AuLib

1.0beta

Generated by Doxygen 1.8.13

Contents

Chapter 1

Main Page

AuLib is a simple lightweight C++ (ISO C++14) class library for audio DSP.

Build

The library and example programs are built with CMake. The only optional dependencies are libsndfile, portaudio, and portmidi, but these are not strictly needed to build and use AuLib.

The code can be used within any audio processing context and it can be incorporated in plugins and other programs with their own IO.

```
$ mkdir build
$ cd build
$ cmake ..
$ make
$ make install
```

Documentation

Documentation can be built using

```
$ make doc
```

HTML, latex and manual files are placed in the ./docs subdirectory under the build directory.

Using

A number of simple examples are supplied to demonstrate how the library can be used. However, these are only very simple programs that only hint at the possible applications. Generally speaking, objects of the different processing classes are instantiated and then used to generate or process vectors of audio data, which are kept the library objects.

It is possible to access the output samples of each object directly, and processing functions can take both objects (const AudioObj&) and audio vectors (const double) as inputs (as well as other parameters, depending on the object type). A number of operations can be applied to objects (such as scaling, offsetting, mixing to, etc.).

A minimal example is shown below where an echo effect is applied to an arbitrary input, using SoundIn, SoundOut, Chn, Pan, SigBus and Delay objects.

```
#include <Chn.h>
#include <Delay.h>
#include <Pan.h>
#include <SigBus.h>
#include <SoundIn.h>
#include <SoundOut.h>
#include <iostream>
#include <numeric>
#include <vector>

using namespace AuLib;
using namespace std;

int main(int argc, const char **argv) {

  if (argc > 2) {

    SoundIn input(argv[1]);
    std::vector<Chn> chn(input.nchnls());
    std::vector<Delay> echo(input.nchnls(),
                            Delay(0.5, 0.5, def_vframes, input.sr()));
    std::vector<Pan> pan(input.nchnls());
    SigBus mix(1. / input.nchnls(), 0., false, 2);
    SoundOut output(argv[2], 2, def_bframes, input.sr());
    uint64_t end = input.dur() + 5 * input.sr(), t = 0;

    std::vector<uint32_t> channels(input.nchnls());
    std::iota(channels.begin(), channels.end(), 0);

    cout << Info::version();

    while (t < end) {
      input();
      for (uint32_t channel : channels) {
        chn[channel](input, channel + 1);
        echo[channel](chn[channel]);
        pan[channel](echo[channel] += chn[channel],
                     (1 + channel) * input.nchnls() / 2.);
        mix(pan[channel]);
      }
      t = output(mix);
      mix.clear();
    }

    return 0;

  } else
    std::cout << "usage: " << argv[0] << " <source> <dest>\n";
  return 1;
}
```

Extending

The library classes can be easily extended. There is quite a bit freedom to do this, as there are not strict prescriptions on the form or signature of processing methods. Usually we would inherit from the AudioBase class to allow easy integration with existing objects, and to avail of basic audio processing facilities provided there. Supplying a processing method that consumes a vector of samples and another that reads from an AudioObj& is the minimum necessary to allow full integration. The recommended approach is to separate interface from implementation, and to provide a means for overriding this.

The following example is a skeleton of an AudioBase-derived class that demonstrates these ideas:

```
namespace AuLib {

class NewClass : public AudioBase {

  virtual const double *dsp();

protected:
  double m_par;

public:
  NewClass(double param = .5,
           uint32_t nchnls = def_nchnls uint32_t vframes = def_vframes,
           double sr = def_sr)
      : m_param(param), AudioBase(nchnls, vframes, sr){};

  const double process(const double sig) {
    return dsp(sig);
 }

  const double process(const double sig, double par) {
    m_par = par;
    return process(sig);
  }

  const NewClass &process(const AudioBase &obj) {
    if (obj.vframes() == m_vframes && obj.nchnls() == m_nchnls) {
      process(obj.vector());
    } else
      m_error = AULIB_ERROR;
    return this;
  }

  const NewClass &process(const AudioBase &obj, double par) {
    m_par = par;
    process(obj);
    return this;
    }

  const NewClass &operator()(const AudioBase &obj) {
     return process(obj);
  }

  const NewClass &operator()(const AudioBase &obj, double par) {
     return process(obj, par);
  }
};
}
```

License

Chapter 2

Namespace Index

2.1 Namespace List

Here is a list of all namespaces with brief descriptions:

Chapter 3

Hierarchical Index

3.1 Class Hierarchy

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

Chapter 4

Class Index

4.1 Class List

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

Chapter 5

File Index

5.1 File List

Here is a list of all files with brief descriptions:

Chapter 6

Namespace Documentation

6.1 AuLib Namespace Reference

Namespaces

- fft
- Info
- midi
- waveset

Classes

- class Adsr
- class AllPass
- class AudioBase
- class Balance
- class BandP
- class BandR
- class BlOsc
- class Chn
- class Circular
- class Delay
- class Envel
- class EnvelTable
- struct Event
- class Expon
- class Fir
- class FourierTable
- class FuncTable
- struct Hamming
- struct Hann
- class HighP
- class Iir
- class Instrument
- class Line
- class LowP
- class MidiIn

- class Note
- class Oscil
- class Oscili
- class Oscilic
- class Pan
- class PConv
- class Phasor
- class Pvoc
- class Reson
- class ResonR
- class ResonZ
- class Rms
- class SamplePlayer
- class SampleTable
- struct SawOsc
- class SawTable
- class Score
- class ScorePlayer
- class Segments
- class SigBus
- class SoundIn
- class SoundOut
- struct SqOsc
- class SquareTable
- class Stft
- class TableRead
- class TableReadi
- class TableReadic
- class TableSet
- class Tap
- class Tapi
- class ToneHP
- class ToneLP
- class TriangleTable
- struct TriOsc

Typedefs

- typedef int(∗ pa_callback_t) (const void ∗, void ∗, unsigned long, const PaStreamCallbackTimeInfo ∗, unsigned long, void ∗)

Enumerations

- enum error_codes { AULIB_NOERROR = 0, AULIB_ERROR }
- enum wave_types { SAW = 1, SQUARE, TRIANGLE }
- enum sampletable_error_codes { AULIB_FILE_ERROR = AULIB_ERROR + 1, AULIB_READ_ERROR }
- enum { SOUNDIN_RT = 1, SOUNDIN_STDIN, SOUNDIN_SNDFILE }
- enum dest_types {
  SOUNDOUT_RT = 1, SOUNDOUT_STDOUT, SOUNDOUT_SNDFILE, SOUNDOUT_RT = 1,
  SOUNDOUT_STDOUT, SOUNDOUT_SNDFILE }

- enum soundout_error_codes {
  AULIB_FOPEN_ERROR = AULIB_ERROR + 1, AULIB_RTINIT_ERROR, AULIB_RTOPEN_ERROR, AU↩LIB_RTSTREAM_ERROR,
  AULIB_NOIO_ERROR, AULIB_SOUNDIO_ERROR, AULIB_FOPEN_ERROR = AULIB_ERROR + 1, AU↩LIB_RTINIT_ERROR,
  AULIB_RTOPEN_ERROR, AULIB_RTSTREAM_ERROR, AULIB_NOIO_ERROR, AULIB_SOUNDIO_ER↩ROR }
- enum dest_types {
  SOUNDOUT_RT = 1, SOUNDOUT_STDOUT, SOUNDOUT_SNDFILE, SOUNDOUT_RT = 1,
  SOUNDOUT_STDOUT, SOUNDOUT_SNDFILE }
- enum soundout_error_codes {
  AULIB_FOPEN_ERROR = AULIB_ERROR + 1, AULIB_RTINIT_ERROR, AULIB_RTOPEN_ERROR, AU↩LIB_RTSTREAM_ERROR,
  AULIB_NOIO_ERROR, AULIB_SOUNDIO_ERROR, AULIB_FOPEN_ERROR = AULIB_ERROR + 1, AU↩LIB_RTINIT_ERROR,
  AULIB_RTOPEN_ERROR, AULIB_RTSTREAM_ERROR, AULIB_NOIO_ERROR, AULIB_SOUNDIO_ER↩ROR }

Functions

- static uint32_t npow2 (uint32_t n)
- int rt_audio (const float ∗input, float ∗output, unsigned long frameCount, const void ∗timeInfo, unsigned long statusFlags, SoundIn ∗userData)
- void audio (SoundIn &obj)
- int rt_audio (const float ∗input, float ∗output, unsigned long frameCount, const void ∗timeInfo, unsigned long statusFlags, SoundOut ∗userData)
- void audio (SoundOut &obj)

Variables

- const uint32_t def_vframes = 64
- const uint32_t def_bframes = 1024
- const double def_sr = 44100.
- const double def_kr = def_sr / def_vframes
- const uint32_t def_nchnls = 1
- const uint32_t def_tframes = 8192
- const uint32_t def_fftsize = 1024
- const uint32_t def_decim = 4
- const double pi = 3.141592653589793
- const double twopi = 6.283185307179586
- const double db_min = std::numeric_limits<double>::min()
- const uint64_t ui64_max = std::numeric_limits<uint64_t>::max()
- const double m120dBfs = 0.000001
- const std::string aulib_error [ ]
- const int32_t octs = 10
- const double base = 20
- const std::string sampletable_error [ ]
- const std::string soundin_error [ ]
- const std::string soundout_error [ ]

6.1.1 Typedef Documentation

6.1.1.1 pa_callback_t

```
typedef int(* AuLib::pa_callback_t)(const void *, void *, unsigned long, const PaStreamCallback↩
TimeInfo *, unsigned long, void *)
```

6.1.2 Enumeration Type Documentation

6.1.2.1 anonymous enum

```
anonymous enum
```

Enumerator

SOUNDIN_RT	
SOUNDIN_STDIN	
SOUNDIN_SNDFILE	

6.1.2.2 dest_types [1/2]

```
enum AuLib::dest_types
```

destinations

Enumerator

SOUNDOUT_RT	
SOUNDOUT_STDOUT	
SOUNDOUT_SNDFILE	
SOUNDOUT_RT	
SOUNDOUT_STDOUT	
SOUNDOUT_SNDFILE	

6.1.2.3 dest_types [2/2]

```
enum AuLib::dest_types
```

destinations

Enumerator

SOUNDOUT_RT	

Enumerator

SOUNDOUT_STDOUT	
SOUNDOUT_SNDFILE	
SOUNDOUT_RT	
SOUNDOUT_STDOUT	
SOUNDOUT_SNDFILE	

6.1.2.4 error_codes

`enum AuLib::error_codes`

General error xodes

Enumerator

AULIB_NOERROR	
AULIB_ERROR	

6.1.2.5 sampletable_error_codes

`enum AuLib::sampletable_error_codes`

Error codes

Enumerator

AULIB_FILE_ERROR	
AULIB_READ_ERROR	

6.1.2.6 soundout_error_codes [1/2]

`enum AuLib::soundout_error_codes`

Error codes

Enumerator

AULIB_FOPEN_ERROR	
AULIB_RTINIT_ERROR	
AULIB_RTOPEN_ERROR	
AULIB_RTSTREAM_ERROR	

Enumerator

AULIB_NOIO_ERROR	
AULIB_SOUNDIO_ERROR	
AULIB_FOPEN_ERROR	
AULIB_RTINIT_ERROR	
AULIB_RTOPEN_ERROR	
AULIB_RTSTREAM_ERROR	
AULIB_NOIO_ERROR	
AULIB_SOUNDIO_ERROR	

6.1.2.7 soundout_error_codes [2/2]

```
enum AuLib::soundout_error_codes
```

Error codes

Enumerator

AULIB_FOPEN_ERROR	
AULIB_RTINIT_ERROR	
AULIB_RTOPEN_ERROR	
AULIB_RTSTREAM_ERROR	
AULIB_NOIO_ERROR	
AULIB_SOUNDIO_ERROR	
AULIB_FOPEN_ERROR	
AULIB_RTINIT_ERROR	
AULIB_RTOPEN_ERROR	
AULIB_RTSTREAM_ERROR	
AULIB_NOIO_ERROR	
AULIB_SOUNDIO_ERROR	

6.1.2.8 wave_types

```
enum AuLib::wave_types
```

Table type constants

Enumerator

SAW	
SQUARE	
TRIANGLE	

6.1.3 Function Documentation

6.1.3.1 audio() [1/2]

```
void AuLib::audio (
             SoundIn & obj )
```

6.1.3.2 audio() [2/2]

```
void AuLib::audio (
             AuLib::SoundOut & obj )
```

6.1.3.3 npow2()

```
static uint32_t AuLib::npow2 (
             uint32_t n )  [inline], [static]
```

return the next power-of-two <= n

6.1.3.4 rt_audio() [1/2]

```
int AuLib::rt_audio (
             const float * input,
             float * output,
             unsigned long frameCount,
             const void * timeInfo,
             unsigned long statusFlags,
             AuLib::SoundOut * userData )
```

6.1.3.5 rt_audio() [2/2]

```
int AuLib::rt_audio (
             const float * input,
             float * output,
             unsigned long frameCount,
             const void * timeInfo,
             unsigned long statusFlags,
             SoundIn * userData )
```

6.1.4 Variable Documentation

6.1.4.1 aulib_error

```
const std::string AuLib::aulib_error[]
```

Initial value:

```
= {"Aulib: no error",
                                        "Aulib: general object error"}
```

Standard Error messages

6.1.4.2 base

```
const double AuLib::base = 20
```

6.1.4.3 db_min

```
const double AuLib::db_min = std::numeric_limits<double>::min()
```

the min. pos. double

6.1.4.4 def_bframes

```
const uint32_t AuLib::def_bframes = 1024
```

default IO buffersize.

6.1.4.5 def_decim

```
const uint32_t AuLib::def_decim = 4
```

default decimation

6.1.4.6 def_fftsize

```
const uint32_t AuLib::def_fftsize = 1024
```

default fftsize

6.1.4.7 def_kr

```
const double AuLib::def_kr = def_sr / def_vframes
```

default control rate

6.1.4.8 def_nchnls

```
const uint32_t AuLib::def_nchnls = 1
```

default audio channels.

6.1.4.9 def_sr

```
const double AuLib::def_sr = 44100.
```

default sampling rate

6.1.4.10 def_tframes

```
const uint32_t AuLib::def_tframes = 8192
```

default function table length

6.1.4.11 def_vframes

```
const uint32_t AuLib::def_vframes = 64
```

default signal vectorsize.

6.1.4.12 m120dBfs

```
const double AuLib::m120dBfs = 0.000001
```

-120dBfs

6.1.4.13 octs

```
const int32_t AuLib::octs = 10
```

6.1.4.14 pi

```
const double AuLib::pi = 3.141592653589793
```

the pi definition.

6.1.4.15 sampletable_error

```
const std::string AuLib::sampletable_error[]
```

Initial value:

```
= {"SampleTable: file open error",
                                        "SampleTable: file read error"}
```

Standard Error messages

6.1.4.16 soundin_error

```
const std::string AuLib::soundin_error[]
```

Initial value:

```
= {
    "SoundIn: file open error",       "SoundIn: RT initialisation error",
    "SoundIn: RT open error",         "SoundIn: RT stream start error",
    "SoundOut: source not available", "SoundIn: general error"}
```

Standard Error messages

6.1.4.17 soundout_error

```
const std::string AuLib::soundout_error
```

Initial value:

```
= {"SoundOut: file open error",
                                     "SoundOut: RT initialisation error",
                                     "SoundOut: RT open error",
                                     "SoundOut: RT stream start error",
                                     "SoundOut: destination not available",
                                     "SoundOut: general error"}
```

Standard Error messages

6.1.4.18 twopi

```
const double AuLib::twopi = 6.283185307179586
```

the two pi definition.

6.1.4.19 ui64_max

```
const uint64_t AuLib::ui64_max = std::numeric_limits<uint64_t>::max()
```

the max uint64_t

6.2 AuLib::fft Namespace Reference

Functions

- void transform (std::vector< std::complex< double >> &data, bool dir)
- void transform (std::vector< std::complex< double >> &out, double ∗in, bool pckd=true)
- void transform (double ∗out, std::vector< std::complex< double >> &in, bool pckd=true)

Variables

- const bool forward = true
- const bool inverse = false
- const bool polar = true
- const bool rectang = false
- const bool packed = true

6.2.1 Function Documentation

6.2.1.1 transform() [1/3]

```
void AuLib::fft::transform (
            std::vector< std::complex< double >> & data,
            bool dir )
```

In-place complex-to-complex FFT
data - data to be transformed
dir - true for forward operation, false for inverse
pckd - true for packed data (first point is DC,Nyq) Size of data is expected to be a power-of-two.

6.2.1.2 transform() [2/3]

```
void AuLib::fft::transform (
            std::vector< std::complex< double >> & out,
            double ∗ in,
            bool pckd = true )
```

Real-to-complex forward FFT (size N)
in - pointer to real array of size N or N + 1
out - complex vector of size N/2 or N/2 + 1
pckd - true for packed data. Size of complex out array is expected to be a power-of-two (pckd = fft::packed) or power-of-two + 1 (pckd = !fftpacked) In packed form im(0) contains the Nyquist coefficient
The in and out data may reside in the same memory location for an in-place transform (but should have enough memory to hold N/2 or N/2

- 1 complex numbers).

6.2.1.3 transform() [3/3]

```
void AuLib::fft::transform (
            double * out,
            std::vector< std::complex< double >> & in,
            bool pckd = true )
```

Complex-to-real inverse FFT (size N)
in - complex vector of size N/2 or N/2 + 1
with im(0) containing the Nyquist coefficient
out - real array of size N or N + 1
Size of complex in array is expected to be a power-of-two (pckd = fft::packed) or power-of-two + 1 (pckd = !fftpacked)
In packed form im(0) contains the Nyquist coefficient
The in and out data may reside in the same memory location for an in-place transform (but should have enough memory to hold N/2 or N/2

- 1 complex numbers).

6.2.2 Variable Documentation

6.2.2.1 forward

```
const bool AuLib::fft::forward = true
```

definition of forward

6.2.2.2 inverse

```
const bool AuLib::fft::inverse = false
```

definition of inverse

6.2.2.3 packed

```
const bool AuLib::fft::packed = true
```

definition of packed

6.2.2.4 polar

```
const bool AuLib::fft::polar = true
```

definition of polar

6.2.2.5 rectang

```
const bool AuLib::fft::rectang = false
```

definition of rectang

6.3 AuLib::Info Namespace Reference

Functions

- static const std::string version ()
- static const std::string copyright ()

Variables

- const uint32_t major_version = AULIB_MAJOR_V
- const uint32_t minor_version = AULIB_MINOR_V

6.3.1 Function Documentation

6.3.1.1 copyright()

```
static const std::string AuLib::Info::copyright ( )  [inline], [static]
```

returns the copyright string

6.3.1.2 version()

```
static const std::string AuLib::Info::version ( )  [inline], [static]
```

returns the version string

6.3.2 Variable Documentation

6.3.2.1 major_version

```
const uint32_t AuLib::Info::major_version = AULIB_MAJOR_V
```

Aulib major version

6.3.2.2 minor_version

```
const uint32_t AuLib::Info::minor_version = AULIB_MINOR_V
```

Aulib minor version

6.4 AuLib::midi Namespace Reference

Variables

- const uint32_t note_on = 0x90
- const uint32_t note_off = 0x80
- const uint32_t ctrl_msg = 0xB0
- const uint32_t aftouch = 0xD0
- const uint32_t poly_aftouch = 0xA0
- const uint32_t prg_msg = 0xC0
- const uint32_t pitchbend = 0xE0

6.4.1 Variable Documentation

6.4.1.1 aftouch

```
const uint32_t AuLib::midi::aftouch = 0xD0
```

MIDI aftertouch

6.4.1.2 ctrl_msg

```
const uint32_t AuLib::midi::ctrl_msg = 0xB0
```

MIDI control change

6.4.1.3 note_off

```
const uint32_t AuLib::midi::note_off = 0x80
```

MIDI note off

6.4.1.4 note_on

```
const uint32_t AuLib::midi::note_on = 0x90
```

MIDI note on

6.4.1.5 pitchbend

```
const uint32_t AuLib::midi::pitchbend = 0xE0
```

MIDI pitchbend

6.4.1.6 poly_aftouch

```
const uint32_t AuLib::midi::poly_aftouch = 0xA0
```

MIDI poly aftertouch

6.4.1.7 prg_msg

```
const uint32_t AuLib::midi::prg_msg = 0xC0
```

MIDI program change

6.5 AuLib::waveset Namespace Reference

Functions

- static const TableSet saw (SAW)
- static const TableSet triangle (TRIANGLE)
- static const TableSet square (SQUARE)

6.5.1 Function Documentation

6.5.1.1 saw()

```
static const TableSet AuLib::waveset::saw (
             SAW  )  [static]
```

Sawtooth table set

6.5.1.2 square()

```
static const TableSet AuLib::waveset::square (
             SQUARE  )  [static]
```

Square table set

6.5.1.3 triangle()

```
static const TableSet AuLib::waveset::triangle (
             TRIANGLE  )  [static]
```

Triangle table set

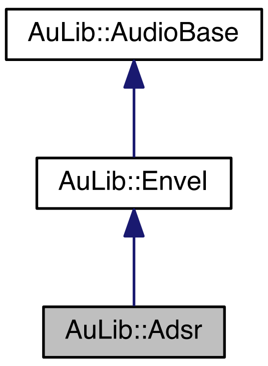
Chapter 7

Class Documentation

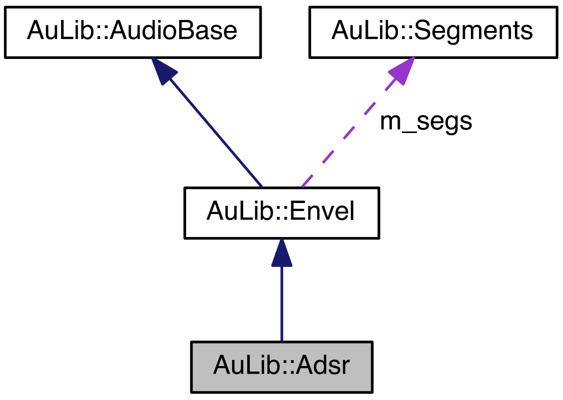
7.1 AuLib::Adsr Class Reference

```
#include <AuLib/Adsr.h>
```

Inheritance diagram for AuLib::Adsr:



Collaboration diagram for AuLib::Adsr:



Public Member Functions

- Adsr (double amp, double att, double dec, double sus, double rel, uint32_t vframes=def_vframes, double sr=def_sr)
- void reset (double amp, double att, double dec, double sus, double rel)

Additional Inherited Members

7.1.1 Detailed Description

Adsr description

7.1.2 Constructor & Destructor Documentation

7.1.2.1 Adsr()

```
AuLib::Adsr::Adsr (
            double amp,
            double att,
            double dec,
            double sus,
            double rel,
            uint32_t vframes = def_vframes,
            double sr = def_sr )  [inline]
```

Adsr constructor

amp - amplitude after attack
att - attack time
dec - decay time
rel - release time
vframes - vector size
sr - sampling rate

7.1.3 Member Function Documentation

7.1.3.1 reset()

```
void AuLib::Adsr::reset (
             double amp,
             double att,
             double dec,
             double sus,
             double rel )  [inline]
```

reset the envelope parameters and retrigger

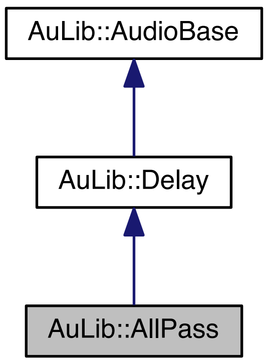
The documentation for this class was generated from the following file:

- Adsr.h

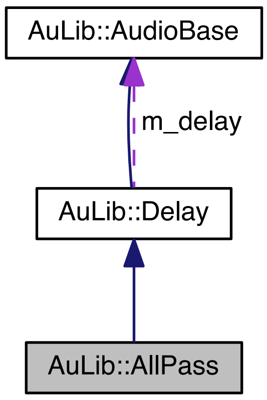
7.2 AuLib::AllPass Class Reference

```
#include <AuLib/AllPass.h>
```

Inheritance diagram for AuLib::AllPass:



Collaboration diagram for AuLib::AllPass:



Public Member Functions

- AllPass (double dtime, double fdb, uint32_t vframes=def_vframes, double sr=def_sr)

Additional Inherited Members

7.2.1 Detailed Description

All-pass filter

7.2.2 Constructor & Destructor Documentation

7.2.2.1 AllPass()

```
AuLib::AllPass::AllPass (
             double dtime,
             double fdb,
             uint32_t vframes = def_vframes,
             double sr = def_sr ) [inline]
```

AllPass constructor

dtime - delay time
fdb - feedback gain
vframes - vector size
sr - sampling rate

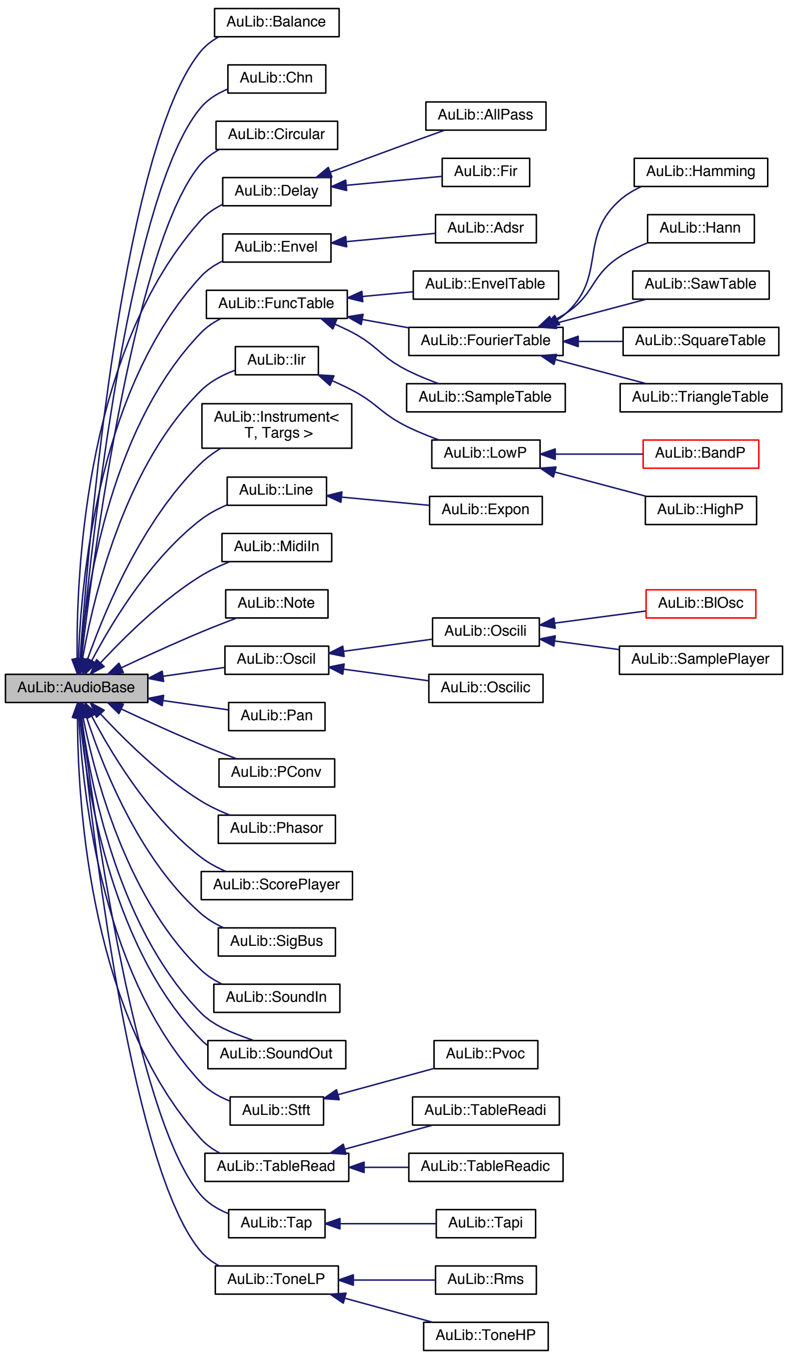
The documentation for this class was generated from the following files:

- AllPass.h
- Delay.cpp

7.3 AuLib::AudioBase Class Reference

```
#include <AuLib/AudioBase.h>
```

Inheritance diagram for AuLib::AudioBase:



Public Types

- typedef std::vector< double >::iterator iterator
- typedef std::vector< double >::const_iterator const_iterator

Public Member Functions

- AudioBase (uint32_t nchnls=def_nchnls, uint32_t vframes=def_vframes, double sr=def_sr)
- virtual const AudioBase & operator*= (double scal)
- virtual const AudioBase & operator*= (const double *sig)
- virtual const AudioBase & operator*= (const AudioBase &obj)
- virtual const AudioBase & operator+= (double offs)
- virtual const AudioBase & operator+= (const double *sig)
- virtual const AudioBase & operator+= (const AudioBase &obj)
- double & operator[ ] (uint32_t ndx)
- const double & operator[ ] (uint32_t ndx) const
- iterator begin ()
- iterator end ()
- const_iterator cbegin () const
- const_iterator cend () const
- const AudioBase & set (const AudioBase &obj)
- const AudioBase & set (const double *sig)
- const double * set (double v)
- double set (double v, uint32_t p)
- operator const std::vector< double > & () const
- operator const double * () const
- const double * vector () const
- double vector (uint32_t frndx, uint32_t chn) const
- uint32_t vframes (uint32_t frames)
- uint32_t resize_exact (uint32_t frames)
- uint32_t vframes () const
- uint32_t vsamps () const
- uint32_t nchnls () const
- uint32_t sr () const
- uint32_t error () const
- virtual const char * error_message () const

Protected Attributes

- uint32_t m_nchnls
- uint32_t m_vframes
- std::vector< double > m_vector
- double m_sr
- uint32_t m_error

Friends

- std::ostream & operator<< (std::ostream &os, const AudioBase &obj)
- std::istream & operator>> (std::istream &is, AudioBase &obj)

7.3.1 Detailed Description

Audio DSP base class

7.3.2 Member Typedef Documentation

7.3.2.1 const_iterator

```
typedef std::vector<double>::const_iterator AuLib::AudioBase::const_iterator
```

const iterator for this class

7.3.2.2 iterator

```
typedef std::vector<double>::iterator AuLib::AudioBase::iterator
```

iterator for this class

7.3.3 Constructor & Destructor Documentation

7.3.3.1 AudioBase()

```
AuLib::AudioBase::AudioBase (
            uint32_t nchnls = def_nchnls,
            uint32_t vframes = def_vframes,
            double sr = def_sr )  [inline]
```

AudioBase constructor
nchnls - number of channels
vframes - number of frames in vector. This is set to next power-of-two no greater than the requested number of frames. Objects requiring arbitrary vector sizes should explicity re-size the vector using the resize_exact() method.
sr - sampling rate

7.3.4 Member Function Documentation

7.3.4.1 begin()

```
iterator AuLib::AudioBase::begin ( )  [inline]
```

returns an iterator to the beginning

7.3.4.2 cbegin()

```
const_iterator AuLib::AudioBase::cbegin ( ) const  [inline]
```

returns a const iterator to the beginning

7.3.4.3 cend()

```
const_iterator AuLib::AudioBase::cend ( ) const  [inline]
```

returns a const iterator to the end

7.3.4.4 end()

```
iterator AuLib::AudioBase::end ( )  [inline]
```

returns an iterator to the end

7.3.4.5 error()

```
uint32_t AuLib::AudioBase::error ( ) const  [inline]
```

Get error code

7.3.4.6 error_message()

```
virtual const char* AuLib::AudioBase::error_message ( ) const  [inline], [virtual]
```

Get error message

Reimplemented in AuLib::SoundOut, AuLib::SoundOut, AuLib::SoundIn, and AuLib::SampleTable.

7.3.4.7 nchnls()

```
uint32_t AuLib::AudioBase::nchnls ( ) const  [inline]
```

Get number of channels

7.3.4.8 operator const double ∗()

```
AuLib::AudioBase::operator const double * ( ) const  [inline], [explicit]
```

Conversion operator for const double∗

7.3.4.9 operator const std::vector< double > &()

```
AuLib::AudioBase::operator const std::vector< double > & ( ) const  [inline]
```

Conversion operator for const std::vector<double>&

7.3.4.10 operator∗=() [1/3]

```
virtual const AudioBase& AuLib::AudioBase::operator∗= (
             double scal )  [inline], [virtual]
```

Scale the data vector

Reimplemented in AuLib::Stft, and AuLib::Pvoc.

7.3.4.11 operator∗=() [2/3]

```
virtual const AudioBase& AuLib::AudioBase::operator∗= (
             const double ∗ sig )  [inline], [virtual]
```

Multiply the data vector by a sig vector

Reimplemented in AuLib::Stft, and AuLib::Pvoc.

7.3.4.12 operator∗=() [3/3]

```
virtual const AudioBase& AuLib::AudioBase::operator∗= (
             const AudioBase & obj )  [inline], [virtual]
```

Multiply the data vector by the vector from obj

7.3.4.13 operator+=() [1/3]

```
virtual const AudioBase& AuLib::AudioBase::operator+= (
             double offs )  [inline], [virtual]
```

DC offset the data vector

Reimplemented in AuLib::Stft, and AuLib::Pvoc.

7.3.4.14 operator+=() [2/3]

```
virtual const AudioBase& AuLib::AudioBase::operator+= (
             const double * sig )  [inline], [virtual]
```

Add a vector sig to the data vector

Reimplemented in AuLib::Stft, and AuLib::Pvoc.

7.3.4.15 operator+=() [3/3]

```
virtual const AudioBase& AuLib::AudioBase::operator+= (
             const AudioBase & obj )  [inline], [virtual]
```

Add a vector sig from obj to the data vector

7.3.4.16 operator[]() [1/2]

```
double& AuLib::AudioBase::operator[] (
             uint32_t ndx )  [inline]
```

Get a reference of a single sample at sample pos ndx off the data vector

7.3.4.17 operator[]() [2/2]

```
const double& AuLib::AudioBase::operator[] (
             uint32_t ndx ) const  [inline]
```

Get a constant reference of a single sample at sample pos ndx off the data vector

7.3.4.18 resize_exact()

```
uint32_t AuLib::AudioBase::resize_exact (
             uint32_t frames )  [inline]
```

Resize the vector to given number of frames, exactly. This is used to set the vector size to values other than power-of-two sizes, for dedicated applications where this is required. Clears the vector and returns the updated vector frame size.

7.3.4.19 set() [1/4]

```
const AudioBase& AuLib::AudioBase::set (
             const AudioBase & obj )  [inline]
```

set the data vector to an input obj vector return the AudioBase obj reference

7.3.4.20 set() [2/4]

```
const AudioBase& AuLib::AudioBase::set (
             const double * sig )  [inline]
```

set the data vector to a sig vector return the AudioBase obj reference

7.3.4.21 set() [3/4]

```
const double* AuLib::AudioBase::set (
             double v )  [inline]
```

set the data vector to a value v return a pointer to the vector

7.3.4.22 set() [4/4]

```
double AuLib::AudioBase::set (
             double v,
             uint32_t p )  [inline]
```

set the data vector to a value v at pos p return the old (replaced) sample

7.3.4.23 sr()

```
uint32_t AuLib::AudioBase::sr ( ) const  [inline]
```

Get sampling reate

7.3.4.24 vector() [1/2]

```
const double* AuLib::AudioBase::vector ( ) const  [inline]
```

Get the audio vector

7.3.4.25 vector() [2/2]

```
double AuLib::AudioBase::vector (
             uint32_t frndx,
             uint32_t chn ) const  [inline]
```

Get a single sample at frame ndx and channel chn off the data vector

7.3.4.26 vframes() [1/2]

```
uint32_t AuLib::AudioBase::vframes (
             uint32_t frames )  [inline]
```

Set the current vector size in frames. Clears the vector and returns the updated vector frame size. Vector is resized if the requested frame size cannot be accommodated. Size is set to next power-of-two no greater than the requested number of frames.

7.3.4.27 vframes() [2/2]

```
uint32_t AuLib::AudioBase::vframes ( ) const  [inline]
```

Get current vector size in frames

7.3.4.28 vsamps()

```
uint32_t AuLib::AudioBase::vsamps ( ) const  [inline]
```

Get (max/allocated) vector size in samples

7.3.5 Friends And Related Function Documentation

7.3.5.1 operator<<

```
std::ostream& operator<< (
            std::ostream & os,
            const AudioBase & obj )  [friend]
```

Stream output operator

7.3.5.2 operator>>

```
std::istream& operator>> (
            std::istream & is,
            AudioBase & obj )  [friend]
```

Stream input operator

7.3.6 Member Data Documentation

7.3.6.1 m_error

```
uint32_t AuLib::AudioBase::m_error  [protected]
```

7.3.6.2 m_nchnls

```
uint32_t AuLib::AudioBase::m_nchnls  [protected]
```

7.3.6.3 m_sr

```
double AuLib::AudioBase::m_sr [protected]
```

7.3.6.4 m_vector

```
std::vector<double> AuLib::AudioBase::m_vector [protected]
```

7.3.6.5 m_vframes

```
uint32_t AuLib::AudioBase::m_vframes [protected]
```

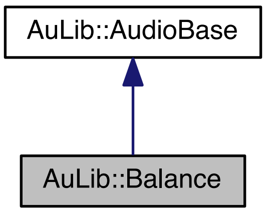
The documentation for this class was generated from the following file:

- AudioBase.h

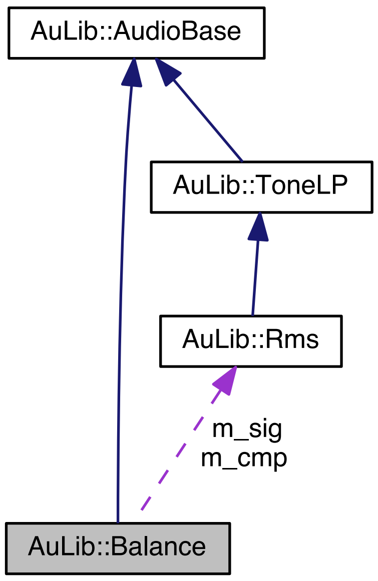
7.4 AuLib::Balance Class Reference

```
#include <AuLib/Balance.h>
```

Inheritance diagram for AuLib::Balance:



Collaboration diagram for AuLib::Balance:



Public Member Functions

- Balance (double cf=10., uint32_t vframes=def_vframes, double sr=def_sr)
- const double ∗ process (const double ∗sig, const double ∗cmp)
- const Balance & process (const AudioBase &obj, const AudioBase &cmp)
- const Balance & operator() (const AudioBase &obj, const AudioBase &cmp)

Protected Attributes

- Rms m_cmp
- Rms m_sig

Additional Inherited Members

7.4.1 Detailed Description

Balance description

7.4.2 Constructor & Destructor Documentation

7.4.2.1 Balance()

```
AuLib::Balance::Balance (
             double cf = 10.,
             uint32_t vframes = def_vframes,
             double sr = def_sr )  [inline]
```

Balance constructor

cf - LP cutoff freq
nchnls - number of channels
vframes - vector size
sr - sampling rate

7.4.3 Member Function Documentation

7.4.3.1 operator()()

```
const Balance& AuLib::Balance::operator() (
             const AudioBase & obj,
             const AudioBase & cmp )  [inline]
```

operator(a,b) convenience method, same as process()

7.4.3.2 process() [1/2]

```
const double* AuLib::Balance::process (
             const double * sig,
             const double * cmp )  [inline]
```

process a sig with a comparator cmp

7.4.3.3 process() [2/2]

```
const Balance& AuLib::Balance::process (
             const AudioBase & obj,
             const AudioBase & cmp )  [inline]
```

process a signal in obj

7.4.4 Member Data Documentation

7.4.4.1 m_cmp

```
Rms AuLib::Balance::m_cmp [protected]
```

7.4.4.2 m_sig

```
Rms AuLib::Balance::m_sig [protected]
```

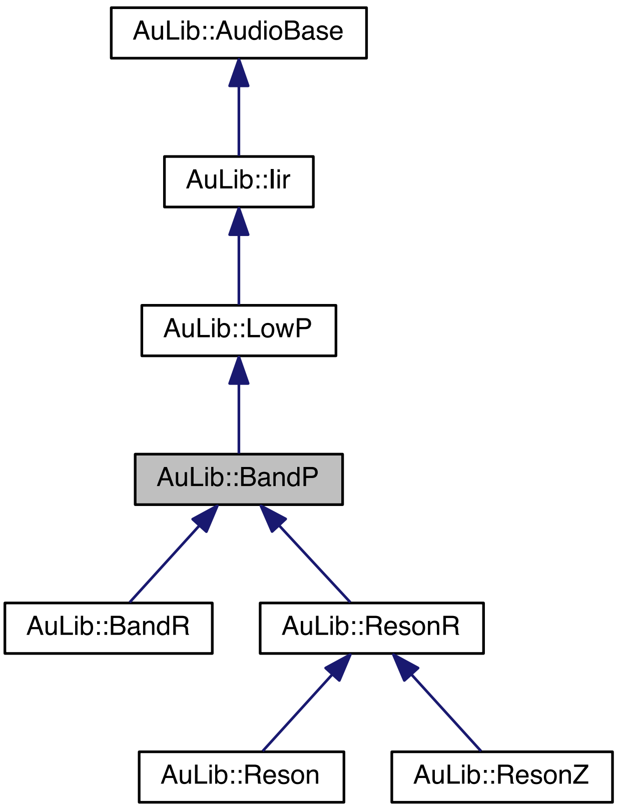
The documentation for this class was generated from the following files:

- Balance.h
- Balance.cpp

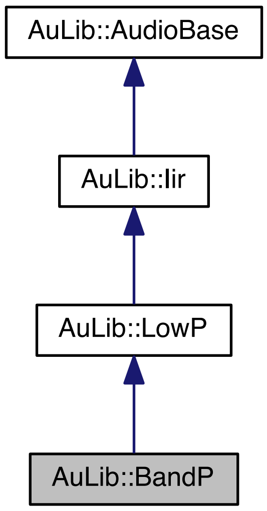
7.5 AuLib::BandP Class Reference

```
#include <AuLib/BandP.h>
```

Inheritance diagram for AuLib::BandP:



Collaboration diagram for AuLib::BandP:



Public Member Functions

- BandP (double cf, double bw, uint32_t vframes=def_vframes, double sr=def_sr)
- const double ∗ process (const double ∗sig, double cf, double bw)
- const BandP & process (const AudioBase &obj, double cf, double bw)

Protected Member Functions

- virtual void update ()

Protected Attributes

- double m_bw

Additional Inherited Members

7.5.1 Detailed Description

2nd-order Butterworth band-pass filter

7.5.2 Constructor & Destructor Documentation

7.5.2.1 BandP()

```
AuLib::BandP::BandP (
             double cf,
             double bw,
             uint32_t vframes = def_vframes,
             double sr = def_sr )  [inline]
```

BandP constructor

cf - centre frequency
bw - bandwidth
vframes - vector size
sr - sampling rate

7.5.3 Member Function Documentation

7.5.3.1 process() [1/2]

```
const double* AuLib::BandP::process (
             const double * sig,
             double cf,
             double bw )  [inline]
```

process a signal sig with cutoff freq cf and bandwidth bw

7.5.3.2 process() [2/2]

```
const BandP& AuLib::BandP::process (
             const AudioBase & obj,
             double cf,
             double bw )  [inline]
```

process a signal in obj with cutoff freq cf and bandwidth bw

7.5.3.3 update()

```
void AuLib::BandP::update ( )  [protected], [virtual]
```

Coefficients update

Reimplemented from AuLib::LowP.

