centerFrequency,
            double bandWidth )
```

Calculates the coefficients

Parameters

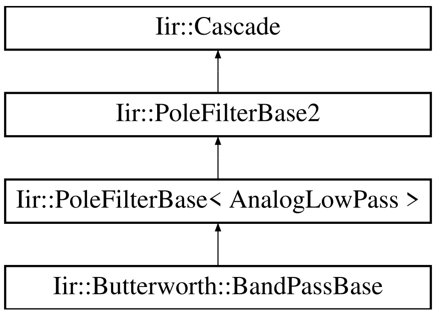
sampleRate	Sampling rate
centerFrequency	Center frequency of the bandpass
bandWidth	Bandwidth of the bandpass

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

- iir/RBJ.h
- iir/RBJ.cpp

6.17 Iir::Butterworth::BandPassBase Struct Reference

Inheritance diagram for Iir::Butterworth::BandPassBase:

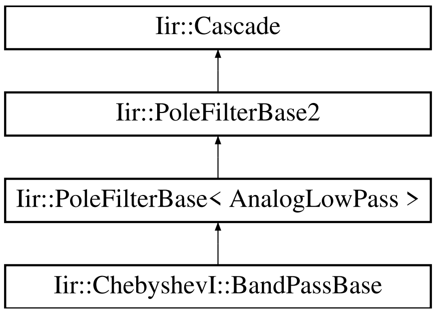

Iir::Cascade
Iir::PoleFilterBase2
Iir::PoleFilterBase< AnalogLowPass >
Iir::Butterworth::BandPassBase

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

- iir/Butterworth.h
- iir/Butterworth.cpp

6.18 Iir::ChebyshevI::BandPassBase Struct Reference

Inheritance diagram for Iir::ChebyshevI::BandPassBase:

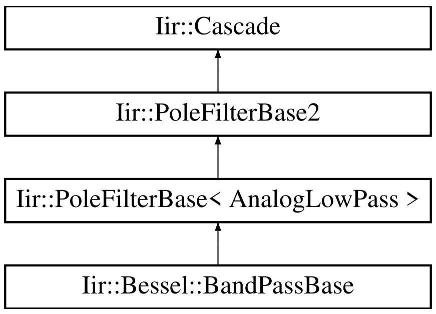

Iir::Cascade
Iir::PoleFilterBase2
Iir::PoleFilterBase< AnalogLowPass >
Iir::ChebyshevI::BandPassBase

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

- iir/ChebyshevI.h
- iir/ChebyshevI.cpp

6.19 Iir::Bessel::BandPassBase Struct Reference

Inheritance diagram for Iir::Bessel::BandPassBase:

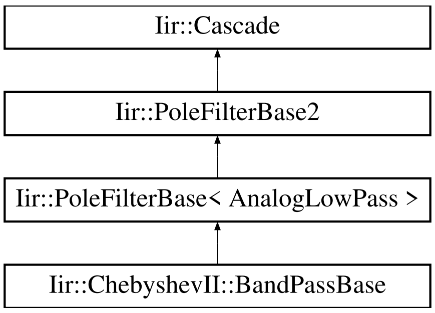

Iir::Cascade
Iir::PoleFilterBase2
Iir::PoleFilterBase< AnalogLowPass >
Iir::Bessel::BandPassBase

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

- iir/Bessel.h
- iir/Bessel.cpp

6.20 Iir::ChebyshevII::BandPassBase Struct Reference

Inheritance diagram for Iir::ChebyshevII::BandPassBase:


Iir::Cascade
Iir::PoleFilterBase2
Iir::PoleFilterBase< AnalogLowPass >
Iir::ChebyshevII::BandPassBase

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

- iir/ChebyshevII.h
- iir/ChebyshevII.cpp

6.21 Iir::BandPassTransform Class Reference

```
#include <PoleFilter.h>
```

6.21.1 Detailed Description

low pass to band pass transform

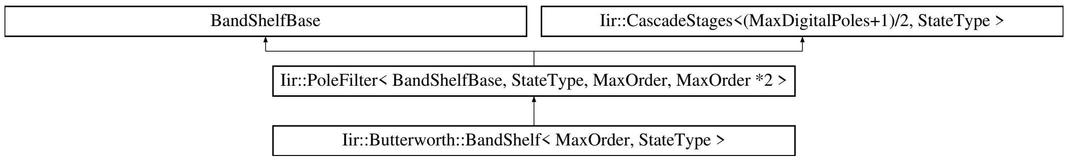
The documentation for this class was generated from the following files:

- iir/PoleFilter.h
- iir/PoleFilter.cpp

6.22 Iir::Butterworth::BandShelf< MaxOrder, StateType > Struct Template Reference

```
#include <Butterworth.h>
```

Inheritance diagram for Iir::Butterworth::BandShelf< MaxOrder, StateType >:


BandShelfBase
Iir::CascadeStages<(MaxDigitalPoles+1)/2, StateType >
Iir::PoleFilter< BandShelfBase, StateType, MaxOrder, MaxOrder *2 >
Iir::Butterworth::BandShelf< MaxOrder, StateType >

Public Member Functions

- void setup (double sampleRate, double centerFrequency, double widthFrequency, double gainDb)

6.22.1 Detailed Description

template<int MaxOrder, class StateType = DEFAULT_STATE>
struct Iir::Butterworth::BandShelf< MaxOrder, StateType >

Butterworth Bandshelf filter: it is a bandpass filter which amplifies at a specified gain in dB the frequencies in the passband.

6.22.2 Member Function Documentation

6.22.2.1 setup()

```
template<int MaxOrder, class StateType = DEFAULT_STATE>
void Iir::Butterworth::BandShelf< MaxOrder, StateType >::setup (
             double sampleRate,
             double centerFrequency,
             double widthFrequency,
             double gainDb )  [inline]
```

Calculates the coefficients

Parameters

sampleRate	Sampling rate
centerFrequency	Centre frequency of the passband
widthFrequency	Width of the passband
gainDb	The gain in the passband

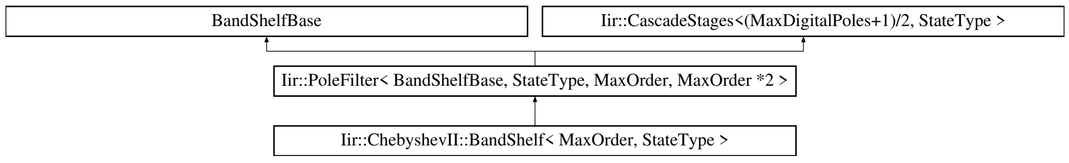
The documentation for this struct was generated from the following file:

- iir/Butterworth.h

6.23 Iir::ChebyshevII::BandShelf< MaxOrder, StateType > Struct Template Reference

```
#include <ChebyshevII.h>
```

Inheritance diagram for Iir::ChebyshevII::BandShelf< MaxOrder, StateType >:


BandShelfBase
Iir::CascadeStages<(MaxDigitalPoles+1)/2, StateType >
Iir::PoleFilter< BandShelfBase, StateType, MaxOrder, MaxOrder *2 >
Iir::ChebyshevII::BandShelf< MaxOrder, StateType >

Public Member Functions

- void setup (double sampleRate, double centerFrequency, double widthFrequency, double gainDb, double stopBandDb)

6.23.1 Detailed Description

template<int MaxOrder, class StateType = DEFAULT_STATE>
struct Iir::ChebyshevII::BandShelf< MaxOrder, StateType >

ChebyshevII bandshelf filter. Bandpass with specified gain and 0 dB gain in the stopband.

6.23.2 Member Function Documentation

6.23.2.1 setup()

```
template<int MaxOrder, class StateType = DEFAULT_STATE>
void Iir::ChebyshevII::BandShelf< MaxOrder, StateType >::setup (
             double sampleRate,
             double centerFrequency,
             double widthFrequency,
             double gainDb,
             double stopBandDb )  [inline]
```

Calculates the coefficients of the filter

Parameters

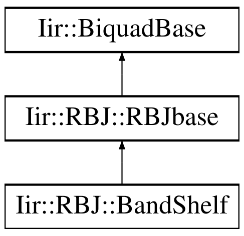
sampleRate	Sampling rate
centerFrequency	Center frequency of the bandpass
widthFrequency	Width of the bandpass
gainDb	Gain in the passband. The stopband has always 0dB.
stopBandDb	Permitted ripples in dB in the stopband

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

- iir/ChebyshevII.h

6.24 Iir::RBJ::BandShelf Struct Reference

Inheritance diagram for Iir::RBJ::BandShelf:



Additional Inherited Members

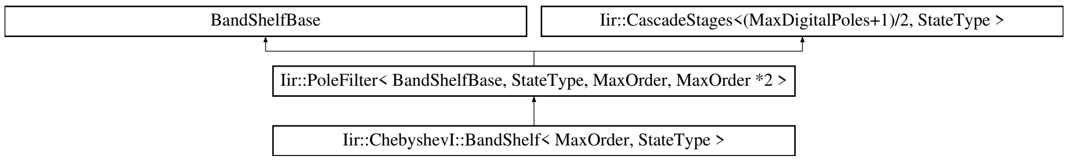
The documentation for this struct was generated from the following files:

- iir/RBJ.h
- iir/RBJ.cpp

6.25 Iir::ChebyshevI::BandShelf< MaxOrder, StateType > Struct Template Reference

```
#include <ChebyshevI.h>
```

Inheritance diagram for Iir::ChebyshevI::BandShelf< MaxOrder, StateType >:


BandShelfBase
Iir::CascadeStages<(MaxDigitalPoles+1)/2, StateType >
Iir::PoleFilter< BandShelfBase, StateType, MaxOrder, MaxOrder *2 >
Iir::ChebyshevI::BandShelf< MaxOrder, StateType >

Public Member Functions

- void setup (double sampleRate, double centerFrequency, double widthFrequency, double gainDb, double rippleDb)

6.25.1 Detailed Description

template<int MaxOrder, class StateType = DEFAULT_STATE>
struct Iir::ChebyshevI::BandShelf< MaxOrder, StateType >

ChebyshevI bandshelf filter. Specified gain in the passband. Otherwise 0 dB.

6.25.2 Member Function Documentation

6.25.2.1 setup()

```
template<int MaxOrder, class StateType = DEFAULT_STATE>
void Iir::ChebyshevI::BandShelf< MaxOrder, StateType >::setup (
             double sampleRate,
             double centerFrequency,
             double widthFrequency,
             double gainDb,
             double rippleDb )  [inline]
```

Calculates the coefficients of the filter

Parameters

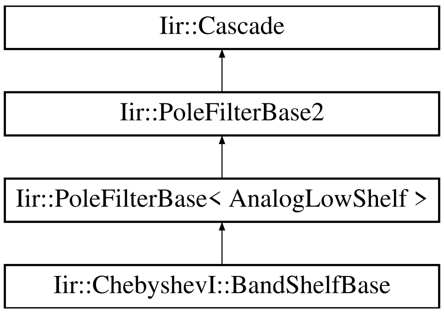
sampleRate	Sampling rate
centerFrequency	Center frequency of the passband
widthFrequency	Width of the passband.
gainDb	Gain in the passband. The stopband has 0 dB.
rippleDb	Permitted ripples in dB in the passband.

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

- iir/ChebyshevI.h

6.26 Iir::ChebyshevI::BandShelfBase Struct Reference

Inheritance diagram for Iir::ChebyshevI::BandShelfBase:

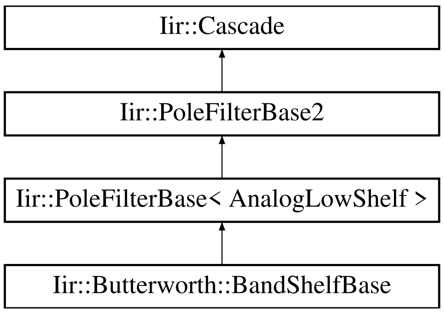

Iir::Cascade
Iir::PoleFilterBase2
Iir::PoleFilterBase< AnalogLowShelf >
Iir::ChebyshevI::BandShelfBase

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

- iir/ChebyshevI.h
- iir/ChebyshevI.cpp

6.27 Iir::Butterworth::BandShelfBase Struct Reference

Inheritance diagram for Iir::Butterworth::BandShelfBase:

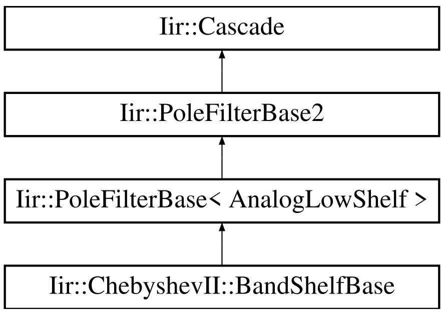

Iir::Cascade
Iir::PoleFilterBase2
Iir::PoleFilterBase< AnalogLowShelf >
Iir::Butterworth::BandShelfBase

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

- iir/Butterworth.h
- iir/Butterworth.cpp

6.28 Iir::ChebyshevII::BandShelfBase Struct Reference

Inheritance diagram for Iir::ChebyshevII::BandShelfBase:


Iir::Cascade
Iir::PoleFilterBase2
Iir::PoleFilterBase< AnalogLowShelf >
Iir::ChebyshevII::BandShelfBase

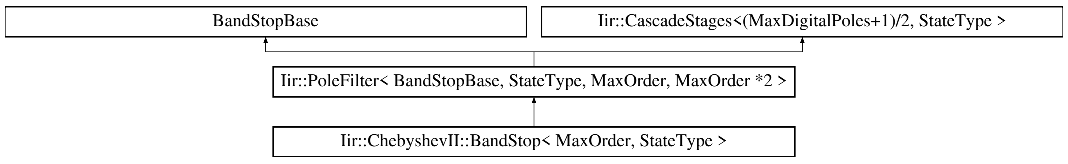
The documentation for this struct was generated from the following files:

- iir/ChebyshevII.h
- iir/ChebyshevII.cpp

6.29 Iir::ChebyshevII::BandStop< MaxOrder, StateType > Struct Template Reference

```
#include <ChebyshevII.h>
```

Inheritance diagram for Iir::ChebyshevII::BandStop< MaxOrder, StateType >:


BandStopBase
Iir::CascadeStages<(MaxDigitalPoles+1)/2, StateType >
Iir::PoleFilter< BandStopBase, StateType, MaxOrder, MaxOrder *2 >
Iir::ChebyshevII::BandStop< MaxOrder, StateType >

Public Member Functions

- void setup (double sampleRate, double centerFrequency, double widthFrequency, double stopBandDb)

6.29.1 Detailed Description

template<int MaxOrder, class StateType = DEFAULT_STATE>
struct Iir::ChebyshevII::BandStop< MaxOrder, StateType >

ChebyshevII bandstop filter.

6.29.2 Member Function Documentation

6.29.2.1 setup()

```
template<int MaxOrder, class StateType = DEFAULT_STATE>
void Iir::ChebyshevII::BandStop< MaxOrder, StateType >::setup (
             double sampleRate,
             double centerFrequency,
             double widthFrequency,
             double stopBandDb )  [inline]
```

Calculates the coefficients of the filter

Parameters

sampleRate	Sampling rate
centerFrequency	Center frequency of the bandstop
widthFrequency	Width of the bandstop
stopBandDb	Permitted ripples in dB in the stopband

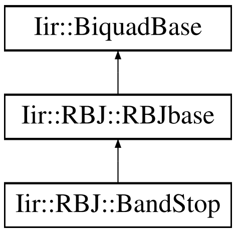
The documentation for this struct was generated from the following file:

- iir/ChebyshevII.h

6.30 Iir::RBJ::BandStop Struct Reference

```
#include <RBJ.h>
```

Inheritance diagram for Iir::RBJ::BandStop:



Public Member Functions

- void setup (double sampleRate, double centerFrequency, double bandWidth)

6.30.1 Detailed Description

Bandstop filter. Warning: the bandwidth might not be accurate for narrow notches.

6.30.2 Member Function Documentation

6.30.2.1 setup()

```
void Iir::RBJ::BandStop::setup (
            double sampleRate,
            double centerFrequency,
            double bandWidth )
```

Calculates the coefficients

Parameters

sampleRate	Sampling rate
centerFrequency	Center frequency of the bandstop
bandWidth	Bandwidth of the bandstop

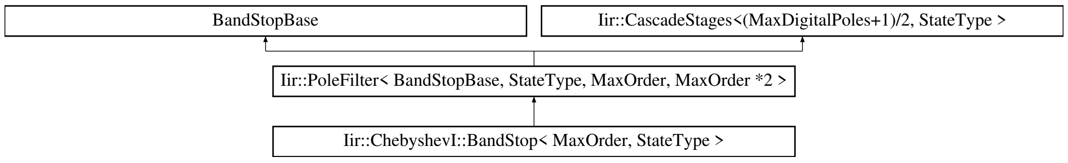
The documentation for this struct was generated from the following files:

- iir/RBJ.h
- iir/RBJ.cpp

6.31 Iir::ChebyshevI::BandStop< MaxOrder, StateType > Struct Template Reference

```
#include <ChebyshevI.h>
```

Inheritance diagram for Iir::ChebyshevI::BandStop< MaxOrder, StateType >:


BandStopBase
Iir::CascadeStages<(MaxDigitalPoles+1)/2, StateType >
Iir::PoleFilter< BandStopBase, StateType, MaxOrder, MaxOrder *2 >
Iir::ChebyshevI::BandStop< MaxOrder, StateType >

Public Member Functions

- void setup (double sampleRate, double centerFrequency, double widthFrequency, double rippleDb)

6.31.1 Detailed Description

template<int MaxOrder, class StateType = DEFAULT_STATE>
struct Iir::ChebyshevI::BandStop< MaxOrder, StateType >

ChebyshevI bandstop filter

6.31.2 Member Function Documentation

6.31.2.1 setup()

```
template<int MaxOrder, class StateType = DEFAULT_STATE>
void Iir::ChebyshevI::BandStop< MaxOrder, StateType >::setup (
            double sampleRate,
            double centerFrequency,
            double widthFrequency,
            double rippleDb ) [inline]
```

Calculates the coefficients of the filter

Parameters

sampleRate	Sampling rate
centerFrequency	Center frequency of the notch
widthFrequency	Frequency with of the notch
rippleDb	Permitted ripples in dB in the passband

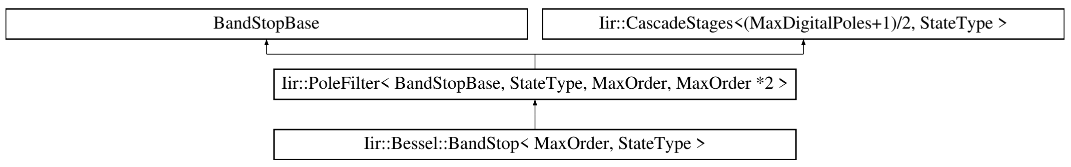
The documentation for this struct was generated from the following file:

- iir/ChebyshevI.h

6.32 Iir::Bessel::BandStop< MaxOrder, StateType > Struct Template Reference

```
#include <Bessel.h>
```

Inheritance diagram for Iir::Bessel::BandStop< MaxOrder, StateType >:


BandStopBase
Iir::CascadeStages<(MaxDigitalPoles+1)/2, StateType >
Iir::PoleFilter< BandStopBase, StateType, MaxOrder, MaxOrder *2 >
Iir::Bessel::BandStop< MaxOrder, StateType >

Public Member Functions

- void setup (double sampleRate, double centerFrequency, double widthFrequency)

6.32.1 Detailed Description

template<int MaxOrder, class StateType = DEFAULT_STATE>
struct Iir::Bessel::BandStop< MaxOrder, StateType >

Bessel bandstop.

6.32.2 Member Function Documentation

6.32.2.1 setup()

```
template<int MaxOrder, class StateType = DEFAULT_STATE>
void Iir::Bessel::BandStop< MaxOrder, StateType >::setup (
            double sampleRate,
            double centerFrequency,
            double widthFrequency )  [inline]
```

Calculate the coefficients

Parameters

sampleRate	Sampling rate
centerFrequency	Center frequency of the bandpass in Hz
widthFrequency	Width of the bandpass in Hz

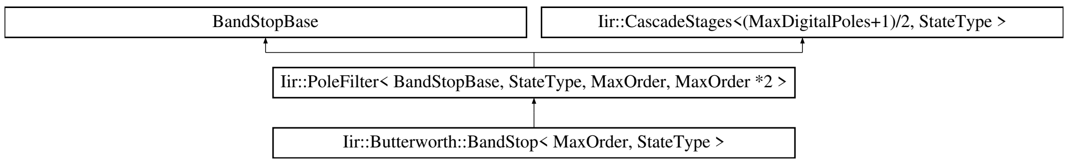
The documentation for this struct was generated from the following file:

- iir/Bessel.h

6.33 Iir::Butterworth::BandStop< MaxOrder, StateType > Struct Template Reference

```
#include <Butterworth.h>
```

Inheritance diagram for Iir::Butterworth::BandStop< MaxOrder, StateType >:


BandStopBase
Iir::CascadeStages<(MaxDigitalPoles+1)/2, StateType >
Iir::PoleFilter< BandStopBase, StateType, MaxOrder, MaxOrder *2 >
Iir::Butterworth::BandStop< MaxOrder, StateType >

Public Member Functions

- void setup (double sampleRate, double centerFrequency, double widthFrequency)

6.33.1 Detailed Description

template<int MaxOrder, class StateType = DEFAULT_STATE>
struct Iir::Butterworth::BandStop< MaxOrder, StateType >

Butterworth Bandstop filter.

6.33.2 Member Function Documentation

6.33.2.1 setup()

```
template<int MaxOrder, class StateType = DEFAULT_STATE>
void Iir::Butterworth::BandStop< MaxOrder, StateType >::setup (
             double sampleRate,
             double centerFrequency,
             double widthFrequency )  [inline]
```

Calculates the coefficients

Parameters

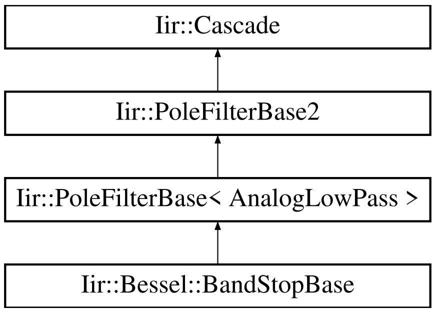
sampleRate	Sampling rate
centerFrequency	Centre frequency of the bandstop
widthFrequency	Width of the bandstop

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

- iir/Butterworth.h

6.34 Iir::Bessel::BandStopBase Struct Reference

Inheritance diagram for Iir::Bessel::BandStopBase:

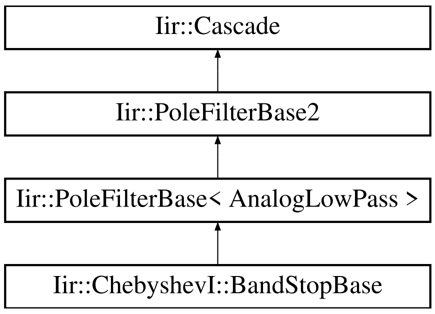

Iir::Cascade
Iir::PoleFilterBase2
Iir::PoleFilterBase< AnalogLowPass >
Iir::Bessel::BandStopBase

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

- iir/Bessel.h
- iir/Bessel.cpp

6.35 Iir::ChebyshevI::BandStopBase Struct Reference

Inheritance diagram for Iir::ChebyshevI::BandStopBase:

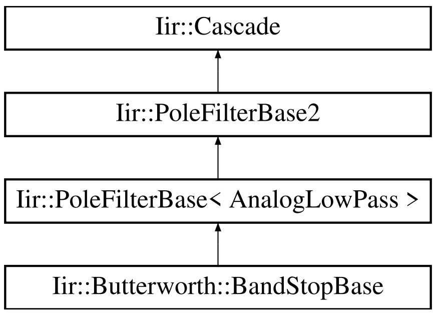

Iir::Cascade
Iir::PoleFilterBase2
Iir::PoleFilterBase< AnalogLowPass >
Iir::ChebyshevI::BandStopBase

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

- iir/ChebyshevI.h
- iir/ChebyshevI.cpp

6.36 Iir::Butterworth::BandStopBase Struct Reference

Inheritance diagram for Iir::Butterworth::BandStopBase:

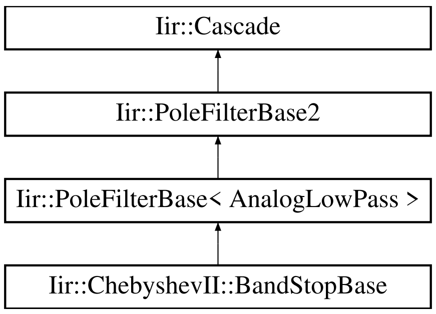

Iir::Cascade
Iir::PoleFilterBase2
Iir::PoleFilterBase< AnalogLowPass >
Iir::Butterworth::BandStopBase

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

- iir/Butterworth.h
- iir/Butterworth.cpp

6.37 Iir::ChebyshevII::BandStopBase Struct Reference

Inheritance diagram for Iir::ChebyshevII::BandStopBase:


Iir::Cascade
Iir::PoleFilterBase2
Iir::PoleFilterBase< AnalogLowPass >
Iir::ChebyshevII::BandStopBase

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

- iir/ChebyshevII.h
- iir/ChebyshevII.cpp

6.38 Iir::BandStopTransform Class Reference