Reimplemented in AuLib::ResonR, AuLib::ResonZ, and AuLib::BandR.

7.5.4 Member Data Documentation

7.5.4.1 m_bw

```
double AuLib::BandP::m_bw  [protected]
```

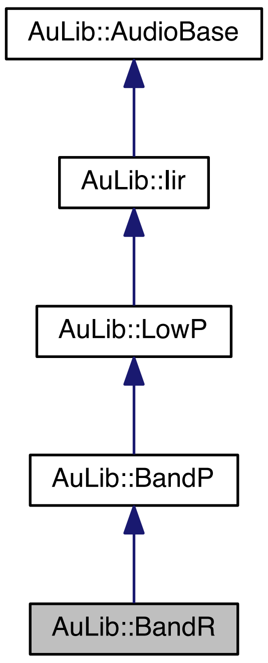
The documentation for this class was generated from the following files:

- BandP.h
- Iir.cpp

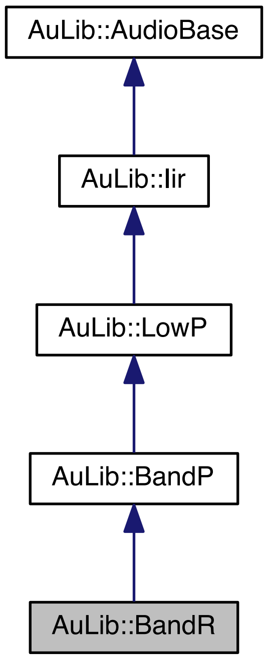
7.6 AuLib::BandR Class Reference

```
#include <AuLib/BandR.h>
```

Inheritance diagram for AuLib::BandR:



Collaboration diagram for AuLib::BandR:



Public Member Functions

- BandR (double cf, double bw, uint32_t vframes=def_vframes, double sr=def_sr)

Protected Member Functions

- virtual void update ()

Additional Inherited Members

7.6.1 Detailed Description

2nd-order Butterworth band-reject filter

7.6.2 Constructor & Destructor Documentation

7.6.2.1 BandR()

```
AuLib::BandR::BandR (
             double cf,
             double bw,
             uint32_t vframes = def_vframes,
             double sr = def_sr )  [inline]
```

BandR constructor

cf - centre frequency
bw - bandwidth
vframes - vector size
sr - sampling rate

7.6.3 Member Function Documentation

7.6.3.1 update()

```
void AuLib::BandR::update ( )  [protected], [virtual]
```

Coefficients update

Reimplemented from AuLib::BandP.

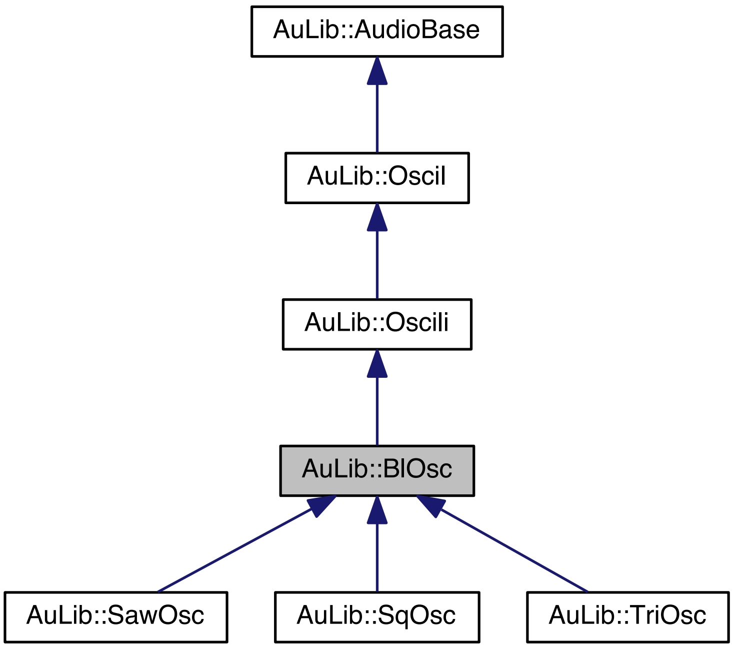
The documentation for this class was generated from the following files:

- BandR.h
- Iir.cpp

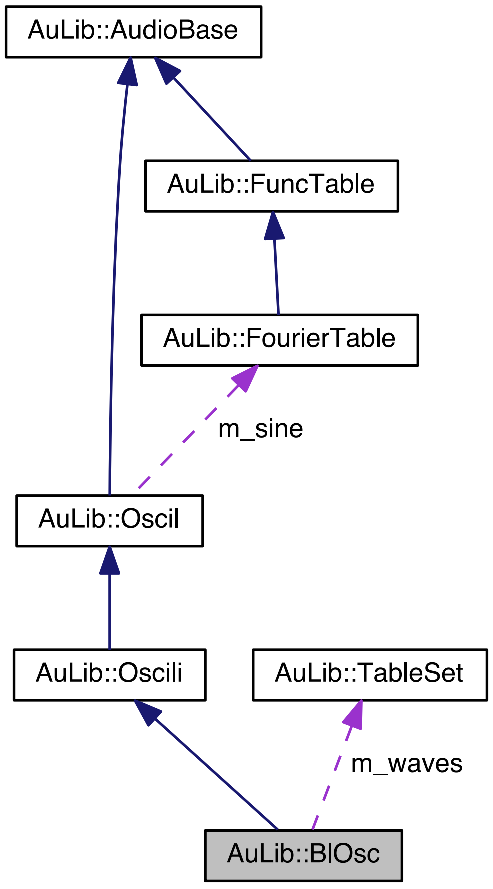
7.7 AuLib::BlOsc Class Reference

```
#include <AuLib/BlOsc.h>
```

Inheritance diagram for AuLib::BlOsc:



Collaboration diagram for AuLib::BlOsc:



Public Member Functions

- BlOsc (double amp, double freq, const TableSet &waveset, double phase=0., uint32_t vframes=def_vframes, double sr=def_sr)

Protected Member Functions

- virtual void am_fm (uint32_t ndx)
- void tselect ()
- virtual void set_incr (double f)

Protected Attributes

- const TableSet & m_waves

Additional Inherited Members

7.7.1 Detailed Description

Bandlimited wavetable oscillator

7.7.2 Constructor & Destructor Documentation

7.7.2.1 BlOsc()

```
AuLib::BlOsc::BlOsc (
            double amp,
            double freq,
            const TableSet & waveset,
            double phase = 0.,
            uint32_t vframes = def_vframes,
            double sr = def_sr )  [inline]
```

BlOsc constructor

amp - amplitude
freq - frequency in Hz
waveset - TableSet reference with the set of bandlimited tables
phase - init phase (0-1)
vframes - vector size
sr - sampling rate

7.7.3 Member Function Documentation

7.7.3.1 am_fm()

```
virtual void AuLib::BlOsc::am_fm (
            uint32_t ndx )  [inline], [protected], [virtual]
```

AM/FM processing

Reimplemented from AuLib::Oscil.

7.7.3.2 set_incr()

```
virtual void AuLib::BlOsc::set_incr (
             double f )  [inline], [protected], [virtual]
```

set the sampling increment

Reimplemented from AuLib::Oscil.

7.7.3.3 tselect()

```
void AuLib::BlOsc::tselect ( )  [inline], [protected]
```

table selection

7.7.4 Member Data Documentation

7.7.4.1 m_waves

```
const TableSet& AuLib::BlOsc::m_waves  [protected]
```

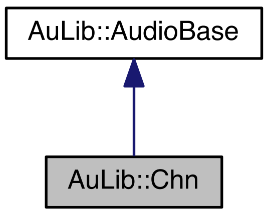
The documentation for this class was generated from the following file:

- BlOsc.h

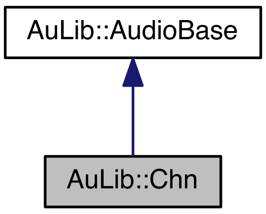
7.8 AuLib::Chn Class Reference

```
#include <AuLib/Chn.h>
```

Inheritance diagram for AuLib::Chn:



Collaboration diagram for AuLib::Chn:



Public Member Functions

- Chn (uint32_t channel=1, uint32_t vframes=def_vframes)
- const double ∗ process (const double ∗sig, uint32_t nchnls)
- const double ∗ process (const double ∗sig, uint32_t chn, uint32_t nchnls)
- const Chn & process (const AudioBase &obj)
- const Chn & process (const AudioBase &obj, uint32_t chn)
- const Chn & operator() (const AudioBase &obj)
- const Chn & operator() (const AudioBase &obj, uint32_t chn)

Protected Attributes

- uint32_t m_chn

Additional Inherited Members

7.8.1 Detailed Description

Extracts a single channel from a multichannel interleaved input

7.8.2 Constructor & Destructor Documentation

7.8.2.1 Chn()

```
AuLib::Chn::Chn (
            uint32_t channel = 1,
            uint32_t vframes = def_vframes )  [inline]
```

Chn constructor

channel - channel to extract
vframes - vector size
sr - sampling rate

7.8.3 Member Function Documentation

7.8.3.1 operator()() [1/2]

```
const Chn& AuLib::Chn::operator() (
             const AudioBase & obj )  [inline]
```

operator(a) convenience method

7.8.3.2 operator()() [2/2]

```
const Chn& AuLib::Chn::operator() (
             const AudioBase & obj,
             uint32_t chn )  [inline]
```

operator(a,b) convenience method

7.8.3.3 process() [1/4]

```
const double* AuLib::Chn::process (
             const double * sig,
             uint32_t nchnls )  [inline]
```

extracts a channel from sig holding nchnls channels

7.8.3.4 process() [2/4]

```
const double* AuLib::Chn::process (
             const double * sig,
             uint32_t chn,
             uint32_t nchnls )  [inline]
```

extracts channel chn from sig holding nchnls channels

7.8.3.5 process() [3/4]

```
const Chn& AuLib::Chn::process (
             const AudioBase & obj )  [inline]
```

extracts a channel from obj

7.8.3.6 process() [4/4]

```
const Chn& AuLib::Chn::process (
             const AudioBase & obj,
             uint32_t chn )  [inline]
```

extracts channel chn from obj

7.8.4 Member Data Documentation

7.8.4.1 m_chn

```
uint32_t AuLib::Chn::m_chn  [protected]
```

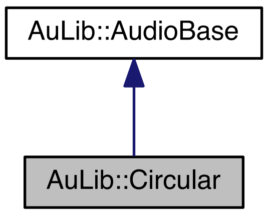
The documentation for this class was generated from the following file:

- Chn.h

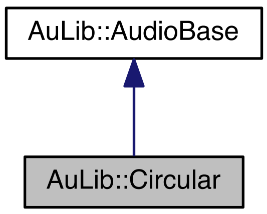
7.9 AuLib::Circular Class Reference

```
#include <AuLib/Circular.h>
```

Inheritance diagram for AuLib::Circular:



Collaboration diagram for AuLib::Circular:



Public Member Functions

- Circular (uint32_t size, uint32_t nchnls=def_nchnls, uint32_t vframes=def_vframes, double sr=def_sr)
- bool writes (const double ∗sig)
- const double ∗ reads ()
- void write (const AudioBase &obj)
- const AudioBase & read ()
- void operator() (const AudioBase &obj)
- const AudioBase & operator() ()
- uint32_t size () const
- bool is_empty () const

Additional Inherited Members

7.9.1 Detailed Description

Circular buffer: implements a buffer with atomic access

7.9.2 Constructor & Destructor Documentation

7.9.2.1 Circular()

```
AuLib::Circular::Circular (
             uint32_t size,
             uint32_t nchnls = def_nchnls,
             uint32_t vframes = def_vframes,
             double sr = def_sr )  [inline]
```

Constructs a circular buffer with a given size in frames

7.9.3 Member Function Documentation

7.9.3.1 is_empty()

```
bool AuLib::Circular::is_empty ( ) const  [inline]
```

check if buffer is empty

7.9.3.2 operator()() [1/2]

```
void AuLib::Circular::operator() (
             const AudioBase & obj )  [inline]
```

Convenience operator(), same as write()

7.9.3.3 operator()() [2/2]

```
const AudioBase& AuLib::Circular::operator() ( ) [inline]
```

Convenience operator(), same as read()

7.9.3.4 read()

```
const AudioBase& AuLib::Circular::read ( ) [inline]
```

Reads a block of frames from the circular buffer, returns a reference to this object, which contains the output.

7.9.3.5 reads()

```
const double* AuLib::Circular::reads ( ) [inline]
```

Reads a block of vframes() samples from the circular buffer into the object vector. If the buffer is empty, nothing is written and the function returns a nullptr. Otherwise, it returns a pointer to the vector.

7.9.3.6 size()

```
uint32_t AuLib::Circular::size ( ) const [inline]
```

Circular buffer size in frames

7.9.3.7 write()

```
void AuLib::Circular::write (
             const AudioBase & obj ) [inline]
```

Writes a block of frames from obj into the circular buffer

7.9.3.8 writes()

```
bool AuLib::Circular::writes (
             const double * sig ) [inline]
```

Writes a block of vframes() samples into the circular buffer. There should be at least vframes() samples in sig. If the buffer is full, nothing is written and the function returns false. Returns true on success.

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

- Circular.h
- Circular.cpp

7.10 AuLib::Score::Cmd Struct Reference

```
#include <Score.h>
```

Public Attributes

- std::string cmd
- bool mode
- uint32_t msg
- uint32_t len

Static Public Attributes

- static constexpr uint32_t omni = true
- static constexpr uint32_t chn = false

7.10.1 Detailed Description

score command

7.10.2 Member Data Documentation

7.10.2.1 chn

```
constexpr uint32_t AuLib::Score::Cmd::chn = false  [static]
```

7.10.2.2 cmd

```
std::string AuLib::Score::Cmd::cmd
```

7.10.2.3 len

```
uint32_t AuLib::Score::Cmd::len
```

7.10.2.4 mode

```
bool AuLib::Score::Cmd::mode
```

7.10.2.5 msg

```
uint32_t AuLib::Score::Cmd::msg
```

7.10.2.6 omni

```
constexpr uint32_t AuLib::Score::Cmd::omni = true  [static]
```

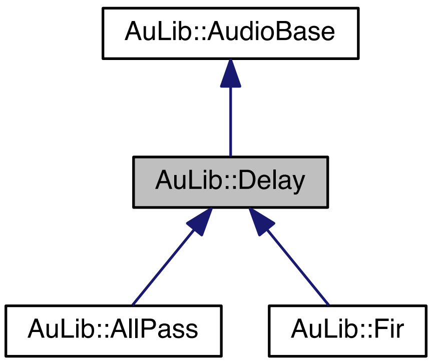
The documentation for this struct was generated from the following file:

- Score.h

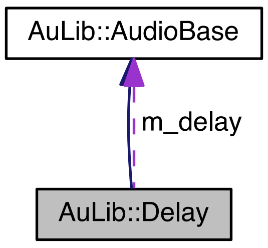
7.11 AuLib::Delay Class Reference

```
#include <AuLib/Delay.h>
```

Inheritance diagram for AuLib::Delay:



Collaboration diagram for AuLib::Delay:



Public Member Functions

- Delay (double dtime, double fdb, uint32_t vframes=def_vframes, double sr=def_sr)
- const double ∗ process (const double ∗sig)
- const double ∗ process (const double ∗sig, double dt)
- const double ∗ process (const double ∗sig, double dt, double fdb)
- const double ∗ process (const double ∗sig, const double ∗dt)
- const double ∗ process (const double ∗sig, const double ∗dt, double fdb)
- const Delay & process (const AudioBase &obj)
- const Delay & process (const AudioBase &obj, double dt)
- const Delay & process (const AudioBase &obj, double dt, double fdb)
- const Delay & process (const AudioBase &obj, const AudioBase &dt)
- const Delay & process (const AudioBase &obj, const AudioBase &dt, double fdb)
- const Delay & operator() (const AudioBase &a, const AudioBase &b, double c)
- const Delay & operator() (const AudioBase &a, const AudioBase &b)
- const Delay & operator() (const AudioBase &a, const double b, double c)
- const Delay & operator() (const AudioBase &a, double b)
- const Delay & operator() (const AudioBase &a)
- uint32_t pos () const
- const AudioBase & delayline () const

Protected Attributes

- double m_fdb
- uint32_t m_pos
- AudioBase m_delay

Additional Inherited Members

7.11.1 Detailed Description

Fixed or variable delay line with optional feedback (Delay, Comb filter, Flanger)

7.11.2 Constructor & Destructor Documentation

7.11.2.1 Delay()

```
AuLib::Delay::Delay (
             double dtime,
             double fdb,
             uint32_t vframes = def_vframes,
             double sr = def_sr )  [inline]
```

Delay constructor

dtime - delay time
vframes - vector size
sr - sampling rate

7.11.3 Member Function Documentation

7.11.3.1 delayline()

```
const AudioBase& AuLib::Delay::delayline ( ) const  [inline]
```

get a reference to the delay line.

7.11.3.2 operator()() [1/5]

```
const Delay& AuLib::Delay::operator() (
             const AudioBase & a,
             const AudioBase & b,
             double c )  [inline]
```

operator(a,b,c) convenience method

7.11.3.3 operator()() [2/5]

```
const Delay& AuLib::Delay::operator() (
             const AudioBase & a,
             const AudioBase & b )  [inline]
```

operator(a,b) convenience method

7.11.3.4 operator()() [3/5]

```
const Delay& AuLib::Delay::operator() (
             const AudioBase & a,
             const double b,
             double c )  [inline]
```

operator(a,b,c) convenience method

7.11.3.5 operator()() [4/5]

```
const Delay& AuLib::Delay::operator() (
             const AudioBase & a,
             double b )  [inline]
```

operator(a,b) convenience method

7.11.3.6 operator()() [5/5]

```
const Delay& AuLib::Delay::operator() (
             const AudioBase & a )  [inline]
```

operator(a) convenience method

7.11.3.7 pos()

```
uint32_t AuLib::Delay::pos ( ) const  [inline]
```

get the current write position

7.11.3.8 process() [1/10]

```
const double* AuLib::Delay::process (
             const double * sig )  [inline]
```

delay a signal sig for a fixed time

7.11.3.9 process() [2/10]

```
const double* AuLib::Delay::process (
             const double * sig,
             double dt )  [inline]
```

delay a signal for dt seconds

7.11.3.10 process() [3/10]

```
const double* AuLib::Delay::process (
             const double * sig,
             double dt,
             double fdb )  [inline]
```

delay a signal for dt seconds and with optional feedback fdb

7.11.3.11 process() [4/10]

```
const double* AuLib::Delay::process (
             const double * sig,
             const double * dt )  [inline]
```

delay a signal for delay time taken from the signal dt

7.11.3.12 process() [5/10]

```
const double* AuLib::Delay::process (
             const double * sig,
             const double * dt,
             double fdb )  [inline]
```

delay a signal for delay time taken from the signal dt and with optional feedback fdb

7.11.3.13 process() [6/10]

```
const Delay& AuLib::Delay::process (
             const AudioBase & obj )  [inline]
```

delay a signal in obj for a fixed time

7.11.3.14 process() [7/10]

```
const Delay& AuLib::Delay::process (
             const AudioBase & obj,
             double dt )  [inline]
```

delay a signal in obj, optionally for dt seconds.

7.11.3.15 process() [8/10]

```
const Delay& AuLib::Delay::process (
             const AudioBase & obj,
             double dt,
             double fdb )  [inline]
```

delay a signal in obj, optionally for dt seconds and with feedback fdb.

7.11.3.16 process() [9/10]

```
const Delay& AuLib::Delay::process (
             const AudioBase & obj,
             const AudioBase & dt )  [inline]
```

delay a signal in obj for dt sec with variable delaytime sig

7.11.3.17 process() [10/10]

```
const Delay& AuLib::Delay::process (
             const AudioBase & obj,
             const AudioBase & dt,
             double fdb )  [inline]
```

delay a signal in obj for dt sec with variable delaytime sig and with optional feedback fdb.

7.11.4 Member Data Documentation

7.11.4.1 m_delay

```
AudioBase AuLib::Delay::m_delay  [protected]
```

7.11.4.2 m_fdb

```
double AuLib::Delay::m_fdb  [protected]
```

7.11.4.3 m_pos

```
uint32_t AuLib::Delay::m_pos  [protected]
```

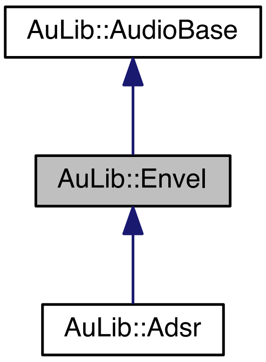
The documentation for this class was generated from the following files:

- Delay.h
- Delay.cpp

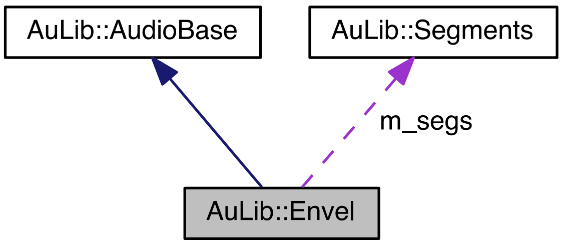
7.12 AuLib::Envel Class Reference

```
#include <AuLib/Envel.h>
```

Inheritance diagram for AuLib::Envel:



Collaboration diagram for AuLib::Envel:



Public Member Functions

- Envel (const Segments &segs, double rel=0.f, uint32_t vframes=def_vframes, double sr=def_sr)
- Envel (double rel=0.f, uint32_t vframes=def_vframes, double sr=def_sr)
- const Envel & process ()
- const Envel & operator() ()
- virtual void retrig ()
- void release ()
- void reset (const Segments &segs, double rel=0.f)
- uint32_t rframes () const
- uint32_t frames () const
- bool is_finished () const

Protected Attributes

- double m_y
- int32_t m_cseg
- uint32_t m_rt
- uint32_t m_cnt
- uint32_t m_time
- double m_incr
- bool m_trig
- bool m_releasing
- bool m_done
- int32_t m_rcnt
- Segments m_segs

Additional Inherited Members

7.12.1 Detailed Description

Multi-segment envelope generator

7.12.2 Constructor & Destructor Documentation

7.12.2.1 Envel() [1/2]

```
AuLib::Envel::Envel (
            const Segments & segs,
            double rel = 0.f,
            uint32_t vframes = def_vframes,
            double sr = def_sr )  [inline]
```

Envel constructor

segs - envelope segments
rel - release time
vframes - vector size
sr - sampling rate

7.12.2.2 Envel() [2/2]

```
AuLib::Envel::Envel (
            double rel = 0.f,
            uint32_t vframes = def_vframes,
            double sr = def_sr )  [inline]
```

Envel constructor

rel - release time
vframes - vector size
sr - sampling rate

7.12.3 Member Function Documentation

7.12.3.1 frames()

```
uint32_t AuLib::Envel::frames ( ) const  [inline]
```

return the env duration in frames (excluding release)

7.12.3.2 is_finished()

```
bool AuLib::Envel::is_finished ( ) const  [inline]
```

return the envelope status

7.12.3.3 operator()()

```
const Envel& AuLib::Envel::operator() ( )  [inline]
```

operator() convenience method, same as process()

7.12.3.4 process()

```
const Envel& AuLib::Envel::process ( )  [inline]
```

process envelope

7.12.3.5 release()

```
void AuLib::Envel::release ( )  [inline]
```

trigger release

7.12.3.6 reset()

```
void AuLib::Envel::reset (
             const Segments & segs,
             double rel = 0.f )  [inline]
```

reset parameters and retrigger

7.12.3.7 retrig()

```
virtual void AuLib::Envel::retrig ( )  [inline], [virtual]
```

retrigger envelope

7.12.3.8 rframes()

```
uint32_t AuLib::Envel::rframes ( ) const  [inline]
```

return the release duration in frames

7.12.4 Member Data Documentation

7.12.4.1 m_cnt

```
uint32_t AuLib::Envel::m_cnt  [protected]
```

7.12.4.2 m_cseg

```
int32_t AuLib::Envel::m_cseg  [protected]
```

7.12.4.3 m_done

```
bool AuLib::Envel::m_done  [protected]
```

7.12.4.4 m_incr

```
double AuLib::Envel::m_incr  [protected]
```

7.12.4.5 m_rcnt

```
int32_t AuLib::Envel::m_rcnt  [protected]
```

7.12.4.6 m_releasing

```
bool AuLib::Envel::m_releasing  [protected]
```

7.12.4.7 m_rt

```
uint32_t AuLib::Envel::m_rt  [protected]
```

7.12.4.8 m_segs

```
Segments AuLib::Envel::m_segs  [protected]
```

7.12.4.9 m_time

```
uint32_t AuLib::Envel::m_time  [protected]
```

7.12.4.10 m_trig

```
bool AuLib::Envel::m_trig  [protected]
```

7.12.4.11 m_y

```
double AuLib::Envel::m_y  [protected]
```

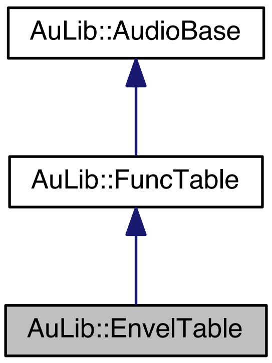
The documentation for this class was generated from the following files:

- Envel.h
- Envel.cpp

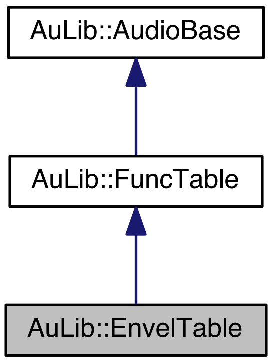
7.13 AuLib::EnvelTable Class Reference

```
#include <EnvelTable.h>
```

Inheritance diagram for AuLib::EnvelTable:



Collaboration diagram for AuLib::EnvelTable:



Public Member Functions

- EnvelTable (const Segments &segs, bool norm=true)

Additional Inherited Members

7.13.1 Detailed Description

Multi-segment Envelope Tables

7.13.2 Constructor & Destructor Documentation

7.13.2.1 EnvelTable()

```
AuLib::EnvelTable::EnvelTable (
             const Segments & segs,
             bool norm = true )
```

EnvelTable constructor

segs - envelope segments norm - normalise table

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

- EnvelTable.h
- EnvelTable.cpp

7.14 AuLib::Event Struct Reference

```
#include <Score.h>
```

Public Attributes

- uint32_t chn
- uint32_t msg
- std::vector< double > data
- double time

7.14.1 Detailed Description

single score event

7.14.2 Member Data Documentation

7.14.2.1 chn

```
uint32_t AuLib::Event::chn
```

7.14.2.2 data

```
std::vector<double> AuLib::Event::data
```

7.14.2.3 msg

```
uint32_t AuLib::Event::msg
```

7.14.2.4 time

```
double AuLib::Event::time
```

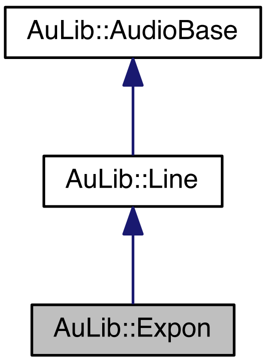
The documentation for this struct was generated from the following file:

- Score.h

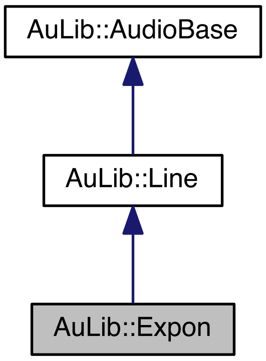
7.15 AuLib::Expon Class Reference

```
#include <AuLib/Expon.h>
```

Inheritance diagram for AuLib::Expon:



Collaboration diagram for AuLib::Expon:



Public Member Functions

- Expon (double start=db_min, double end=1., double time=1., uint32_t vframes=def_vframes, double sr=def↩ _sr)

Protected Member Functions

- virtual void dsp ()
- virtual void restart ()

Additional Inherited Members

7.15.1 Detailed Description

Generates a signal based on an exponential curve between two points over a given duration.

7.15.2 Constructor & Destructor Documentation

7.15.2.1 Expon()

```
AuLib::Expon::Expon (
             double start = db_min,
             double end = 1.,
             double time = 1.,
             uint32_t vframes = def_vframes,
             double sr = def_sr )  [inline]
```

Expon constructor

start - start value
end - end value
time - duration(s)
vframes - vector size
sr - sampling rate

7.15.3 Member Function Documentation

7.15.3.1 dsp()

```
virtual void AuLib::Expon::dsp ( )  [inline], [protected], [virtual]
```

process the output vector

Reimplemented from AuLib::Line.

7.15.3.2 restart()

```
virtual void AuLib::Expon::restart ( )  [inline], [protected], [virtual]
```

Reimplemented from AuLib::Line.

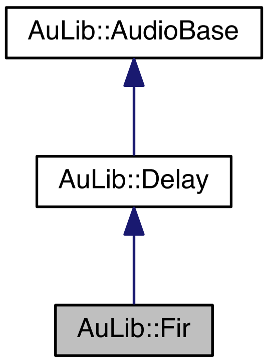
The documentation for this class was generated from the following file:

- Expon.h

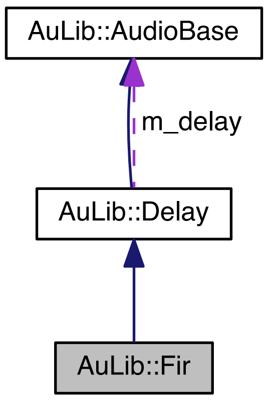
7.16 AuLib::Fir Class Reference

```
#include <AuLib/Fir.h>
```

Inheritance diagram for AuLib::Fir:



Collaboration diagram for AuLib::Fir:



Public Member Functions

- Fir (const FuncTable &ir, uint32_t chn=0, uint32_t len=0, uint32_t vframes=def_vframes, double sr=def_sr)

Protected Attributes

- const double ∗ m_ir
- uint32_t m_ir_nchnls
- uint32_t m_chn

Additional Inherited Members

7.16.1 Detailed Description

This class implements a direct convolution engine using an impulse response defined in a function table.

7.16.2 Constructor & Destructor Documentation

7.16.2.1 Fir()

```
AuLib::Fir::Fir (
            const FuncTable & ir,
            uint32_t chn = 0,
            uint32_t len = 0,
            uint32_t vframes = def_vframes,
            double sr = def_sr )  [inline]
```

Fir constructor

ir - impulse response len - if $>$ 0, set the FIR length vframes - vector size
sr - sampling rate

7.16.3 Member Data Documentation

7.16.3.1 m_chn

```
uint32_t AuLib::Fir::m_chn  [protected]
```

7.16.3.2 m_ir

```
const double∗ AuLib::Fir::m_ir  [protected]
```

7.16.3.3 m_ir_nchnls

```
uint32_t AuLib::Fir::m_ir_nchnls  [protected]
```

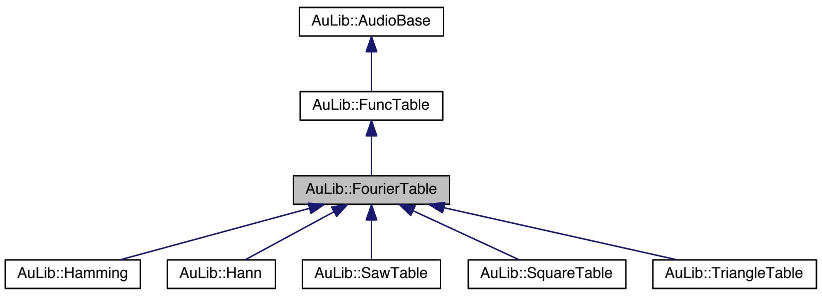
The documentation for this class was generated from the following files:

- Fir.h
- Delay.cpp

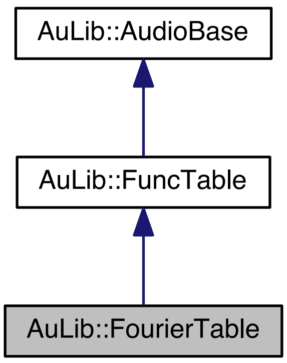
7.17 AuLib::FourierTable Class Reference

```
#include <AuLib/FourierTable.h>
```

Inheritance diagram for AuLib::FourierTable:



Collaboration diagram for AuLib::FourierTable:



Public Member Functions

- FourierTable (uint32_t harms=1, double ∗amps=nullptr, double phase=0., uint64_t tframes=def_tframes)
- FourierTable (uint32_t harms, uint32_t type, uint64_t tframes=def_tframes)

Protected Member Functions

- void create (uint32_t harms, double ∗amps, double phase)

Additional Inherited Members

7.17.1 Detailed Description

Function tables based on Fourier Series

7.17.2 Constructor & Destructor Documentation

7.17.2.1 FourierTable() [1/2]

```
AuLib::FourierTable::FourierTable (
            uint32_t harms = 1,
            double * amps = nullptr,
            double phase = 0.,
            uint64_t tframes = def_tframes )
```

FourierTable constructor

harms - number of harmonics
amps - array of harmonic amplitudes
phase - phase offset
tframes - table size

7.17.2.2 FourierTable() [2/2]

```
AuLib::FourierTable::FourierTable (
            uint32_t harms,
            uint32_t type,
            uint64_t tframes = def_tframes )
```

FourierTable constructor

harms - number of harmonics
type - wave type
tframes - table size

7.17.3 Member Function Documentation

7.17.3.1 create()

```
void AuLib::FourierTable::create (
            uint32_t harms,
            double * amps,
            double phase )  [protected]
```

Create the table

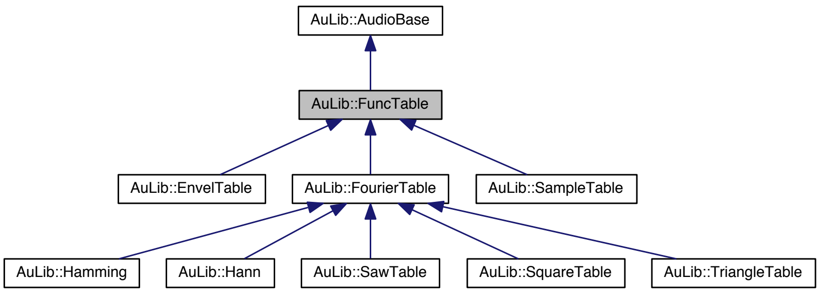
The documentation for this class was generated from the following files:

- FourierTable.h
- FourierTable.cpp

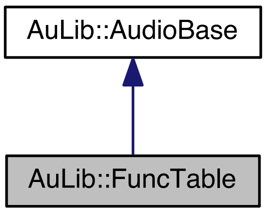
7.18 AuLib::FuncTable Class Reference

```
#include <AuLib/FuncTable.h>
```

Inheritance diagram for AuLib::FuncTable:



Collaboration diagram for AuLib::FuncTable:



Public Member Functions

- FuncTable (uint32_t tframes=def_tframes, uint32_t nchnls=def_nchnls, uint32_t sr=def_sr)
- FuncTable (const double ∗src, bool norm=false, uint32_t tframes=def_tframes, uint32_t nchnls=def_nchnls, uint32_t sr=def_sr)
- uint64_t tframes () const
- void rescale (double max)

Static Public Attributes

- static constexpr bool normalised = true

Protected Member Functions

- void normalise_table (double s=1.0)

Protected Attributes

- uint64_t m_tframes

Additional Inherited Members

7.18.1 Detailed Description

Function table base class

7.18.2 Constructor & Destructor Documentation

7.18.2.1 FuncTable() [1/2]

```
AuLib::FuncTable::FuncTable (
            uint32_t tframes = def_tframes,
            uint32_t nchnls = def_nchnls,
            uint32_t sr = def_sr )  [inline]
```

FuncTable constructor

tframes - table size
nchnls - number of channels
sr - sampling rate

7.18.2.2 FuncTable() [2/2]

```
AuLib::FuncTable::FuncTable (
            const double * src,
            bool norm = false,
            uint32_t tframes = def_tframes,
            uint32_t nchnls = def_nchnls,
            uint32_t sr = def_sr )  [inline]
```

FuncTable constructor from vector

src - source vector
norm - normalise table tframes - table size
nchnls - number of channels
sr - sampling rate

7.18.3 Member Function Documentation

7.18.3.1 normalise_table()

```
void AuLib::FuncTable::normalise_table (
            double s = 1.0 )  [inline], [protected]
```

Normalise the table

7.18.3.2 rescale()

```
void AuLib::FuncTable::rescale (
            double max )  [inline]
```

rescale the table for an absolute max

7.18.3.3 tframes()

```
uint64_t AuLib::FuncTable::tframes ( ) const  [inline]
```

return the nominal table size, ie. the total vectorsize minus the two guard points.