```
#include <PoleFilter.h>
```

6.38.1 Detailed Description

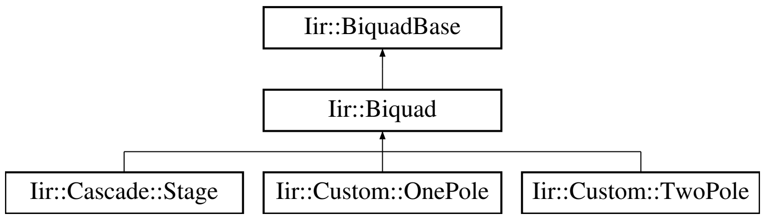
low pass to band stop transform

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

- iir/PoleFilter.h
- iir/PoleFilter.cpp

6.39 Iir::Biquad Class Reference

Inheritance diagram for Iir::Biquad:

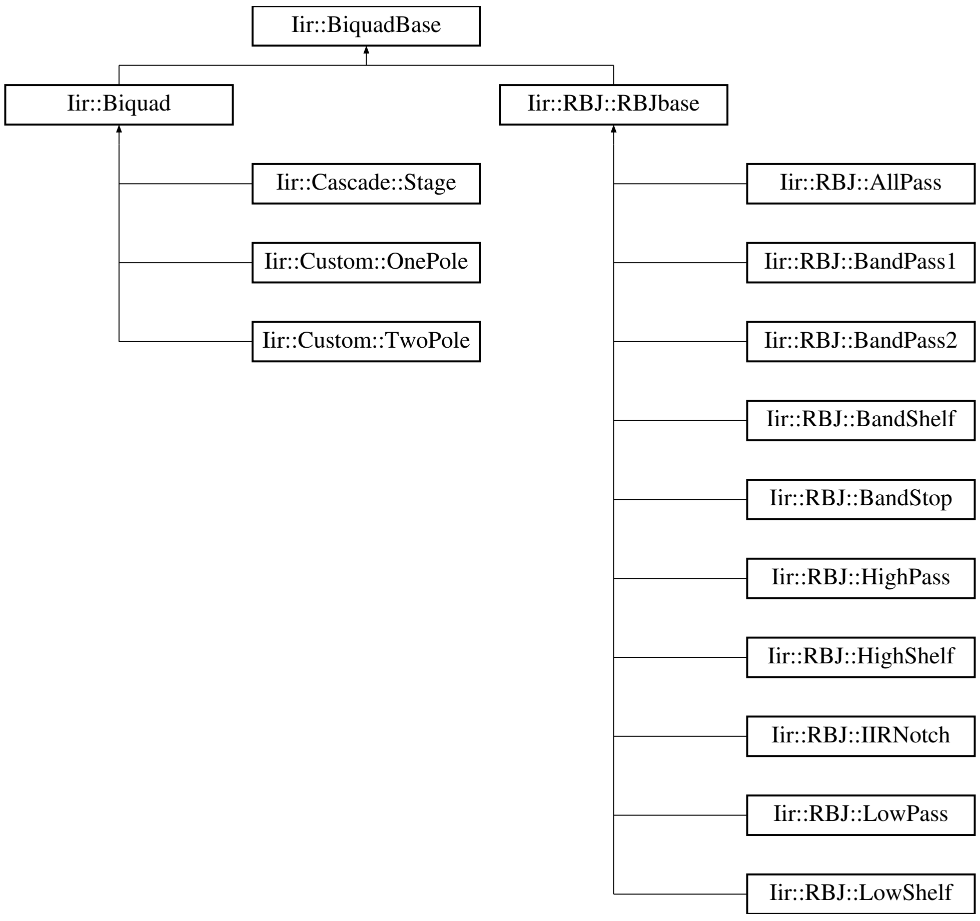


The documentation for this class was generated from the following files:

- iir/Biquad.h
- iir/Biquad.cpp

6.40 Iir::BiquadBase Class Reference

Inheritance diagram for Iir::BiquadBase:



Classes

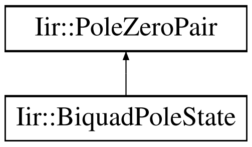
- struct State

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

- iir/Biquad.h
- iir/Biquad.cpp

6.41 Iir::BiquadPoleState Struct Reference

Inheritance diagram for Iir::BiquadPoleState:

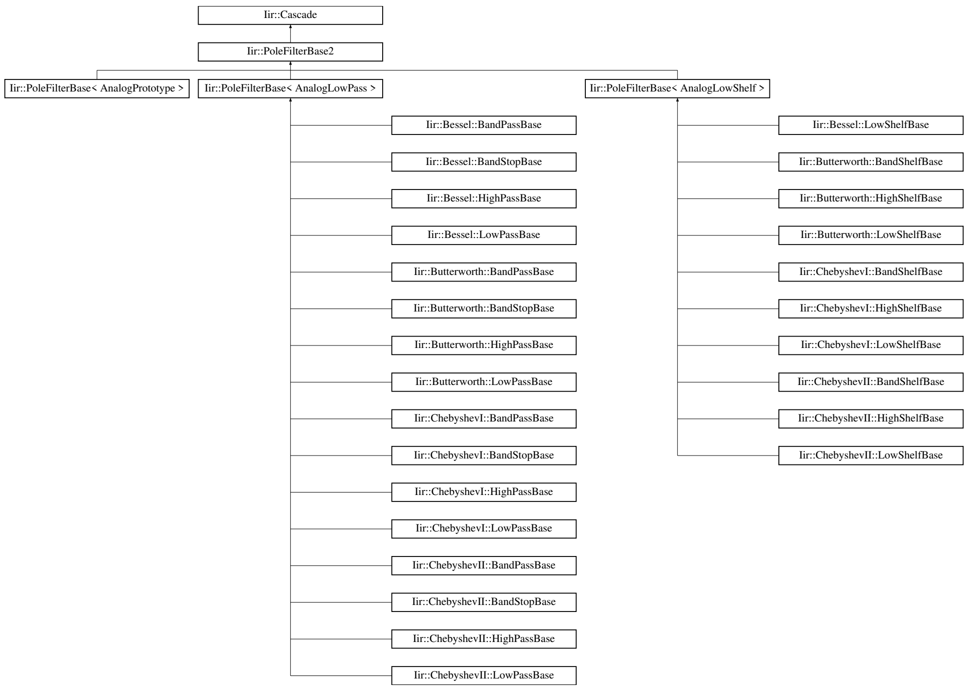


The documentation for this struct was generated from the following files:

- iir/Biquad.h
- iir/Biquad.cpp

6.42 Iir::Cascade Class Reference

Inheritance diagram for Iir::Cascade:


Iir::Cascade
Iir::PoleFilterBase2
Iir::PoleFilterBase< AnalogPrototype >
Iir::PoleFilterBase< AnalogLowPass >
Iir::PoleFilterBase< AnalogLowShelf >
Iir::Bessel::BandPassBase
Iir::Bessel::BandStopBase
Iir::Bessel::HighPassBase
Iir::Bessel::LowPassBase
Iir::Butterworth::BandPassBase
Iir::Butterworth::BandStopBase
Iir::Butterworth::HighPassBase
Iir::Butterworth::LowPassBase
Iir::ChebyshevI::BandPassBase
Iir::ChebyshevI::BandStopBase
Iir::ChebyshevI::HighPassBase
Iir::ChebyshevI::LowPassBase
Iir::ChebyshevII::BandPassBase
Iir::ChebyshevII::BandStopBase
Iir::ChebyshevII::HighPassBase
Iir::ChebyshevII::LowPassBase
Iir::Bessel::LowShelfBase
Iir::Butterworth::BandShelfBase
Iir::Butterworth::HighShelfBase
Iir::Butterworth::LowShelfBase
Iir::ChebyshevI::BandShelfBase
Iir::ChebyshevI::HighShelfBase
Iir::ChebyshevI::LowShelfBase
Iir::ChebyshevII::BandShelfBase
Iir::ChebyshevII::HighShelfBase
Iir::ChebyshevII::LowShelfBase

Classes

- struct Stage
- struct Storage

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

- iir/Cascade.h
- iir/Cascade.cpp

6.43 Iir::CascadeStages< MaxStages, StateType > Class Template Reference

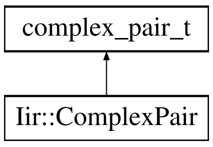
The documentation for this class was generated from the following file:

- iir/Cascade.h

6.44 Iir::ComplexPair Struct Reference

```
#include <Types.h>
```

Inheritance diagram for Iir::ComplexPair:



6.44.1 Detailed Description

A conjugate or real pair

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

- iir/Types.h

6.45 Iir::DirectFormI Class Reference

```
#include <State.h>
```

6.45.1 Detailed Description

State for applying a second order section to a sample using Direct Form I

Difference equation:

y[n] = (b0/a0)∗x[n] + (b1/a0)∗x[n-1] + (b2/a0)∗x[n-2]

- (a1/a0)∗y[n-1] - (a2/a0)∗y[n-2]

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

- iir/State.h

6.46 Iir::DirectFormII Class Reference

```
#include <State.h>
```

6.46.1 Detailed Description

State for applying a second order section to a sample using Direct Form II

Difference equation:

v[n] = x[n] - (a1/a0)∗v[n-1] - (a2/a0)∗v[n-2] y(n) = (b0/a0)∗v[n] + (b1/a0)∗v[n-1] + (b2/a0)∗v[n-2]

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

- iir/State.h

6.47 Iir::EnvelopeFollower< Channels, Value > Class Template Reference

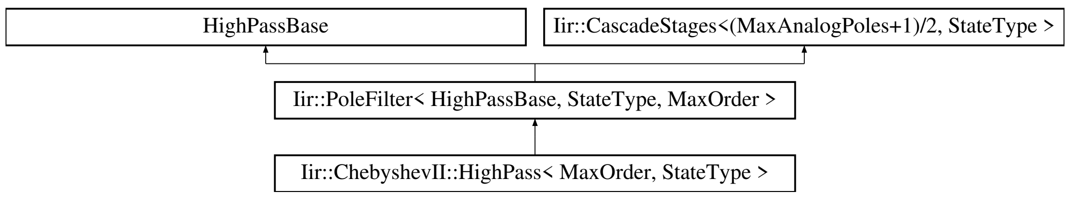
The documentation for this class was generated from the following file:

- iir/Utilities.h

6.48 Iir::ChebyshevII::HighPass< MaxOrder, StateType > Struct Template Reference

```
#include <ChebyshevII.h>
```

Inheritance diagram for Iir::ChebyshevII::HighPass< MaxOrder, StateType >:


HighPassBase
Iir::CascadeStages<(MaxAnalogPoles+1)/2, StateType >
Iir::PoleFilter< HighPassBase, StateType, MaxOrder >
Iir::ChebyshevII::HighPass< MaxOrder, StateType >

Public Member Functions

- void setup (double sampleRate, double cutoffFrequency, double stopBandDb)

6.48.1 Detailed Description

template<int MaxOrder, class StateType = DEFAULT_STATE>
struct Iir::ChebyshevII::HighPass< MaxOrder, StateType >

ChebyshevII highpass filter

6.48.2 Member Function Documentation

6.48.2.1 setup()