7.18.4 Member Data Documentation

7.18.4.1 m_tframes

```
uint64_t AuLib::FuncTable::m_tframes  [protected]
```

7.18.4.2 normalised

```
constexpr bool AuLib::FuncTable::normalised = true  [static]
```

define normalisation (rescaling)

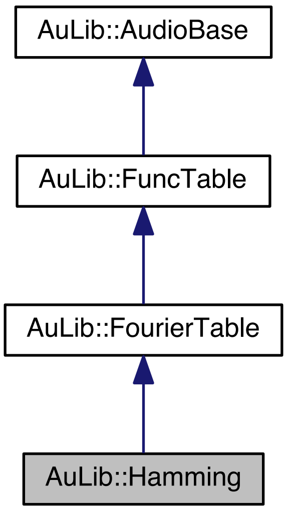
The documentation for this class was generated from the following file:

- FuncTable.h

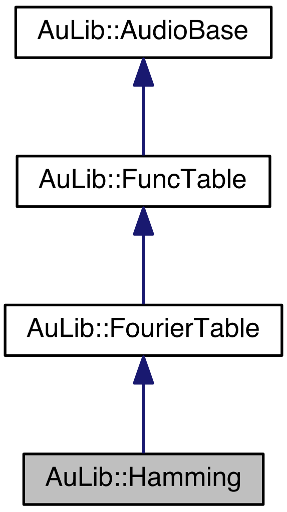
7.19 AuLib::Hamming Struct Reference

```
#include <Wintabs.h>
```

Inheritance diagram for AuLib::Hamming:



Collaboration diagram for AuLib::Hamming:



Public Member Functions

- Hamming (uint64_t tframes=def_fftsize)

Additional Inherited Members

7.19.1 Detailed Description

Hamming window

7.19.2 Constructor & Destructor Documentation

7.19.2.1 Hamming()

```
AuLib::Hamming::Hamming (
             uint64_t tframes = def_fftsize )  [inline]
```

Hamming constructor

tframes - table size

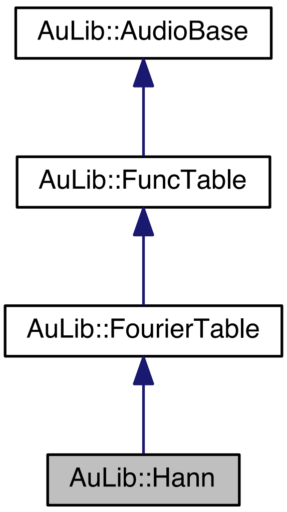
The documentation for this struct was generated from the following file:

- Wintabs.h

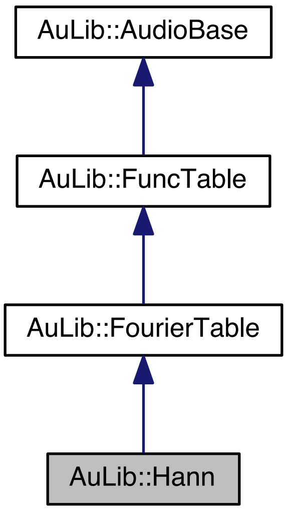
7.20 AuLib::Hann Struct Reference

```
#include <Wintabs.h>
```

Inheritance diagram for AuLib::Hann:



Collaboration diagram for AuLib::Hann:



Public Member Functions

- Hann (uint64_t tframes=def_fftsize)

Additional Inherited Members

7.20.1 Detailed Description

von Hann window

7.20.2 Constructor & Destructor Documentation

7.20.2.1 Hann()

```
AuLib::Hann::Hann (
             uint64_t tframes = def_fftsize )  [inline]
```

Hann constructor

tframes - table size

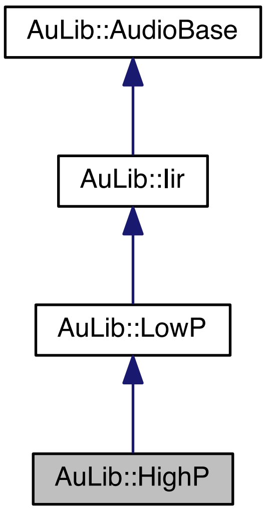
The documentation for this struct was generated from the following file:

- Wintabs.h

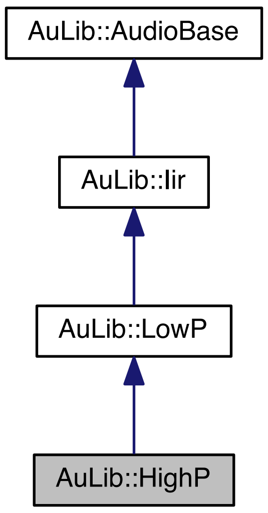
7.21 AuLib::HighP Class Reference

```
#include <AuLib/HighP.h>
```

Inheritance diagram for AuLib::HighP:



Collaboration diagram for AuLib::HighP:



Public Member Functions

- HighP (double cf, uint32_t vframes=def_vframes, double sr=def_sr)

Protected Member Functions

- virtual void update ()

Additional Inherited Members

7.21.1 Detailed Description

2nd-order Butterworth high-pass filter

7.21.2 Constructor & Destructor Documentation

7.21.2.1 HighP()

```
AuLib::HighP::HighP (
             double cf,
             uint32_t vframes = def_vframes,
             double sr = def_sr )  [inline]
```

Highp constructor

cf - cutoff frequency
vframes - vector size
sr - sampling rate

7.21.3 Member Function Documentation

7.21.3.1 update()

```
void AuLib::HighP::update ( )  [protected], [virtual]
```

Coefficients update

Reimplemented from AuLib::LowP.

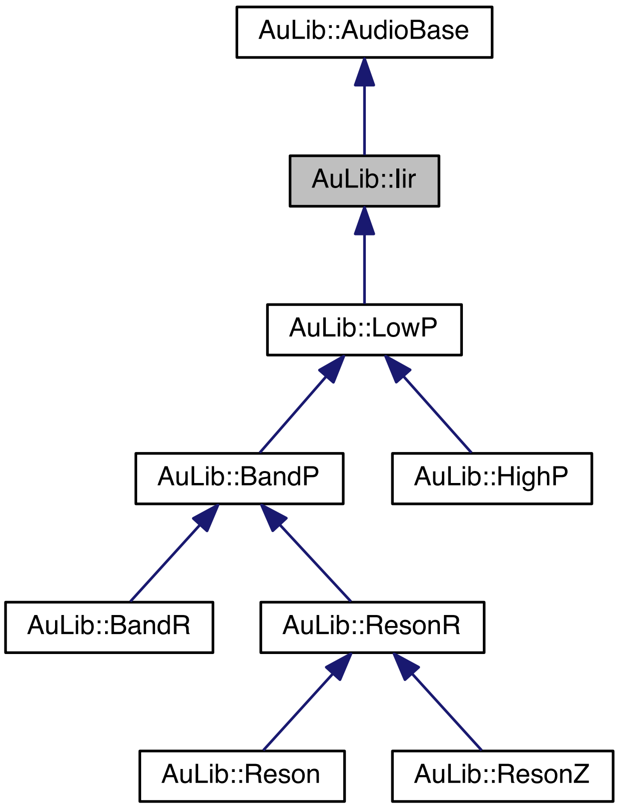
The documentation for this class was generated from the following files:

- HighP.h
- Iir.cpp

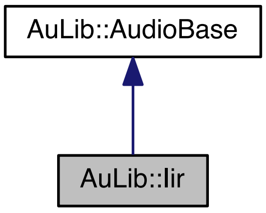
7.22 AuLib::Iir Class Reference

```
#include <AuLib/Iir.h>
```

Inheritance diagram for AuLib::Iir:



Collaboration diagram for AuLib::Iir:



Public Member Functions

- Iir (const double ∗a, const double ∗b, uint32_t vframes=def_vframes, double sr=def_sr)

- Iir (uint32_t vframes=def_vframes, double sr=def_sr)
- const double ∗ process (const double ∗sig)
- const double ∗ process (const double ∗sig, const double ∗a, const double ∗b)
- const Iir & process (const AudioBase &obj)
- const Iir & process (const AudioBase &obj, const double ∗c, bool iir=true)
- const Iir & process (const AudioBase &obj, const double ∗a, const double ∗b)
- const Iir & operator() (const AudioBase &obj)
- template<typename T >
  const Iir & operator() (const AudioBase &obj, T a)
- template<typename T , typename U >
  const Iir & operator() (const AudioBase &obj, T a, U b)

Protected Member Functions

- virtual const double ∗ dsp (const double ∗sig)
- virtual void update ()

Protected Attributes

- double m_del [2]
- double m_a [3]
- double m_b [2]
- double m_scal

Additional Inherited Members

7.22.1 Detailed Description

General-purpose 2nd-order IIR filter section

7.22.2 Constructor & Destructor Documentation

7.22.2.1 Iir() [1/2]

```
AuLib::Iir::Iir (
            const double * a,
            const double * b,
            uint32_t vframes = def_vframes,
            double sr = def_sr )  [inline]
```

Iir constructor

a - feedforward coef list (a0,a1,a2) b - feedback coefs (b1, b2) vframes - vector size
sr - sampling rate

7.22.2.2 Iir() [2/2]

```
AuLib::Iir::Iir (
            uint32_t vframes = def_vframes,
            double sr = def_sr )  [inline]
```

Iir constructor

vframes - vector size
sr - sampling rate

7.22.3 Member Function Documentation

7.22.3.1 dsp()

```
const double * AuLib::Iir::dsp (
            const double * sig )  [protected], [virtual]
```

Filter kernel

7.22.3.2 operator()() [1/3]

```
const Iir& AuLib::Iir::operator() (
            const AudioBase & obj )  [inline]
```

operator(a) convenience

7.22.3.3 operator()() [2/3]

```
template<typename T >
const Iir& AuLib::Iir::operator() (
            const AudioBase & obj,
            T a )  [inline]
```

operator(a, b) convenience

7.22.3.4 operator()() [3/3]

```
template<typename T , typename U >
const Iir& AuLib::Iir::operator() (
            const AudioBase & obj,
            T a,
            U b )  [inline]
```

operator(a, b, c) convenience

7.22.3.5 process() [1/5]

```
const double* AuLib::Iir::process (
             const double * sig )  [inline]
```

process a signal sig

7.22.3.6 process() [2/5]

```
const double* AuLib::Iir::process (
             const double * sig,
             const double * a,
             const double * b )  [inline]
```

process a signal sig with coefficients lists a and b

7.22.3.7 process() [3/5]

```
const Iir& AuLib::Iir::process (
             const AudioBase & obj )  [inline]
```

process a signal in obj

7.22.3.8 process() [4/5]

```
const Iir& AuLib::Iir::process (
             const AudioBase & obj,
             const double * c,
             bool iir = true )  [inline]
```

process a signal in obj with coefficient lists c as an iir or fir

7.22.3.9 process() [5/5]

```
const Iir& AuLib::Iir::process (
             const AudioBase & obj,
             const double * a,
             const double * b )  [inline]
```

process a signal in obj with coefficients lists a and b

7.22.3.10 update()

```
virtual void AuLib::Iir::update ( )  [inline], [protected], [virtual]
```

Coefficients update

Reimplemented in AuLib::BandP, AuLib::LowP, AuLib::ResonR, AuLib::ResonZ, AuLib::BandR, and AuLib::HighP.

7.22.4 Member Data Documentation

7.22.4.1 m_a

```
double AuLib::Iir::m_a[3]  [protected]
```

7.22.4.2 m_b

```
double AuLib::Iir::m_b[2]  [protected]
```

7.22.4.3 m_del

```
double AuLib::Iir::m_del[2]  [protected]
```

7.22.4.4 m_scal

```
double AuLib::Iir::m_scal  [protected]
```

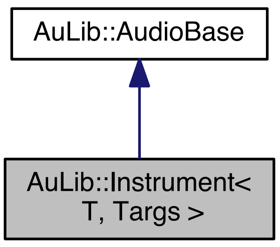
The documentation for this class was generated from the following files:

- Iir.h
- Iir.cpp

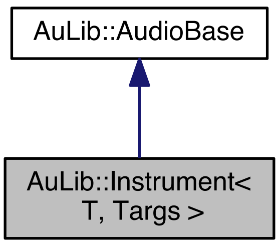
7.23 AuLib::Instrument< T, Targs > Class Template Reference

```
#include <AuLib/Instrument.h>
```

Inheritance diagram for AuLib::Instrument< T, Targs >:



Collaboration diagram for AuLib::Instrument< T, Targs >:



Public Member Functions

- Instrument (uint32_t nvoices, int32_t chn, Targs... args)
- void dispatch (uint32_t msg, uint32_t chn, double data1, double data2, uint64_t stamp)
- void dispatch (uint32_t msg, uint32_t chn, const std::vector< double > &data, uint64_t stamp)
- virtual const Instrument & process ()

Additional Inherited Members

7.23.1 Detailed Description

template<typename T, typename... Targs>
class AuLib::Instrument< T, Targs >

Instrument template class:
Creates an instrument with n voices when instantiated with a Note-derived class with optional extra parameters.

7.23.2 Constructor & Destructor Documentation

7.23.2.1 Instrument()

```
template<typename T , typename...  Targs>
AuLib::Instrument< T, Targs >::Instrument (
            uint32_t nvoices,
            int32_t chn,
            Targs...  args )  [inline]
```

Construct an instrument with nvoices polyphony

7.23.3 Member Function Documentation

7.23.3.1 dispatch() [1/2]

```
template<typename T , typename...  Targs>
void AuLib::Instrument< T, Targs >::dispatch (
            uint32_t msg,
            uint32_t chn,
            double data1,
            double data2,
            uint64_t stamp )  [inline]
```

Dispatch a channel message using a MIDI-like format.
msg - message type
chn - channel
data1 - message data 1
data2 - message data 2
stamp - time stamp

Note that message types are not constrained to MIDI ones; the Note class can be specialised to handle any pre-defined messages.

7.23.3.2 dispatch() [2/2]

```
template<typename T , typename...  Targs>
void AuLib::Instrument< T, Targs >::dispatch (
            uint32_t msg,
            uint32_t chn,
            const std::vector< double > & data,
            uint64_t stamp )  [inline]
```

Dispatch a channel message of unspecified length
msg - type
chn - chn
data - message data
tstamp - time stamp

7.23.3.3 process()

```
template<typename T , typename...  Targs>
virtual const Instrument& AuLib::Instrument< T, Targs >::process ( )  [inline], [virtual]
```

processing interface

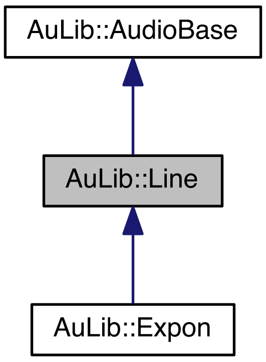
The documentation for this class was generated from the following file:

- Instrument.h

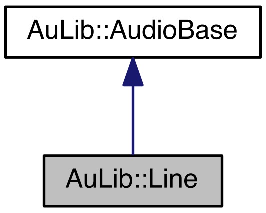
7.24 AuLib::Line Class Reference

```
#include <AuLib/Line.h>
```

Inheritance diagram for AuLib::Line:



Collaboration diagram for AuLib::Line:



Public Member Functions

- Line (double start=.0, double end=1., double time=1., uint32_t vframes=def_vframes, double sr=def_sr)
- const Line & process ()
- const Line & operator() ()
- void retrig ()
- void reset (double start, double end, double time)

Protected Member Functions

- virtual void dsp ()
- virtual void restart ()

Protected Attributes

- double m_y
- double m_y0
- double m_y1
- uint32_t m_x1
- double m_incr
- uint64_t m_cnt

Additional Inherited Members

7.24.1 Detailed Description

Generates a signal based on a linear curve between two points over a given duration.

7.24.2 Constructor & Destructor Documentation

7.24.2.1 Line()

```
AuLib::Line::Line (
             double start = .0,
             double end = 1.,
             double time = 1.,
             uint32_t vframes = def_vframes,
             double sr = def_sr )  [inline]
```

Line constructor

start - start value
end - end value
time - duration(s)
vframes - vector size
sr - sampling rate

7.24.3 Member Function Documentation

7.24.3.1 dsp()

```
virtual void AuLib::Line::dsp ( )  [inline], [protected], [virtual]
```

process the output vector

Reimplemented in AuLib::Expon.

7.24.3.2 operator()()

```
const Line& AuLib::Line::operator() ( )  [inline]
```

operator () convenience method

7.24.3.3 process()

```
const Line& AuLib::Line::process ( )  [inline]
```

process and return a reference to the object

7.24.3.4 reset()

```
void AuLib::Line::reset (
             double start,
             double end,
             double time )  [inline]
```

reset and retrigger

7.24.3.5 restart()

```
virtual void AuLib::Line::restart ( )  [inline], [protected], [virtual]
```

Reimplemented in AuLib::Expon.

7.24.3.6 retrig()

```
void AuLib::Line::retrig ( )  [inline]
```

retrigger

7.24.4 Member Data Documentation

7.24.4.1 m_cnt

```
uint64_t AuLib::Line::m_cnt  [protected]
```

7.24.4.2 m_incr

```
double AuLib::Line::m_incr  [protected]
```

7.24.4.3 m_x1

```
uint32_t AuLib::Line::m_x1  [protected]
```

7.24.4.4 m_y

```
double AuLib::Line::m_y  [protected]
```

7.24.4.5 m_y0

```
double AuLib::Line::m_y0  [protected]
```

7.24.4.6 m_y1

```
double AuLib::Line::m_y1  [protected]
```

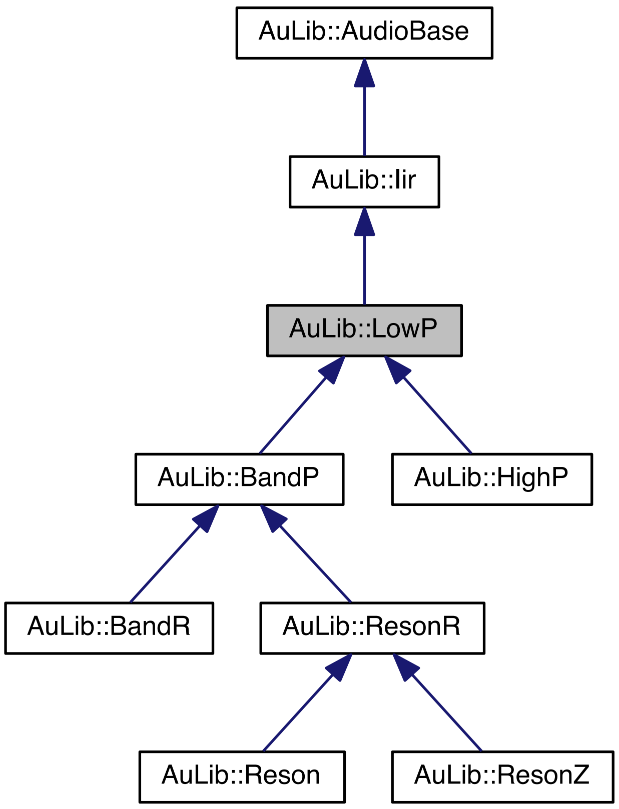
The documentation for this class was generated from the following file:

- Line.h

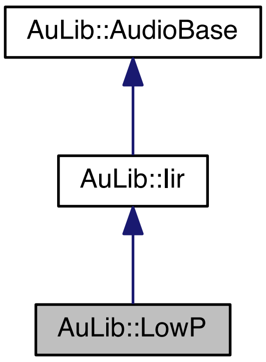
7.25 AuLib::LowP Class Reference

```
#include <AuLib/LowP.h>
```

Inheritance diagram for AuLib::LowP:



Collaboration diagram for AuLib::LowP:



Public Member Functions

- LowP (double cf, uint32_t vframes=def_vframes, double sr=def_sr)
- const double ∗ process (const double ∗sig, double cf)
- const LowP & process (const AudioBase &obj, double cf)
- const LowP & operator() (const AudioBase &obj, double cf)

Protected Member Functions

- virtual void update ()

Protected Attributes

- double m_freq

Additional Inherited Members

7.25.1 Detailed Description

2nd-order Butterworth low-pass filter

7.25.2 Constructor & Destructor Documentation

7.25.2.1 LowP()

```
AuLib::LowP::LowP (
            double cf,
            uint32_t vframes = def_vframes,
            double sr = def_sr )  [inline]
```

LowP constructor

cf - cutoff freq
vframes - vector size
sr - sampling rate

7.25.3 Member Function Documentation

7.25.3.1 operator()()

```
const LowP& AuLib::LowP::operator() (
            const AudioBase & obj,
            double cf )  [inline]
```

7.25.3.2 process() [1/2]

```
const double* AuLib::LowP::process (
            const double * sig,
            double cf )  [inline]
```

process a signal sig with cutoff freq cf

7.25.3.3 process() [2/2]

```
const LowP& AuLib::LowP::process (
            const AudioBase & obj,
            double cf )  [inline]
```

process a signal in obj with cutoff freq cf

7.25.3.4 update()

```
void AuLib::LowP::update ( )  [protected], [virtual]
```

Coefficients update

Reimplemented from AuLib::Iir.

Reimplemented in AuLib::BandP, AuLib::ResonR, AuLib::ResonZ, AuLib::BandR, and AuLib::HighP.

7.25.4 Member Data Documentation

7.25.4.1 m_freq

```
double AuLib::LowP::m_freq  [protected]
```

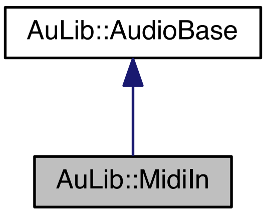
The documentation for this class was generated from the following files:

- LowP.h
- Iir.cpp

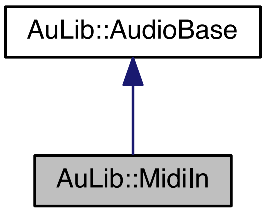
7.26 AuLib::MidiIn Class Reference

```
#include <MidiIn.h>
```

Inheritance diagram for AuLib::MidiIn:



Collaboration diagram for AuLib::MidiIn:



Public Member Functions

- MidiIn ()
- ∼MidiIn ()
- uint32_t open (int dev)
- void close ()
- template<typename... Targs>
  const MidiIn & listen (Targs &... args)
- double ctlval (int32_t chn, uint32_t num)
- double aftval (int32_t chn, uint32_t num)
- double aftval (int32_t chn)
- uint32_t program (int32_t chn)
- uint32_t read ()
- const std::vector< std::string > & device_list ()

Additional Inherited Members

7.26.1 Detailed Description

This class implements a MIDI input listener

7.26.2 Constructor & Destructor Documentation

7.26.2.1 MidiIn()

```
AuLib::MidiIn::MidiIn ( )
```

Constructs an empty object

7.26.2.2 ∼MidiIn()

```
AuLib::MidiIn::∼MidiIn ( )
```

Destructor

7.26.3 Member Function Documentation

7.26.3.1 aftval() [1/2]

```
double AuLib::MidiIn::aftval (
             int32_t chn,
             uint32_t num ) [inline]
```

returns a MIDI aftertouch value for channel chn and note num, chn < 0 means all channels.
Results in the range [0.,1.)

7.26.3.2 aftval() [2/2]

```
double AuLib::MidiIn::aftval (
             int32_t chn )  [inline]
```

returns a MIDI aftertouch value for channel chn, chn $<$ 0 means all channels.
Results in the range [0.,1.)

7.26.3.3 close()

```
void AuLib::MidiIn::close ( )
```

Close MIDI connection

7.26.3.4 ctlval()

```
double AuLib::MidiIn::ctlval (
             int32_t chn,
             uint32_t num )  [inline]
```

returns a MIDI control value for channel chn and control num, chn $<$ 0 means all channels.
Results in the range [0.,1.)

7.26.3.5 device_list()

```
const std::vector<std::string>& AuLib::MidiIn::device_list ( )  [inline]
```

Returns a list of devices names/numbers as strings

7.26.3.6 listen()

```
template<typename...  Targs>
const MidiIn& AuLib::MidiIn::listen (
             Targs &...  args )  [inline]
```

Implements a listener for an obj inst dispatching any received data. Classes implementing the dispatch & process methods are valid types for arguments here.

7.26.3.7 open()

```
uint32_t AuLib::MidiIn::open (
             int dev )
```

Open MIDI input device dev

7.26.3.8 program()

```
uint32_t AuLib::MidiIn::program (
            int32_t chn ) [inline]
```

returns a MIDI program value for channel chn, chn $<$ 0 means all channels.

7.26.3.9 read()

```
uint32_t AuLib::MidiIn::read ( )
```

read MIDI input, returning number of MIDI events read.
Only needs to be called if listen() is not used

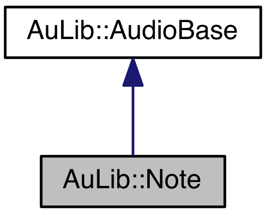
The documentation for this class was generated from the following files:

- MidiIn.h
- MidiIn.cpp

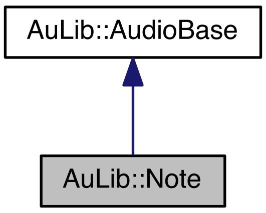
7.27 AuLib::Note Class Reference

```
#include <AuLib/Note.h>
```

Inheritance diagram for AuLib::Note:



Collaboration diagram for AuLib::Note:



Public Member Functions

- Note (int32_t chn=-1)
- const Note & process ()
- bool is_on () const
- uint64_t time_stamp () const
- bool note_on (int32_t chn, double num, double vel, uint64_t tstamp)
- bool note_off (int32_t chn, double num, double vel)
- bool note_off ()
- void ctrl_msg (int32_t chn, uint32_t msg, const std::vector< double > &data, uint64_t tstamp)
- void set_chn (uint32_t chn)

Protected Member Functions

- void clear ()

Protected Attributes

- uint64_t m_tstamp
- double m_num
- double m_vel
- int32_t m_chn
- bool m_on
- double m_cps
- double m_amp

Additional Inherited Members

7.27.1 Detailed Description

Note class:
Base class for modelling synthesiser notes

7.27.2 Constructor & Destructor Documentation

7.27.2.1 Note()

```
AuLib::Note::Note (
             int32_t chn = -1 )  [inline]
```

Constructs a note for a given channel (-1 means omni, channels are ignored; this is the default)

7.27.3 Member Function Documentation

7.27.3.1 clear()

```
void AuLib::Note::clear ( )  [inline], [protected]
```

7.27.3.2 ctrl_msg()

```
void AuLib::Note::ctrl_msg (
             int32_t chn,
             uint32_t msg,
             const std::vector< double > & data,
             uint64_t tstamp )  [inline]
```

called to pass a msg with a list of data items

7.27.3.3 is_on()

```
bool AuLib::Note::is_on ( ) const  [inline]
```

checks whether the note is on or off

7.27.3.4 note_off() [1/2]

```
bool AuLib::Note::note_off (
             int32_t chn,
             double num,
             double vel )
```

called to turn a note off for a matching channel chn and number num.

7.27.3.5 note_off() [2/2]

```
bool AuLib::Note::note_off ( )
```

called to turn this note off, regardless of channel or note number.

7.27.3.6 note_on()

```
bool AuLib::Note::note_on (
             int32_t chn,
             double num,
             double vel,
             uint64_t tstamp )
```

called to turn a note on

7.27.3.7 process()

```
const Note& AuLib::Note::process ( )  [inline]
```

processing method: call it to produce one vector of audio

7.27.3.8 set_chn()

```
void AuLib::Note::set_chn (
             uint32_t chn )  [inline]
```

7.27.3.9 time_stamp()

```
uint64_t AuLib::Note::time_stamp ( ) const  [inline]
```

onset time for this note

7.27.4 Member Data Documentation

7.27.4.1 m_amp

```
double AuLib::Note::m_amp  [protected]
```

7.27.4.2 m_chn

```
int32_t AuLib::Note::m_chn  [protected]
```

7.27.4.3 m_cps

```
double AuLib::Note::m_cps  [protected]
```

7.27.4.4 m_num

```
double AuLib::Note::m_num  [protected]
```

7.27.4.5 m_on

```
bool AuLib::Note::m_on [protected]
```

7.27.4.6 m_tstamp

```
uint64_t AuLib::Note::m_tstamp [protected]
```

7.27.4.7 m_vel

```
double AuLib::Note::m_vel [protected]
```

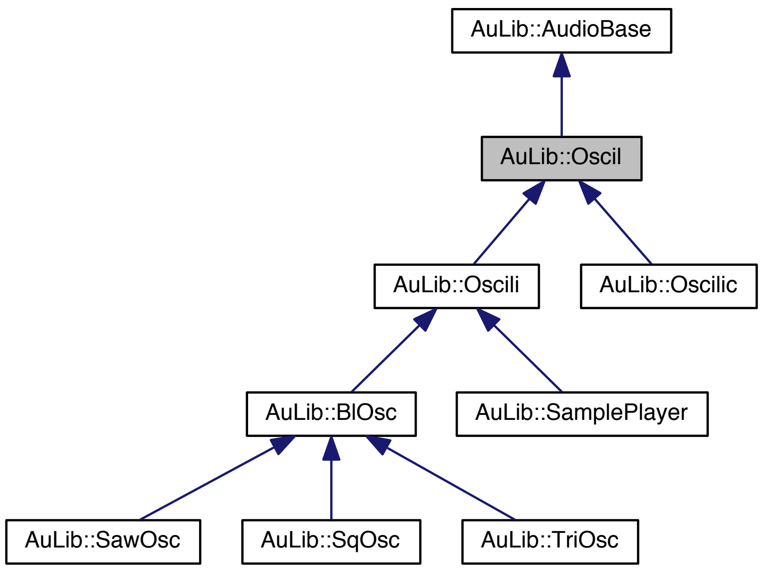
The documentation for this class was generated from the following files:

- Note.h
- Note.cpp

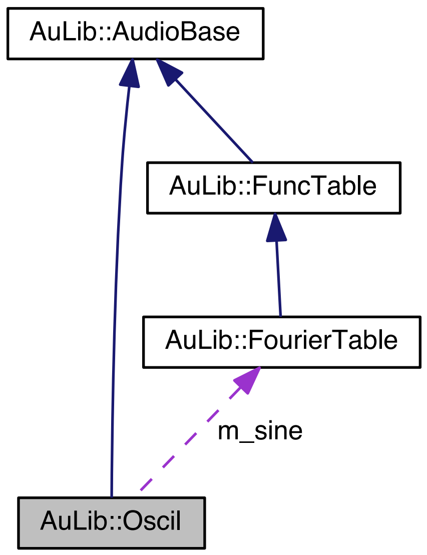
7.28 AuLib::Oscil Class Reference

```
#include <AuLib/Oscil.h>
```

Inheritance diagram for AuLib::Oscil:



Collaboration diagram for AuLib::Oscil:



Public Member Functions

- Oscil (double amp=0., double freq=0., double phase=0., uint32_t vframes=def_vframes, double sr=def_sr)
- Oscil (double amp, double freq, const FuncTable &ftable, double phase=0., uint32_t vframes=def_vframes, double sr=def_sr)
- const Oscil & process ()
- const Oscil & process (double amp)
- const Oscil & process (double amp, double freq)
- const double ∗ process (const double ∗amp, const double ∗freq)
- const double ∗ process (const double ∗amp)
- const double ∗ process (double amp, const double ∗freq)
- const double ∗ process (const double ∗amp, double freq)
- const Oscil & process (const AudioBase &obja)
- const Oscil & process (const AudioBase &obja, double freq)
- const Oscil & process (double amp, const AudioBase &objf)
- const Oscil & process (const AudioBase &obja, const AudioBase &objf)
- const Oscil & operator() (const AudioBase &a, const AudioBase &b)
- const Oscil & operator() (double a, const AudioBase &b)
- const Oscil & operator() (const AudioBase &a)
- const Oscil & operator() (const AudioBase &a, double b)
- const Oscil & operator() (double a)
- const Oscil & operator() ()

Protected Member Functions

- virtual void dsp ()
- virtual void am_fm (uint32_t ndx)
- virtual void set_incr (double f)
- void mod ()

Protected Attributes

- const double ∗ m_table
- double m_phs
- double m_amp
- double m_freq
- double m_incr
- const double ∗ m_am
- const double ∗ m_fm
- uint64_t m_tframes

Static Protected Attributes

- static const FourierTable m_sine

Additional Inherited Members

7.28.1 Detailed Description

Truncating oscillator

7.28.2 Constructor & Destructor Documentation

7.28.2.1 Oscil() [1/2]

```
AuLib::Oscil::Oscil (
             double amp = 0.,
             double freq = 0.,
             double phase = 0.,
             uint32_t vframes = def_vframes,
             double sr = def_sr )
```

Sinusoidal Oscil constructor

amp - amplitude
freq - frequency in Hz
phase - init phase (0-1)
vframes - vector size
sr - sampling rate

Uses internal sine wave table

7.28.2.2 Oscil() [2/2]

```
AuLib::Oscil::Oscil (
             double amp,
             double freq,
             const FuncTable & ftable,
             double phase = 0.,
             uint32_t vframes = def_vframes,
             double sr = def_sr )
```

Oscil constructor

amp - amplitude
freq - frequency in Hz
ftable - function table
phase - init phase (0-1)
vframes - vector size
sr - sampling rate

7.28.3 Member Function Documentation

7.28.3.1 am_fm()

```
virtual void AuLib::Oscil::am_fm (
             uint32_t ndx )  [inline], [protected], [virtual]
```

AM/FM processing

Reimplemented in AuLib::BlOsc, and AuLib::SamplePlayer.

7.28.3.2 dsp()

```
void AuLib::Oscil::dsp ( )  [protected], [virtual]
```

truncating oscillator process

Reimplemented in AuLib::SamplePlayer, AuLib::Oscili, and AuLib::Oscilic.

7.28.3.3 mod()

```
void AuLib::Oscil::mod ( )  [inline], [protected]
```

modulo function

7.28.3.4 operator()() [1/6]

```
const Oscil& AuLib::Oscil::operator() (
             const AudioBase & a,
             const AudioBase & b )  [inline]
```

operator(a,b) convenience method

7.28.3.5 operator()() [2/6]

```
const Oscil& AuLib::Oscil::operator() (
             double a,
             const AudioBase & b )  [inline]
```

operator(a,b) convenience method

7.28.3.6 operator()() [3/6]

```
const Oscil& AuLib::Oscil::operator() (
             const AudioBase & a )  [inline]
```

operator(a) convenience method

7.28.3.7 operator()() [4/6]

```
const Oscil& AuLib::Oscil::operator() (
             const AudioBase & a,
             double b )  [inline]
```

operator(a,b) convenience method

7.28.3.8 operator()() [5/6]

```
const Oscil& AuLib::Oscil::operator() (
             double a )  [inline]
```

operator(a) convenience method

7.28.3.9 operator()() [6/6]

```
const Oscil& AuLib::Oscil::operator() ( )  [inline]
```

operator() convenience method

7.28.3.10 process() [1/11]

```
const Oscil& AuLib::Oscil::process ( )  [inline]
```

Process one vector of audio

7.28.3.11 process() [2/11]

```
const Oscil& AuLib::Oscil::process (
             double amp )  [inline]
```

Process one vector of audio with amplitude amp

7.28.3.12 process() [3/11]

```
const Oscil& AuLib::Oscil::process (
             double amp,
             double freq )  [inline]
```

Process one vector of audio with amplitude amp and frequency freq

7.28.3.13 process() [4/11]

```
const double* AuLib::Oscil::process (
             const double * amp,
             const double * freq )  [inline]
```

Process one vector of audio with amplitude and freq modulation

7.28.3.14 process() [5/11]

```
const double* AuLib::Oscil::process (
             const double * amp )  [inline]
```

Process one vector of audio with amplitude modulation

7.28.3.15 process() [6/11]

```
const double* AuLib::Oscil::process (
             double amp,
             const double * freq )  [inline]
```

Process one vector of audio with amplitude amp and freq modulation

7.28.3.16 process() [7/11]

```
const double* AuLib::Oscil::process (
             const double * amp,
             double freq )  [inline]
```

Process one vector of audio with amplitude modulation and frequency freq

7.28.3.17 process() [8/11]

```
const Oscil& AuLib::Oscil::process (
             const AudioBase & obja )  [inline]
```

Process one vector of audio with amplitude modulation from obja

7.28.3.18 process() [9/11]

```
const Oscil& AuLib::Oscil::process (
             const AudioBase & obja,
             double freq )  [inline]
```

Process one vector of audio with amplitude modulation from obja and frequency freq

7.28.3.19 process() [10/11]

```
const Oscil& AuLib::Oscil::process (
             double amp,
             const AudioBase & objf )  [inline]
```

Process one vector of audio with amplitude amp and freq modulation from objf

7.28.3.20 process() [11/11]

```
const Oscil& AuLib::Oscil::process (
             const AudioBase & obja,
             const AudioBase & objf )  [inline]
```

Process one vector of audio with amplitude from obja and freq modulation from objf

7.28.3.21 set_incr()

```
virtual void AuLib::Oscil::set_incr (
             double f )  [inline], [protected], [virtual]
```

set the sampling increment

Reimplemented in AuLib::BlOsc, and AuLib::SamplePlayer.

7.28.4 Member Data Documentation

7.28.4.1 m_am

```
const double* AuLib::Oscil::m_am  [protected]
```

7.28.4.2 m_amp

`double AuLib::Oscil::m_amp [protected]`

7.28.4.3 m_fm

`const double* AuLib::Oscil::m_fm [protected]`

7.28.4.4 m_freq

`double AuLib::Oscil::m_freq [protected]`

7.28.4.5 m_incr

`double AuLib::Oscil::m_incr [protected]`

7.28.4.6 m_phs

`double AuLib::Oscil::m_phs [protected]`

7.28.4.7 m_sine

`const AuLib::FourierTable AuLib::Oscil::m_sine [static], [protected]`

7.28.4.8 m_table

`const double* AuLib::Oscil::m_table [protected]`

7.28.4.9 m_tframes

```
uint64_t AuLib::Oscil::m_tframes [protected]
```

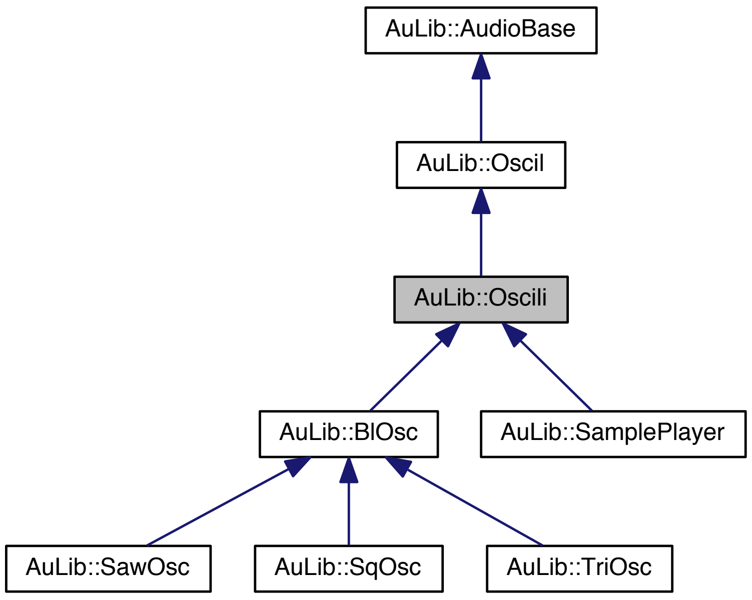
The documentation for this class was generated from the following files:

- Oscil.h
- Oscil.cpp

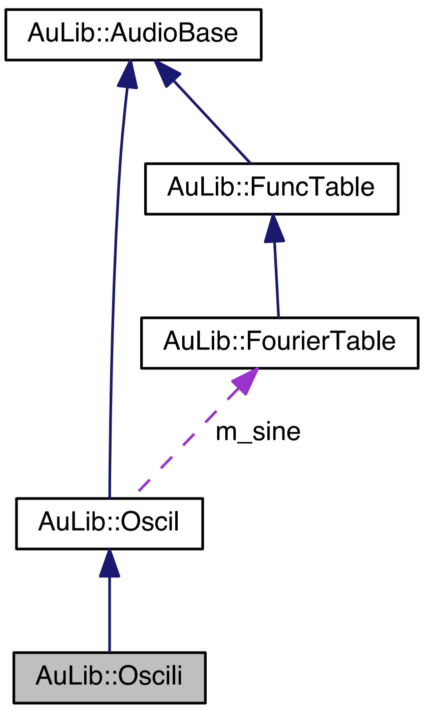
7.29 AuLib::Oscili Class Reference

```
#include <AuLib/Oscili.h>
```

Inheritance diagram for AuLib::Oscili:



Collaboration diagram for AuLib::Oscili:



Public Member Functions

- Oscili (double amp=0., double freq=0., double phase=0., uint32_t vframes=def_vframes, double sr=def_sr)
- Oscili (double amp, double freq, const FuncTable &ftable, double phase=.0, uint32_t vframes=def_vframes, double sr=def_sr)

Protected Member Functions

- virtual void dsp ()

Additional Inherited Members

7.29.1 Detailed Description

Linear interpolation oscillator

7.29.2 Constructor & Destructor Documentation

7.29.2.1 Osciili() [1/2]

```
AuLib::Oscili::Oscili (
            double amp = 0.,
            double freq = 0.,
            double phase = 0.,
            uint32_t vframes = def_vframes,
            double sr = def_sr )  [inline]
```

Sinusoidal Oscili constructor

amp - amplitude
freq - frequency in Hz
phase - init phase (0-1)
vframes - vector size
sr - sampling rate

Uses internal sine wave table

7.29.2.2 Oscili() [2/2]

```
AuLib::Oscili::Oscili (
            double amp,
            double freq,
            const FuncTable & ftable,
            double phase = .0,
            uint32_t vframes = def_vframes,
            double sr = def_sr )  [inline]
```

Oscili constructor

amp - amplitude
freq - frequency in Hz
ftable - function table
phase - init phase (0-1)
vframes - vector size
sr - sampling rate

7.29.3 Member Function Documentation

7.29.3.1 dsp()

```
void AuLib::Oscili::dsp ( )  [protected], [virtual]
```

interpolating oscillator process

Reimplemented from AuLib::Oscil.