```
template<int MaxOrder, class StateType = DEFAULT_STATE>
void Iir::ChebyshevII::HighPass< MaxOrder, StateType >::setup (
             double sampleRate,
             double cutoffFrequency,
             double stopBandDb )  [inline]
```

Calculates the coefficients of the filter

Parameters

sampleRate	Sampling rate
cutoffFrequency	Cutoff frequency.
stopBandDb	Permitted ripples in dB in the stopband

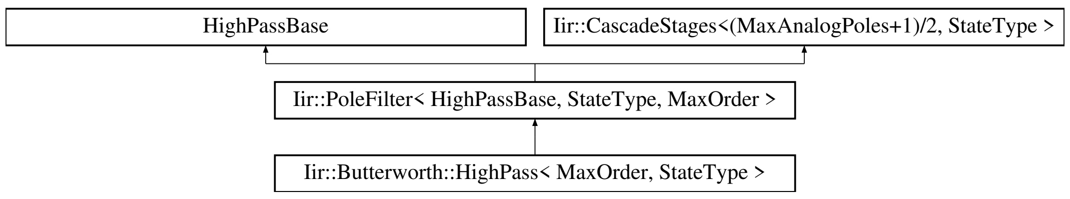
The documentation for this struct was generated from the following file:

- iir/ChebyshevII.h

6.49 Iir::Butterworth::HighPass< MaxOrder, StateType > Struct Template Reference

```
#include <Butterworth.h>
```

Inheritance diagram for Iir::Butterworth::HighPass< MaxOrder, StateType >:


HighPassBase
Iir::CascadeStages<(MaxAnalogPoles+1)/2, StateType >
Iir::PoleFilter< HighPassBase, StateType, MaxOrder >
Iir::Butterworth::HighPass< MaxOrder, StateType >

Public Member Functions

- void setup (double sampleRate, double cutoffFrequency)

6.49.1 Detailed Description

template<int MaxOrder, class StateType = DEFAULT_STATE>
struct Iir::Butterworth::HighPass< MaxOrder, StateType >

Butterworth Highpass filter.

6.49.2 Member Function Documentation

6.49.2.1 setup()

```
template<int MaxOrder, class StateType = DEFAULT_STATE>
void Iir::Butterworth::HighPass< MaxOrder, StateType >::setup (
            double sampleRate,
            double cutoffFrequency )  [inline]
```

Calculates the coefficients

Parameters

sampleRate	Sampling rate
cutoffFrequency	Cutoff

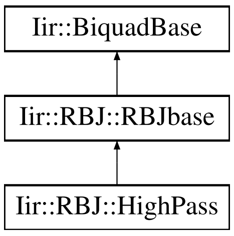
The documentation for this struct was generated from the following file:

- iir/Butterworth.h

6.50 Iir::RBJ::HighPass Struct Reference

```
#include <RBJ.h>
```

Inheritance diagram for Iir::RBJ::HighPass:



Public Member Functions

- void setup (double sampleRate, double cutoffFrequency, double q)

6.50.1 Detailed Description

Highpass.

6.50.2 Member Function Documentation

6.50.2.1 setup()

```
void Iir::RBJ::HighPass::setup (
             double sampleRate,
             double cutoffFrequency,
             double q )
```

Calculates the coefficients

Parameters

sampleRate	Sampling rate
cutoffFrequency	Cutoff frequency
q	Q factor determines the resonance peak at the cutoff.

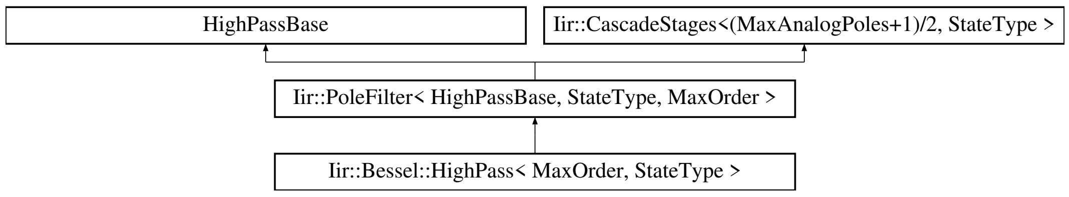
The documentation for this struct was generated from the following files:

- iir/RBJ.h
- iir/RBJ.cpp

6.51 Iir::Bessel::HighPass< MaxOrder, StateType > Struct Template Reference

```
#include <Bessel.h>
```

Inheritance diagram for Iir::Bessel::HighPass< MaxOrder, StateType >:


HighPassBase
Iir::CascadeStages<(MaxAnalogPoles+1)/2, StateType >
Iir::PoleFilter< HighPassBase, StateType, MaxOrder >
Iir::Bessel::HighPass< MaxOrder, StateType >

Public Member Functions

- void setup (double sampleRate, double cutoffFrequency)

6.51.1 Detailed Description

template<int MaxOrder, class StateType = DEFAULT_STATE>
struct Iir::Bessel::HighPass< MaxOrder, StateType >

Bessel Highpass.

6.51.2 Member Function Documentation

6.51.2.1 setup()

```
template<int MaxOrder, class StateType = DEFAULT_STATE>
void Iir::Bessel::HighPass< MaxOrder, StateType >::setup (
             double sampleRate,
             double cutoffFrequency )  [inline]
```

Calculate the coefficients

Parameters

sampleRate	Sampling rate
cutoffFrequency	Cutoff frequency

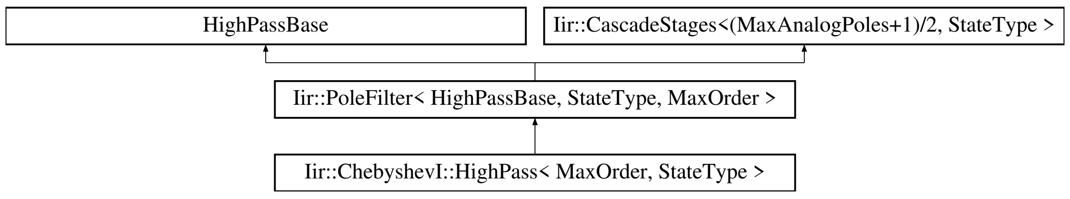
The documentation for this struct was generated from the following file:

- iir/Bessel.h

6.52 Iir::ChebyshevI::HighPass< MaxOrder, StateType > Struct Template Reference

```
#include <ChebyshevI.h>
```

Inheritance diagram for Iir::ChebyshevI::HighPass< MaxOrder, StateType >:


HighPassBase
Iir::CascadeStages<(MaxAnalogPoles+1)/2, StateType >
Iir::PoleFilter< HighPassBase, StateType, MaxOrder >
Iir::ChebyshevI::HighPass< MaxOrder, StateType >

Public Member Functions

- void setup (double sampleRate, double cutoffFrequency, double rippleDb)

6.52.1 Detailed Description

template<int MaxOrder, class StateType = DEFAULT_STATE>
struct Iir::ChebyshevI::HighPass< MaxOrder, StateType >

ChebyshevI highpass filter

6.52.2 Member Function Documentation

6.52.2.1 setup()

```
template<int MaxOrder, class StateType = DEFAULT_STATE>
void Iir::ChebyshevI::HighPass< MaxOrder, StateType >::setup (
             double sampleRate,
             double cutoffFrequency,
             double rippleDb )  [inline]
```

Calculates the coefficients of the filter

Parameters

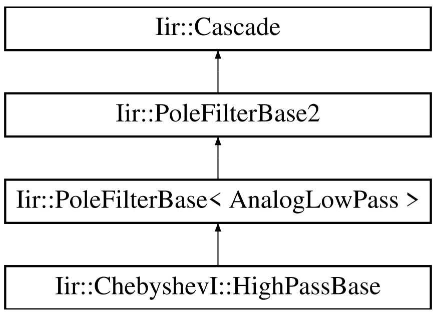
sampleRate	Sampling rate
cutoffFrequency	Cutoff frequency.
rippleDb	Permitted ripples in dB in the passband

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

- iir/ChebyshevI.h

6.53 Iir::ChebyshevI::HighPassBase Struct Reference

Inheritance diagram for Iir::ChebyshevI::HighPassBase:

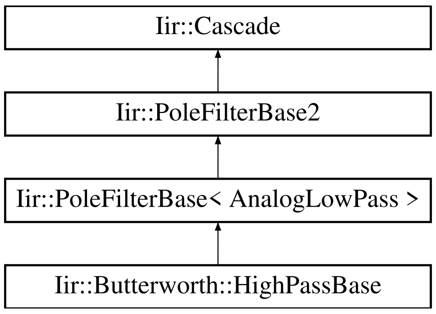

Iir::Cascade
Iir::PoleFilterBase2
Iir::PoleFilterBase< AnalogLowPass >
Iir::ChebyshevI::HighPassBase

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

- iir/ChebyshevI.h
- iir/ChebyshevI.cpp

6.54 Iir::Butterworth::HighPassBase Struct Reference

Inheritance diagram for Iir::Butterworth::HighPassBase:

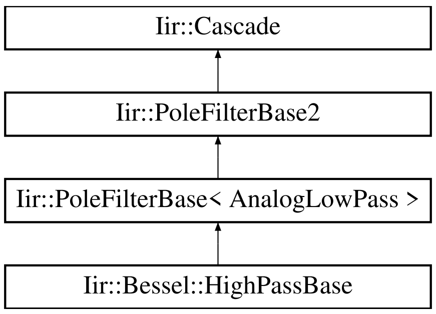

Iir::Cascade
Iir::PoleFilterBase2
Iir::PoleFilterBase< AnalogLowPass >
Iir::Butterworth::HighPassBase

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

- iir/Butterworth.h
- iir/Butterworth.cpp

6.55 Iir::Bessel::HighPassBase Struct Reference

Inheritance diagram for Iir::Bessel::HighPassBase:

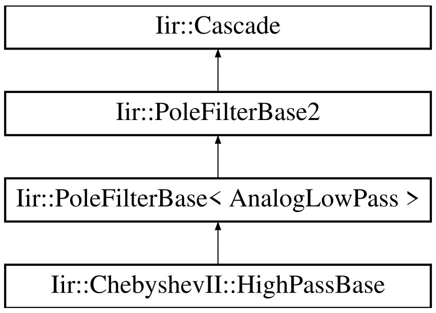

Iir::Cascade
Iir::PoleFilterBase2
Iir::PoleFilterBase< AnalogLowPass >
Iir::Bessel::HighPassBase

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

- iir/Bessel.h
- iir/Bessel.cpp

6.56 Iir::ChebyshevII::HighPassBase Struct Reference

Inheritance diagram for Iir::ChebyshevII::HighPassBase:


Iir::Cascade
Iir::PoleFilterBase2
Iir::PoleFilterBase< AnalogLowPass >
Iir::ChebyshevII::HighPassBase

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

- iir/ChebyshevII.h
- iir/ChebyshevII.cpp

6.57 Iir::HighPassTransform Class Reference

```
#include <PoleFilter.h>
```

6.57.1 Detailed Description

low pass to high pass

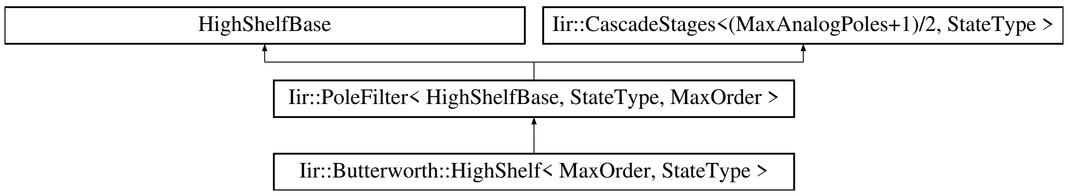
The documentation for this class was generated from the following files:

- iir/PoleFilter.h
- iir/PoleFilter.cpp

6.58 Iir::Butterworth::HighShelf< MaxOrder, StateType > Struct Template Reference

```
#include <Butterworth.h>
```

Inheritance diagram for Iir::Butterworth::HighShelf< MaxOrder, StateType >:


HighShelfBase
Iir::CascadeStages<(MaxAnalogPoles+1)/2, StateType >
Iir::PoleFilter< HighShelfBase, StateType, MaxOrder >
Iir::Butterworth::HighShelf< MaxOrder, StateType >

Public Member Functions

- void setup (double sampleRate, double cutoffFrequency, double gainDb)

6.58.1 Detailed Description

template<int MaxOrder, class StateType = DEFAULT_STATE>
struct Iir::Butterworth::HighShelf< MaxOrder, StateType >

Butterworth high shelf filter. Above the cutoff the filter has a specified gain and below it has 0 dB.

6.58.2 Member Function Documentation

6.58.2.1 setup()

```
template<int MaxOrder, class StateType = DEFAULT_STATE>
void Iir::Butterworth::HighShelf< MaxOrder, StateType >::setup (
            double sampleRate,
            double cutoffFrequency,
            double gainDb )  [inline]
```

Calculates the coefficients

Parameters

sampleRate	Sampling rate
cutoffFrequency	Cutoff
gainDb	Gain in dB of the filter in the passband

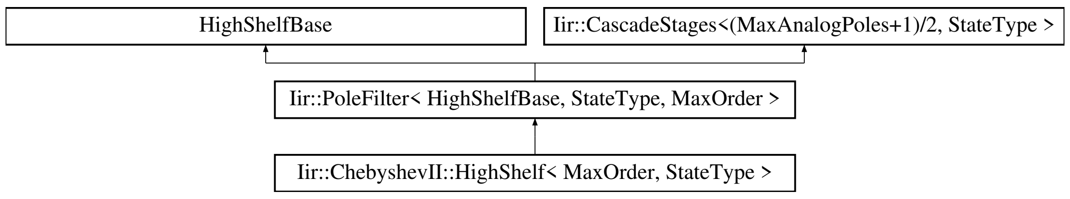
The documentation for this struct was generated from the following file:

- iir/Butterworth.h

6.59 Iir::ChebyshevII::HighShelf< MaxOrder, StateType > Struct Template Reference

```
#include <ChebyshevII.h>
```

Inheritance diagram for Iir::ChebyshevII::HighShelf< MaxOrder, StateType >:


HighShelfBase
Iir::CascadeStages<(MaxAnalogPoles+1)/2, StateType >
Iir::PoleFilter< HighShelfBase, StateType, MaxOrder >
Iir::ChebyshevII::HighShelf< MaxOrder, StateType >

Public Member Functions

- void setup (double sampleRate, double cutoffFrequency, double gainDb, double stopBandDb)

6.59.1 Detailed Description

template<int MaxOrder, class StateType = DEFAULT_STATE>
struct Iir::ChebyshevII::HighShelf< MaxOrder, StateType >

ChebyshevII high shelf filter. Specified gain in the passband and 0dB in the stopband.

6.59.2 Member Function Documentation

6.59.2.1 setup()

```
template<int MaxOrder, class StateType = DEFAULT_STATE>
void Iir::ChebyshevII::HighShelf< MaxOrder, StateType >::setup (
             double sampleRate,
             double cutoffFrequency,
             double gainDb,
             double stopBandDb )  [inline]
```

Calculates the coefficients of the filter

Parameters

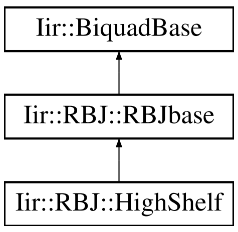
sampleRate	Sampling rate
cutoffFrequency	Cutoff frequency.
gainDb	Gain the passbard. The stopband has 0 dB gain.
stopBandDb	Permitted ripples in dB in the stopband

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

- iir/ChebyshevII.h

6.60 Iir::RBJ::HighShelf Struct Reference

Inheritance diagram for Iir::RBJ::HighShelf:



Additional Inherited Members

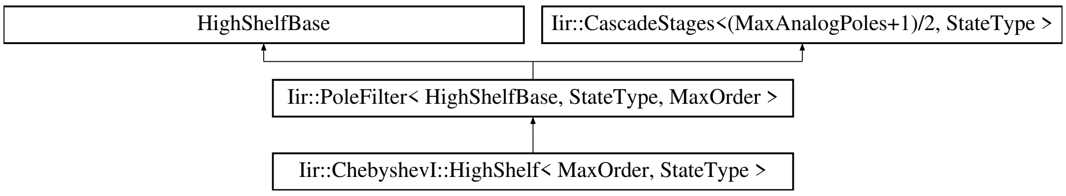
The documentation for this struct was generated from the following files:

- iir/RBJ.h
- iir/RBJ.cpp

6.61 Iir::ChebyshevI::HighShelf< MaxOrder, StateType > Struct Template Reference

```
#include <ChebyshevI.h>
```

Inheritance diagram for Iir::ChebyshevI::HighShelf< MaxOrder, StateType >:


HighShelfBase
Iir::CascadeStages<(MaxAnalogPoles+1)/2, StateType >
Iir::PoleFilter< HighShelfBase, StateType, MaxOrder >
Iir::ChebyshevI::HighShelf< MaxOrder, StateType >

Public Member Functions

- void setup (double sampleRate, double cutoffFrequency, double gainDb, double rippleDb)

6.61.1 Detailed Description

template<int MaxOrder, class StateType = DEFAULT_STATE>
struct Iir::ChebyshevI::HighShelf< MaxOrder, StateType >

ChebyshevI high shelf filter. Specified gain in the passband. Otherwise 0 dB.

6.61.2 Member Function Documentation

6.61.2.1 setup()

```
template<int MaxOrder, class StateType = DEFAULT_STATE>
void Iir::ChebyshevI::HighShelf< MaxOrder, StateType >::setup (
            double sampleRate,
            double cutoffFrequency,
            double gainDb,
            double rippleDb ) [inline]
```

Calculates the coefficients of the filter

Parameters

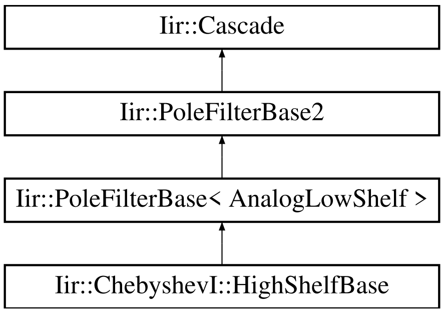
sampleRate	Sampling rate
cutoffFrequency	Cutoff frequency.
gainDb	Gain in the passband
rippleDb	Permitted ripples in dB in the passband

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

- iir/ChebyshevI.h

6.62 Iir::ChebyshevI::HighShelfBase Struct Reference

Inheritance diagram for Iir::ChebyshevI::HighShelfBase:

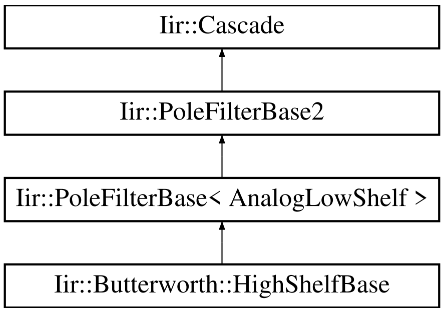

Iir::Cascade
Iir::PoleFilterBase2
Iir::PoleFilterBase< AnalogLowShelf >
Iir::ChebyshevI::HighShelfBase

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

- iir/ChebyshevI.h
- iir/ChebyshevI.cpp

6.63 Iir::Butterworth::HighShelfBase Struct Reference

Inheritance diagram for Iir::Butterworth::HighShelfBase:

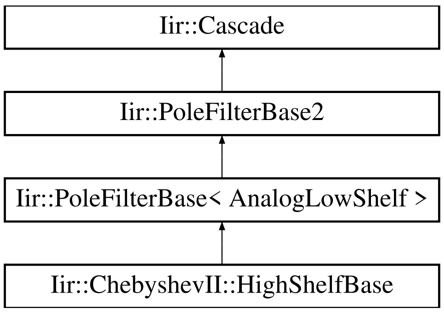

Iir::Cascade
Iir::PoleFilterBase2
Iir::PoleFilterBase< AnalogLowShelf >
Iir::Butterworth::HighShelfBase

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

- iir/Butterworth.h
- iir/Butterworth.cpp

6.64 Iir::ChebyshevII::HighShelfBase Struct Reference

Inheritance diagram for Iir::ChebyshevII::HighShelfBase:


Iir::Cascade
Iir::PoleFilterBase2
Iir::PoleFilterBase< AnalogLowShelf >
Iir::ChebyshevII::HighShelfBase

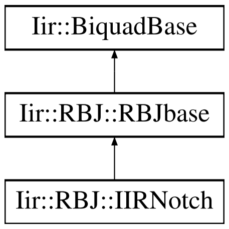
The documentation for this struct was generated from the following files:

- iir/ChebyshevII.h
- iir/ChebyshevII.cpp

6.65 Iir::RBJ::IIRNotch Struct Reference

```
#include <RBJ.h>
```

Inheritance diagram for Iir::RBJ::IIRNotch:



Public Member Functions

- void setup (double sampleRate, double centerFrequency, double q_factor)

6.65.1 Detailed Description

Bandstop with Q factor: the higher the Q factor the more narrow is the notch. However, a narrow notch has a long impulse response (= ringing) and numerical problems might prevent perfect damping. Practical values of the Q factor are about Q = 10 to 20. In terms of the design the Q factor defines the radius of the poles as r = exp(-pi∗(centerFrequency/sampleRate)/q_factor) whereas the angles of the poles/zeros define the bandstop frequency. The higher Q the closer r moves towards the unit circle.

6.65.2 Member Function Documentation

6.65.2.1 setup()

```
void Iir::RBJ::IIRNotch::setup (
             double sampleRate,
             double centerFrequency,
             double q_factor )
```

Calculates the coefficients

Parameters

sampleRate	Sampling rate
centerFrequency	Center frequency of the notch
q_factor	Q factor of the notch (1 to ∼20)

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

- iir/RBJ.h
- iir/RBJ.cpp

6.66 Iir::Layout< MaxPoles > Class Template Reference

```
#include <Layout.h>
```

6.66.1 Detailed Description

template<int MaxPoles>
class Iir::Layout< MaxPoles >

Storage for Layout

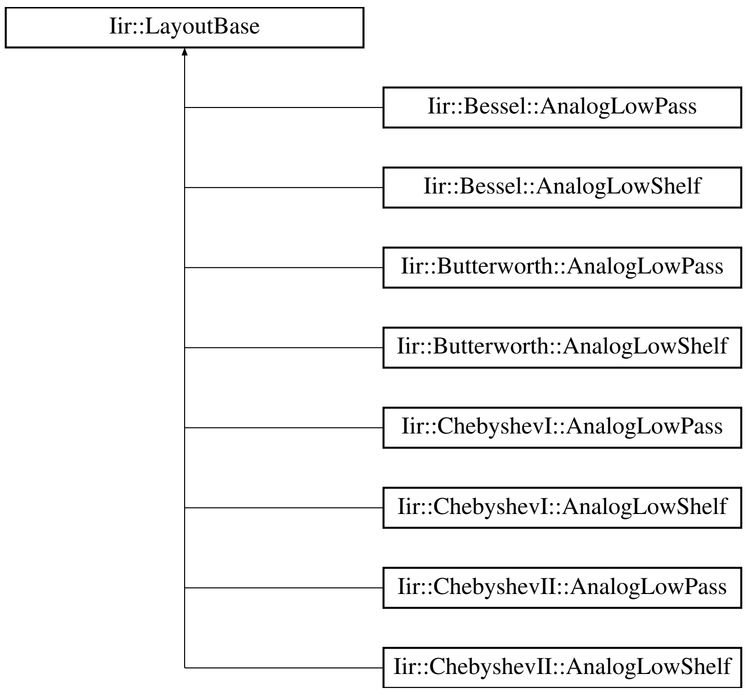
The documentation for this class was generated from the following file:

- iir/Layout.h

6.67 Iir::LayoutBase Class Reference