Reimplemented in AuLib::SamplePlayer.

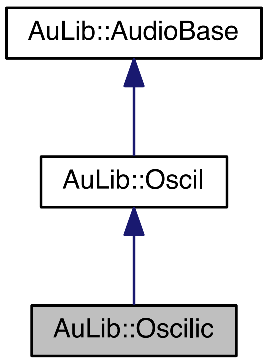
The documentation for this class was generated from the following files:

- Oscili.h
- Oscil.cpp

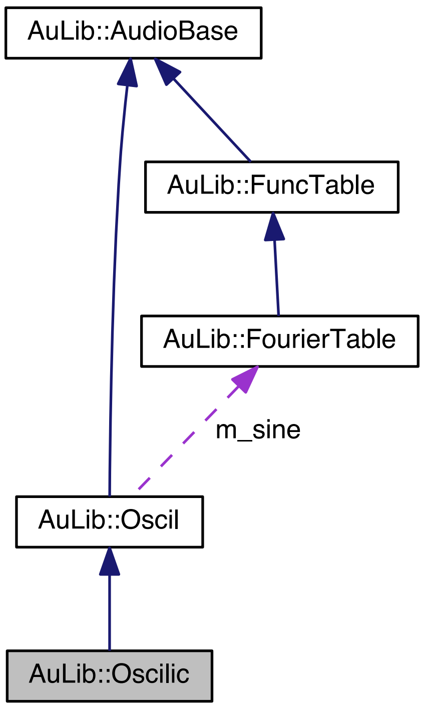
7.30 AuLib::Oscilic Class Reference

```
#include <AuLib/Oscil.h>
```

Inheritance diagram for AuLib::Oscilic:



Collaboration diagram for AuLib::Oscilic:



Public Member Functions

- Oscilic (double amp=0., double freq=0., double phase=0., uint32_t vframes=def_vframes, double sr=def_sr)
- Oscilic (double amp, double freq, const FuncTable &ftable, double phase=0., uint32_t vframes=def_vframes, double sr=def_sr)

Protected Member Functions

- virtual void dsp ()

Additional Inherited Members

7.30.1 Detailed Description

Cubic interpolation oscillator

7.30.2 Constructor & Destructor Documentation

7.30.2.1 Oscilic() [1/2]

```
AuLib::Oscilic::Oscilic (
             double amp = 0.,
             double freq = 0.,
             double phase = 0.,
             uint32_t vframes = def_vframes,
             double sr = def_sr )  [inline]
```

Sinusoidal Oscilic constructor

amp - amplitude
freq - frequency in Hz
phase - init phase (0-1)
vframes - vector size
sr - sampling rate

Uses internal sine wave table

7.30.2.2 Oscilic() [2/2]

```
AuLib::Oscilic::Oscilic (
             double amp,
             double freq,
             const FuncTable & ftable,
             double phase = 0.,
             uint32_t vframes = def_vframes,
             double sr = def_sr )  [inline]
```

Oscilic constructor

amp - amplitude
freq - frequency in Hz
ftable - function table
phase - init phase (0-1)
vframes - vector size
sr - sampling rate

7.30.3 Member Function Documentation

7.30.3.1 dsp()

```
void AuLib::Oscilic::dsp ( )  [protected], [virtual]
```

cubic oscillator process

Reimplemented from AuLib::Oscil.

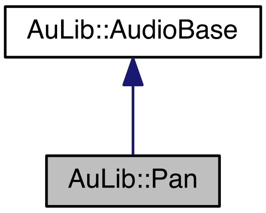
The documentation for this class was generated from the following files:

- Oscilic.h
- Oscil.cpp

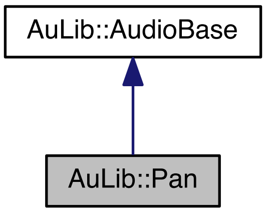
7.31 AuLib::Pan Class Reference

```
#include <AuLib/Pan.h>
```

Inheritance diagram for AuLib::Pan:



Collaboration diagram for AuLib::Pan:



Public Member Functions

- Pan (double pos=.5, uint32_t vframes=def_vframes)
- const double ∗ process (const double ∗sig)
- const double ∗ process (const double ∗sig, double pos)
- const Pan & process (const AudioBase &obj)
- const Pan & process (const AudioBase &obj, double pos)
- const Pan & operator() (const AudioBase &obj)
- const Pan & operator() (const AudioBase &obj, double pos)

Protected Attributes

- double m_pos

Additional Inherited Members

7.31.1 Detailed Description

Pan a mono input signal using an equal-power algorithm

7.31.2 Constructor & Destructor Documentation

7.31.2.1 Pan()

```
AuLib::Pan::Pan (
            double pos = .5,
            uint32_t vframes = def_vframes )  [inline]
```

Pan constructor

pos - pan position (0 - 1)
vframes - vector size

7.31.3 Member Function Documentation

7.31.3.1 operator()() [1/2]

```
const Pan& AuLib::Pan::operator() (
            const AudioBase & obj )  [inline]
```

operator(a) convenience method

7.31.3.2 operator()() [2/2]

```
const Pan& AuLib::Pan::operator() (
             const AudioBase & obj,
             double pos )  [inline]
```

operator(a,b) convenience method

7.31.3.3 process() [1/4]

```
const double* AuLib::Pan::process (
             const double * sig )  [inline]
```

Pan a signal sig

7.31.3.4 process() [2/4]

```
const double* AuLib::Pan::process (
             const double * sig,
             double pos )  [inline]
```

Pan a signal sig according to position pos (0-1)

7.31.3.5 process() [3/4]

```
const Pan& AuLib::Pan::process (
             const AudioBase & obj )  [inline]
```

Pan a signal in obj

7.31.3.6 process() [4/4]

```
const Pan& AuLib::Pan::process (
             const AudioBase & obj,
             double pos )  [inline]
```

Pan a signal in obj according to position pos (0-1)

7.31.4 Member Data Documentation

7.31.4.1 m_pos

```
double AuLib::Pan::m_pos  [protected]
```

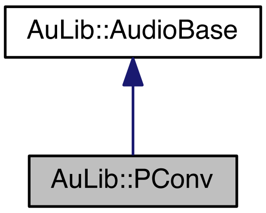
The documentation for this class was generated from the following files:

- Pan.h
- Pan.cpp

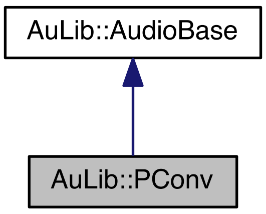
7.32 AuLib::PConv Class Reference

```
#include <AuLib/PConv.h>
```

Inheritance diagram for AuLib::PConv:



Collaboration diagram for AuLib::PConv:



Public Member Functions

- PConv (const FuncTable &ir, uint32_t psize=256, uint32_t chn=0, uint32_t begin=0, uint32_t end=0, uint32_t vframes=def_vframes, double sr=def_sr)
- const double ∗ process (const double ∗sig)
- const PConv & process (const AudioBase &obj)
- const PConv & operator() (const AudioBase &obj)

Additional Inherited Members

7.32.1 Constructor & Destructor Documentation

7.32.1.1 PConv()

```
AuLib::PConv::PConv (
             const FuncTable & ir,
             uint32_t psize = 256,
             uint32_t chn = 0,
             uint32_t begin = 0,
             uint32_t end = 0,
             uint32_t vframes = def_vframes,
             double sr = def_sr )
```

7.32.2 Member Function Documentation

7.32.2.1 operator()()

```
const PConv& AuLib::PConv::operator() (
             const AudioBase & obj )  [inline]
```

function operator convenience method, same as process()

7.32.2.2 process() [1/2]

```
const double* AuLib::PConv::process (
             const double * sig )  [inline]
```

Apply partitioned convolution to an input signal sig

7.32.2.3 process() [2/2]

```
const PConv& AuLib::PConv::process (
             const AudioBase & obj )  [inline]
```

Apply partitioned convolution to a signal in obj

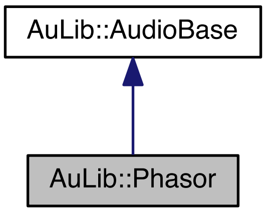
The documentation for this class was generated from the following files:

- PConv.h
- PConv.cpp

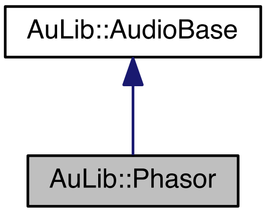
7.33 AuLib::Phasor Class Reference

`#include <AuLib/Phasor.h>`

Inheritance diagram for AuLib::Phasor:



Collaboration diagram for AuLib::Phasor:



Public Member Functions

- Phasor (double freq=0., double phase=0., uint32_t vframes=def_vframes, double sr=def_sr)
- const Phasor & process ()
- const Phasor & process (double freq)
- const Phasor & operator() ()
- const Phasor & operator() (double freq)

Protected Member Functions

- void mod1 ()

Protected Attributes

- double m_freq
- double m_phs
- double m_incr

Additional Inherited Members

7.33.1 Detailed Description

Phase signal (ramp) generator

7.33.2 Constructor & Destructor Documentation

7.33.2.1 Phasor()

```
AuLib::Phasor::Phasor (
             double freq = 0.,
             double phase = 0.,
             uint32_t vframes = def_vframes,
             double sr = def_sr )
```

Phasor constructor

freq - frequency in Hz
phase - init phase (0-1)
sr - sampling rate
vframes - vector size

7.33.3 Member Function Documentation

7.33.3.1 mod1()

```
void AuLib::Phasor::mod1 ( )  [inline], [protected]
```

phase modulo

7.33.3.2 operator()() [1/2]

```
const Phasor& AuLib::Phasor::operator() ( )  [inline]
```

operator () convenience

7.33.3.3 operator()() [2/2]

```
const Phasor& AuLib::Phasor::operator() (
             double freq )  [inline]
```

operator () convenience

7.33.3.4 process() [1/2]

```
const Phasor& AuLib::Phasor::process ( )  [inline]
```

Process one vector of audio

7.33.3.5 process() [2/2]

```
const Phasor& AuLib::Phasor::process (
             double freq )  [inline]
```

Process one vector of audio with frequency freq

7.33.4 Member Data Documentation

7.33.4.1 m_freq

```
double AuLib::Phasor::m_freq  [protected]
```

7.33.4.2 m_incr

```
double AuLib::Phasor::m_incr  [protected]
```

7.33.4.3 m_phs

```
double AuLib::Phasor::m_phs  [protected]
```

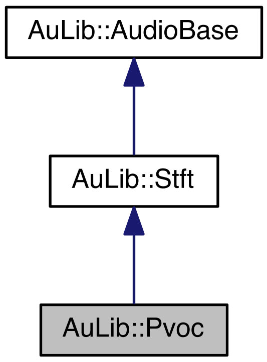
The documentation for this class was generated from the following files:

- Phasor.h
- Phasor.cpp

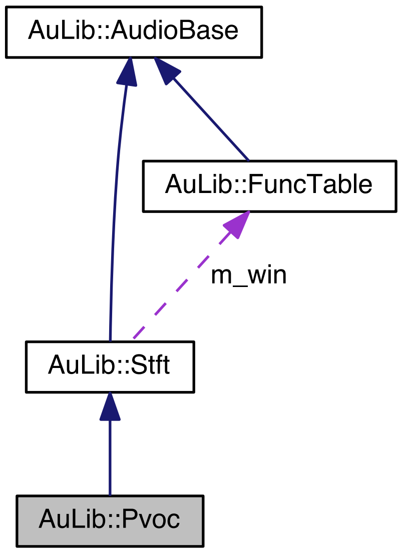
7.34 AuLib::Pvoc Class Reference

```
#include <AuLib/Pvoc.h>
```

Inheritance diagram for AuLib::Pvoc:



Collaboration diagram for AuLib::Pvoc:



Public Member Functions

- Pvoc (const FuncTable &win, bool dir, uint32_t decim=def_decim, uint32_t vframes=def_vframes, double sr=def_sr)

- virtual const std::vector< std::complex< double > > & spectrum ()
- virtual const AudioBase & operator*= (double scal)
- virtual const AudioBase & operator*= (const double *sig)
- virtual const AudioBase & operator*= (const Pvoc &obj)
- virtual const AudioBase & operator+= (double num)
- virtual const AudioBase & operator+= (const double *sig)
- virtual const AudioBase & operator+= (const Stft &obj)

Additional Inherited Members

7.34.1 Constructor & Destructor Documentation

7.34.1.1 Pvoc()

```
AuLib::Pvoc::Pvoc (
            const FuncTable & win,
            bool dir,
            uint32_t decim = def_decim,
            uint32_t vframes = def_vframes,
            double sr = def_sr )  [inline]
```

Pvoc constructor

win - analysis window
decim - decimation
fwd - true for forward, false for inverse
vframes - vector size
sr - sampling rate

7.34.2 Member Function Documentation

7.34.2.1 operator*=() [1/3]

```
virtual const AudioBase& AuLib::Pvoc::operator*= (
            double scal )  [inline], [virtual]
```

Scale the spectral data vector

Reimplemented from AuLib::Stft.

7.34.2.2 operator*=() [2/3]

```
virtual const AudioBase& AuLib::Pvoc::operator*= (
              const double * sig )  [inline], [virtual]
```

Multiply the data vector by a spectral vector

Reimplemented from AuLib::Stft.

7.34.2.3 operator*=() [3/3]

```
virtual const AudioBase& AuLib::Pvoc::operator*= (
              const Pvoc & obj )  [inline], [virtual]
```

Multiply the data vector by the vector from obj

7.34.2.4 operator+=() [1/3]

```
virtual const AudioBase& AuLib::Pvoc::operator+= (
              double num )  [inline], [virtual]
```

Add a double to the magnitude of spectral data

Reimplemented from AuLib::Stft.

7.34.2.5 operator+=() [2/3]

```
virtual const AudioBase& AuLib::Pvoc::operator+= (
              const double * sig )  [inline], [virtual]
```

Mix a spectral vector into this object

Reimplemented from AuLib::Stft.

7.34.2.6 operator+=() [3/3]

```
virtual const AudioBase& AuLib::Pvoc::operator+= (
              const Stft & obj )  [inline], [virtual]
```

Mix a vector sig from obj into this object

Reimplemented from AuLib::Stft.

7.34.2.7 spectrum()

```
virtual const std::vector<std::complex<double> >& AuLib::Pvoc::spectrum ( )  [inline], [virtual]
```

return spectrum as a complex<double> vector ref (only meaningful in forward transforms)

Reimplemented from AuLib::Stft.

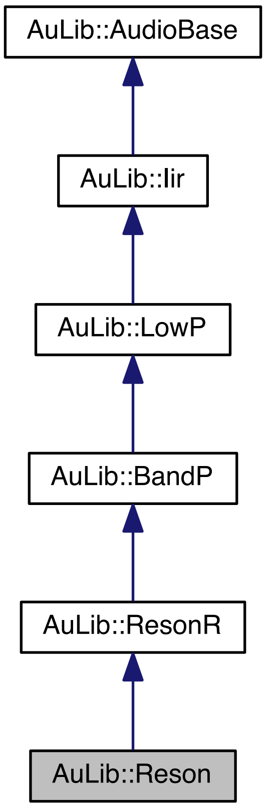
The documentation for this class was generated from the following files:

- Pvoc.h
- Stft.cpp

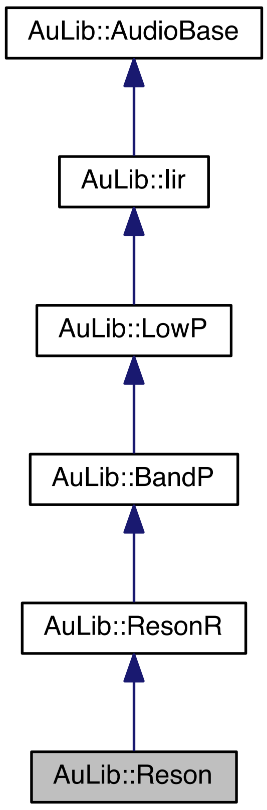
7.35 AuLib::Reson Class Reference

```
#include <AuLib/Reson.h>
```

Inheritance diagram for AuLib::Reson:



Collaboration diagram for AuLib::Reson:



Public Member Functions

- Reson (double cf, double bw, uint32_t vframes=def_vframes, double sr=def_sr)

Protected Member Functions

- virtual const double ∗ filter (const double ∗sig)

Additional Inherited Members

7.35.1 Detailed Description

Original band-pass resonator design

7.35.2 Constructor & Destructor Documentation

7.35.2.1 Reson()

```
AuLib::Reson::Reson (
            double cf,
            double bw,
            uint32_t vframes = def_vframes,
            double sr = def_sr )  [inline]
```

Reson constructor

cf - centre frequency
bw - bandwidth
vframes - vector size
sr - sampling rate

7.35.3 Member Function Documentation

7.35.3.1 filter()

```
const double * AuLib::Reson::filter (
            const double * sig )  [protected], [virtual]
```

Filter kernel

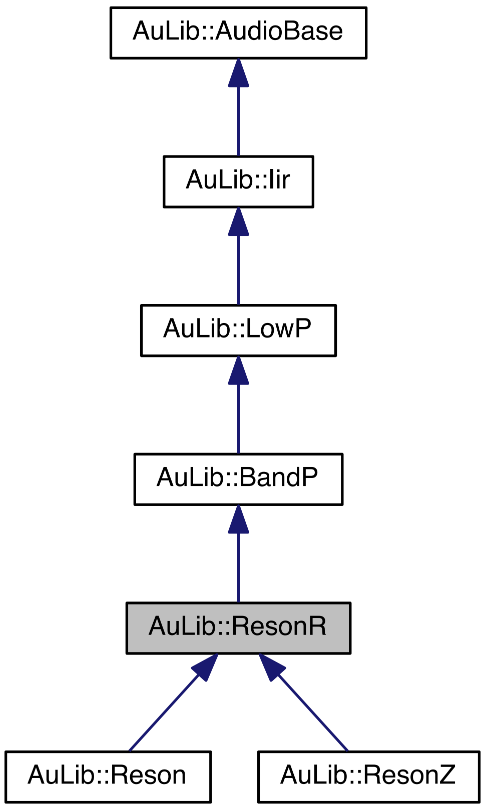
The documentation for this class was generated from the following files:

- Reson.h
- Iir.cpp

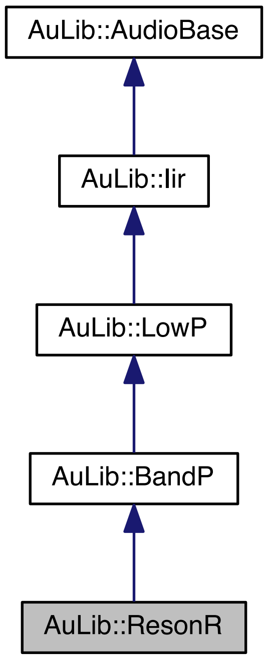
7.36 AuLib::ResonR Class Reference

```
#include <AuLib/ResonR.h>
```

Inheritance diagram for AuLib::ResonR:



Collaboration diagram for AuLib::ResonR:



Public Member Functions

- ResonR (double cf, double bw, uint32_t vframes=def_vframes, double sr=def_sr)

Protected Member Functions

- virtual void update ()

Additional Inherited Members

7.36.1 Detailed Description

Band-pass resonator with better amplitude response at low frequencies.

7.36.2 Constructor & Destructor Documentation

7.36.2.1 ResonR()

```
AuLib::ResonR::ResonR (
             double cf,
             double bw,
             uint32_t vframes = def_vframes,
             double sr = def_sr )  [inline]
```

ResonR constructor

cf - centre frequency
bw - bandwidth
vframes - vector size
sr - sampling rate

7.36.3 Member Function Documentation

7.36.3.1 update()

```
void AuLib::ResonR::update ( )  [protected], [virtual]
```

Coefficients update

Reimplemented from AuLib::BandP.

Reimplemented in AuLib::ResonZ.

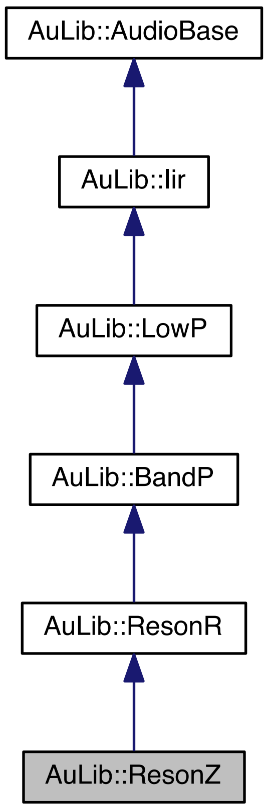
The documentation for this class was generated from the following files:

- ResonR.h
- Iir.cpp

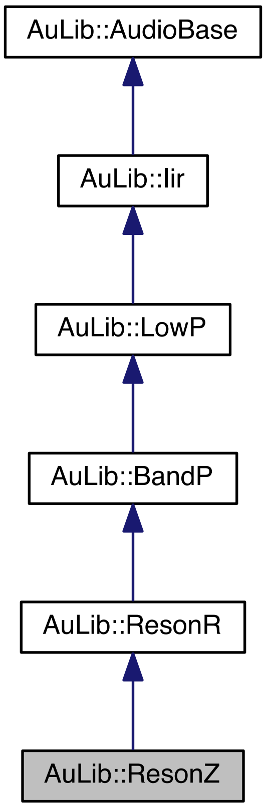
7.37 AuLib::ResonZ Class Reference

```
#include <AuLib/ResonZ.h>
```

Inheritance diagram for AuLib::ResonZ:



Collaboration diagram for AuLib::ResonZ:



Public Member Functions

- ResonZ (double cf, double bw, uint32_t vframes=def_vframes, double sr=def_sr)

Protected Member Functions

- virtual void update ()

Additional Inherited Members

7.37.1 Detailed Description

Band-pass resonator with better amplitude response at low frequencies (alternative design).

7.37.2 Constructor & Destructor Documentation

7.37.2.1 ResonZ()

```
AuLib::ResonZ::ResonZ (
            double cf,
            double bw,
            uint32_t vframes = def_vframes,
            double sr = def_sr )  [inline]
```

ResonZ constructor

cf - centre frequency
bw - bandwidth
vframes - vector size
sr - sampling rate

7.37.3 Member Function Documentation

7.37.3.1 update()

```
virtual void AuLib::ResonZ::update ( )  [inline], [protected], [virtual]
```

Coefficients update

Reimplemented from AuLib::ResonR.

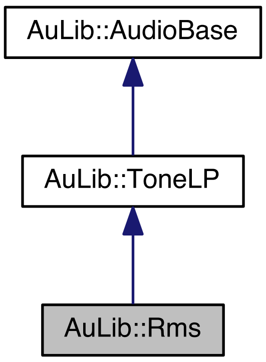
The documentation for this class was generated from the following file:

- ResonZ.h

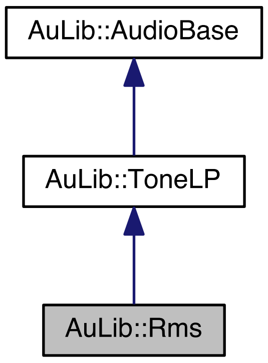
7.38 AuLib::Rms Class Reference

```
#include <AuLib/Rms.h>
```

Inheritance diagram for AuLib::Rms:



Collaboration diagram for AuLib::Rms:



Public Member Functions

- Rms (double cf=10., uint32_t vframes=def_vframes, double sr=def_sr)

Protected Member Functions

- double rect (double x)
- virtual const double ∗ dsp (const double ∗sig)

Additional Inherited Members

7.38.1 Detailed Description

Estimates the root-means-square of a signal

7.38.2 Constructor & Destructor Documentation

7.38.2.1 Rms()

```
AuLib::Rms::Rms (
             double cf = 10.,
             uint32_t vframes = def_vframes,
             double sr = def_sr )  [inline]
```

Rms constructor

cf - cutoff frequency
vframes - vector size
sr - sampling rate

7.38.3 Member Function Documentation

7.38.3.1 dsp()

```
virtual const double* AuLib::Rms::dsp (
            const double * sig )  [inline], [protected], [virtual]
```

RMS processing

Reimplemented from AuLib::ToneLP.

7.38.3.2 rect()

```
double AuLib::Rms::rect (
            double x )  [inline], [protected]
```

rectification

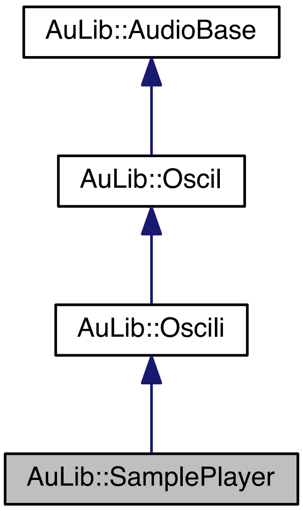
The documentation for this class was generated from the following file:

- Rms.h

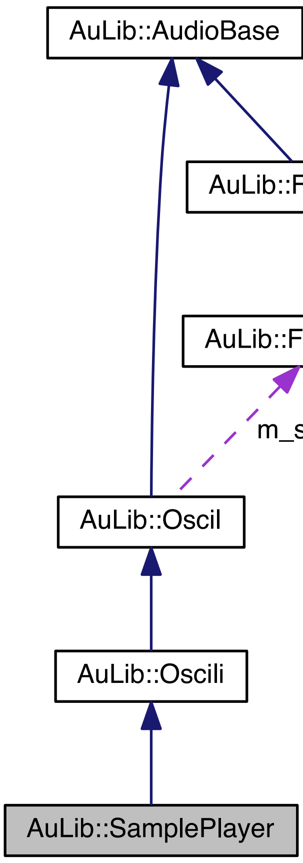
7.39 AuLib::SamplePlayer Class Reference

```
#include <AuLib/SamplePlayer.h>
```

Inheritance diagram for AuLib::SamplePlayer:



Collaboration diagram for AuLib::SamplePlayer:



Public Member Functions

- SamplePlayer (const FuncTable &ftable, double amp=0., double pitch=0., double phase=0., uint32_↩ t vframs=def_vframes, double sr=def_sr)

Protected Member Functions

- virtual void dsp ()
- virtual void am_fm (uint32_t ndx)
- virtual void set_incr (double f)

Additional Inherited Members

7.39.1 Detailed Description

Sample playback oscillator with linear interpolation

7.39.2 Constructor & Destructor Documentation

7.39.2.1 SamplePlayer()

```
AuLib::SamplePlayer::SamplePlayer (
             const FuncTable & ftable,
             double amp = 0.,
             double pitch = 0.,
             double phase = 0.,
             uint32_t vframs = def_vframes,
             double sr = def_sr )  [inline]
```

SamplePlayer constructor

table - function table
amp - amplitude
pitch - playback pitch
phase - init phase (0-1)
vframes - vector size
sr - sampling rate

7.39.3 Member Function Documentation

7.39.3.1 am_fm()

```
virtual void AuLib::SamplePlayer::am_fm (
             uint32_t ndx )  [inline], [protected], [virtual]
```

AM/FM processing

Reimplemented from AuLib::Oscil.

7.39.3.2 dsp()

```
void AuLib::SamplePlayer::dsp ( )  [protected], [virtual]
```

Oscillator for multichannel tables

Reimplemented from AuLib::Oscili.

7.39.3.3 set_incr()

```
virtual void AuLib::SamplePlayer::set_incr (
             double f )  [inline], [protected], [virtual]
```

set the sampling increment

Reimplemented from AuLib::Oscil.

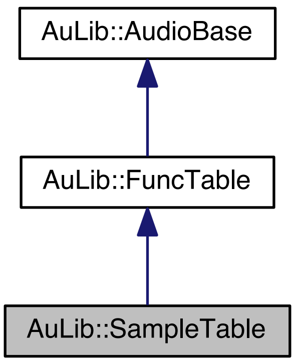
The documentation for this class was generated from the following files:

- SamplePlayer.h
- Oscil.cpp

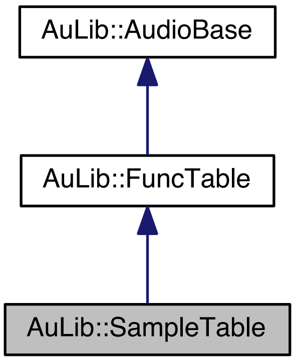
7.40 AuLib::SampleTable Class Reference

```
#include <AuLib/SampleTable.h>
```

Inheritance diagram for AuLib::SampleTable:



Collaboration diagram for AuLib::SampleTable:



Public Member Functions

- SampleTable (const char ∗name, uint32_t chn=1)
- virtual const char ∗ error_message () const

Additional Inherited Members

7.40.1 Detailed Description

Sampled-sound table from a soundfile

7.40.2 Constructor & Destructor Documentation

7.40.2.1 SampleTable()

```
AuLib::SampleTable::SampleTable (
             const char ∗ name,
             uint32_t chn = 1 )
```

SampleTable constructor

name - filename
chn - channel to load (0 = all channels)

7.40.3 Member Function Documentation

7.40.3.1 error_message()

```
virtual const char* AuLib::SampleTable::error_message ( ) const  [inline], [virtual]
```

Get error message

Reimplemented from AuLib::AudioBase.

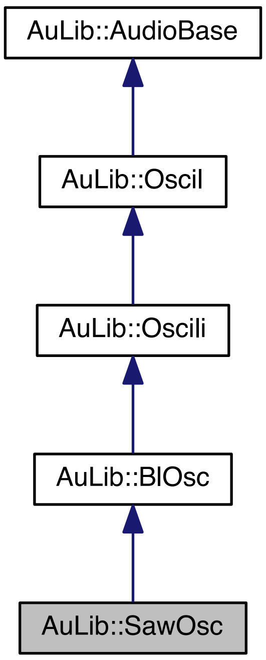
The documentation for this class was generated from the following files:

- SampleTable.h
- SampleTable.cpp

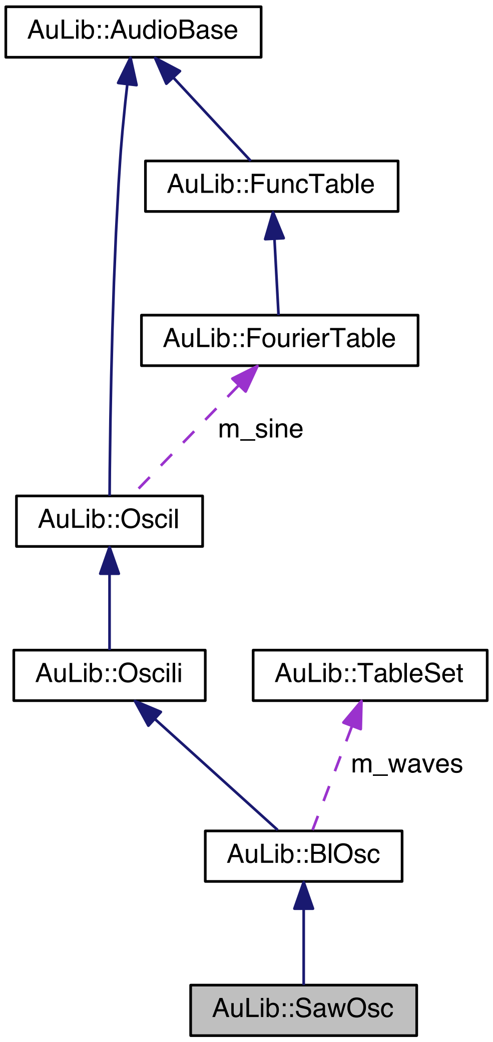
7.41 AuLib::SawOsc Struct Reference

```
#include <BlOsc.h>
```

Inheritance diagram for AuLib::SawOsc:



Collaboration diagram for AuLib::SawOsc:



Public Member Functions

- SawOsc (double amp=0.f, double freq=440.f, double phase=0., uint32_t vframes=def_vframes, double sr=def_sr)

Additional Inherited Members

7.41.1 Constructor & Destructor Documentation

7.41.1.1 SawOsc()

```
AuLib::SawOsc::SawOsc (
             double amp = 0.f,
             double freq = 440.f,
             double phase = 0.,
             uint32_t vframes = def_vframes,
             double sr = def_sr )  [inline]
```

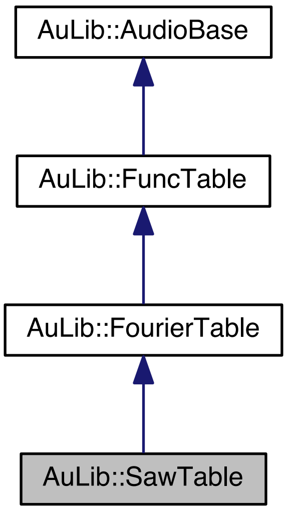
The documentation for this struct was generated from the following file:

- BlOsc.h

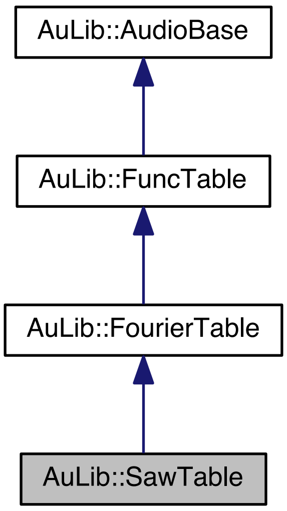
7.42 AuLib::SawTable Class Reference

```
#include <AuLib/WaveTables.h>
```

Inheritance diagram for AuLib::SawTable:



Collaboration diagram for AuLib::SawTable:



Public Member Functions

- SawTable (uint32_t harms, uint64_t tframes=def_tframes)

Additional Inherited Members

7.42.1 Detailed Description

Sawtooth wave table

7.42.2 Constructor & Destructor Documentation

7.42.2.1 SawTable()

```
AuLib::SawTable::SawTable (
            uint32_t harms,
            uint64_t tframes = def_tframes )  [inline]
```

SawTable constructor

harms - number of harmonics
tframes - table size

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

- WaveTables.h

7.43 AuLib::Score Class Reference

```
#include <AuLib/Score.h>
```

Classes

- struct Cmd

Public Member Functions

- Score ()
- void add_cmd (const Cmd &c)
- void read (std::ifstream &input)
- void read (std::string &input)
- void read (std::istream &input)
- void clear ()
- const std::list< Event > & score () const

Static Public Attributes

- static constexpr uint32_t end = 0

7.43.1 Detailed Description

Basic score model

7.43.2 Constructor & Destructor Documentation

7.43.2.1 Score()

```
AuLib::Score::Score ( )  [inline]
```

create a score

7.43.3 Member Function Documentation

7.43.3.1 add_cmd()

```
void AuLib::Score::add_cmd (
             const Cmd & c )  [inline]
```

Add command with the following format
{ name, // string
type, // Score::Cmd::omni or !Score::Cmd::omni
msg, // uint32_t message code
len // uint32_t number of data items (doubles) in cmd
}

Scores are required to define the termination command:

{name, Score::Cmd::omni, Score::end, 0},

where name is a user-defined name (std::string) for the command

7.43.3.2 clear()

```
void AuLib::Score::clear ( )  [inline]
```

clear the stored score

7.43.3.3 read() [1/3]

```
void AuLib::Score::read (
             std::ifstream & input )  [inline]
```

read file stream with commands

7.43.3.4 read() [2/3]

```
void AuLib::Score::read (
             std::string & input )  [inline]
```

read string with commands

7.43.3.5 read() [3/3]

```
void AuLib::Score::read (
             std::istream & input )  [inline]
```

read stream with commands

7.43.3.6 score()

```
const std::list<Event>& AuLib::Score::score ( ) const  [inline]
```

get the score

7.43.4 Member Data Documentation

7.43.4.1 end

```
constexpr uint32_t AuLib::Score::end = 0  [static]
```

end command message

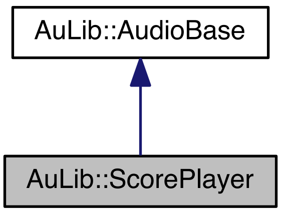
The documentation for this class was generated from the following file:

- Score.h

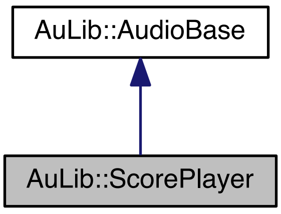
7.44 AuLib::ScorePlayer Class Reference

```
#include <AuLib/ScorePlayer.h>
```

Inheritance diagram for AuLib::ScorePlayer:



Collaboration diagram for AuLib::ScorePlayer:



Public Member Functions

- ScorePlayer (const Score &sc, uint32_t nchnls=def_nchnls, uint32_t vframes=def_vframes, double sr=def_sr)
- template<typename... Targs>
  const ScorePlayer & process (Targs &... args)
- void insert (const Score &input, double offset=0.)
- uint64_t score_time () const
- void reset ()
- void score_time (uint64_t t)
- void score_time (double t)
- void prepare ()
- template<typename T , typename... Targs>
  bool play (T &dest, Targs... args)
- bool check_score () const

Additional Inherited Members

7.44.1 Constructor & Destructor Documentation

7.44.1.1 ScorePlayer()

```
AuLib::ScorePlayer::ScorePlayer (
             const Score & sc,
             uint32_t nchnls = def_nchnls,
             uint32_t vframes = def_vframes,
             double sr = def_sr )  [inline]
```

create a score

7.44.2 Member Function Documentation

7.44.2.1 check_score()

```
bool AuLib::ScorePlayer::check_score ( ) const  [inline]
```

check score for termination msg (Score::end)

7.44.2.2 insert()

```
void AuLib::ScorePlayer::insert (
             const Score & input,
             double offset = 0.  )  [inline]
```

insert score at the current running time with optional offset.

7.44.2.3 play()

```
template<typename T , typename...  Targs>
bool AuLib::ScorePlayer::play (
             T & dest,
             Targs...  args )  [inline]
```

play loaded score from start position to end into dest object.

7.44.2.4 prepare()

```
void AuLib::ScorePlayer::prepare ( )  [inline]
```

prepare for playback of score object. Also resets playback time.