```
#include <Layout.h>
```

Inheritance diagram for Iir::LayoutBase:


Iir::LayoutBase
Iir::Bessel::AnalogLowPass
Iir::Bessel::AnalogLowShelf
Iir::Butterworth::AnalogLowPass
Iir::Butterworth::AnalogLowShelf
Iir::ChebyshevI::AnalogLowPass
Iir::ChebyshevI::AnalogLowShelf
Iir::ChebyshevII::AnalogLowPass
Iir::ChebyshevII::AnalogLowShelf

6.67.1 Detailed Description

Base uses pointers to reduce template instantiations

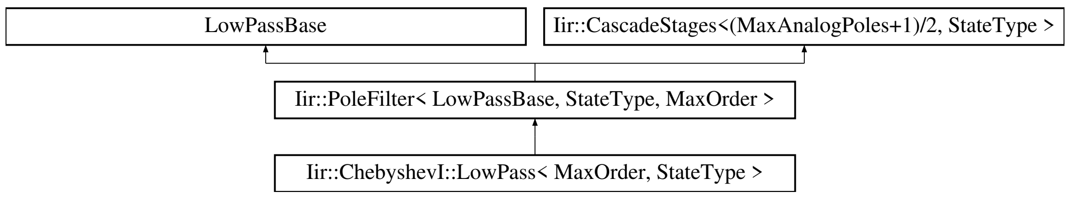
The documentation for this class was generated from the following file:

- iir/Layout.h

6.68 Iir::ChebyshevI::LowPass< MaxOrder, StateType > Struct Template Reference

```
#include <ChebyshevI.h>
```

Inheritance diagram for Iir::ChebyshevI::LowPass< MaxOrder, StateType >:


LowPassBase
Iir::CascadeStages<(MaxAnalogPoles+1)/2, StateType >
Iir::PoleFilter< LowPassBase, StateType, MaxOrder >
Iir::ChebyshevI::LowPass< MaxOrder, StateType >

Public Member Functions

- void setup (double sampleRate, double cutoffFrequency, double rippleDb)

6.68.1 Detailed Description

template<int MaxOrder, class StateType = DEFAULT_STATE>
struct Iir::ChebyshevI::LowPass< MaxOrder, StateType >

ChebyshevI lowpass filter

6.68.2 Member Function Documentation

6.68.2.1 setup()

```
template<int MaxOrder, class StateType = DEFAULT_STATE>
void Iir::ChebyshevI::LowPass< MaxOrder, StateType >::setup (
              double sampleRate,
              double cutoffFrequency,
              double rippleDb )  [inline]
```

Calculates the coefficients of the filter

Parameters

sampleRate	Sampling rate
cutoffFrequency	Cutoff frequency.
rippleDb	Permitted ripples in dB in the passband

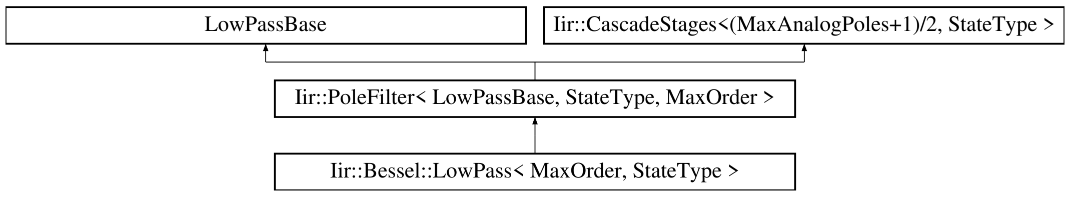
The documentation for this struct was generated from the following file:

- iir/ChebyshevI.h

6.69 Iir::Bessel::LowPass< MaxOrder, StateType > Struct Template Reference

```
#include <Bessel.h>
```

Inheritance diagram for Iir::Bessel::LowPass< MaxOrder, StateType >:


LowPassBase
Iir::CascadeStages<(MaxAnalogPoles+1)/2, StateType >
Iir::PoleFilter< LowPassBase, StateType, MaxOrder >
Iir::Bessel::LowPass< MaxOrder, StateType >

Public Member Functions

- void setup (double sampleRate, double cutoffFrequency)

6.69.1 Detailed Description

template<int MaxOrder, class StateType = DEFAULT_STATE>
struct Iir::Bessel::LowPass< MaxOrder, StateType >

Bessel Lowpass

6.69.2 Member Function Documentation

6.69.2.1 setup()

```
template<int MaxOrder, class StateType = DEFAULT_STATE>
void Iir::Bessel::LowPass< MaxOrder, StateType >::setup (
             double sampleRate,
             double cutoffFrequency )  [inline]
```

Calculate the coefficients

Parameters

sampleRate	Sampling rate
cutoffFrequency	Cutoff frequency

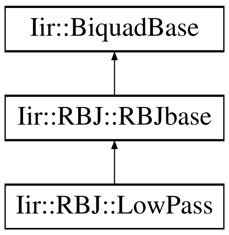
The documentation for this struct was generated from the following file:

- iir/Bessel.h

6.70 Iir::RBJ::LowPass Struct Reference

```
#include <RBJ.h>
```

Inheritance diagram for Iir::RBJ::LowPass:



Public Member Functions

- void setup (double sampleRate, double cutoffFrequency, double q)

6.70.1 Detailed Description

Lowpass.

6.70.2 Member Function Documentation

6.70.2.1 setup()

```
void Iir::RBJ::LowPass::setup (
             double sampleRate,
             double cutoffFrequency,
             double q )
```

Calculates the coefficients

Parameters

sampleRate	Sampling rate
cutoffFrequency	Cutoff frequency
q	Q factor determines the resonance peak at the cutoff.

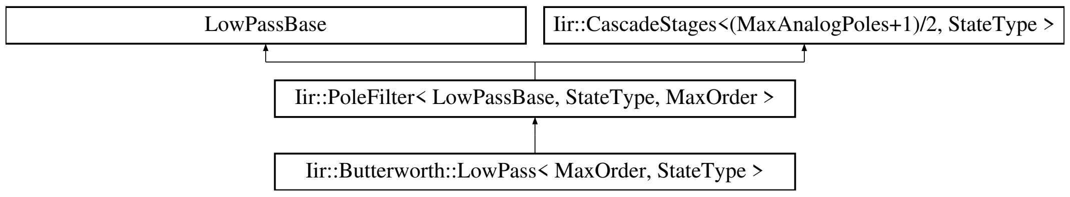
The documentation for this struct was generated from the following files:

- iir/RBJ.h
- iir/RBJ.cpp

6.71 Iir::Butterworth::LowPass< MaxOrder, StateType > Struct Template Reference

```
#include <Butterworth.h>
```

Inheritance diagram for Iir::Butterworth::LowPass< MaxOrder, StateType >:


LowPassBase
Iir::CascadeStages<(MaxAnalogPoles+1)/2, StateType >
Iir::PoleFilter< LowPassBase, StateType, MaxOrder >
Iir::Butterworth::LowPass< MaxOrder, StateType >

Public Member Functions

- void setup (double sampleRate, double cutoffFrequency)

6.71.1 Detailed Description

template<int MaxOrder, class StateType = DEFAULT_STATE>
struct Iir::Butterworth::LowPass< MaxOrder, StateType >

Butterworth Lowpass filter.

6.71.2 Member Function Documentation

6.71.2.1 setup()

```
template<int MaxOrder, class StateType = DEFAULT_STATE>
void Iir::Butterworth::LowPass< MaxOrder, StateType >::setup (
            double sampleRate,
            double cutoffFrequency )  [inline]
```

Calculates the coefficients

Parameters

sampleRate	Sampling rate
cutoffFrequency	Cutoff

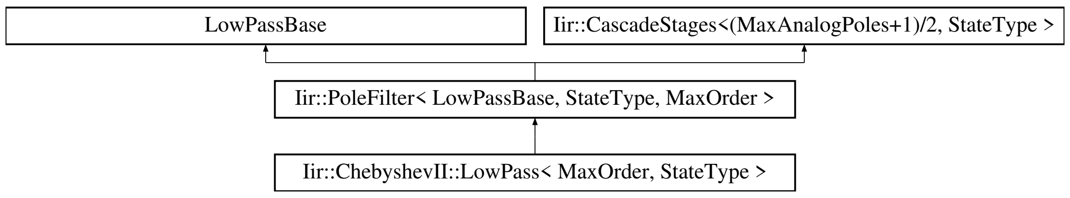
The documentation for this struct was generated from the following file:

- iir/Butterworth.h

6.72 Iir::ChebyshevII::LowPass< MaxOrder, StateType > Struct Template Reference

```
#include <ChebyshevII.h>
```

Inheritance diagram for Iir::ChebyshevII::LowPass< MaxOrder, StateType >:


LowPassBase
Iir::CascadeStages<(MaxAnalogPoles+1)/2, StateType >
Iir::PoleFilter< LowPassBase, StateType, MaxOrder >
Iir::ChebyshevII::LowPass< MaxOrder, StateType >

Public Member Functions

- void setup (double sampleRate, double cutoffFrequency, double stopBandDb)

6.72.1 Detailed Description

template<int MaxOrder, class StateType = DEFAULT_STATE>
struct Iir::ChebyshevII::LowPass< MaxOrder, StateType >

ChebyshevII lowpass filter

6.72.2 Member Function Documentation

6.72.2.1 setup()

```
template<int MaxOrder, class StateType = DEFAULT_STATE>
void Iir::ChebyshevII::LowPass< MaxOrder, StateType >::setup (
            double sampleRate,
            double cutoffFrequency,
            double stopBandDb )  [inline]
```

Calculates the coefficients of the filter

Parameters

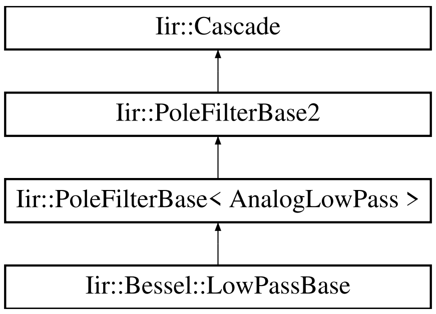
sampleRate	Sampling rate
cutoffFrequency	Cutoff frequency.
stopBandDb	Permitted ripples in dB in the stopband

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

- iir/ChebyshevII.h

6.73 Iir::Bessel::LowPassBase Struct Reference

Inheritance diagram for Iir::Bessel::LowPassBase:

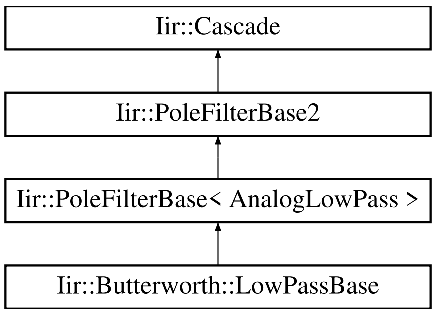


The documentation for this struct was generated from the following files:

- iir/Bessel.h
- iir/Bessel.cpp

6.74 Iir::Butterworth::LowPassBase Struct Reference

Inheritance diagram for Iir::Butterworth::LowPassBase:

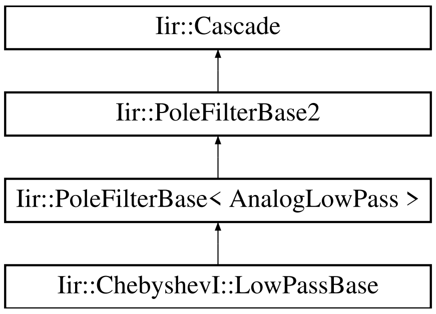

Iir::Cascade
Iir::PoleFilterBase2
Iir::PoleFilterBase< AnalogLowPass >
Iir::Butterworth::LowPassBase

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

- iir/Butterworth.h
- iir/Butterworth.cpp

6.75 Iir::ChebyshevI::LowPassBase Struct Reference

Inheritance diagram for Iir::ChebyshevI::LowPassBase:

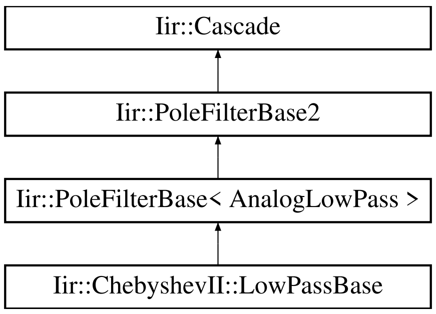

Iir::Cascade
Iir::PoleFilterBase2
Iir::PoleFilterBase< AnalogLowPass >
Iir::ChebyshevI::LowPassBase

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

- iir/ChebyshevI.h
- iir/ChebyshevI.cpp

6.76 Iir::ChebyshevII::LowPassBase Struct Reference

Inheritance diagram for Iir::ChebyshevII::LowPassBase:


Iir::Cascade
Iir::PoleFilterBase2
Iir::PoleFilterBase< AnalogLowPass >
Iir::ChebyshevII::LowPassBase

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

- iir/ChebyshevII.h
- iir/ChebyshevII.cpp

6.77 Iir::LowPassTransform Class Reference

```
#include <PoleFilter.h>
```

6.77.1 Detailed Description

s-plane to z-plane transforms

For pole filters, an analog prototype is created via placement of poles and zeros in the s-plane. The analog prototype is either a halfband low pass or a halfband low shelf. The poles, zeros, and normalization parameters are transformed into the z-plane using variants of the bilinear transformation. low pass to low pass

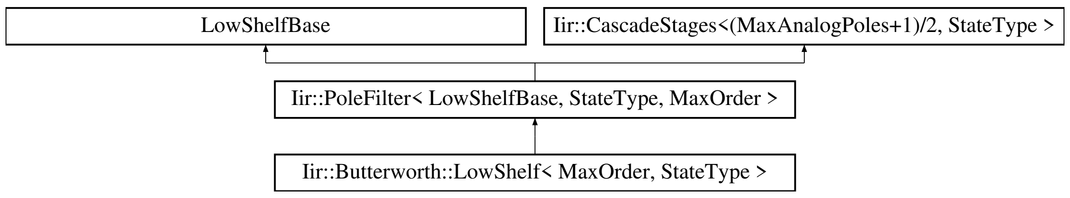
The documentation for this class was generated from the following files:

- iir/PoleFilter.h
- iir/PoleFilter.cpp

6.78 Iir::Butterworth::LowShelf< MaxOrder, StateType > Struct Template Reference

```
#include <Butterworth.h>
```

Inheritance diagram for Iir::Butterworth::LowShelf< MaxOrder, StateType >:


LowShelfBase
Iir::CascadeStages<(MaxAnalogPoles+1)/2, StateType >
Iir::PoleFilter< LowShelfBase, StateType, MaxOrder >
Iir::Butterworth::LowShelf< MaxOrder, StateType >

Public Member Functions

- void setup (double sampleRate, double cutoffFrequency, double gainDb)

6.78.1 Detailed Description

template<int MaxOrder, class StateType = DEFAULT_STATE>
struct Iir::Butterworth::LowShelf< MaxOrder, StateType >

Butterworth low shelf filter: below the cutoff it has a specified gain and above the cutoff the gain is 0 dB.

6.78.2 Member Function Documentation

6.78.2.1 setup()

```
template<int MaxOrder, class StateType = DEFAULT_STATE>
void Iir::Butterworth::LowShelf< MaxOrder, StateType >::setup (
            double sampleRate,
            double cutoffFrequency,
            double gainDb )  [inline]
```

Calculates the coefficients

Parameters

sampleRate	Sampling rate
cutoffFrequency	Cutoff
gainDb	Gain in dB of the filter in the passband

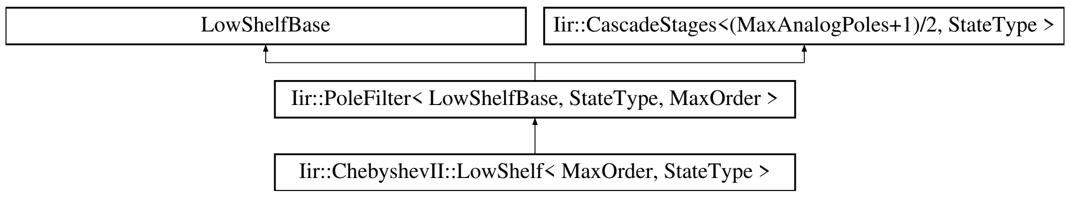
The documentation for this struct was generated from the following file:

- iir/Butterworth.h

6.79 Iir::ChebyshevII::LowShelf< MaxOrder, StateType > Struct Template Reference

```
#include <ChebyshevII.h>
```

Inheritance diagram for Iir::ChebyshevII::LowShelf< MaxOrder, StateType >:


LowShelfBase
Iir::CascadeStages<(MaxAnalogPoles+1)/2, StateType >
Iir::PoleFilter< LowShelfBase, StateType, MaxOrder >
Iir::ChebyshevII::LowShelf< MaxOrder, StateType >

Public Member Functions

- void setup (double sampleRate, double cutoffFrequency, double gainDb, double stopBandDb)

6.79.1 Detailed Description

template<int MaxOrder, class StateType = DEFAULT_STATE>
struct Iir::ChebyshevII::LowShelf< MaxOrder, StateType >

ChebyshevII low shelf filter. Specified gain in the passband and 0dB in the stopband.

6.79.2 Member Function Documentation

6.79.2.1 setup()

```
template<int MaxOrder, class StateType = DEFAULT_STATE>
void Iir::ChebyshevII::LowShelf< MaxOrder, StateType >::setup (
            double sampleRate,
            double cutoffFrequency,
            double gainDb,
            double stopBandDb )  [inline]
```

Calculates the coefficients of the filter

Parameters

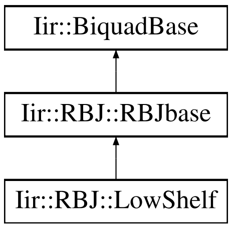
sampleRate	Sampling rate
cutoffFrequency	Cutoff frequency.
gainDb	Gain the passbard. The stopband has 0 dB gain.
stopBandDb	Permitted ripples in dB in the stopband

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

- iir/ChebyshevII.h

6.80 Iir::RBJ::LowShelf Struct Reference

Inheritance diagram for Iir::RBJ::LowShelf:



Additional Inherited Members

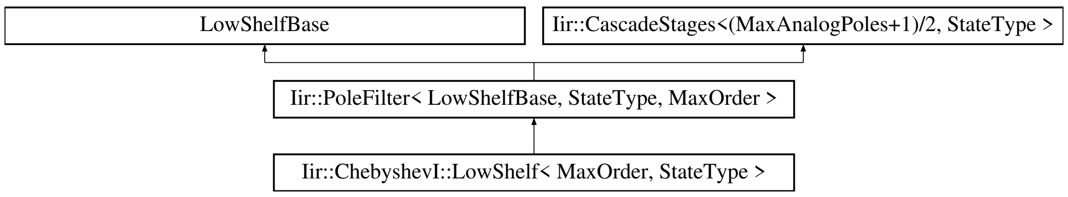
The documentation for this struct was generated from the following files:

- iir/RBJ.h
- iir/RBJ.cpp

6.81 Iir::ChebyshevI::LowShelf< MaxOrder, StateType > Struct Template Reference

```
#include <ChebyshevI.h>
```

Inheritance diagram for Iir::ChebyshevI::LowShelf< MaxOrder, StateType >:


LowShelfBase
Iir::CascadeStages<(MaxAnalogPoles+1)/2, StateType >
Iir::PoleFilter< LowShelfBase, StateType, MaxOrder >
Iir::ChebyshevI::LowShelf< MaxOrder, StateType >

Public Member Functions

- void setup (double sampleRate, double cutoffFrequency, double gainDb, double rippleDb)

6.81.1 Detailed Description

template< int MaxOrder, class StateType = DEFAULT_STATE>
struct Iir::ChebyshevI::LowShelf< MaxOrder, StateType >

ChebyshevI low shelf filter. Specified gain in the passband. Otherwise 0 dB.

6.81.2 Member Function Documentation

6.81.2.1 setup()

```
template<int MaxOrder, class StateType = DEFAULT_STATE>
void Iir::ChebyshevI::LowShelf< MaxOrder, StateType >::setup (
            double sampleRate,
            double cutoffFrequency,
            double gainDb,
            double rippleDb )  [inline]
```

Calculates the coefficients of the filter

Parameters

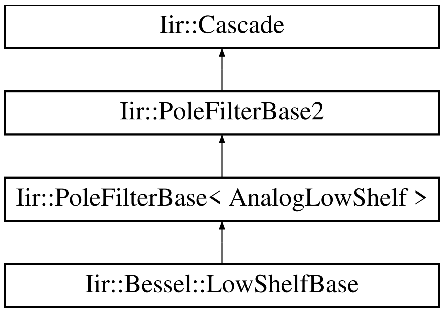
sampleRate	Sampling rate
cutoffFrequency	Cutoff frequency.
gainDb	Gain in the passband
rippleDb	Permitted ripples in dB in the passband

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

- iir/ChebyshevI.h

6.82 Iir::Bessel::LowShelfBase Struct Reference

Inheritance diagram for Iir::Bessel::LowShelfBase:

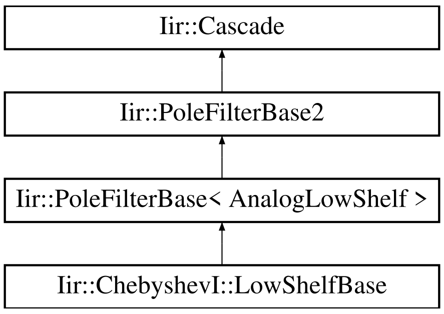

Iir::Cascade
Iir::PoleFilterBase2
Iir::PoleFilterBase< AnalogLowShelf >
Iir::Bessel::LowShelfBase

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

- iir/Bessel.h
- iir/Bessel.cpp

6.83 Iir::ChebyshevI::LowShelfBase Struct Reference

Inheritance diagram for Iir::ChebyshevI::LowShelfBase:

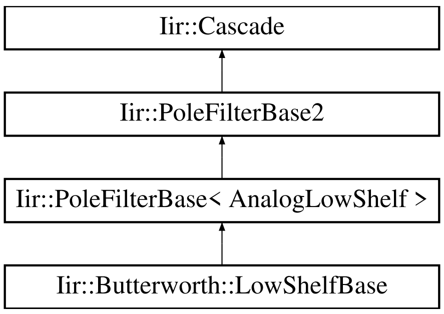

Iir::Cascade
Iir::PoleFilterBase2
Iir::PoleFilterBase< AnalogLowShelf >
Iir::ChebyshevI::LowShelfBase

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

- iir/ChebyshevI.h
- iir/ChebyshevI.cpp

6.84 Iir::Butterworth::LowShelfBase Struct Reference

Inheritance diagram for Iir::Butterworth::LowShelfBase:

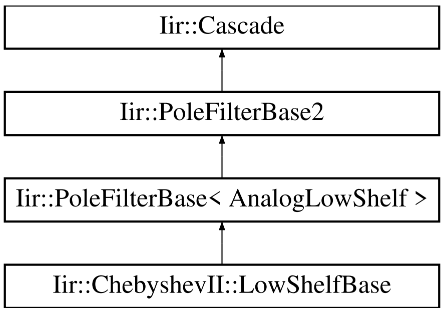

Iir::Cascade
Iir::PoleFilterBase2
Iir::PoleFilterBase< AnalogLowShelf >
Iir::Butterworth::LowShelfBase

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

- iir/Butterworth.h
- iir/Butterworth.cpp

6.85 Iir::ChebyshevII::LowShelfBase Struct Reference

Inheritance diagram for Iir::ChebyshevII::LowShelfBase:

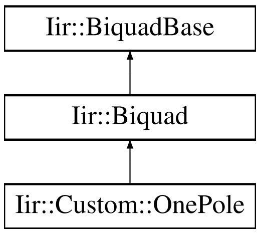

Iir::Cascade
Iir::PoleFilterBase2
Iir::PoleFilterBase< AnalogLowShelf >
Iir::ChebyshevII::LowShelfBase

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

- iir/ChebyshevII.h
- iir/ChebyshevII.cpp

6.86 Iir::Custom::OnePole Struct Reference

Inheritance diagram for Iir::Custom::OnePole:



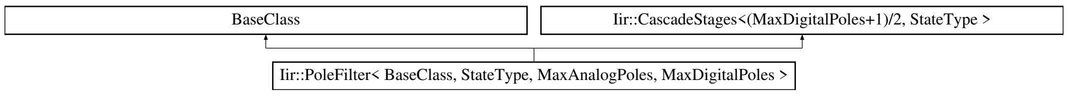
The documentation for this struct was generated from the following files:

- iir/Custom.h
- iir/Custom.cpp

6.87 Iir::PoleFilter< BaseClass, StateType, MaxAnalogPoles, MaxDigitalPoles > Struct Template Reference