7.44.2.5 process()

```
template<typename...  Targs>
const ScorePlayer & AuLib::ScorePlayer::process (
             Targs &...  args )
```

process instruments obj ... dispatching score commands, for one vector of audio, returning the mixed score audio

7.44.2.6 reset()

```
void AuLib::ScorePlayer::reset ( )  [inline]
```

reset playback time counter

7.44.2.7 score_time() [1/3]

```
uint64_t AuLib::ScorePlayer::score_time ( ) const  [inline]
```

get current score time in frames

7.44.2.8 score_time() [2/3]

```
void AuLib::ScorePlayer::score_time (
             uint64_t t )  [inline]
```

set current score time in frames

7.44.2.9 score_time() [3/3]

```
void AuLib::ScorePlayer::score_time (
             double t )  [inline]
```

set current score time in secs

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

- ScorePlayer.h

7.45 AuLib::Segments Class Reference

```
#include <Envel.h>
```

Public Member Functions

- Segments (double start, const std::vector< double > endpts, const std::vector< double > times, bool linear=true, double sr=def_sr)
- Segments ()
- void reset (double start, const std::vector< double > endpts, const std::vector< double > times, double sr)
- const double ∗ incrs () const
- const uint32_t ∗ durs () const
- const double ∗ endpts () const
- uint32_t nsegs () const
- double start () const
- bool isLinear () const
- uint32_t frames () const

Static Public Attributes

- static constexpr bool linear = true
- static constexpr bool exponential = false

7.45.1 Detailed Description

Curve segments for envelope generators and tables

7.45.2 Constructor & Destructor Documentation

7.45.2.1 Segments() [1/2]

```
AuLib::Segments::Segments (
             double start,
             const std::vector< double > endpts,
             const std::vector< double > times,
             bool linear = true,
             double sr = def_sr )
```

7.45.2.2 Segments() [2/2]

```
AuLib::Segments::Segments ( )  [inline]
```

7.45.3 Member Function Documentation

7.45.3.1 durs()

```
const uint32_t* AuLib::Segments::durs ( ) const  [inline]
```

Get the durations array

7.45.3.2 endpts()

```
const double* AuLib::Segments::endpts ( ) const  [inline]
```

Get the endpts array

7.45.3.3 frames()

```
uint32_t AuLib::Segments::frames ( ) const  [inline]
```

Get the envelope duration in frames

7.45.3.4 incrs()

```
const double* AuLib::Segments::incrs ( ) const  [inline]
```

Get the increments array

7.45.3.5 isLinear()

```
bool AuLib::Segments::isLinear ( ) const  [inline]
```

check if the envelope is designed to be linear or else exponential

7.45.3.6 nsegs()

```
uint32_t AuLib::Segments::nsegs ( ) const  [inline]
```

Get the number of segments

7.45.3.7 reset()

```
void AuLib::Segments::reset (
             double start,
             const std::vector< double > endpts,
             const std::vector< double > times,
             double sr )
```

7.45.3.8 start()

```
double AuLib::Segments::start ( ) const  [inline]
```

Get the start value

7.45.4 Member Data Documentation

7.45.4.1 exponential

```
constexpr bool AuLib::Segments::exponential = false  [static]
```

define exponential segments

7.45.4.2 linear

```
constexpr bool AuLib::Segments::linear = true  [static]
```

define linear segments

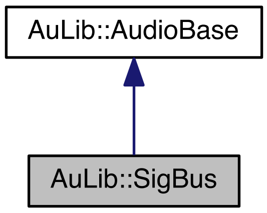
The documentation for this class was generated from the following files:

- Envel.h
- Envel.cpp

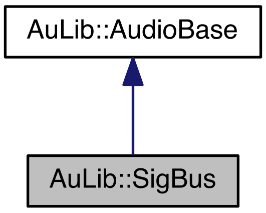
7.46 AuLib::SigBus Class Reference

```
#include <AuLib/SigBus.h>
```

Inheritance diagram for AuLib::SigBus:



Collaboration diagram for AuLib::SigBus:



Public Member Functions

- SigBus (double scal=1., double offs=0., bool overwrite=false, uint32_t nchnls=def_nchnls, uint32_↩
  t vframes=def_vframes)
- const double ∗ process (const double ∗sig)
- const SigBus & process (const double ∗sig, double scal, double offs=0., bool overwrite=true)
- const SigBus & process (const AudioBase &obj)
- const SigBus & process (const AudioBase &obj, double scal, double offs=0., bool overwrite=true)
- const SigBus & operator() (const AudioBase &obj)
- const SigBus & operator() (const AudioBase &obj, double scal, double offs=0., bool overwrite=true)
- void clear ()

Protected Attributes

- double m_scal
- double m_offs
- bool m_ovw

Additional Inherited Members

7.46.1 Detailed Description

A signal bus with optional scaling, offset and overwriting

7.46.2 Constructor & Destructor Documentation

7.46.2.1 SigBus()

```
AuLib::SigBus::SigBus (
            double scal = 1.,
            double offs = 0.,
            bool overwrite = false,
            uint32_t nchnls = def_nchnls,
            uint32_t vframes = def_vframes )  [inline]
```

SigBus constructor

scal - amplitude scaling
offs - amplitude offset
overwrite - overwrite bus
nchnls - number of channels
vframes - vector size

7.46.3 Member Function Documentation

7.46.3.1 clear()

```
void AuLib::SigBus::clear ( )  [inline]
```

Clears the vector vector

7.46.3.2 operator()() [1/2]

```
const SigBus& AuLib::SigBus::operator() (
            const AudioBase & obj )  [inline]
```

operator (a) convenience method

7.46.3.3 operator()() [2/2]

```
const SigBus& AuLib::SigBus::operator() (
            const AudioBase & obj,
            double scal,
            double offs = 0.,
            bool overwrite = true )  [inline]
```

operator (a,b,c,d) convenience method

7.46.3.4 process() [1/4]

```
const double* AuLib::SigBus::process (
            const double * sig )  [inline]
```

Adds a signal vector to the bus. Requires an explicit call to clear() if overwrite is switched off once the bus data has been consumed

7.46.3.5 process() [2/4]

```
const SigBus& AuLib::SigBus::process (
            const double * sig,
            double scal,
            double offs = 0.,
            bool overwrite = true )  [inline]
```

Applies a gain scaling and optional offset to a signal vector. If overwrite is true, the vector vector is overwritten and a call to clear() is not required.

7.46.3.6 process() [3/4]

```
const SigBus& AuLib::SigBus::process (
            const AudioBase & obj )  [inline]
```

Adds a signal vector to the bus from obj. Requires an explicit call to clear() once the bus data has been consumed

7.46.3.7 process() [4/4]

```
const SigBus& AuLib::SigBus::process (
            const AudioBase & obj,
            double scal,
            double offs = 0.,
            bool overwrite = true )  [inline]
```

Applies a gain scaling and optional offset to a signal vector from obj. If overwrite is true, the vector vector is overwritten and a call to clear() is not required.

7.46.4 Member Data Documentation

7.46.4.1 m_offs

```
double AuLib::SigBus::m_offs  [protected]
```

7.46.4.2 m_ovw

```
bool AuLib::SigBus::m_ovw  [protected]
```

7.46.4.3 m_scal

```
double AuLib::SigBus::m_scal [protected]
```

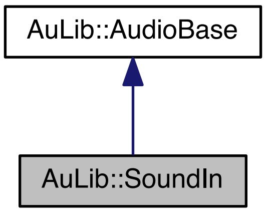
The documentation for this class was generated from the following file:

- SigBus.h

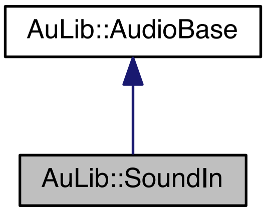
7.47 AuLib::SoundIn Class Reference

```
#include <AuLib/SoundIn.h>
```

Inheritance diagram for AuLib::SoundIn:



Collaboration diagram for AuLib::SoundIn:



Public Member Functions

- SoundIn (const char ∗src, uint32_t nchnls=def_nchnls, uint32_t vframes=def_vframes, uint64_↩ t bframes=def_bframes, double sr=def_sr)
- ∼SoundIn ()
- const double ∗ read (uint32_t frames=def_vframes)
- double time () const
- const char ∗ src () const
- uint64_t dur () const
- virtual const char ∗ error_message () const
- const double ∗ operator() (uint32_t frames=def_vframes)

Friends

- int rt_audio (const float ∗input, float ∗output, unsigned long frameCount, const void ∗timeInfo, unsigned long statusFlags, SoundIn ∗userData)
- void audio (SoundIn &obj)

Additional Inherited Members

7.47.1 Detailed Description

Audio input class

7.47.2 Constructor & Destructor Documentation

7.47.2.1 SoundIn()

```
AuLib::SoundIn::SoundIn (
             const char * src,
             uint32_t nchnls = def_nchnls,
             uint32_t vframes = def_vframes,
             uint64_t bframes = def_bframes,
             double sr = def_sr )
```

SoundIn constructor

src - input source ("adc", "stdin", or file path)
vframes - vector size
bframes - buffer size in frames

7.47.2.2 ∼SoundIn()

```
AuLib::SoundIn::∼SoundIn ( )
```

SoundOut destructor

7.47.3 Member Function Documentation

7.47.3.1 dur()

```
uint64_t AuLib::SoundIn::dur ( ) const  [inline]
```

Get the file duration in frames

7.47.3.2 error_message()

```
virtual const char* AuLib::SoundIn::error_message ( ) const  [inline], [virtual]
```

Get error message

Reimplemented from AuLib::AudioBase.

7.47.3.3 operator()()

```
const double* AuLib::SoundIn::operator() (
             uint32_t frames = def_vframes )  [inline]
```

operator () convenience method

7.47.3.4 read()

```
const double * AuLib::SoundIn::read (
             uint32_t frames = def_vframes )
```

Reads frames of audio to output

7.47.3.5 src()

```
const char* AuLib::SoundIn::src ( ) const  [inline]
```

Get the source name

7.47.3.6 time()

```
double AuLib::SoundIn::time ( ) const  [inline]
```

Get the current vector stream time

7.47.4 Friends And Related Function Documentation

7.47.4.1 audio

```
void audio (
             SoundIn & obj )  [friend]
```

7.47.4.2 rt_audio

```
int rt_audio (
             const float * input,
             float * output,
             unsigned long frameCount,
             const void * timeInfo,
             unsigned long statusFlags,
             SoundIn * userData )  [friend]
```

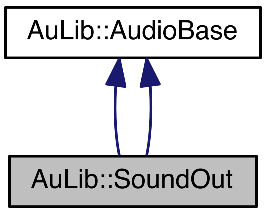
The documentation for this class was generated from the following files:

- SoundIn.h
- SoundIn.cpp

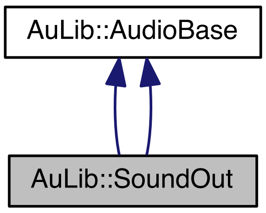
7.48 AuLib::SoundOut Class Reference

```
#include <AuLib/SoundOut.h>
```

Inheritance diagram for AuLib::SoundOut:



Collaboration diagram for AuLib::SoundOut:



Public Member Functions

- SoundOut (const char ∗dest, uint32_t nchnls=def_nchnls, uint32_t vframes=def_bframes, double sr=def_sr)
- ∼SoundOut ()
- uint64_t write (const double ∗sig, uint32_t samples)
- uint64_t write (const AudioBase &obj)
- uint64_t write (uint32_t chn, const AudioBase &obj)
- double time () const
- uint64_t timestamp () const
- const char ∗ dest () const
- virtual const char ∗ error_message () const
- uint64_t operator() (const AudioBase &obj)
- uint64_t operator() (uint32_t chn, const AudioBase &obj)
- SoundOut (const char ∗dest, uint32_t nchnls=def_nchnls, uint32_t vframes=def_bframes, double sr=def_sr)
- ∼SoundOut ()
- uint64_t write (const double ∗sig, uint32_t frames=def_vframes)
- uint64_t write (const AudioBase &obj)
- double time () const
- const char ∗ dest () const
- virtual const char ∗ error_message () const
- uint64_t operator() (const AudioBase &obj)

Friends

- int rt_audio (const float ∗input, float ∗output, unsigned long frameCount, const void ∗timeInfo, unsigned long statusFlags, SoundOut ∗userData)
- void audio (SoundOut &obj)
- int rt_audio (const float ∗input, float ∗output, unsigned long frameCount, const void ∗timeInfo, unsigned long statusFlags, SoundOut ∗userData)
- void audio (AuLib::SoundOut &obj)

Additional Inherited Members

7.48.1 Detailed Description

Audio output class

7.48.2 Constructor & Destructor Documentation

7.48.2.1 SoundOut() [1/2]

```
AuLib::SoundOut::SoundOut (
             const char * dest,
             uint32_t nchnls = def_nchnls,
             uint32_t vframes = def_bframes,
             double sr = def_sr )
```

SoundOut constructor

dest - vector destination ("dac", "stdout", or file path)
nchnls - number of channels
sr - sampling rate
vframes - vector size

7.48.2.2 ∼SoundOut() [1/2]

```
AuLib::SoundOut::∼SoundOut ( )
```

SoundOut destructor

7.48.2.3 SoundOut() [2/2]

```
AuLib::SoundOut::SoundOut (
            const char * dest,
            uint32_t nchnls = def_nchnls,
            uint32_t vframes = def_bframes,
            double sr = def_sr )
```

SoundOut constructor

dest - vector destination ("dac", "stdout", or file path)
nchnls - number of channels
sr - sampling rate
vframes - vector size

7.48.2.4 ∼SoundOut() [2/2]

```
AuLib::SoundOut::∼SoundOut ( )
```

SoundOut destructor

7.48.3 Member Function Documentation

7.48.3.1 dest() [1/2]

```
const char* AuLib::SoundOut::dest ( ) const [inline]
```

Get the destination name

7.48.3.2 dest() [2/2]

```
const char* AuLib::SoundOut::dest ( ) const [inline]
```

Get the destination name

7.48.3.3 error_message() [1/2]

```
virtual const char* AuLib::SoundOut::error_message ( ) const  [inline], [virtual]
```

Get error message

Reimplemented from AuLib::AudioBase.

7.48.3.4 error_message() [2/2]

```
virtual const char* AuLib::SoundOut::error_message ( ) const  [inline], [virtual]
```

Get error message

Reimplemented from AuLib::AudioBase.

7.48.3.5 operator()() [1/3]

```
uint64_t AuLib::SoundOut::operator() (
             const AudioBase & obj )  [inline]
```

operator() overload convenience method

7.48.3.6 operator()() [2/3]

```
uint64_t AuLib::SoundOut::operator() (
             const AudioBase & obj )  [inline]
```

operator() overload convenience method

7.48.3.7 operator()() [3/3]

```
uint64_t AuLib::SoundOut::operator() (
             uint32_t chn,
             const AudioBase & obj )  [inline]
```

operator() overload convenience method

7.48.3.8 time() [1/2]

```
double AuLib::SoundOut::time ( ) const  [inline]
```

Get the current vector stream time

7.48.3.9 time() [2/2]

```
double AuLib::SoundOut::time ( ) const  [inline]
```

Get the current output stream time

7.48.3.10 timestamp()

```
uint64_t AuLib::SoundOut::timestamp ( ) const  [inline]
```

Get the current output stream time in frames

7.48.3.11 write() [1/5]

```
uint64_t AuLib::SoundOut::write (
              const double * sig,
              uint32_t frames = def_vframes )
```

Writes sig to the vector destination, returning the vector current framecount.

7.48.3.12 write() [2/5]

```
uint64_t AuLib::SoundOut::write (
              const double * sig,
              uint32_t samples )  [inline]
```

Writes sig containing samples to the vector destination, returning the vector current framecount.

7.48.3.13 write() [3/5]

```
uint64_t AuLib::SoundOut::write (
              const AudioBase & obj )  [inline]
```

Writes the audio vector in obj to the vector destination, returning the vector current framecount.

7.48.3.14 write() [4/5]

```
uint64_t AuLib::SoundOut::write (
              const AudioBase & obj )  [inline]
```

Writes the audio vector in obj to the vector destination, returning the vector current framecount.

7.48.3.15 write() [5/5]

```
uint64_t AuLib::SoundOut::write (
              uint32_t chn,
              const AudioBase & obj )  [inline]
```

Writes the audio vector in obj to chn of destination, returning the vector current framecount.

7.48.4 Friends And Related Function Documentation

7.48.4.1 audio [1/2]

```
void audio (
             AuLib::SoundOut & obj )  [friend]
```

7.48.4.2 audio [2/2]

```
void audio (
             SoundOut & obj )  [friend]
```

7.48.4.3 rt_audio [1/2]

```
int rt_audio (
             const float * input,
             float * output,
             unsigned long frameCount,
             const void * timeInfo,
             unsigned long statusFlags,
             SoundOut * userData )  [friend]
```

7.48.4.4 rt_audio [2/2]

```
int rt_audio (
             const float * input,
             float * output,
             unsigned long frameCount,
             const void * timeInfo,
             unsigned long statusFlags,
             SoundOut * userData )  [friend]
```

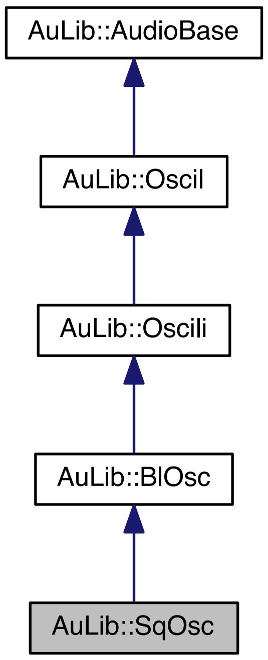
The documentation for this class was generated from the following files:

- SoundOut.h
- SoundOut_o.h
- SoundOut.cpp
- SoundOut_o.cpp

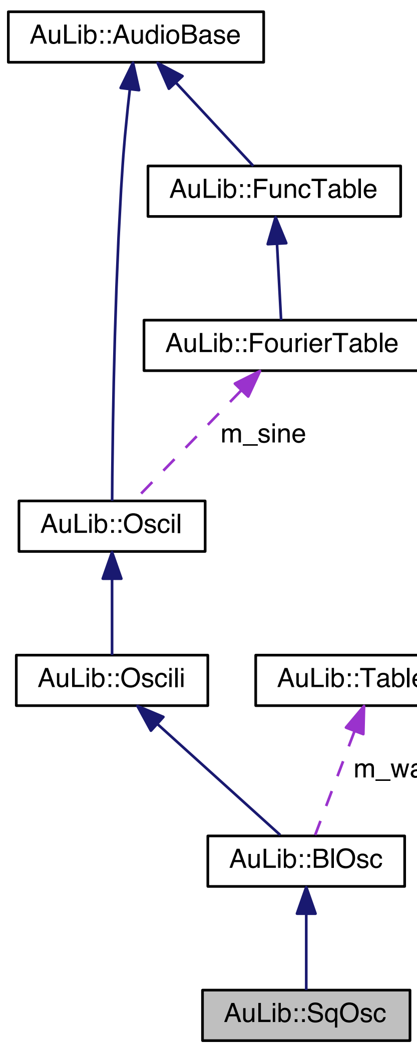
7.49 AuLib::SqOsc Struct Reference

```
#include <BlOsc.h>
```

Inheritance diagram for AuLib::SqOsc:



Collaboration diagram for AuLib::SqOsc:



Public Member Functions

- SqOsc (double amp=0.f, double freq=440.f, double phase=0., uint32_t vframes=def_vframes, double sr=def←
 _sr)

Additional Inherited Members

7.49.1 Constructor & Destructor Documentation

7.49.1.1 SqOsc()

```
AuLib::SqOsc::SqOsc (
             double amp = 0.f,
             double freq = 440.f,
             double phase = 0.,
             uint32_t vframes = def_vframes,
             double sr = def_sr )  [inline]
```

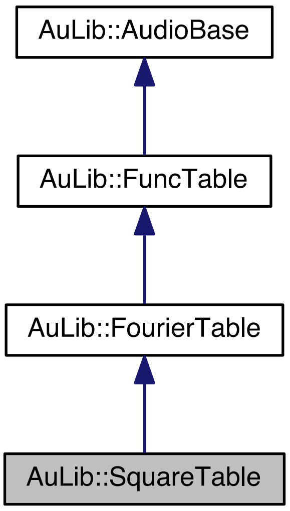
The documentation for this struct was generated from the following file:

- BlOsc.h

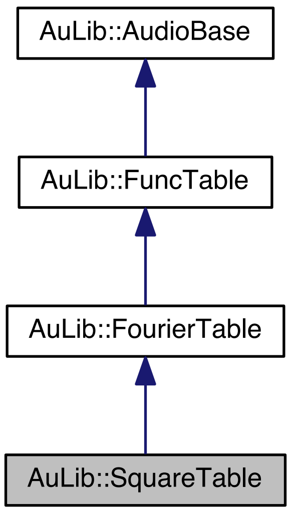
7.50 AuLib::SquareTable Class Reference

```
#include <AuLib/WaveTables.h>
```

Inheritance diagram for AuLib::SquareTable:



Collaboration diagram for AuLib::SquareTable:



Public Member Functions

- SquareTable (uint32_t harms, uint64_t tframes=def_tframes)

Additional Inherited Members

7.50.1 Detailed Description

Square wave table

7.50.2 Constructor & Destructor Documentation

7.50.2.1 SquareTable()

```
AuLib::SquareTable::SquareTable (
             uint32_t harms,
             uint64_t tframes = def_tframes )  [inline]
```

SquareTable constructor

harms - number of harmonics
tframes - table size

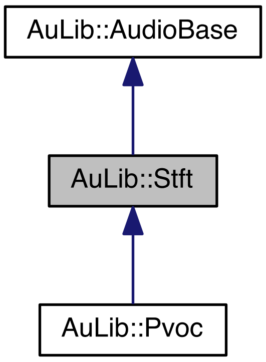
The documentation for this class was generated from the following file:

- WaveTables.h

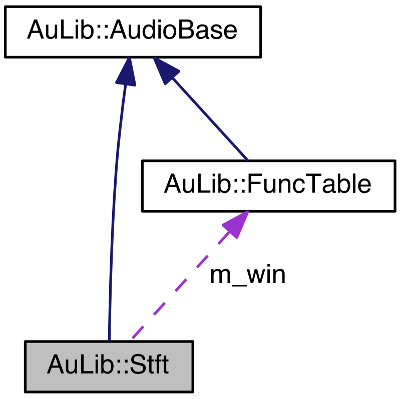
7.51 AuLib::Stft Class Reference

```
#include <AuLib/Stft.h>
```

Inheritance diagram for AuLib::Stft:



Collaboration diagram for AuLib::Stft:



Public Member Functions

- Stft (const FuncTable &win, bool dir, uint32_t decim=def_decim, bool repr=fft::rectang, uint32_↩ t vframes=def_vframes, double sr=def_sr)
- const double ∗ process (const double ∗sig, uint32_t vframes=def_vframes)
- const Stft & process (const AudioBase &obj)

- const Stft & operator() (const AudioBase &obj)
- bool repr () const
- bool framecount () const
- virtual const std::vector< std::complex< double > > & spectrum ()
- const std::complex< double > & bin (uint32_t n)
- virtual const AudioBase & operator*= (double scal)
- virtual const AudioBase & operator*= (const double *sig)
- virtual const AudioBase & operator*= (const Stft &obj)
- virtual const AudioBase & operator+= (double num)
- virtual const AudioBase & operator+= (const double *sig)
- virtual const AudioBase & operator+= (const Stft &obj)

Protected Member Functions

- virtual const double * transform (const double *sig, uint32_t vframes)

Protected Attributes

- const std::complex< double > m_z
- uint32_t m_N
- uint32_t m_H
- uint32_t m_D
- bool m_dir
- bool m_repr
- uint64_t m_framecount
- const FuncTable & m_win
- std::vector< std::vector< double > > m_framebufs
- std::vector< uint32_t > m_pos
- std::vector< std::complex< double > > m_cdata

Additional Inherited Members

7.51.1 Detailed Description

Stft implements a streaming Short-time Fourier Transform overlapped windows are taken from the input (forward mode) and analysis frames in either rectangular or polar formats are produced in the output (N/2 bins, first bin containing both DC and Nyq points) containing the non-negative spectrum.
On inverse mode, spectral data is read from the input and transformed back to the output, using an overlap-add algorithm.

7.51.2 Constructor & Destructor Documentation

7.51.2.1 Stft()

```
AuLib::Stft::Stft (
             const FuncTable & win,
             bool dir,
             uint32_t decim = def_decim,
             bool repr = fft::rectang,
             uint32_t vframes = def_vframes,
             double sr = def_sr )  [inline]
```

Stft constructor

win - analysis window
decim - decimation
dir - transform direction, forward or inverse
repr - data format, fft::polar or fft::rectang
vframes - vector size
sr - sampling rate

7.51.3 Member Function Documentation

7.51.3.1 bin()

```
const std::complex<double>& AuLib::Stft::bin (
             uint32_t n )  [inline]
```

return bin data as a complex<double> ref (only meaningful in forward transforms)

7.51.3.2 framecount()

```
bool AuLib::Stft::framecount ( ) const  [inline]
```

return analysis frame count

7.51.3.3 operator()()

```
const Stft& AuLib::Stft::operator() (
             const AudioBase & obj )  [inline]
```

operator (a) convenience method

7.51.3.4 operator∗=() [1/3]

```
virtual const AudioBase& AuLib::Stft::operator∗= (
             double scal )  [inline], [virtual]
```

Scale the spectral data vector

Reimplemented from AuLib::AudioBase.

Reimplemented in AuLib::Pvoc.

7.51.3.5 operator∗=() [2/3]

```
virtual const AudioBase& AuLib::Stft::operator∗= (
             const double ∗ sig )  [inline], [virtual]
```

Multiply the data vector by a spectral vector

Reimplemented from AuLib::AudioBase.

Reimplemented in AuLib::Pvoc.

7.51.3.6 operator∗=() [3/3]

```
virtual const AudioBase& AuLib::Stft::operator∗= (
             const Stft & obj )  [inline], [virtual]
```

Multiply the data vector by the vector from obj

7.51.3.7 operator+=() [1/3]

```
virtual const AudioBase& AuLib::Stft::operator+= (
             double num )  [inline], [virtual]
```

Add a double to the spectral data (non-op for polar repr)

Reimplemented from AuLib::AudioBase.

Reimplemented in AuLib::Pvoc.

7.51.3.8 operator+=() [2/3]

```
virtual const AudioBase& AuLib::Stft::operator+= (
             const double * sig )  [inline], [virtual]
```

Add a vector sig to the data vector (non-op for polar repr)

Reimplemented from AuLib::AudioBase.

Reimplemented in AuLib::Pvoc.

7.51.3.9 operator+=() [3/3]

```
virtual const AudioBase& AuLib::Stft::operator+= (
             const Stft & obj )  [inline], [virtual]
```

Add a vector sig from obj to the data vector (non-op for polar repr)

Reimplemented in AuLib::Pvoc.

7.51.3.10 process() [1/2]

```
const double* AuLib::Stft::process (
             const double * sig,
             uint32_t vframes = def_vframes )  [inline]
```

transform an signal sig, vframes is the (time-domain) processing vector size and it is required to be >= analysis hopsize, otherwise no processing takes place.

7.51.3.11 process() [2/2]

```
const Stft& AuLib::Stft::process (
             const AudioBase & obj )  [inline]
```

transform a signal in obj

7.51.3.12 repr()

```
bool AuLib::Stft::repr ( ) const  [inline]
```

return data format (fft::polar or fft::rectang)

7.51.3.13 spectrum()

```
virtual const std::vector<std::complex<double> >& AuLib::Stft::spectrum ( )  [inline], [virtual]
```

spectrum as a complex<double> vector ref (only meaningful in forward transforms)

Reimplemented in AuLib::Pvoc.

7.51.3.14 transform()

```
const double * AuLib::Stft::transform (
             const double * sig,
             uint32_t vframes )  [protected], [virtual]
```

STFT transform

7.51.4 Member Data Documentation

7.51.4.1 m_cdata

```
std::vector<std::complex<double> > AuLib::Stft::m_cdata  [protected]
```

7.51.4.2 m_D

```
uint32_t AuLib::Stft::m_D  [protected]
```

7.51.4.3 m_dir

```
bool AuLib::Stft::m_dir  [protected]
```

7.51.4.4 m_framebufs

```
std::vector<std::vector<double> > AuLib::Stft::m_framebufs  [protected]
```

7.51.4.5 m_framecount

```
uint64_t AuLib::Stft::m_framecount  [protected]
```

7.51.4.6 m_H

```
uint32_t AuLib::Stft::m_H  [protected]
```

7.51.4.7 m_N

```
uint32_t AuLib::Stft::m_N  [protected]
```

7.51.4.8 m_pos

```
std::vector<uint32_t> AuLib::Stft::m_pos  [protected]
```

7.51.4.9 m_repr

```
bool AuLib::Stft::m_repr  [protected]
```

7.51.4.10 m_win

```
const FuncTable& AuLib::Stft::m_win  [protected]
```

7.51.4.11 m_z

```
const std::complex<double> AuLib::Stft::m_z  [protected]
```

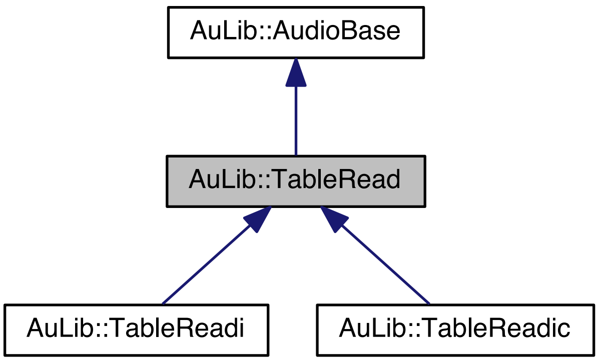
The documentation for this class was generated from the following files:

- Stft.h
- Stft.cpp

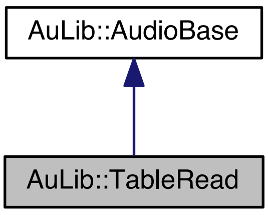
7.52 AuLib::TableRead Class Reference

```
#include <AuLib/TableRead.h>
```

Inheritance diagram for AuLib::TableRead:



Collaboration diagram for AuLib::TableRead:



Public Member Functions

- TableRead (const FuncTable &ftable, double phase=0., bool norm=true, bool wrap=true, uint32_↩
  t vframes=def_vframes, double sr=def_sr)
- const double ∗ process (const double ∗phs)
- const TableRead & process (const AudioBase &obj)
- const TableRead & operator() (const AudioBase &obj)

Protected Member Functions

- virtual void dsp (const double ∗phs)
- double mod (double pos)

Protected Attributes

- const double ∗ m_table
- double m_phs
- bool m_norm
- bool m_wrap
- uint64_t m_tframes

Additional Inherited Members

7.52.1 Detailed Description

Table reader with truncated lookup

7.52.2 Constructor & Destructor Documentation

7.52.2.1 TableRead()

```
AuLib::TableRead::TableRead (
             const FuncTable & ftable,
             double phase = 0.,
             bool norm = true,
             bool wrap = true,
             uint32_t vframes = def_vframes,
             double sr = def_sr )  [inline]
```

TableRead constructor

table - function table
phase - initial phase
norm - normalisation switch
wrap - wraparound switch
vframes - vector size
sr - sampling rate

7.52.3 Member Function Documentation

7.52.3.1 dsp()

```
void AuLib::TableRead::dsp (
             const double ∗ phs )  [protected], [virtual]
```

truncated table lookup

Reimplemented in AuLib::TableReadi, and AuLib::TableReadic.

7.52.3.2 mod()

```
double AuLib::TableRead::mod (
             double pos )  [inline], [protected]
```

modulo function

7.52.3.3 operator()()

```
const TableRead& AuLib::TableRead::operator() (
             const AudioBase & obj )  [inline]
```

operator () convenience

7.52.3.4 process() [1/2]

```
const double* AuLib::TableRead::process (
             const double * phs )  [inline]
```

takes in a frame of phase values and lookups up the table values

7.52.3.5 process() [2/2]

```
const TableRead& AuLib::TableRead::process (
             const AudioBase & obj )  [inline]
```

takes in a frame of phase values and lookups up the table values

7.52.4 Member Data Documentation

7.52.4.1 m_norm

```
bool AuLib::TableRead::m_norm  [protected]
```

7.52.4.2 m_phs

```
double AuLib::TableRead::m_phs  [protected]
```

7.52.4.3 m_table

```
const double* AuLib::TableRead::m_table  [protected]
```

7.52.4.4 m_tframes

```
uint64_t AuLib::TableRead::m_tframes  [protected]
```

7.52.4.5 m_wrap

```
bool AuLib::TableRead::m_wrap  [protected]
```

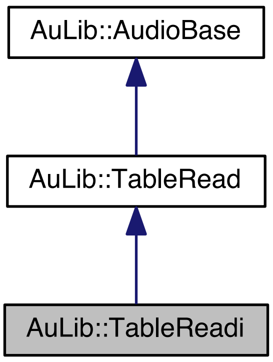
The documentation for this class was generated from the following files:

- TableRead.h
- TableRead.cpp

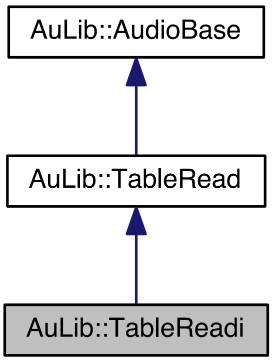
7.53 AuLib::TableReadi Class Reference

```
#include <AuLib/TableReadi.h>
```

Inheritance diagram for AuLib::TableReadi:



Collaboration diagram for AuLib::TableReadi:



Public Member Functions

- TableReadi (const FuncTable &ftable, double phase=0., bool norm=true, bool wrap=true, uint32_↩
t vframes=def_vframes)

Protected Member Functions

- virtual void dsp (const double ∗phs)

Additional Inherited Members

7.53.1 Detailed Description

Table reader with linear interpolation

7.53.2 Constructor & Destructor Documentation

7.53.2.1 TableReadi()

```
AuLib::TableReadi::TableReadi (
             const FuncTable & ftable,
             double phase = 0.,
             bool norm = true,
             bool wrap = true,
             uint32_t vframes = def_vframes )  [inline]
```

TableReadi constructor

ftable - function table
phase - initial phase
norm - normalisation switch
wrap - wraparound switch
vframes - vector size

7.53.3 Member Function Documentation

7.53.3.1 dsp()

```
void AuLib::TableReadi::dsp (
             const double * phs )  [protected], [virtual]
```

truncated table lookup

Reimplemented from AuLib::TableRead.

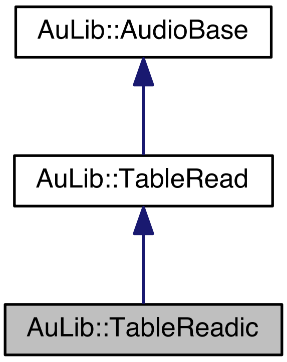
The documentation for this class was generated from the following files:

- TableReadi.h
- TableRead.cpp

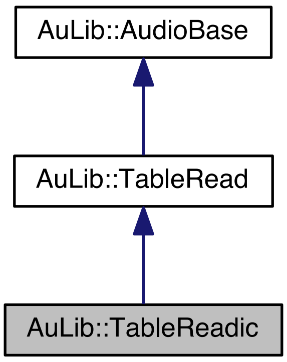
7.54 AuLib::TableReadic Class Reference

```
#include <AuLib/TableReadic.h>
```

Inheritance diagram for AuLib::TableReadic:



Collaboration diagram for AuLib::TableReadic:



Public Member Functions

- TableReadic (const FuncTable &ftable, double phase=0., bool norm=true, bool wrap=true, uint64_↩
t tframes=def_tframes, uint32_t vframes=def_vframes)

Protected Member Functions

- virtual void dsp (const double ∗phs)

Additional Inherited Members

7.54.1 Detailed Description

Table reader with cubic interpolation

7.54.2 Constructor & Destructor Documentation

7.54.2.1 TableReadic()

```
AuLib::TableReadic::TableReadic (
             const FuncTable & ftable,
             double phase = 0.,
             bool norm = true,
             bool wrap = true,
             uint64_t tframes = def_tframes,
             uint32_t vframes = def_vframes )  [inline]
```

TableReadic constructor

ftable - function table
phase - initial phase
norm - normalisation switch
wrap - wraparound switch
vframes - vector size

7.54.3 Member Function Documentation

7.54.3.1 dsp()

```
void AuLib::TableReadic::dsp (
             const double * phs )  [protected], [virtual]
```

truncated table lookup

Reimplemented from AuLib::TableRead.

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

- TableReadic.h
- TableRead.cpp

7.55 AuLib::TableSet Class Reference

```
#include <BlOsc.h>
```

Public Member Functions

- TableSet (uint32_t type, double sr=def_sr)
- const FuncTable & operator[ ] (uint32_t num) const

7.55.1 Detailed Description

A set of tables for BlOsc

7.55.2 Constructor & Destructor Documentation

7.55.2.1 TableSet()

```
AuLib::TableSet::TableSet (
            uint32_t type,
            double sr = def_sr )  [inline]
```

TableSet constructor type - SAW, SQUARE or TRIANGLE sr - sampling rate

7.55.3 Member Function Documentation

7.55.3.1 operator[]()

```
const FuncTable& AuLib::TableSet::operator[ ] (
            uint32_t num ) const  [inline]
```

returns a function table in the set.

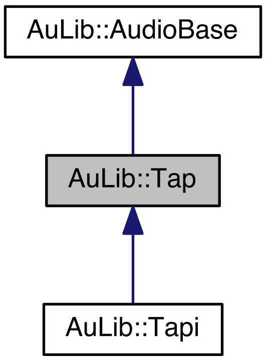
The documentation for this class was generated from the following file:

- BlOsc.h

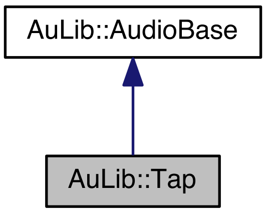
7.56 AuLib::Tap Class Reference

```
#include <AuLib/Tap.h>
```

Inheritance diagram for AuLib::Tap:



Collaboration diagram for AuLib::Tap:



Public Member Functions

- Tap (uint32_t vframes=def_vframes, double sr=def_sr)
- const Tap & process (const Delay &obj, double time)
- const double ∗ process (const Delay &obj, const double ∗time)
- const Tap & process (const Delay &del, const AudioBase &obj)
- const Tap & operator() (const Delay &del, const AudioBase &b)
- const Tap & operator() (const Delay &del, double b)

Additional Inherited Members

7.56.1 Detailed Description

Creates a tap for a Delay object truncating readout.

7.56.2 Constructor & Destructor Documentation

7.56.2.1 Tap()

```
AuLib::Tap::Tap (
            uint32_t vframes = def_vframes,
            double sr = def_sr )  [inline]
```

Tap constructor

vframes - vector size
sr - sampling rate

7.56.3 Member Function Documentation

7.56.3.1 operator()() [1/2]

```
const Tap& AuLib::Tap::operator() (
            const Delay & del,
            const AudioBase & b )  [inline]
```

operator () convenience method

7.56.3.2 operator()() [2/2]

```
const Tap& AuLib::Tap::operator() (
            const Delay & del,
            double b )  [inline]
```

operator () convenience method

7.56.3.3 process() [1/3]

```
const Tap& AuLib::Tap::process (
            const Delay & obj,
            double time )  [inline]
```

tap a delay object at time secs

7.56.3.4 process() [2/3]

```
const double* AuLib::Tap::process (
             const Delay & obj,
             const double * time )  [inline]
```

tap a delay object according to time signal in secs

7.56.3.5 process() [3/3]

```
const Tap& AuLib::Tap::process (
             const Delay & del,
             const AudioBase & obj )  [inline]
```

tap a delay object according to time signal from obj in secs

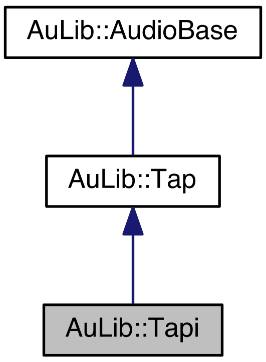
The documentation for this class was generated from the following files:

- Tap.h
- Tap.cpp

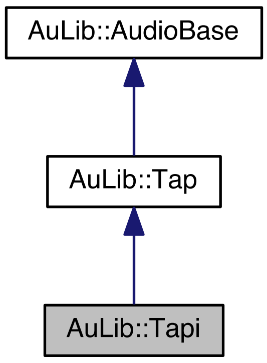
7.57 AuLib::Tapi Class Reference

```
#include <AuLib/Tapi.h>
```

Inheritance diagram for AuLib::Tapi:



Collaboration diagram for AuLib::Tapi:



Public Member Functions

- Tapi (uint32_t vframes=def_vframes, double sr=def_sr)

Additional Inherited Members

7.57.1 Detailed Description

Creates an tap for a Delay object, using interpolated readout.