```
#include <PoleFilter.h>
```

Inheritance diagram for Iir::PoleFilter< BaseClass, StateType, MaxAnalogPoles, MaxDigitalPoles >:


BaseClass
Iir::CascadeStages<(MaxDigitalPoles+1)/2, StateType >
Iir::PoleFilter< BaseClass, StateType, MaxAnalogPoles, MaxDigitalPoles >

6.87.1 Detailed Description

template<class BaseClass, class StateType, int MaxAnalogPoles, int MaxDigitalPoles = MaxAnalogPoles>
struct Iir::PoleFilter< BaseClass, StateType, MaxAnalogPoles, MaxDigitalPoles >

Storage for pole filters

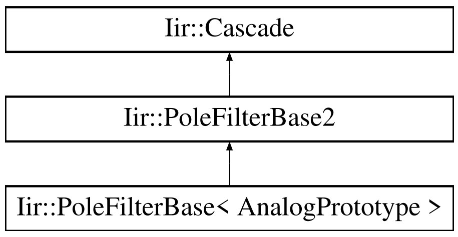
The documentation for this struct was generated from the following file:

- iir/PoleFilter.h

6.88 Iir::PoleFilterBase< AnalogPrototype > Class Template Reference

`#include <PoleFilter.h>`

Inheritance diagram for Iir::PoleFilterBase< AnalogPrototype >:



6.88.1 Detailed Description

template<class AnalogPrototype>
class Iir::PoleFilterBase< AnalogPrototype >

Serves a container to hold the analog prototype and the digital pole/zero layout.

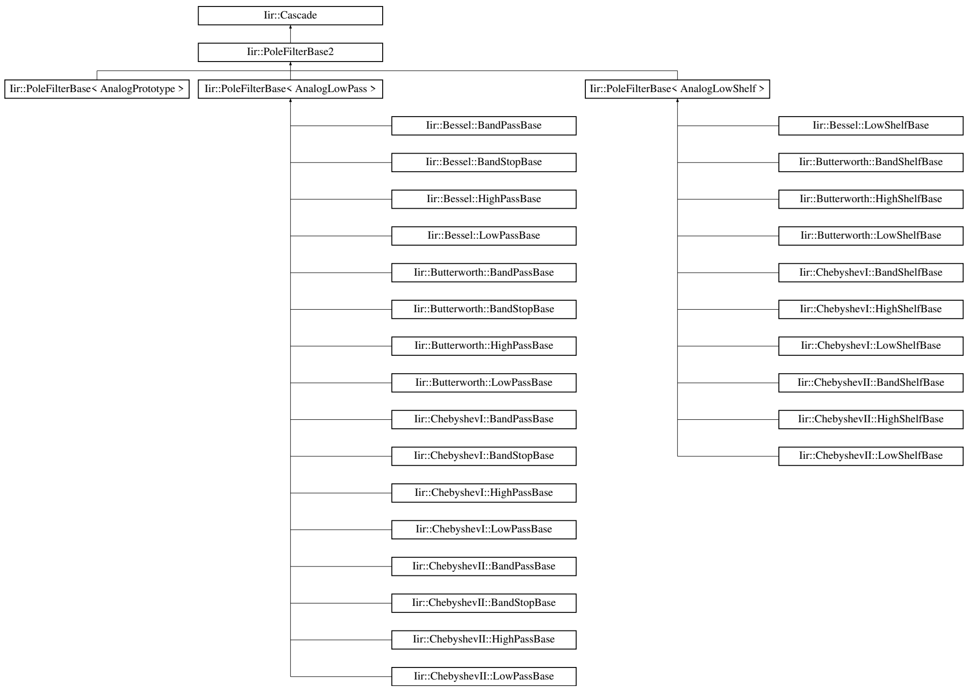
The documentation for this class was generated from the following file:

- iir/PoleFilter.h

6.89 Iir::PoleFilterBase2 Class Reference

```
#include <PoleFilter.h>
```

Inheritance diagram for Iir::PoleFilterBase2:


Iir::Cascade
Iir::PoleFilterBase2
Iir::PoleFilterBase< AnalogPrototype >
Iir::PoleFilterBase< AnalogLowPass >
Iir::PoleFilterBase< AnalogLowShelf >
Iir::Bessel::BandPassBase
Iir::Bessel::BandStopBase
Iir::Bessel::HighPassBase
Iir::Bessel::LowPassBase
Iir::Butterworth::BandPassBase
Iir::Butterworth::BandStopBase
Iir::Butterworth::HighPassBase
Iir::Butterworth::LowPassBase
Iir::ChebyshevI::BandPassBase
Iir::ChebyshevI::BandStopBase
Iir::ChebyshevI::HighPassBase
Iir::ChebyshevI::LowPassBase
Iir::ChebyshevII::BandPassBase
Iir::ChebyshevII::BandStopBase
Iir::ChebyshevII::HighPassBase
Iir::ChebyshevII::LowPassBase
Iir::Bessel::LowShelfBase
Iir::Butterworth::BandShelfBase
Iir::Butterworth::HighShelfBase
Iir::Butterworth::LowShelfBase
Iir::ChebyshevI::BandShelfBase
Iir::ChebyshevI::HighShelfBase
Iir::ChebyshevI::LowShelfBase
Iir::ChebyshevII::BandShelfBase
Iir::ChebyshevII::HighShelfBase
Iir::ChebyshevII::LowShelfBase

6.89.1 Detailed Description

Factored implementations to reduce template instantiations

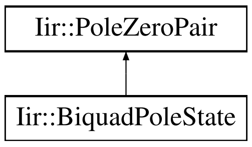
The documentation for this class was generated from the following file:

- iir/PoleFilter.h

6.90 Iir::PoleZeroPair Struct Reference

```
#include <Types.h>
```

Inheritance diagram for Iir::PoleZeroPair:



6.90.1 Detailed Description

A pair of pole/zeros. This fits in a biquad (but is missing the gain)

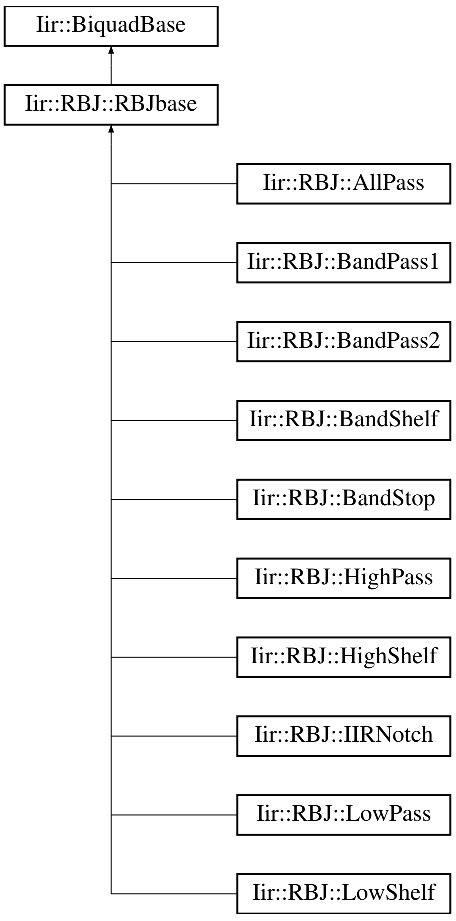
The documentation for this struct was generated from the following file:

- iir/Types.h

6.91 Iir::RBJ::RBJbase Struct Reference

```
#include <RBJ.h>
```

Inheritance diagram for Iir::RBJ::RBJbase:



Public Member Functions

- template<typename Sample >
  Sample filter (Sample s)

  filter operation
- void reset ()

  resets the delay lines to zero
- const DirectFormI & getState ()

  gets the delay lines (=state) of the filter

6.91.1 Detailed Description

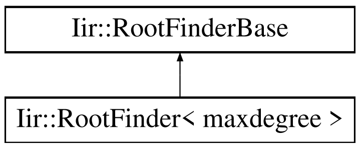
The base class of all RBJ filters

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

- iir/RBJ.h

6.92 Iir::RootFinder< maxdegree > Struct Template Reference

Inheritance diagram for Iir::RootFinder< maxdegree >:



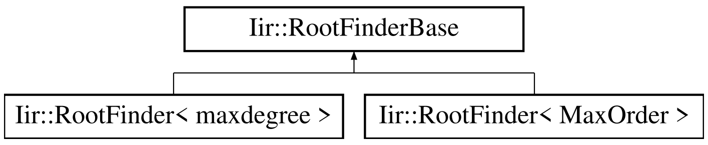
The documentation for this struct was generated from the following file:

- iir/RootFinder.h

6.93 Iir::RootFinderBase Class Reference

```
#include <RootFinder.h>
```

Inheritance diagram for Iir::RootFinderBase:



Classes

- struct Array

6.93.1 Detailed Description

Finds the complex roots of the given polynomial with complex-valued coefficients using a numerical method.

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

- iir/RootFinder.h
- iir/RootFinder.cpp

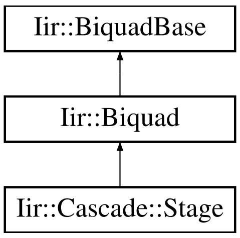
6.94 Iir::SlopeDetector< Channels, Value > Class Template Reference

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

- iir/Utilities.h

6.95 Iir::Cascade::Stage Struct Reference

Inheritance diagram for Iir::Cascade::Stage:

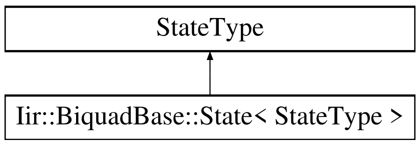


The documentation for this struct was generated from the following file:

- iir/Cascade.h

6.96 Iir::BiquadBase::State< StateType > Struct Template Reference

Inheritance diagram for Iir::BiquadBase::State< StateType >:



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

- iir/Biquad.h

6.97 Iir::Cascade::Storage Struct Reference

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

- iir/Cascade.h

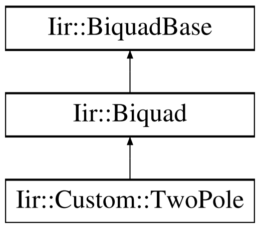
6.98 Iir::TransposedDirectFormII Class Reference

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

- iir/State.h

6.99 Iir::Custom::TwoPole Struct Reference

Inheritance diagram for Iir::Custom::TwoPole:

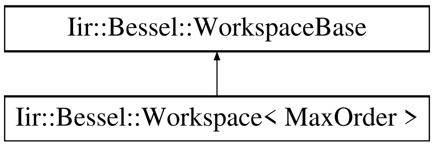


The documentation for this struct was generated from the following files:

- iir/Custom.h
- iir/Custom.cpp

6.100 Iir::Bessel::Workspace$<$ MaxOrder $>$ Struct Template Reference

Inheritance diagram for Iir::Bessel::Workspace$<$ MaxOrder $>$:

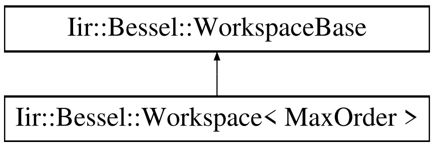

Iir::Bessel::WorkspaceBase
Iir::Bessel::Workspace< MaxOrder >

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

- iir/Bessel.h

6.101 Iir::Bessel::WorkspaceBase Struct Reference

Inheritance diagram for Iir::Bessel::WorkspaceBase:


Iir::Bessel::WorkspaceBase
Iir::Bessel::Workspace< MaxOrder >

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

- iir/Bessel.h

Index