7.57.2 Constructor & Destructor Documentation

7.57.2.1 Tapi()

```
AuLib::Tapi::Tapi (
              uint32_t vframes = def_vframes,
              double sr = def_sr )  [inline]
```

Tapi constructor

vframes - vector size
sr - sampling rate

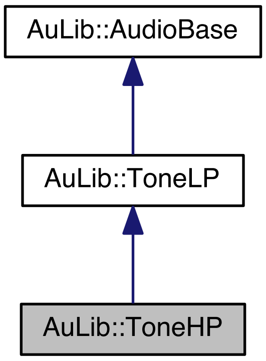
The documentation for this class was generated from the following files:

- Tapi.h
- Tap.cpp

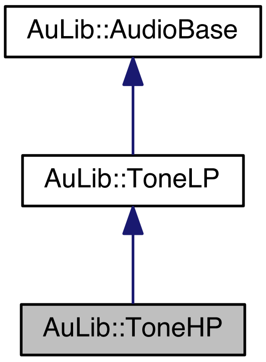
7.58 AuLib::ToneHP Class Reference

```
#include <ToneHP.h>
```

Inheritance diagram for AuLib::ToneHP:



Collaboration diagram for AuLib::ToneHP:



Public Member Functions

- ToneHP (double cf, uint32_t vframes=def_vframes, double sr=def_sr)

Protected Member Functions

- virtual void update ()

Additional Inherited Members

7.58.1 Detailed Description

Simple first-order high-pass filter

7.58.2 Constructor & Destructor Documentation

7.58.2.1 ToneHP()

```
AuLib::ToneHP::ToneHP (
             double cf,
             uint32_t vframes = def_vframes,
             double sr = def_sr )  [inline]
```

ToneHP constructor

cf - cutoff frequency
vframes - vector size
sr - sampling rate

7.58.3 Member Function Documentation

7.58.3.1 update()

```
void AuLib::ToneHP::update ( )  [protected], [virtual]
```

Update filter coefficients

Reimplemented from AuLib::ToneLP.

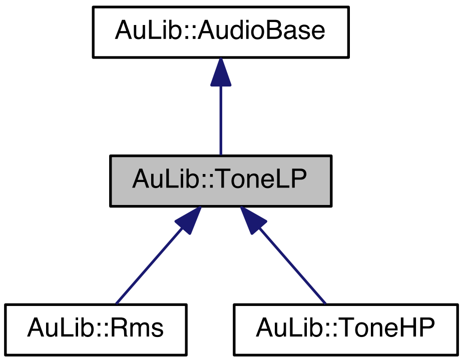
The documentation for this class was generated from the following files:

- ToneHP.h
- Tone.cpp

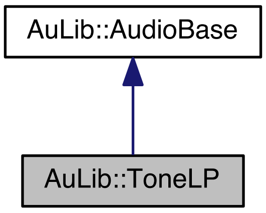
7.59 AuLib::ToneLP Class Reference

```
#include <AuLib/ToneLP.h>
```

Inheritance diagram for AuLib::ToneLP:



Collaboration diagram for AuLib::ToneLP:



Public Member Functions

- ToneLP (double cf, uint32_t vframes=def_vframes, double sr=def_sr)
- const double ∗ process (const double ∗sig)
- const double ∗ process (const double ∗sig, double cf)
- const ToneLP & process (const AudioBase &obj)
- const ToneLP & process (const AudioBase &obj, double cf)
- const ToneLP & operator() (const AudioBase &obj)
- const ToneLP & operator() (const AudioBase &obj, double cf)

Protected Member Functions

- virtual void update ()
- virtual const double ∗ dsp (const double ∗sig)

Protected Attributes

- double m_freq
- double m_del
- double m_a
- double m_b

Additional Inherited Members

7.59.1 Detailed Description

Simple first-order low-pass filter

7.59.2 Constructor & Destructor Documentation

7.59.2.1 ToneLP()

```
AuLib::ToneLP::ToneLP (
            double cf,
            uint32_t vframes = def_vframes,
            double sr = def_sr )  [inline]
```

ToneLP constructor

cf - cutoff frequency
vframes - vector size
sr - sampling rate

7.59.3 Member Function Documentation

7.59.3.1 dsp()

```
virtual const double* AuLib::ToneLP::dsp (
            const double * sig )  [inline], [protected], [virtual]
```

filter kernel

Reimplemented in AuLib::Rms.

7.59.3.2 operator()() [1/2]

```
const ToneLP& AuLib::ToneLP::operator() (
            const AudioBase & obj )  [inline]
```

operator () convenience method

7.59.3.3 operator()() [2/2]

```
const ToneLP& AuLib::ToneLP::operator() (
            const AudioBase & obj,
            double cf )  [inline]
```

operator () convenience method

7.59.3.4 process() [1/4]

```
const double* AuLib::ToneLP::process (
            const double * sig )  [inline]
```

process a signal sig

7.59.3.5 process() [2/4]

```
const double* AuLib::ToneLP::process (
            const double * sig,
            double cf )  [inline]
```

process a signal sig with cutoff freq cf

7.59.3.6 process() [3/4]

```
const ToneLP& AuLib::ToneLP::process (
            const AudioBase & obj )  [inline]
```

process a signal in obj

7.59.3.7 process() [4/4]

```
const ToneLP& AuLib::ToneLP::process (
            const AudioBase & obj,
            double cf )  [inline]
```

process a signal in obj with cutoff freq cf

7.59.3.8 update()

```
void AuLib::ToneLP::update ( )  [protected], [virtual]
```

Update filter coefficients

Reimplemented in AuLib::ToneHP.

7.59.4 Member Data Documentation

7.59.4.1 m_a

```
double AuLib::ToneLP::m_a  [protected]
```

7.59.4.2 m_b

```
double AuLib::ToneLP::m_b  [protected]
```

7.59.4.3 m_del

```
double AuLib::ToneLP::m_del  [protected]
```

7.59.4.4 m_freq

```
double AuLib::ToneLP::m_freq  [protected]
```

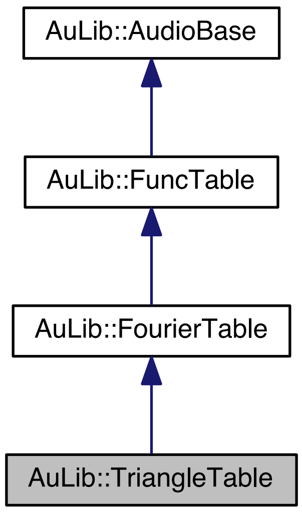
The documentation for this class was generated from the following files:

- ToneLP.h
- Tone.cpp

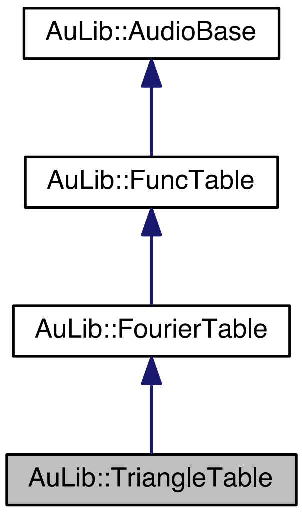
7.60 AuLib::TriangleTable Class Reference

```
#include <AuLib/WaveTables.h>
```

Inheritance diagram for AuLib::TriangleTable:



Collaboration diagram for AuLib::TriangleTable:



Public Member Functions

- TriangleTable (uint32_t harms, uint64_t tframes=def_tframes)

Additional Inherited Members

7.60.1 Detailed Description

Triangle wave table

7.60.2 Constructor & Destructor Documentation

7.60.2.1 TriangleTable()

```
AuLib::TriangleTable::TriangleTable (
            uint32_t harms,
            uint64_t tframes = def_tframes )  [inline]
```

TriangleTable constructor

harms - number of harmonics
tframes - table size

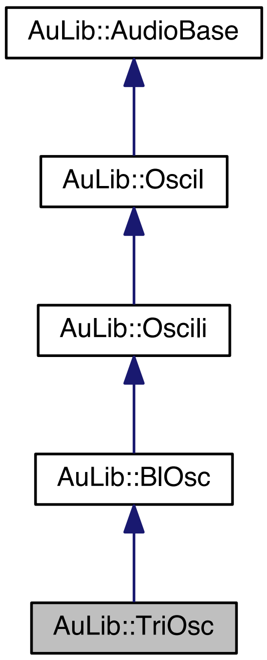
The documentation for this class was generated from the following file:

- WaveTables.h

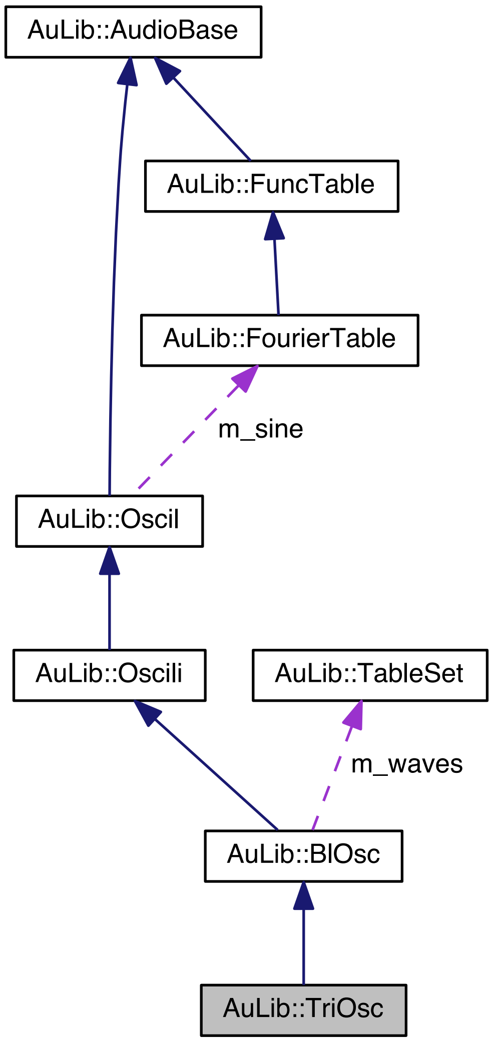
7.61 AuLib::TriOsc Struct Reference

`#include <BlOsc.h>`

Inheritance diagram for AuLib::TriOsc:



Collaboration diagram for AuLib::TriOsc:



Public Member Functions

- TriOsc (double amp=0.f, double freq=440.f, double phase=0., uint32_t vframes=def_vframes, double sr=def←_sr)

Additional Inherited Members

7.61.1 Constructor & Destructor Documentation

7.61.1.1 TriOsc()

```
AuLib::TriOsc::TriOsc (
            double amp = 0.f,
            double freq = 440.f,
            double phase = 0.,
            uint32_t vframes = def_vframes,
            double sr = def_sr )  [inline]
```

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

- BlOsc.h

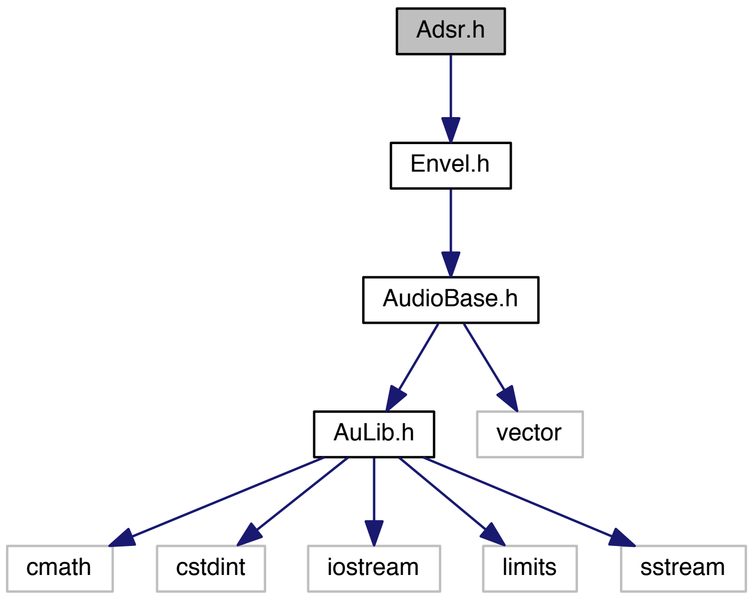
Chapter 8

File Documentation

8.1 Adsr.h File Reference

`#include "Envel.h"`

Include dependency graph for Adsr.h:



Classes

- class AuLib::Adsr

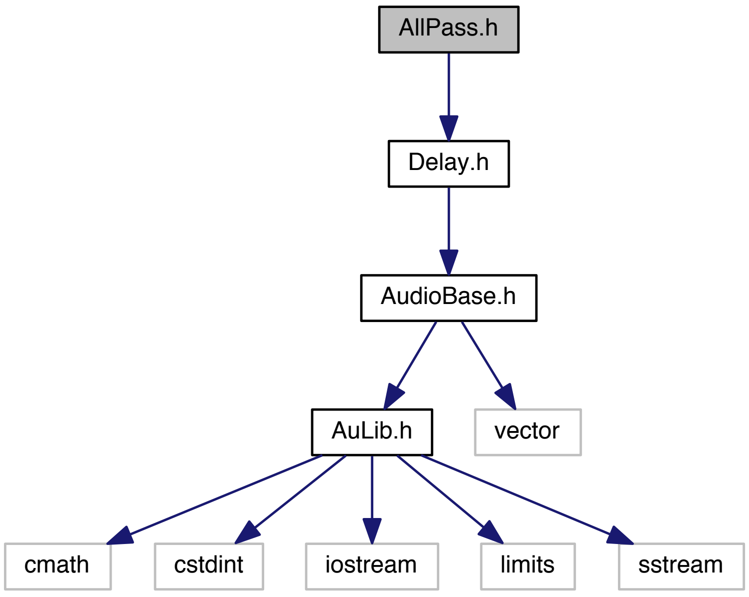
Namespaces

- AuLib

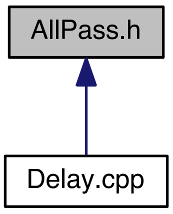
8.2 AllPass.h File Reference

```
#include "Delay.h"
```

Include dependency graph for AllPass.h:



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



Classes

- class AuLib::AllPass

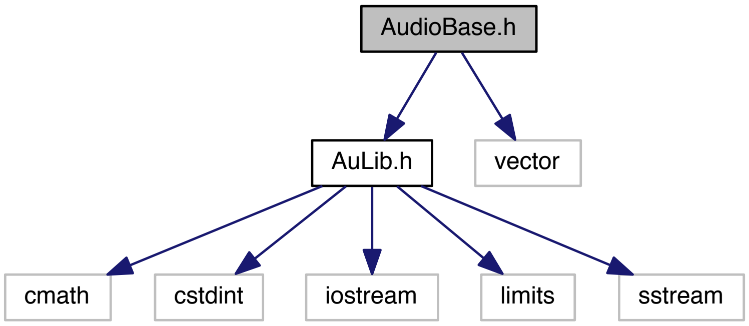
Namespaces

- AuLib

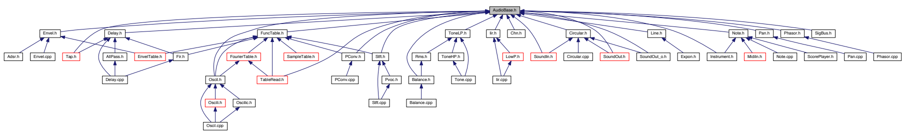
8.3 AudioBase.h File Reference

```
#include "AuLib.h"
#include <vector>
```

Include dependency graph for AudioBase.h:



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



Classes

- class AuLib::AudioBase

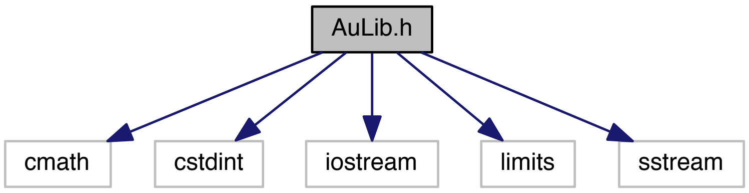
Namespaces

- AuLib

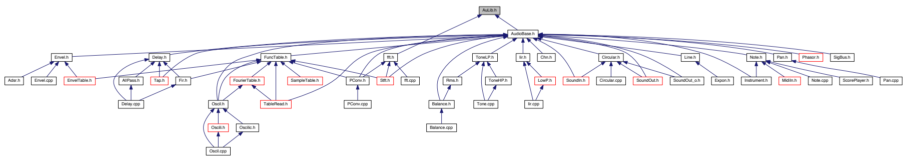
8.4 AuLib.h File Reference

```
#include <cmath>
#include <cstdint>
#include <iostream>
#include <limits>
```

`#include <sstream>`
Include dependency graph for AuLib.h:



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



Namespaces

- AuLib
- AuLib::Info
- AuLib::midi

Macros

- #define AULIB_MAJOR_V 0
- #define AULIB_MINOR_V 0
- #define BETA 1
- #define NONCOPYABLE(A)

Enumerations

- enum AuLib::error_codes { AuLib::AULIB_NOERROR = 0, AuLib::AULIB_ERROR }

Functions

- static const std::string AuLib::Info::version ()
- static const std::string AuLib::Info::copyright ()
- static uint32_t AuLib::npow2 (uint32_t n)

Variables

- const uint32_t AuLib::Info::major_version = AULIB_MAJOR_V
- const uint32_t AuLib::Info::minor_version = AULIB_MINOR_V
- const uint32_t AuLib::def_vframes = 64
- const uint32_t AuLib::def_bframes = 1024
- const double AuLib::def_sr = 44100.
- const double AuLib::def_kr = def_sr / def_vframes
- const uint32_t AuLib::def_nchnls = 1
- const uint32_t AuLib::def_tframes = 8192
- const uint32_t AuLib::def_fftsize = 1024
- const uint32_t AuLib::def_decim = 4
- const double AuLib::pi = 3.141592653589793
- const double AuLib::twopi = 6.283185307179586
- const double AuLib::db_min = std::numeric_limits<double>::min()
- const uint64_t AuLib::ui64_max = std::numeric_limits<uint64_t>::max()
- const double AuLib::m120dBfs = 0.000001
- const uint32_t AuLib::midi::note_on = 0x90
- const uint32_t AuLib::midi::note_off = 0x80
- const uint32_t AuLib::midi::ctrl_msg = 0xB0
- const uint32_t AuLib::midi::aftouch = 0xD0
- const uint32_t AuLib::midi::poly_aftouch = 0xA0
- const uint32_t AuLib::midi::prg_msg = 0xC0
- const uint32_t AuLib::midi::pitchbend = 0xE0
- const std::string AuLib::aulib_error [ ]

8.4.1 Macro Definition Documentation

8.4.1.1 AULIB_MAJOR_V

```
#define AULIB_MAJOR_V 0
```

8.4.1.2 AULIB_MINOR_V

```
#define AULIB_MINOR_V 0
```

8.4.1.3 BETA

```
#define BETA 1
```

8.4.1.4 NONCOPYABLE

```
#define NONCOPYABLE(
                A )
```

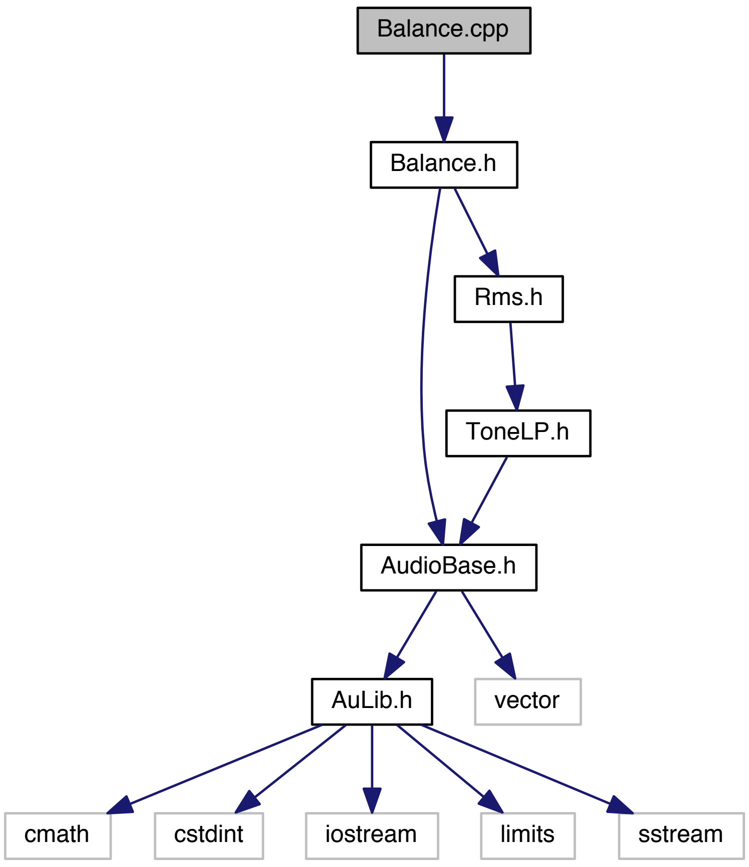
Value:

```
\
public:                                                                    \
  A(const A &) = delete;                                                   \
  A &operator=(A) = delete
```

8.5 Balance.cpp File Reference

`#include "Balance.h"`

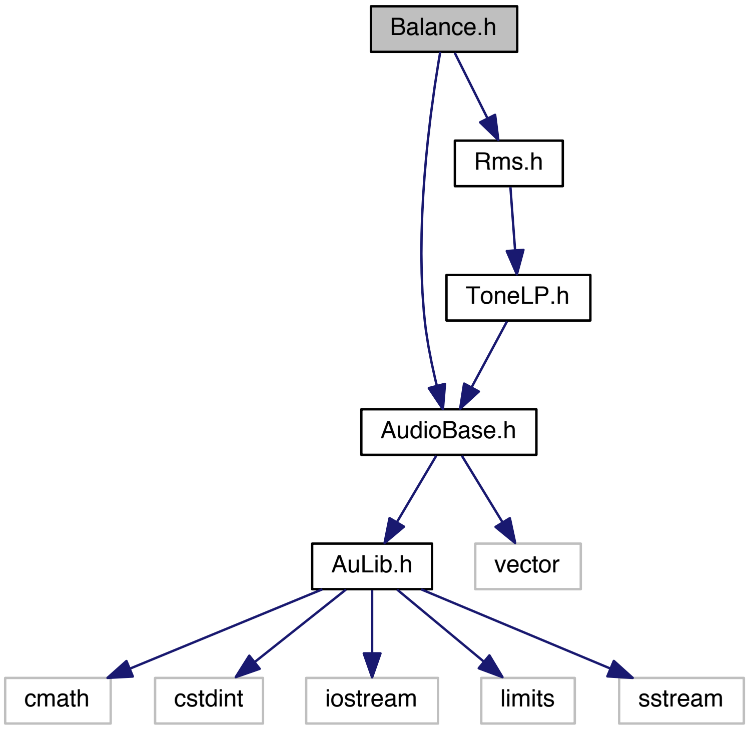
Include dependency graph for Balance.cpp:



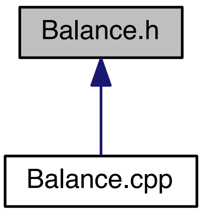
8.6 Balance.h File Reference

```
#include "AudioBase.h"
#include "Rms.h"
```

Include dependency graph for Balance.h:



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



Classes

- class AuLib::Balance

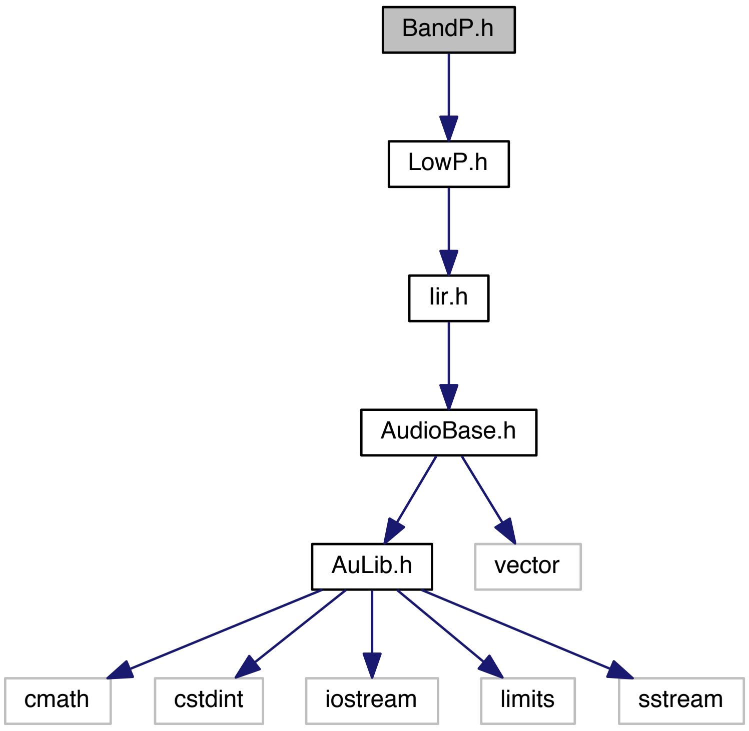
Namespaces

- AuLib

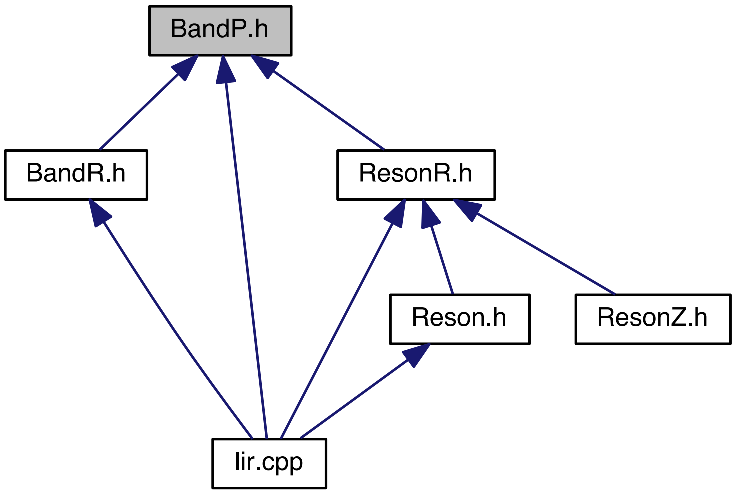
8.7 BandP.h File Reference

```
#include "LowP.h"
```

Include dependency graph for BandP.h:



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



Classes

- class AuLib::BandP

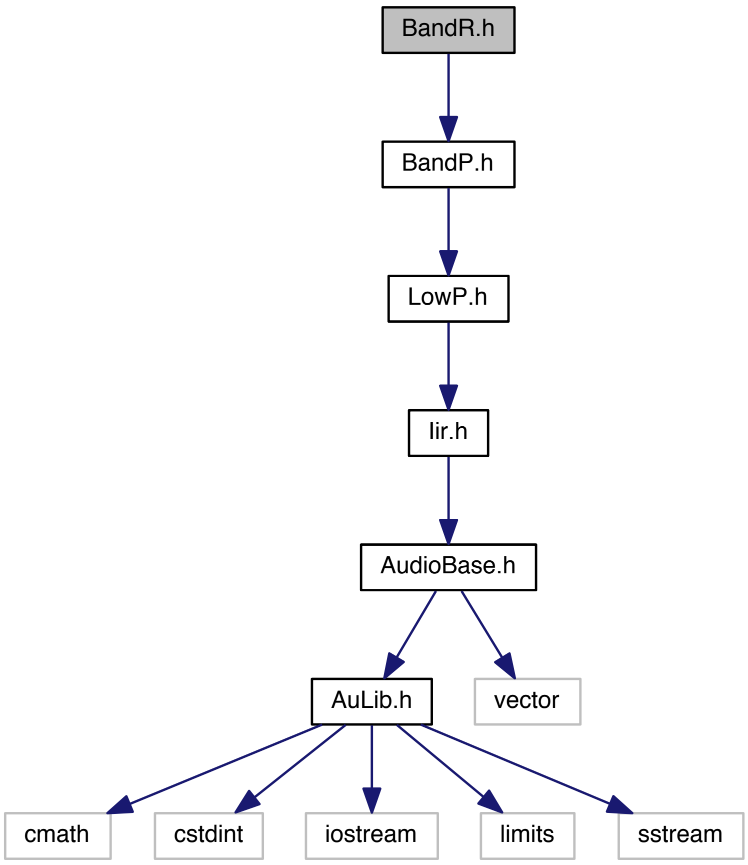
Namespaces

- AuLib

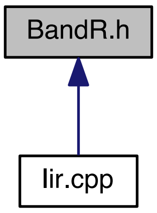
8.8 BandR.h File Reference

```
#include "BandP.h"
```

Include dependency graph for BandR.h:



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



Classes

- class AuLib::BandR

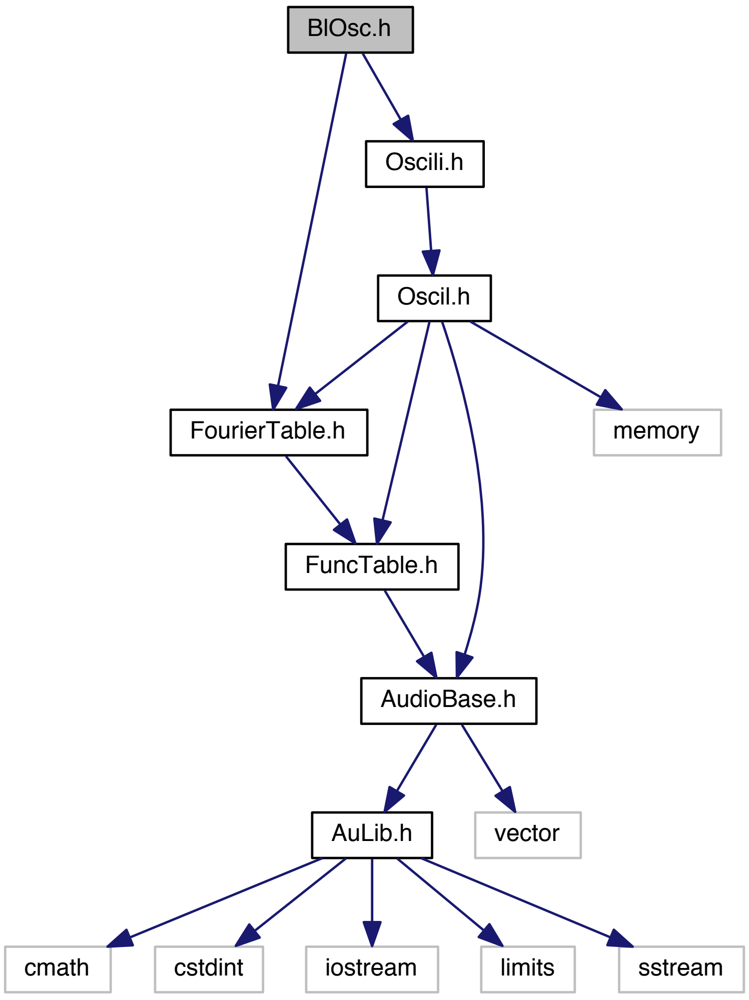
Namespaces

- AuLib

8.9 BlOsc.h File Reference

```
#include "FourierTable.h"
#include "Oscili.h"
```

Include dependency graph for BlOsc.h:



Classes

- class AuLib::TableSet
- class AuLib::BlOsc
- struct AuLib::SqOsc
- struct AuLib::TriOsc
- struct AuLib::SawOsc

Namespaces

- AuLib
- AuLib::waveset

Functions

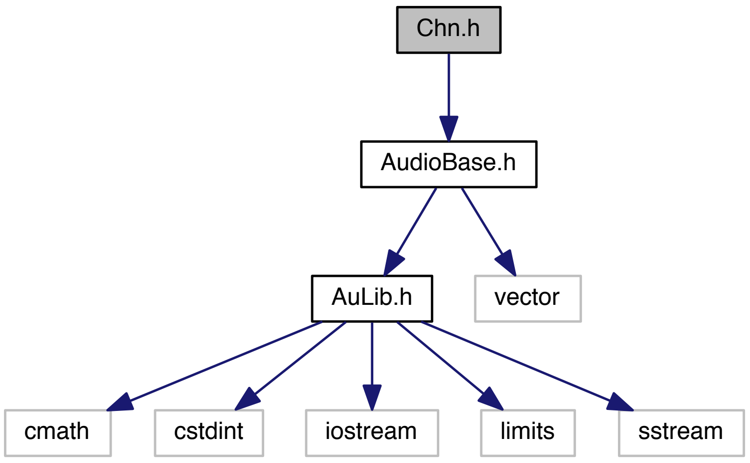
- static const TableSet AuLib::waveset::saw (SAW)
- static const TableSet AuLib::waveset::triangle (TRIANGLE)
- static const TableSet AuLib::waveset::square (SQUARE)

Variables

- const int32_t AuLib::octs = 10
- const double AuLib::base = 20

8.10 Chn.h File Reference

```
#include "AudioBase.h"
```
Include dependency graph for Chn.h:



Classes

- class AuLib::Chn

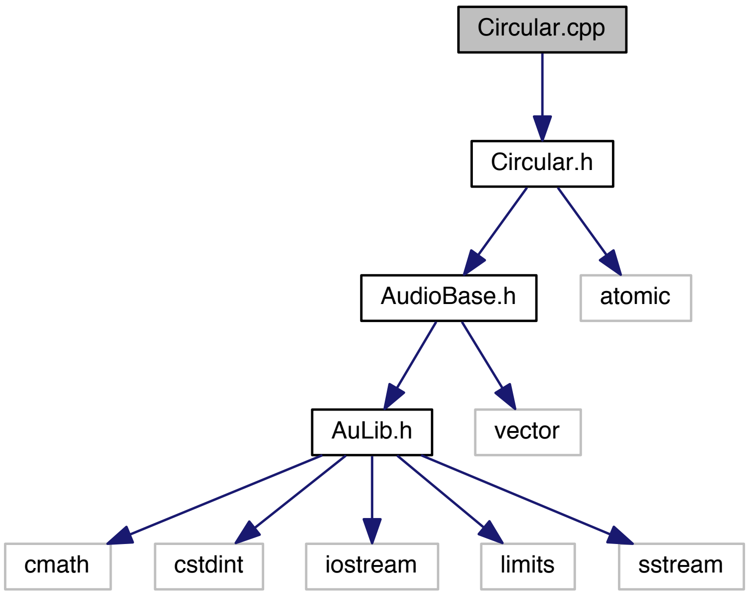
Namespaces

- AuLib

8.11 Circular.cpp File Reference

```
#include "Circular.h"
```

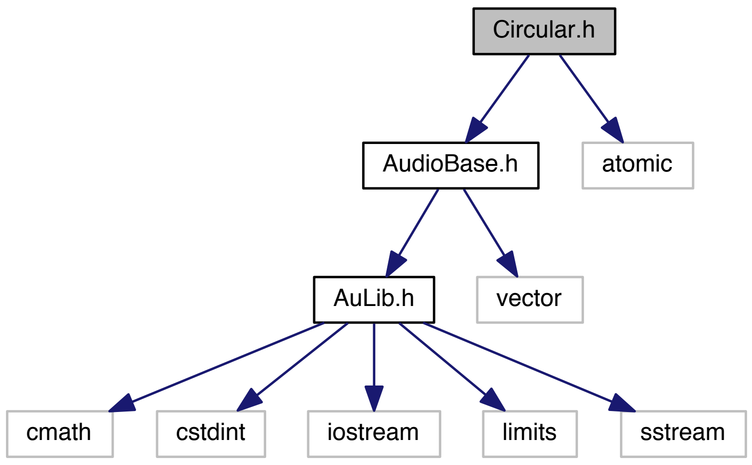
Include dependency graph for Circular.cpp:



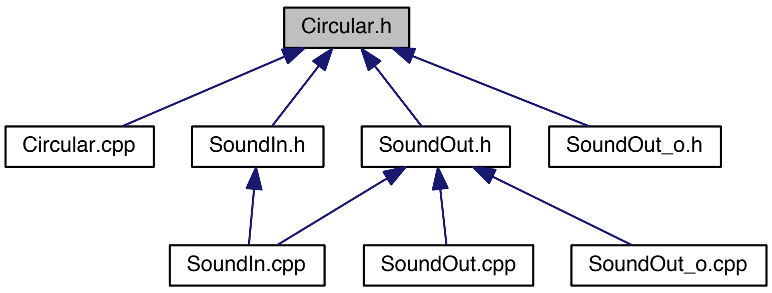
8.12 Circular.h File Reference

```
#include <AudioBase.h>
#include <atomic>
```

Include dependency graph for Circular.h:



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



Classes

- class AuLib::Circular

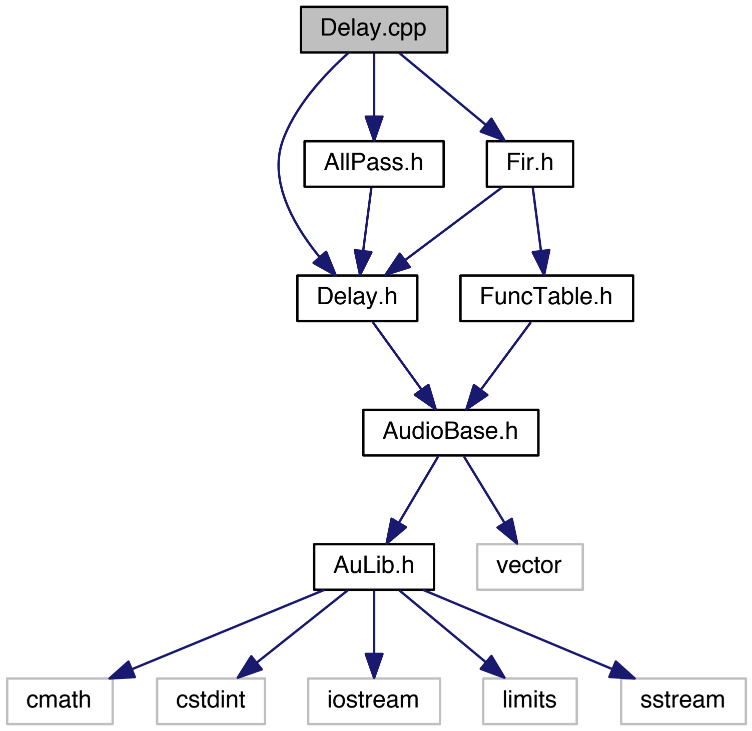
Namespaces

- AuLib

8.13 Delay.cpp File Reference

```
#include "Delay.h"
#include "AllPass.h"
#include "Fir.h"
```

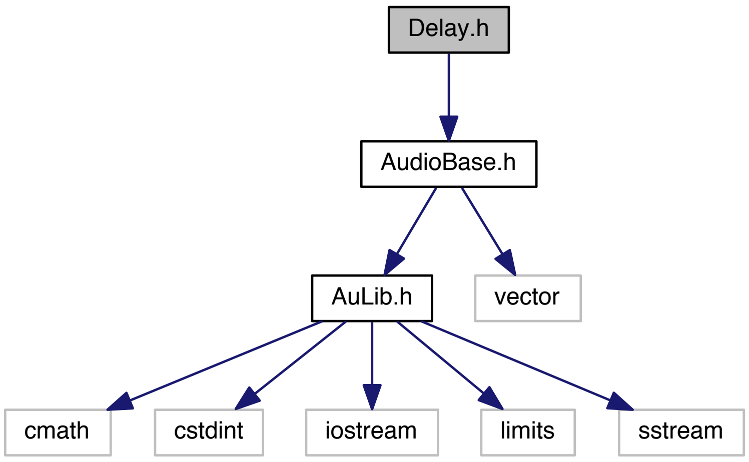
Include dependency graph for Delay.cpp:



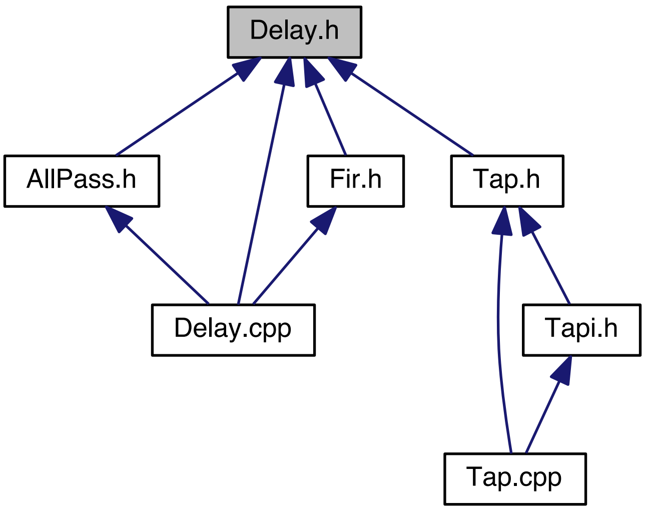
8.14 Delay.h File Reference

```
#include "AudioBase.h"
```

Include dependency graph for Delay.h:



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



Classes

- class AuLib::Delay

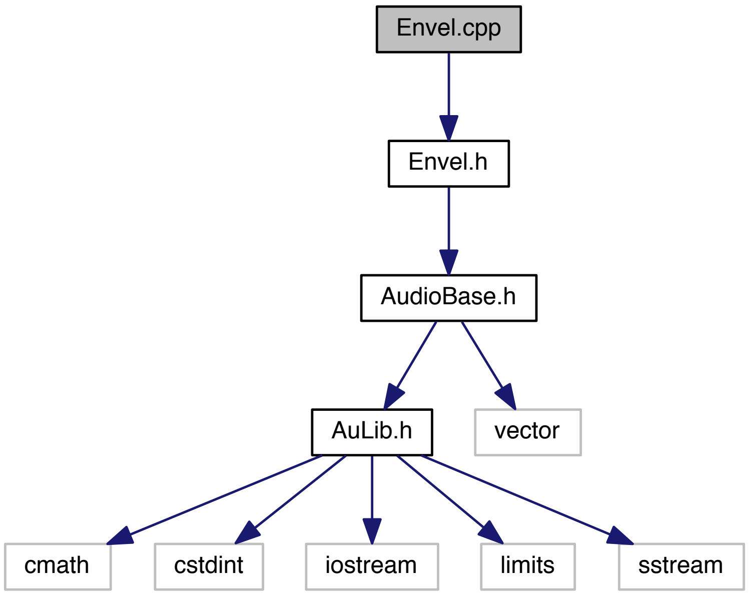
Namespaces

- AuLib

8.15 Envel.cpp File Reference

```
#include "Envel.h"
```

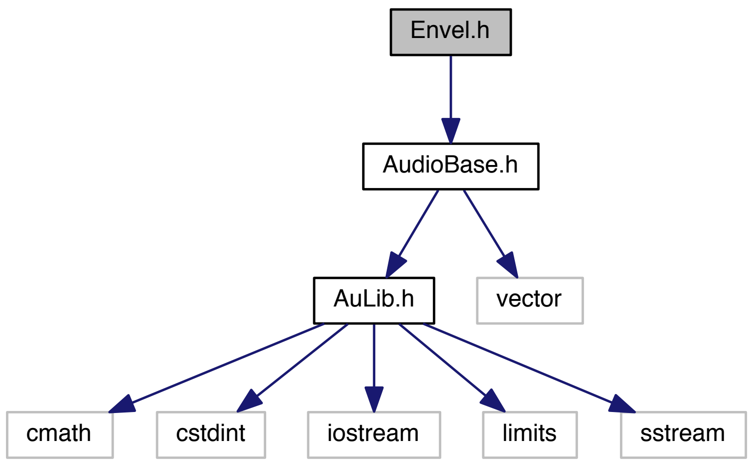
Include dependency graph for Envel.cpp:



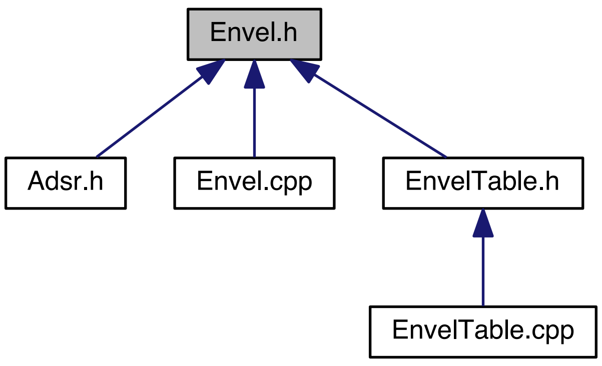
8.16 Envel.h File Reference

```
#include "AudioBase.h"
```

Include dependency graph for Envel.h:



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



Classes

- class AuLib::Segments
- class AuLib::Envel

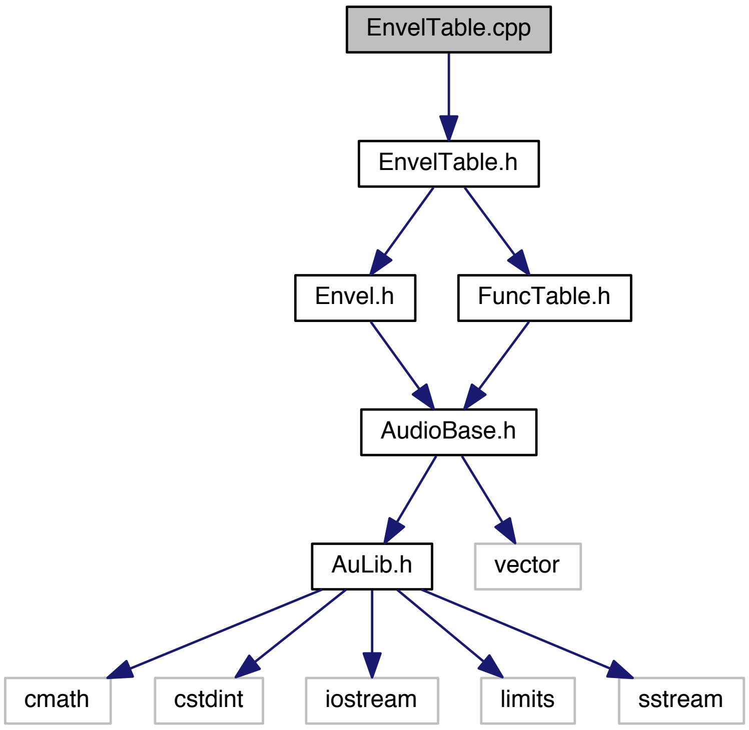
Namespaces

- AuLib

8.17 EnvelTable.cpp File Reference

```
#include "EnvelTable.h"
```

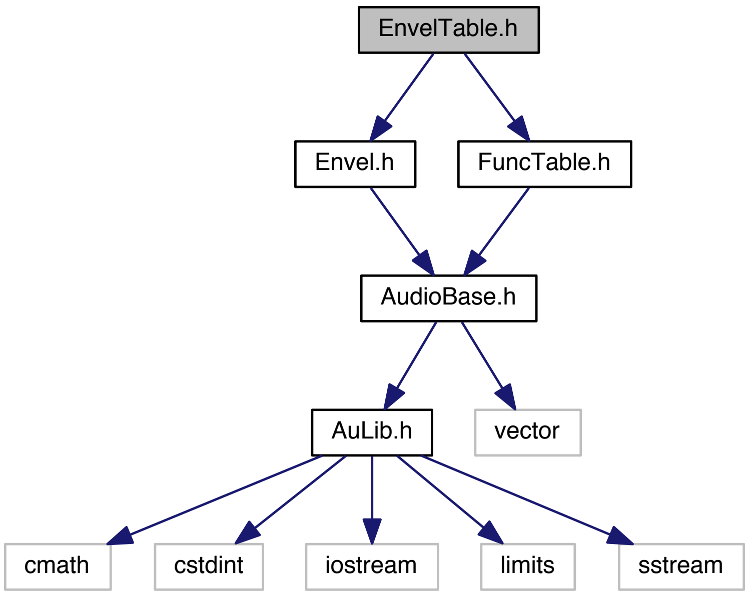
Include dependency graph for EnvelTable.cpp:



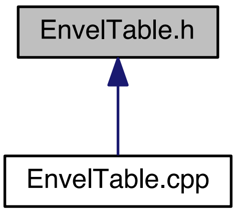
8.18 EnvelTable.h File Reference

```
#include "Envel.h"
#include "FuncTable.h"
```

Include dependency graph for EnvelTable.h:



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



Classes

- class AuLib::EnvelTable

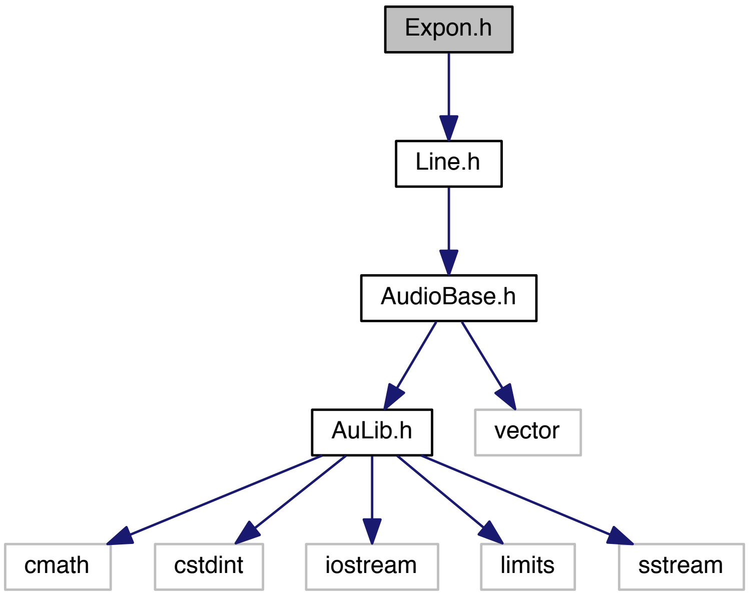
Namespaces

- AuLib

8.19 Expon.h File Reference

```
#include "Line.h"
```

Include dependency graph for Expon.h:



Classes

- class AuLib::Expon

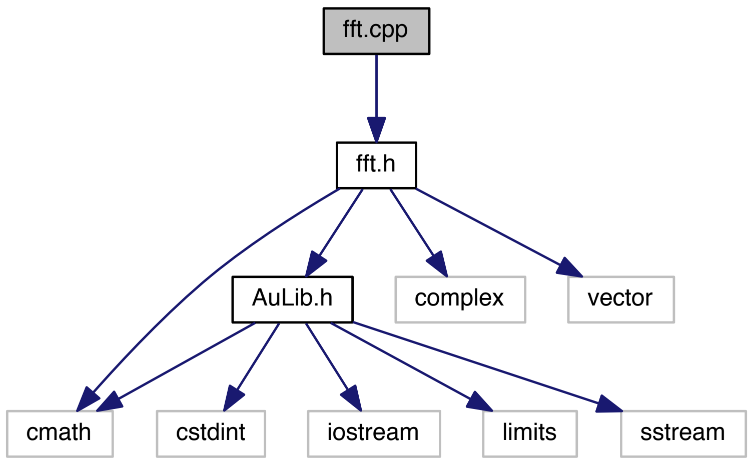
Namespaces

- AuLib

8.20 fft.cpp File Reference

```
#include "fft.h"
```

Include dependency graph for fft.cpp:



Functions

- static void reorder (std::vector< std::complex< double >> &s)

8.20.1 Function Documentation

8.20.1.1 reorder()

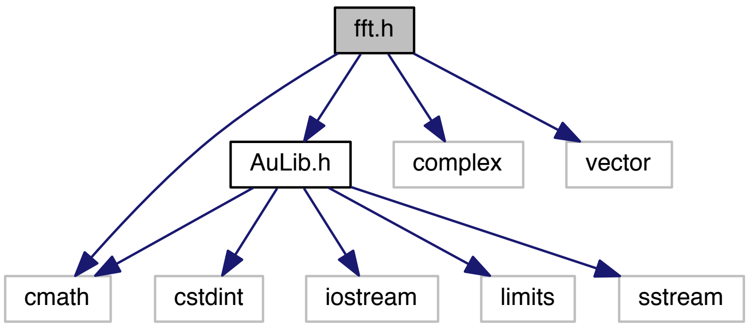
```
static void reorder (
            std::vector< std::complex< double >> & s )  [static]
```

8.21 fft.h File Reference

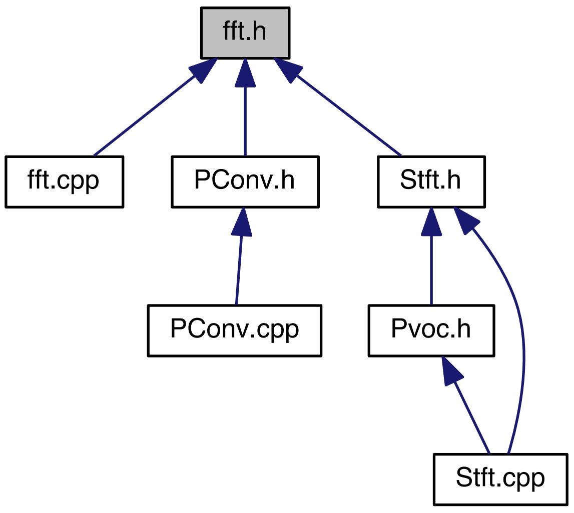
```
#include "AuLib.h"
#include <cmath>
#include <complex>
```

```
#include <vector>
```

Include dependency graph for fft.h:



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



Namespaces

- AuLib
- AuLib::fft

Functions

- void AuLib::fft::transform (std::vector< std::complex< double >> &data, bool dir)
- void AuLib::fft::transform (std::vector< std::complex< double >> &out, double ∗in, bool pckd=true)
- void AuLib::fft::transform (double ∗out, std::vector< std::complex< double >> &in, bool pckd=true)

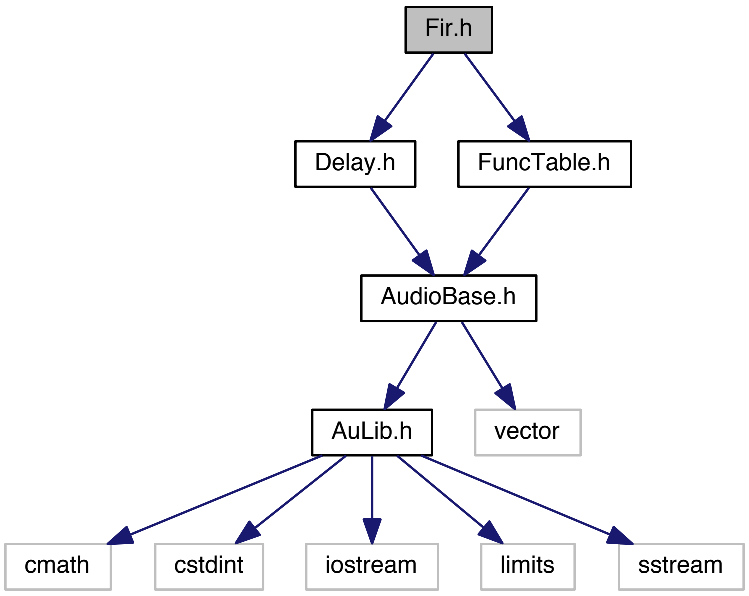
Variables

- const bool AuLib::fft::forward = true
- const bool AuLib::fft::inverse = false
- const bool AuLib::fft::polar = true
- const bool AuLib::fft::rectang = false
- const bool AuLib::fft::packed = true

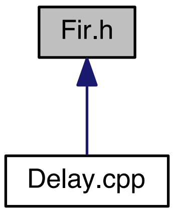
8.22 Fir.h File Reference

```
#include "Delay.h"
#include "FuncTable.h"
```

Include dependency graph for Fir.h:



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



Classes

- class AuLib::Fir

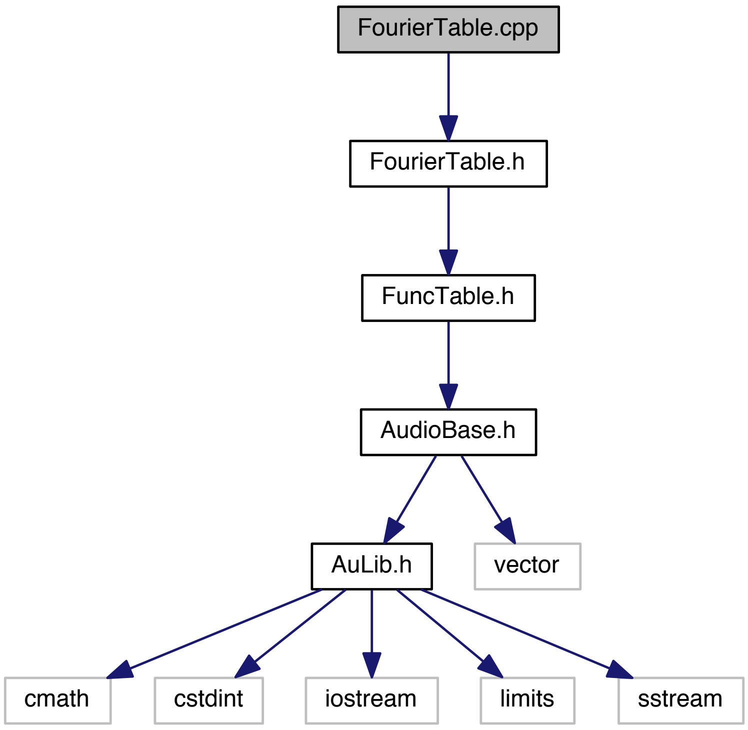
Namespaces

- AuLib

8.23 FourierTable.cpp File Reference

```
#include "FourierTable.h"
```

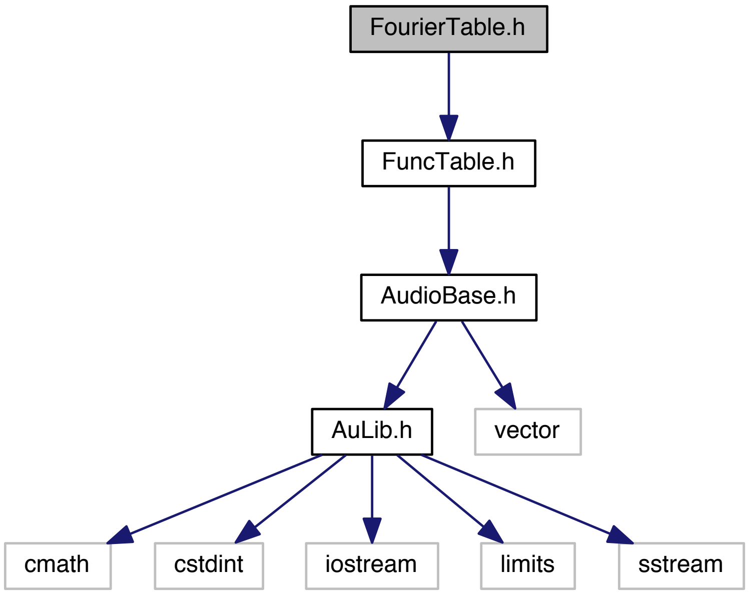
Include dependency graph for FourierTable.cpp:



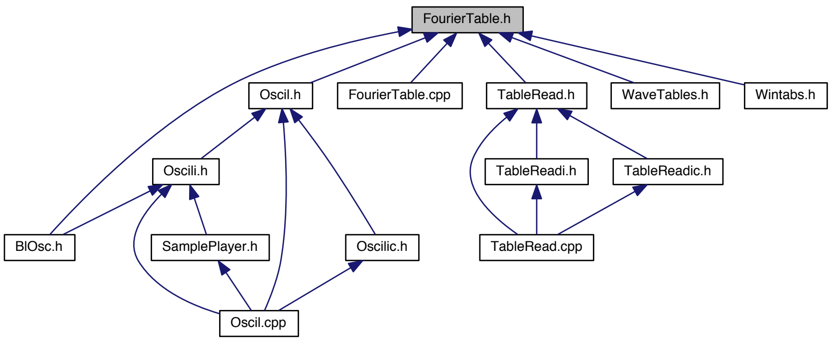
8.24 FourierTable.h File Reference

```
#include "FuncTable.h"
```

Include dependency graph for FourierTable.h:



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



Classes

- class AuLib::FourierTable

Namespaces

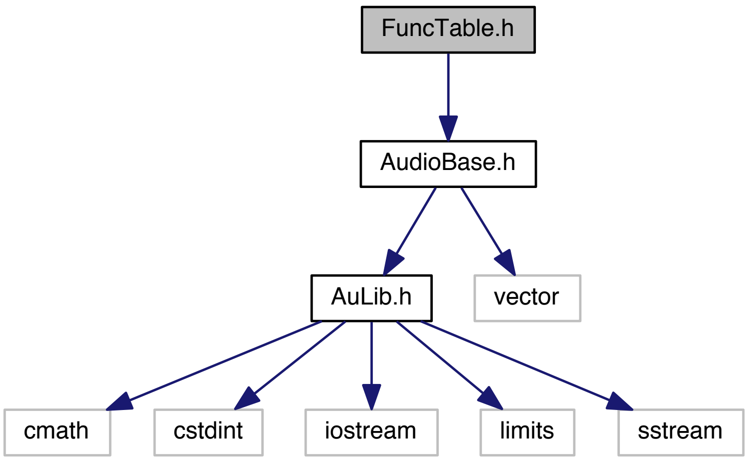
- AuLib

Enumerations

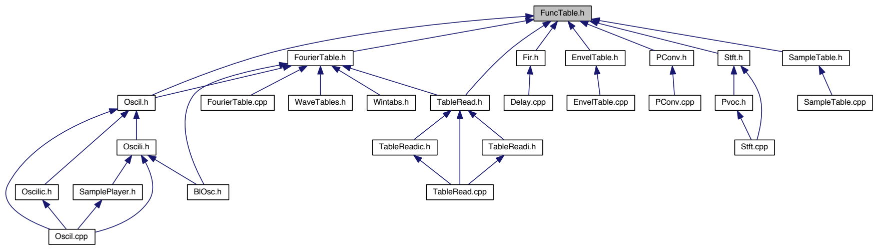
- enum AuLib::wave_types { AuLib::SAW = 1, AuLib::SQUARE, AuLib::TRIANGLE }

8.25 FuncTable.h File Reference

`#include "AudioBase.h"`
Include dependency graph for FuncTable.h:



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



Classes

- class AuLib::FuncTable

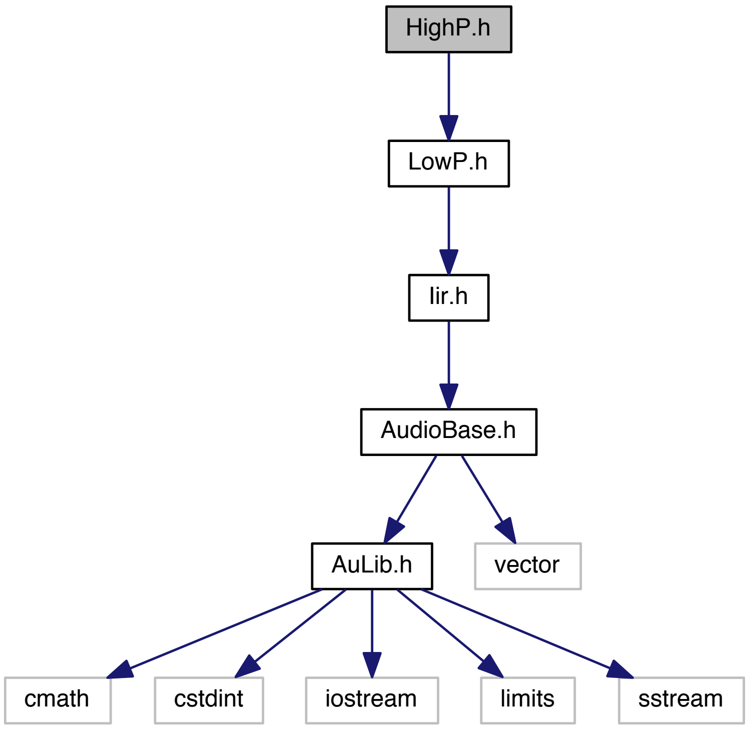
Namespaces

- AuLib

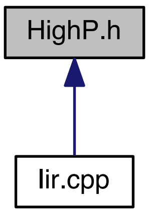
8.26 HighP.h File Reference

```
#include "LowP.h"
```

Include dependency graph for HighP.h:



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



Classes

- class AuLib::HighP

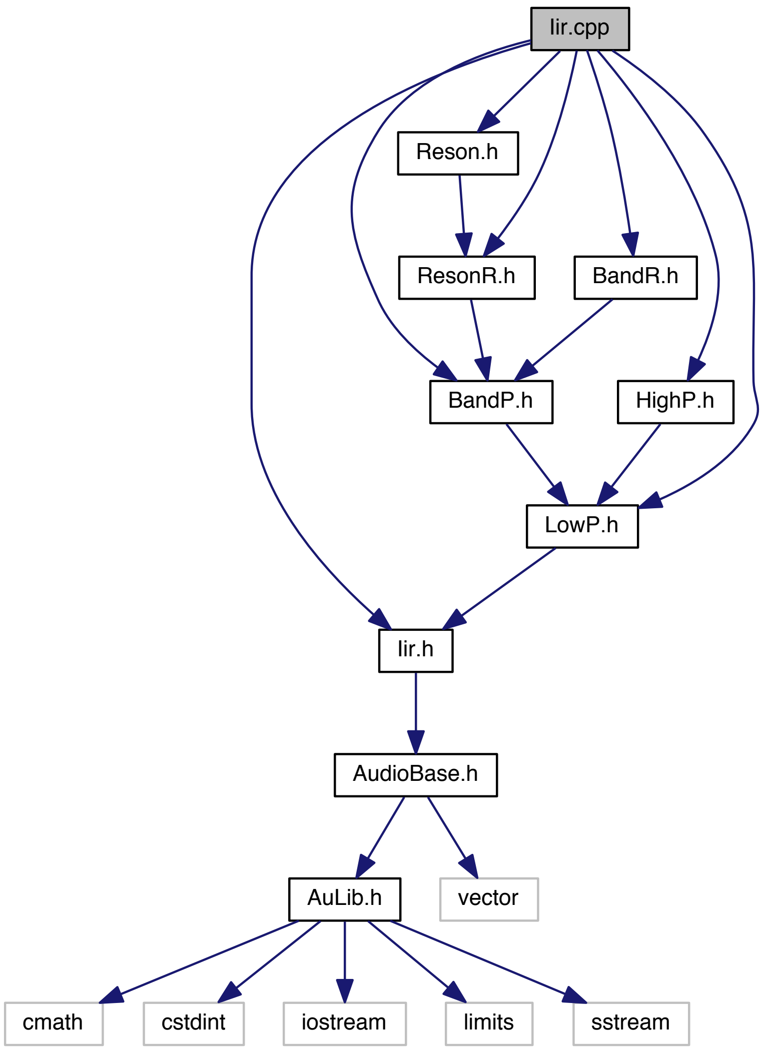
Namespaces

- AuLib

8.27 Iir.cpp File Reference

```
#include "Iir.h"
#include "BandP.h"
#include "BandR.h"
#include "HighP.h"
#include "LowP.h"
#include "Reson.h"
#include "ResonR.h"
```

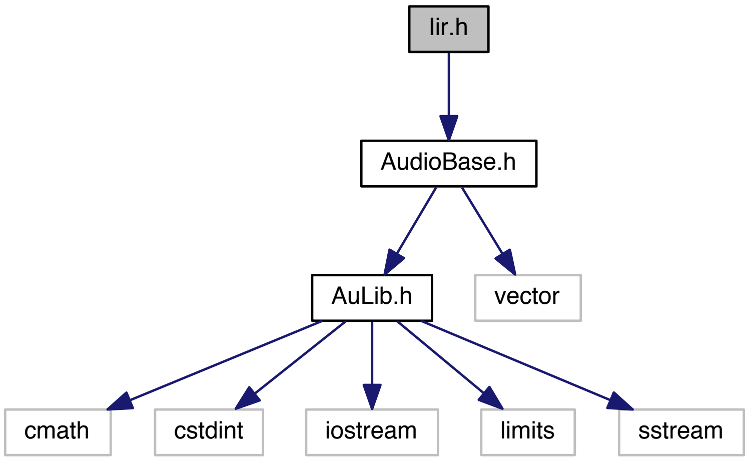
Include dependency graph for Iir.cpp:



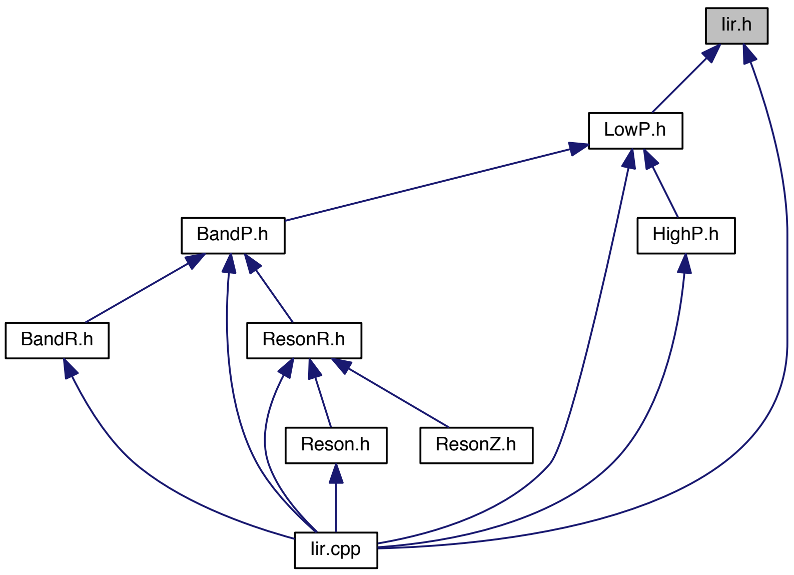
8.28 Iir.h File Reference

```
#include "AudioBase.h"
```

Include dependency graph for Iir.h:



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



Classes

- class AuLib::Iir

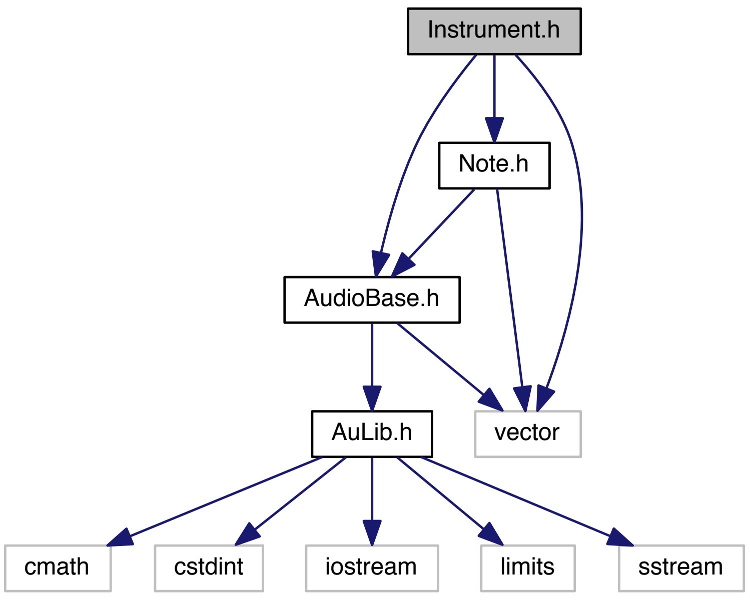
Namespaces

- AuLib

8.29 Instrument.h File Reference

```
#include <AudioBase.h>
#include <Note.h>
#include <vector>
```

Include dependency graph for Instrument.h:



Classes

- class AuLib::Instrument< T, Targs >

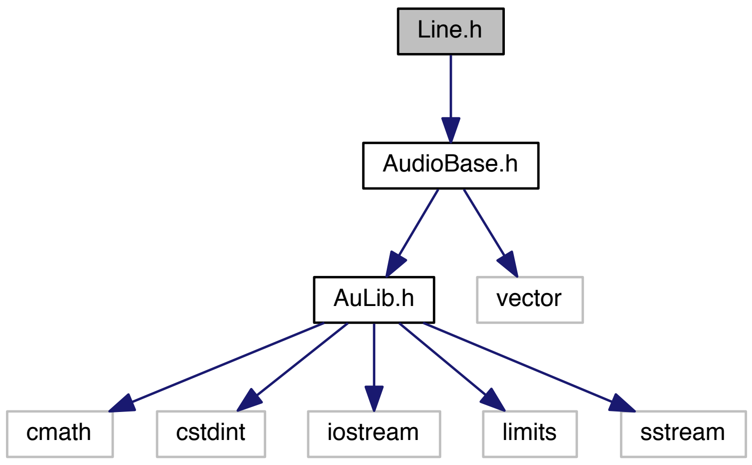
Namespaces

- AuLib

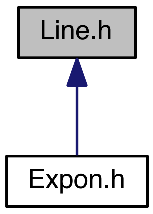
8.30 Line.h File Reference

```
#include "AudioBase.h"
```

Include dependency graph for Line.h:



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



Classes

- class AuLib::Line

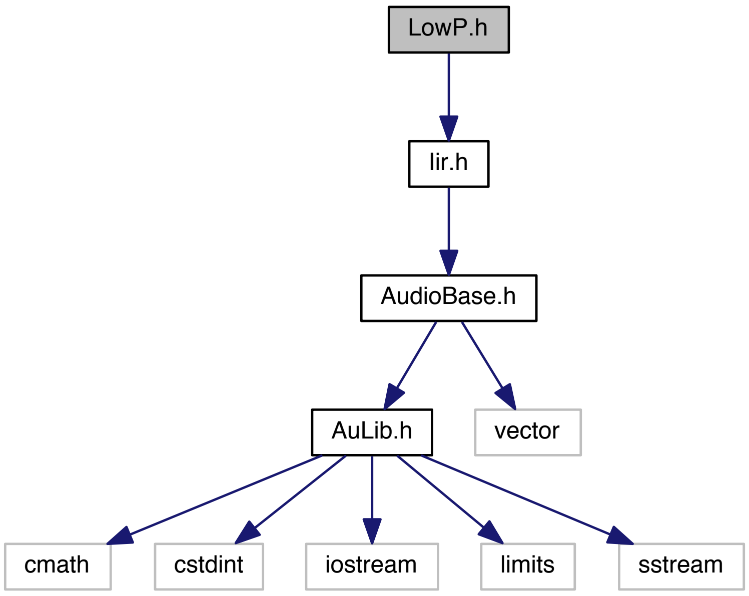
Namespaces

- AuLib

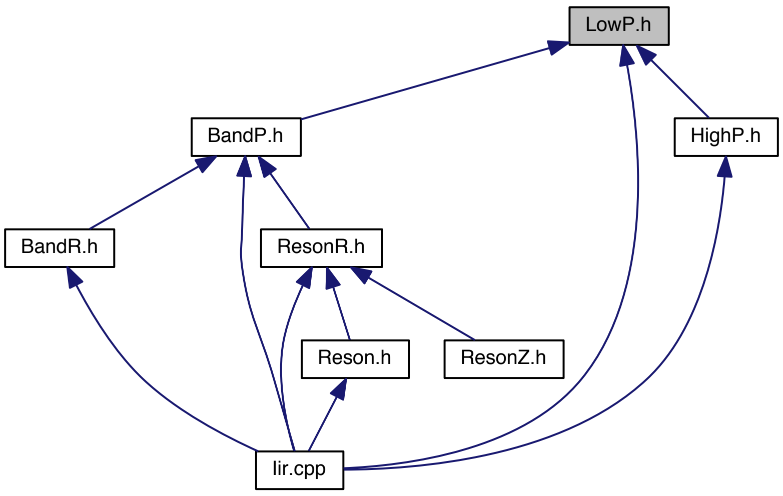
8.31 LowP.h File Reference

```
#include "Iir.h"
```

Include dependency graph for LowP.h:



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



Classes

- class AuLib::LowP

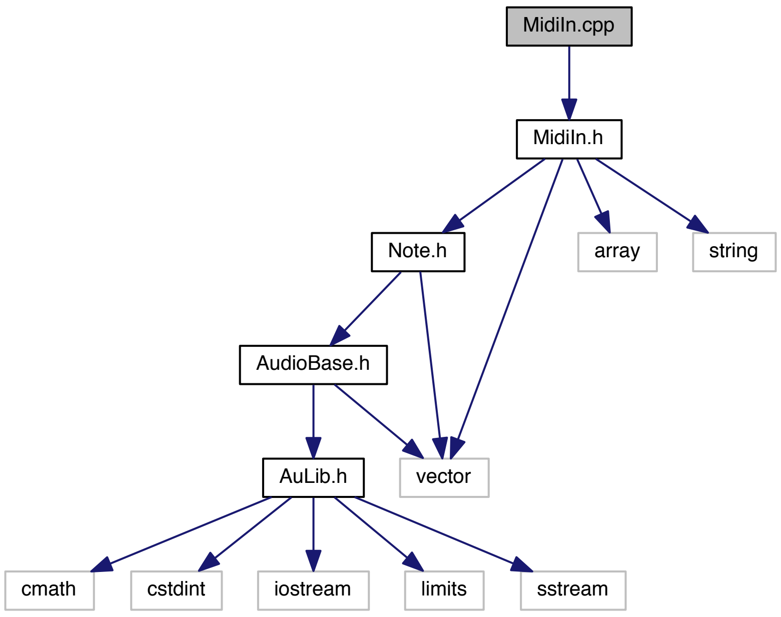
Namespaces

- AuLib

8.32 MidiIn.cpp File Reference

```
#include <MidiIn.h>
```

Include dependency graph for MidiIn.cpp:

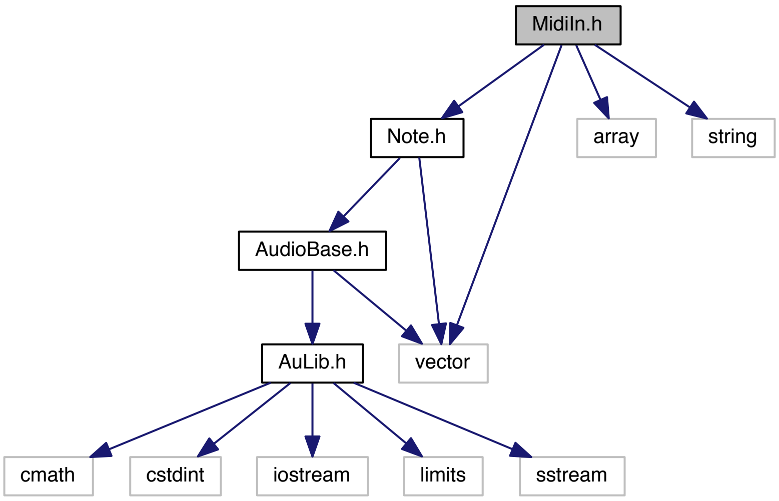


8.33 MidiIn.h File Reference

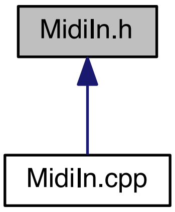
```
#include <Note.h>
#include <array>
#include <string>
```

`#include <vector>`

Include dependency graph for MidiIn.h:



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



Classes

- class AuLib::MidiIn

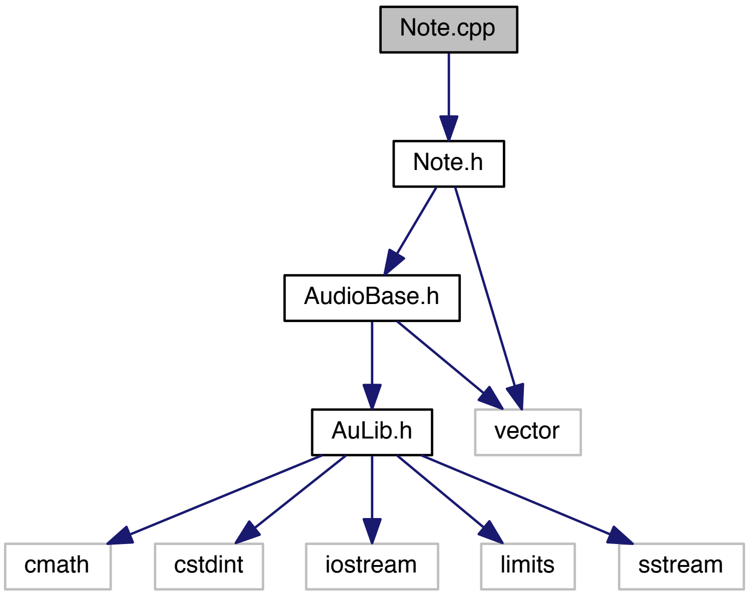
Namespaces

- AuLib

8.34 Note.cpp File Reference

```
#include <Note.h>
```

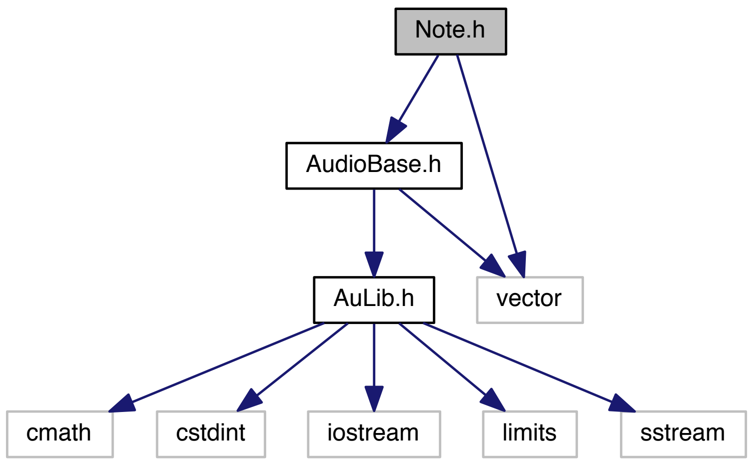
Include dependency graph for Note.cpp:



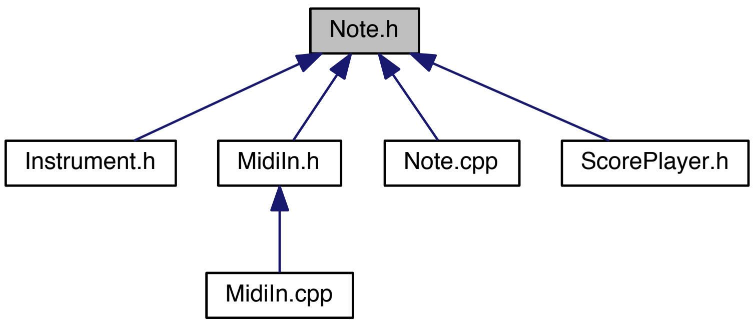
8.35 Note.h File Reference

```
#include <AudioBase.h>
#include <vector>
```

Include dependency graph for Note.h:



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



Classes

- class AuLib::Note

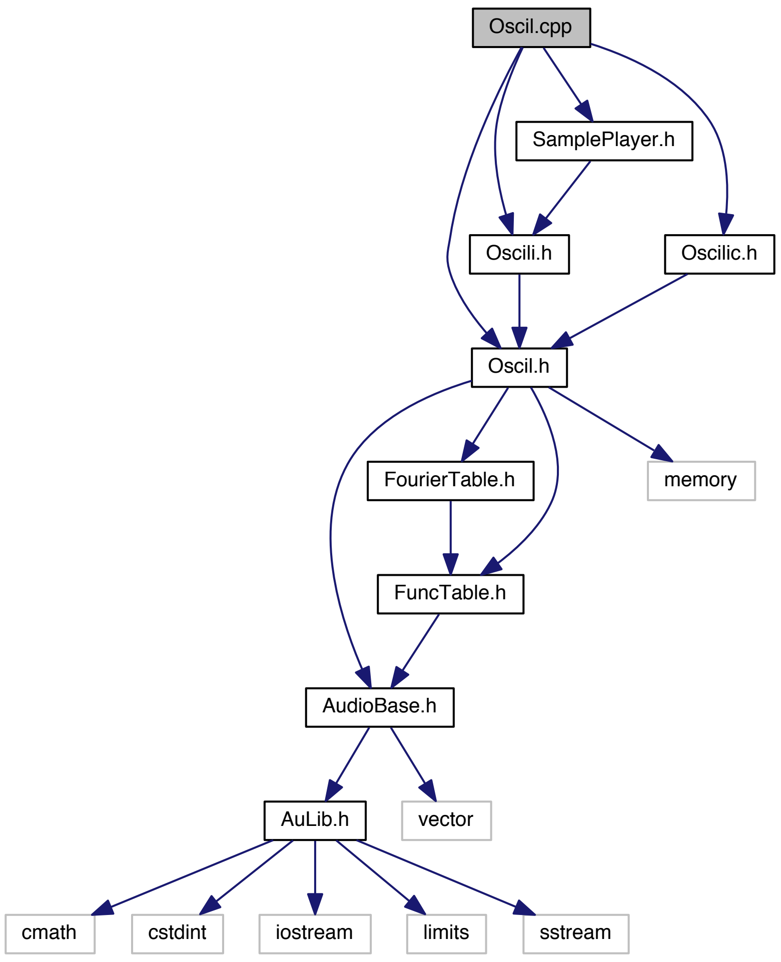
Namespaces

- AuLib

8.36 Oscil.cpp File Reference

```
#include "Oscil.h"
#include "Oscili.h"
#include "Oscilic.h"
#include "SamplePlayer.h"
```

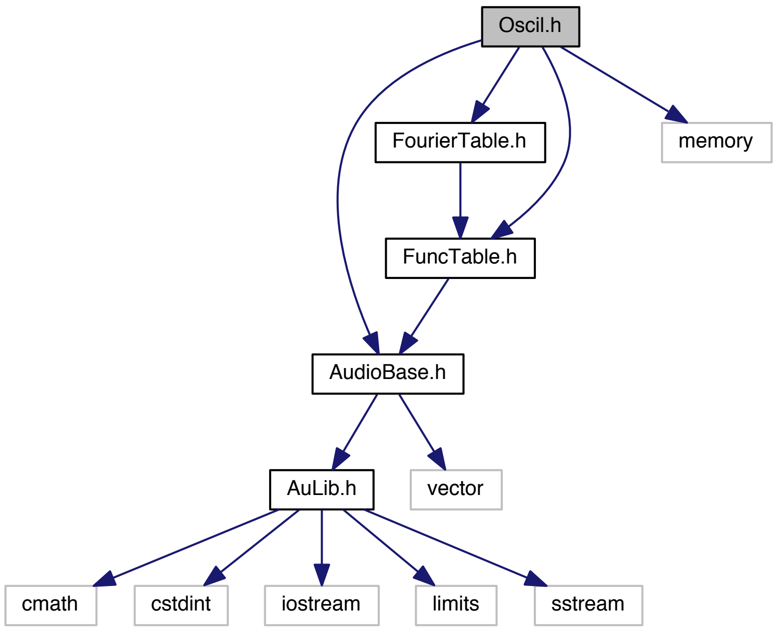
Include dependency graph for Oscil.cpp:



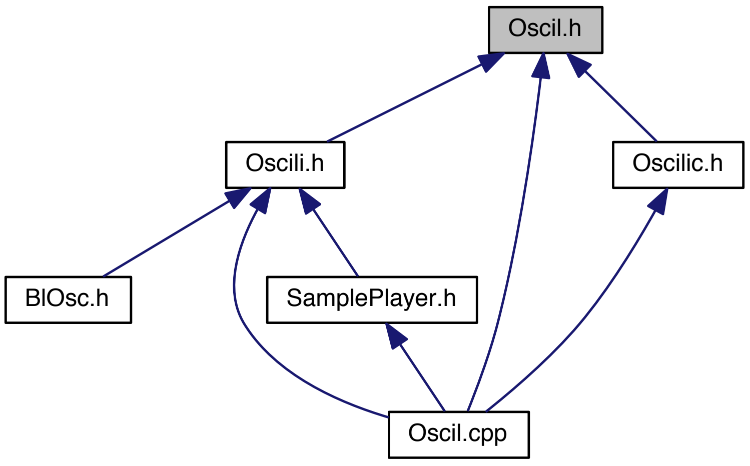
8.37 Oscil.h File Reference

```
#include "AudioBase.h"
#include "FourierTable.h"
#include "FuncTable.h"
#include <memory>
```

Include dependency graph for Oscil.h:



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



Classes

- class AuLib::Oscil

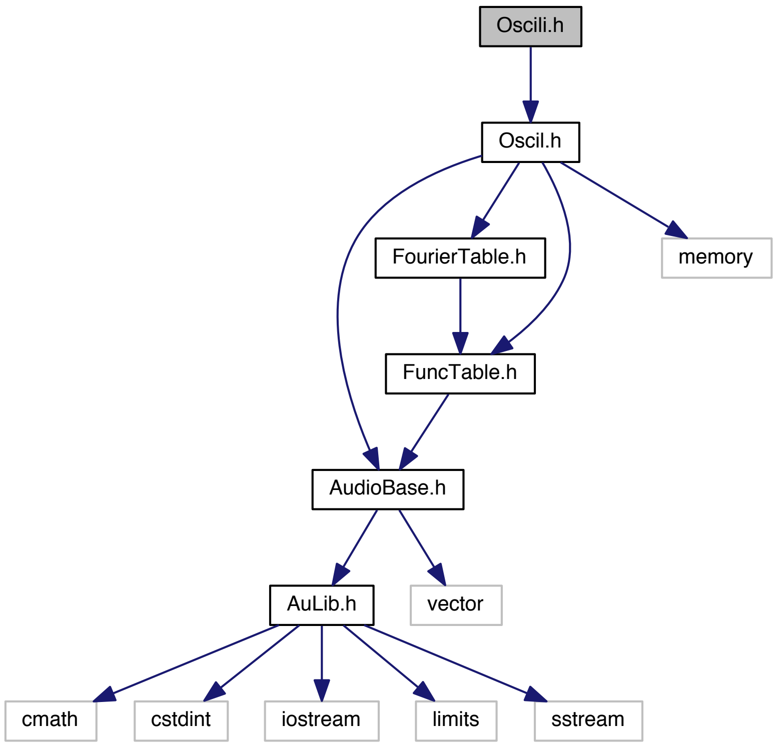
Namespaces

- AuLib

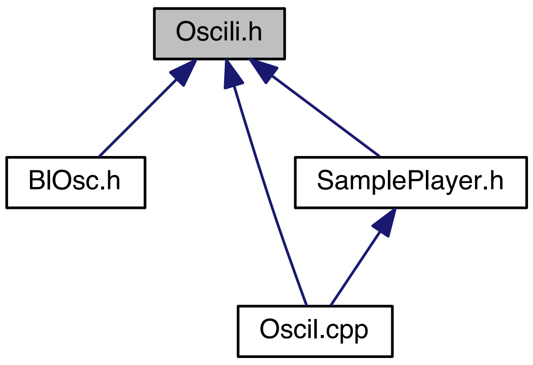
8.38 Oscili.h File Reference

```
#include "Oscil.h"
```

Include dependency graph for Oscili.h:



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



Classes

- class AuLib::Oscili

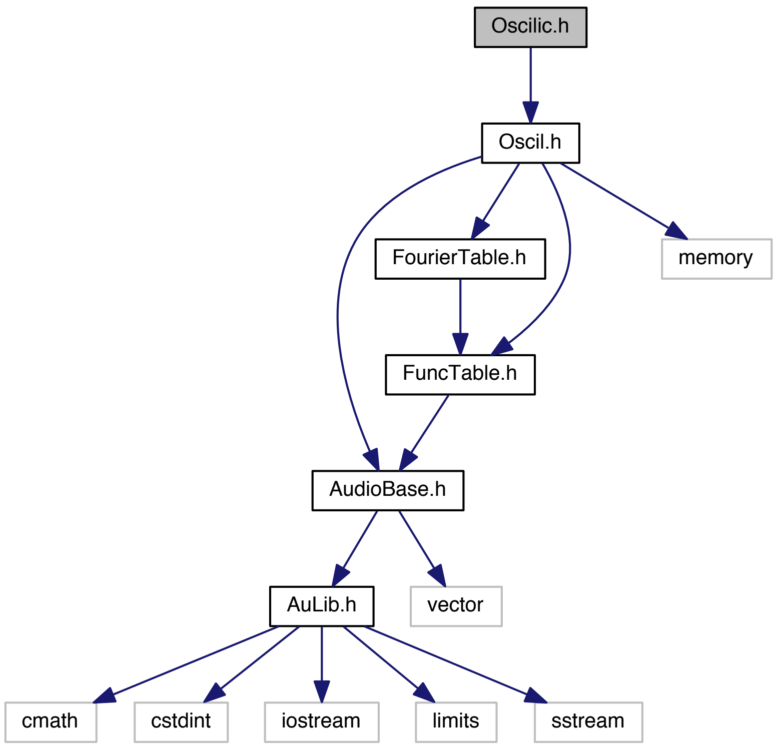
Namespaces

- AuLib

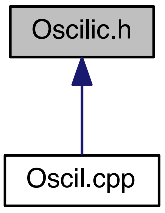
8.39 Oscilic.h File Reference

```
#include "Oscil.h"
```

Include dependency graph for Oscilic.h:



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



Classes

- class AuLib::Oscilic

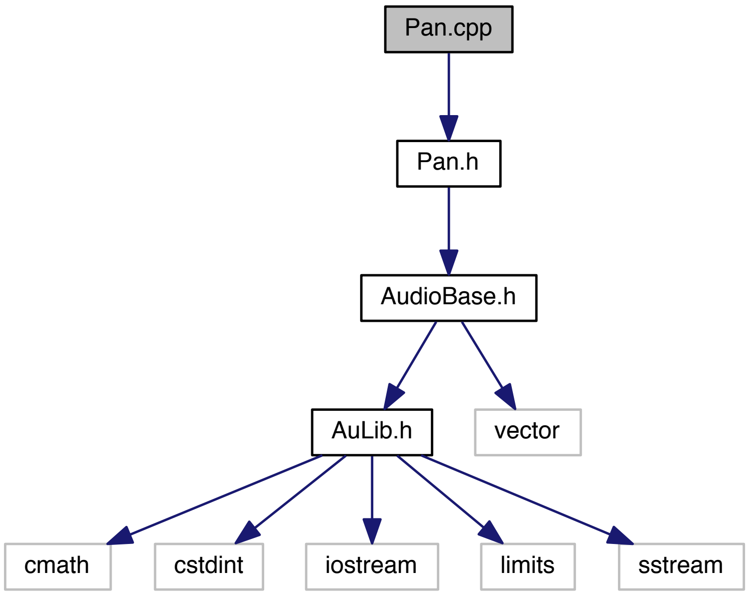
Namespaces

- AuLib

8.40 Pan.cpp File Reference

```
#include "Pan.h"
```

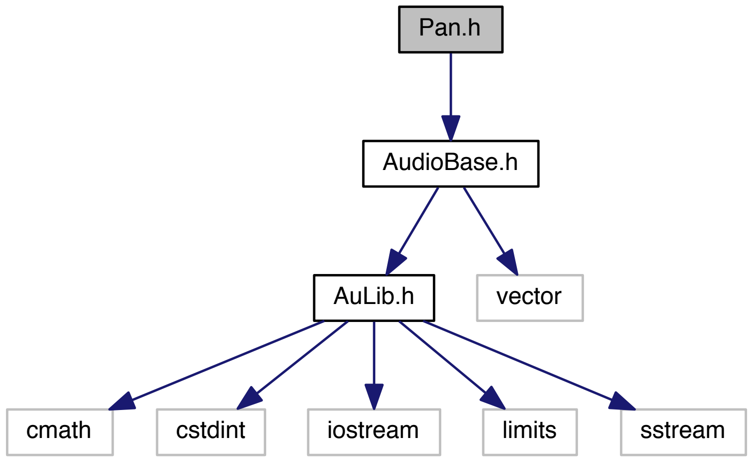
Include dependency graph for Pan.cpp:



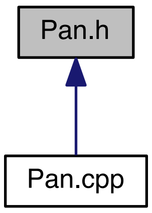
8.41 Pan.h File Reference

```
#include "AudioBase.h"
```

Include dependency graph for Pan.h:



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



Classes

- class AuLib::Pan

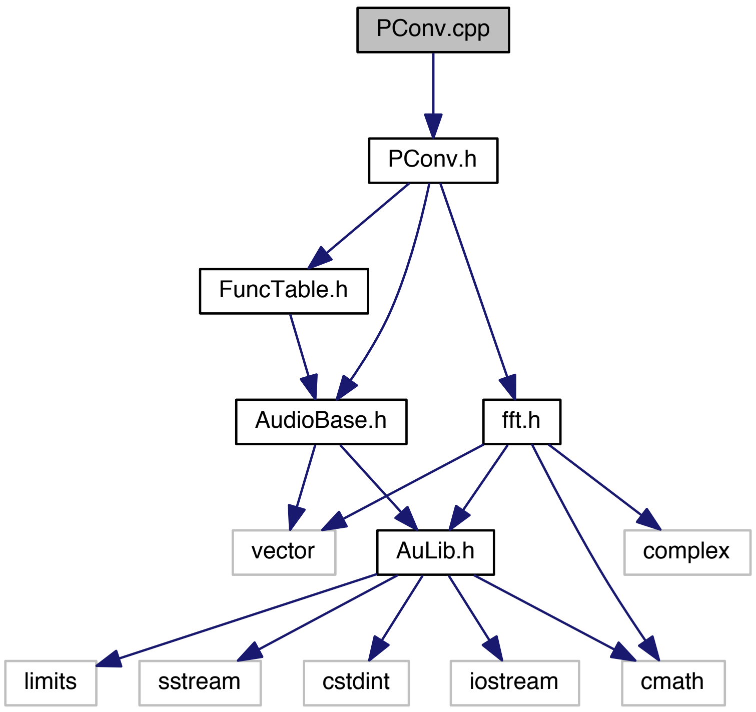
Namespaces

- AuLib

8.42 PConv.cpp File Reference

```
#include <PConv.h>
```

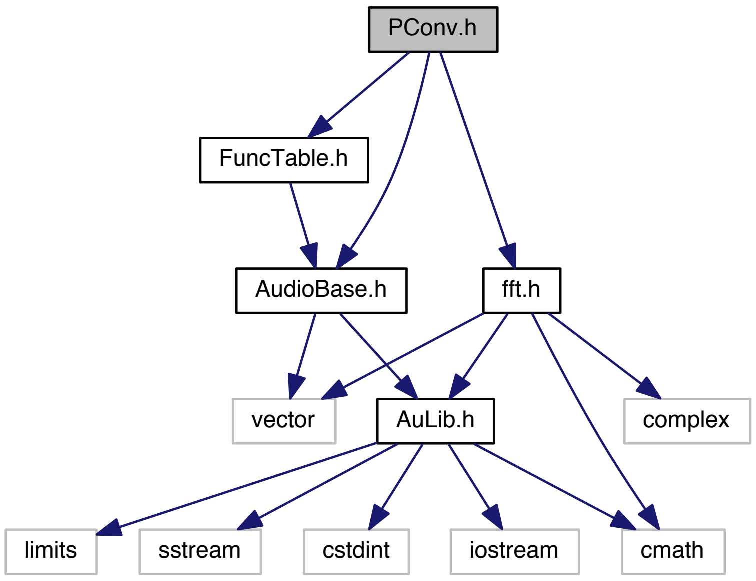
Include dependency graph for PConv.cpp:



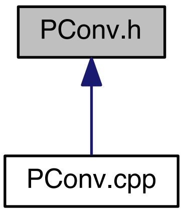
8.43 PConv.h File Reference

```
#include <AudioBase.h>
#include <FuncTable.h>
#include <fft.h>
```

Include dependency graph for PConv.h:



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



Classes

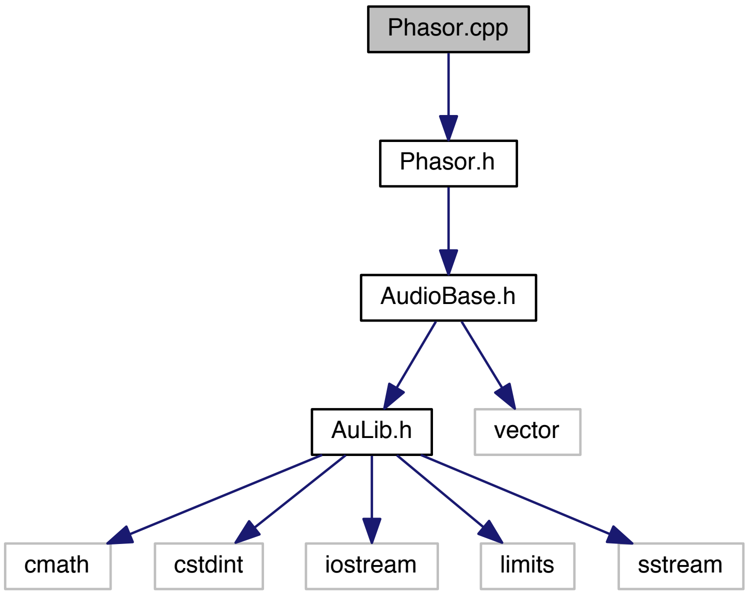
- class AuLib::PConv

Namespaces

- AuLib

8.44 Phasor.cpp File Reference

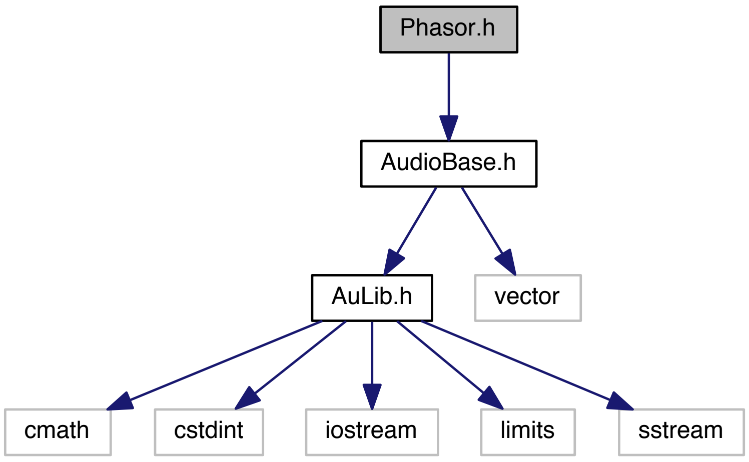
```
#include "Phasor.h"
```
Include dependency graph for Phasor.cpp:



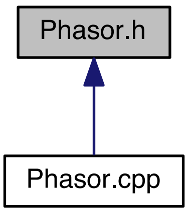
8.45 Phasor.h File Reference

```
#include "AudioBase.h"
```

Include dependency graph for Phasor.h:



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



Classes

- class AuLib::Phasor

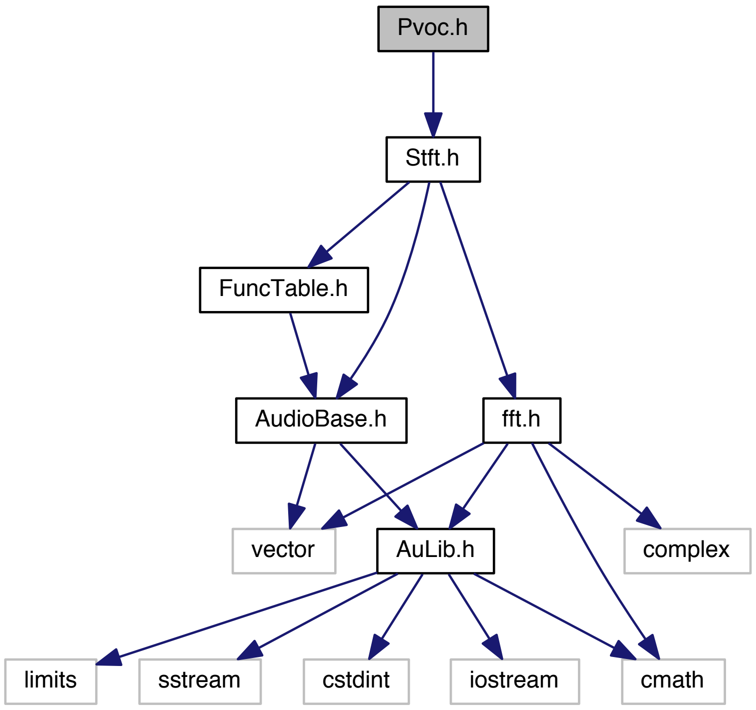
Namespaces

- AuLib

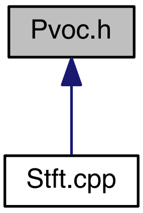
8.46 Pvoc.h File Reference

`#include "Stft.h"`

Include dependency graph for Pvoc.h:



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



Classes

- class AuLib::Pvoc

Namespaces

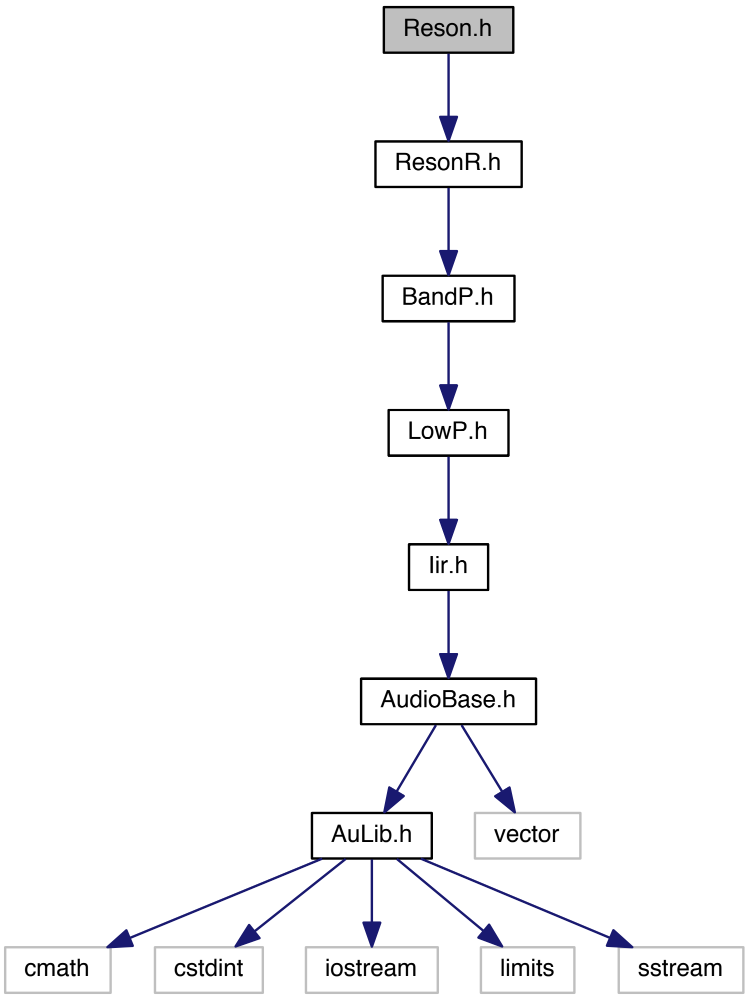
- AuLib

8.47 README.md File Reference

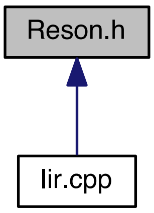
8.48 Reson.h File Reference

`#include "ResonR.h"`

Include dependency graph for Reson.h:



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



Classes

- class AuLib::Reson

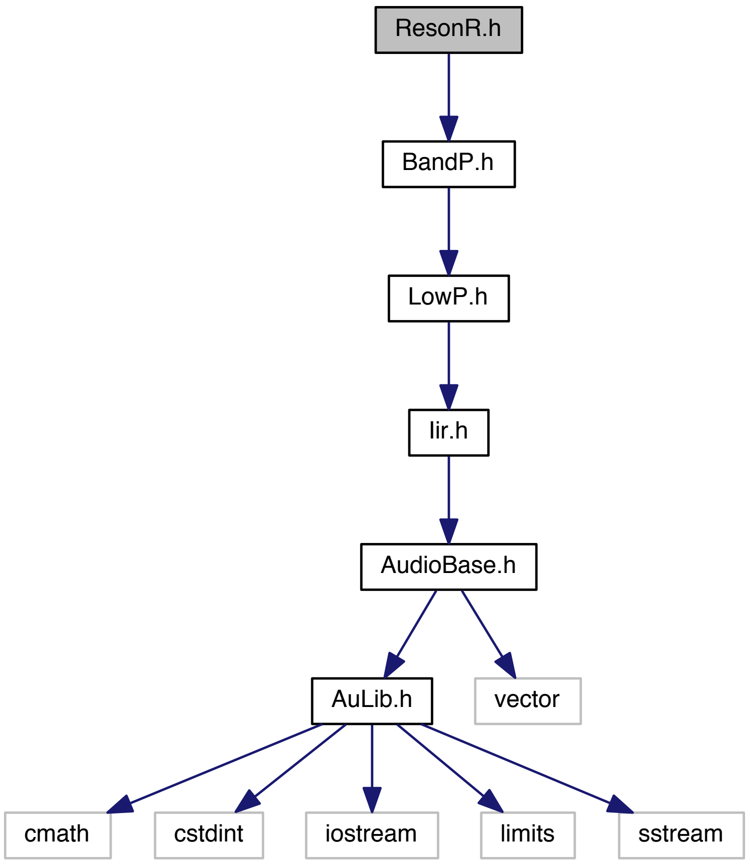
Namespaces

- AuLib

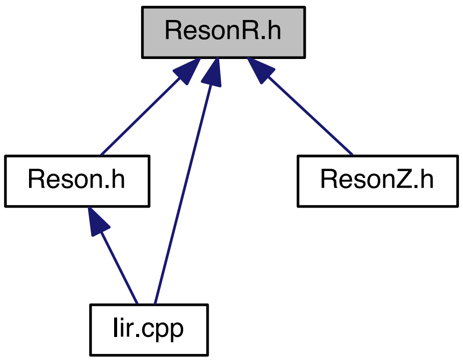
8.49 ResonR.h File Reference

```
#include "BandP.h"
```

Include dependency graph for ResonR.h:



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



Classes

- class AuLib::ResonR

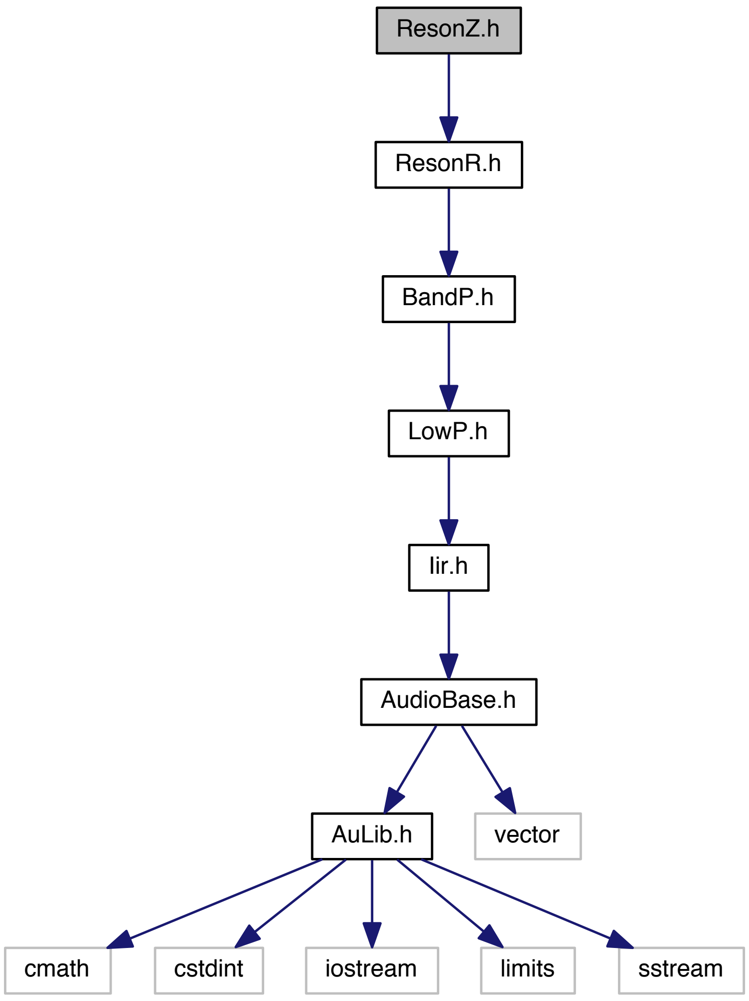
Namespaces

- AuLib

8.50 ResonZ.h File Reference

`#include "ResonR.h"`

Include dependency graph for ResonZ.h:



Classes

- class AuLib::ResonZ

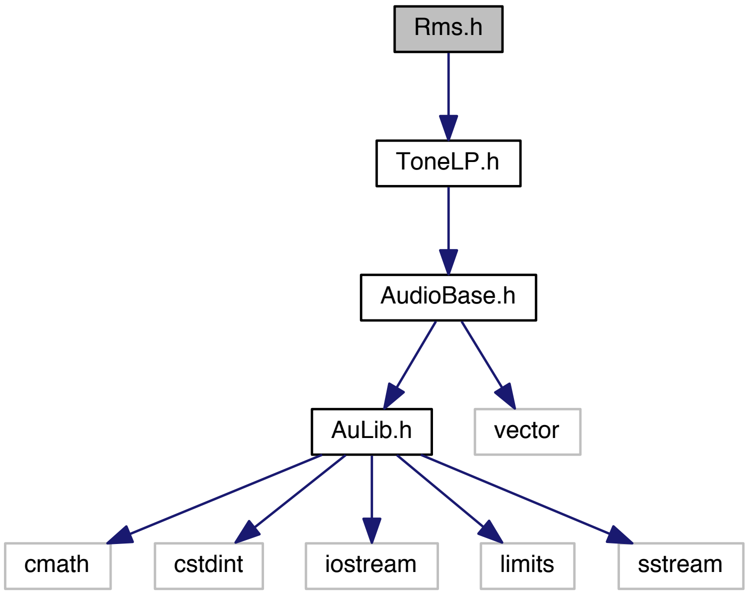
Namespaces

- AuLib

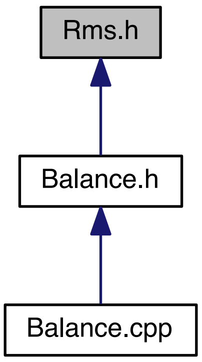
8.51 Rms.h File Reference

`#include "ToneLP.h"`

Include dependency graph for Rms.h:



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



Classes

- class AuLib::Rms

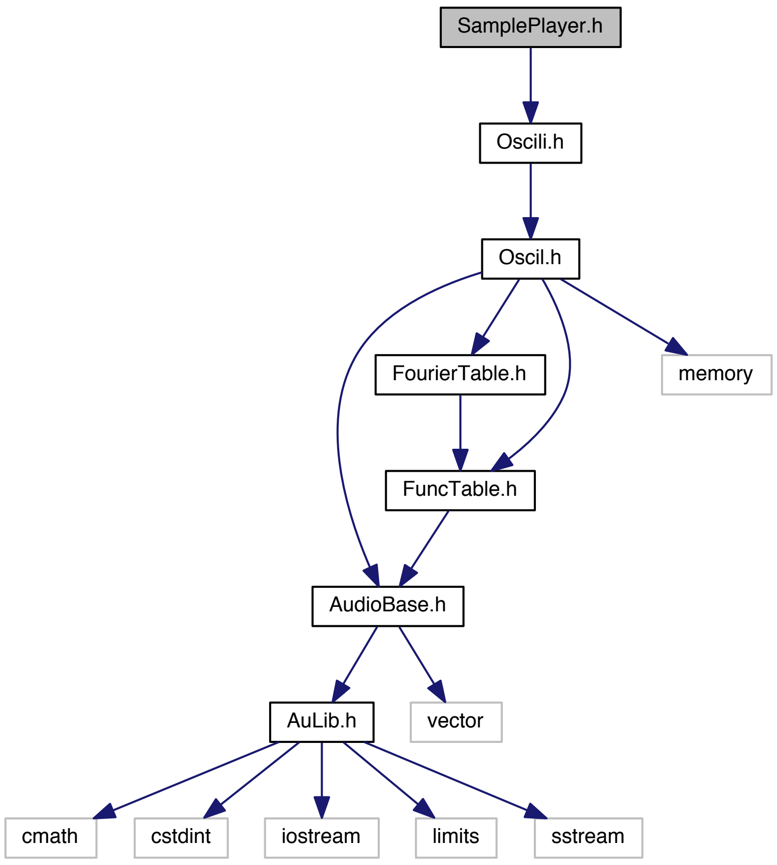
Namespaces

- AuLib

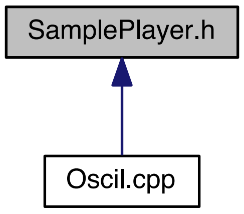
8.52 SamplePlayer.h File Reference

```
#include "Oscili.h"
```

Include dependency graph for SamplePlayer.h:



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



Classes

- class AuLib::SamplePlayer

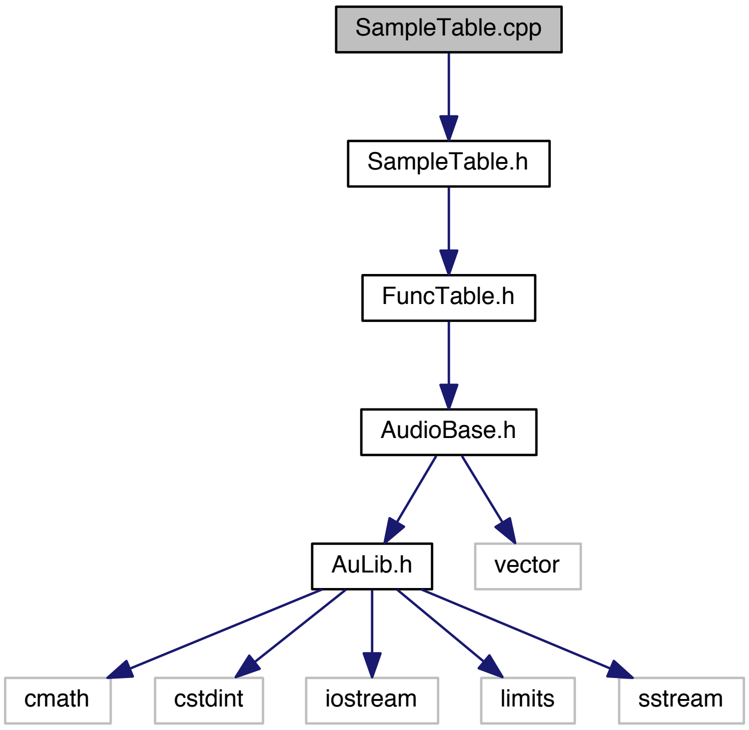
Namespaces

- AuLib

8.53 SampleTable.cpp File Reference

```
#include "SampleTable.h"
```

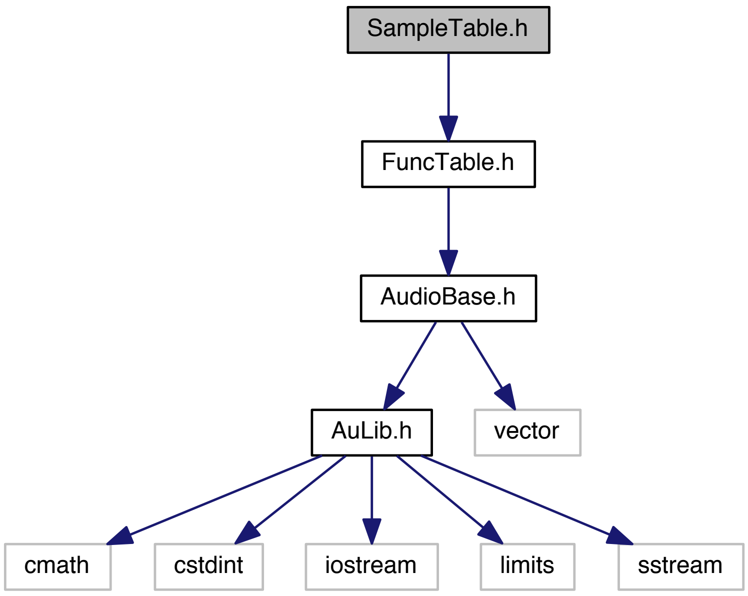
Include dependency graph for SampleTable.cpp:



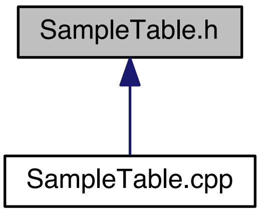
8.54 SampleTable.h File Reference

```
#include "FuncTable.h"
```

Include dependency graph for SampleTable.h:



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



Classes

- class AuLib::SampleTable

Namespaces

- AuLib

Enumerations

- enum AuLib::sampletable_error_codes { AuLib::AULIB_FILE_ERROR = AULIB_ERROR + 1, AuLib::AULI←
B_READ_ERROR }

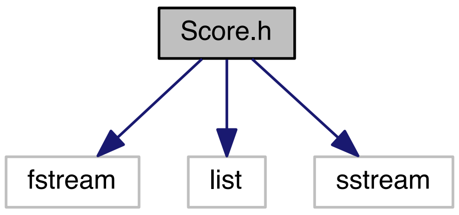
Variables

- const std::string AuLib::sampletable_error [ ]

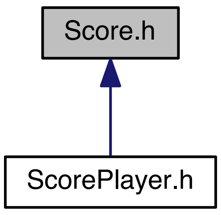
8.55 Score.h File Reference

```
#include <fstream>
#include <list>
#include <sstream>
```

Include dependency graph for Score.h:



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



Classes

- struct AuLib::Event
- class AuLib::Score
- struct AuLib::Score::Cmd

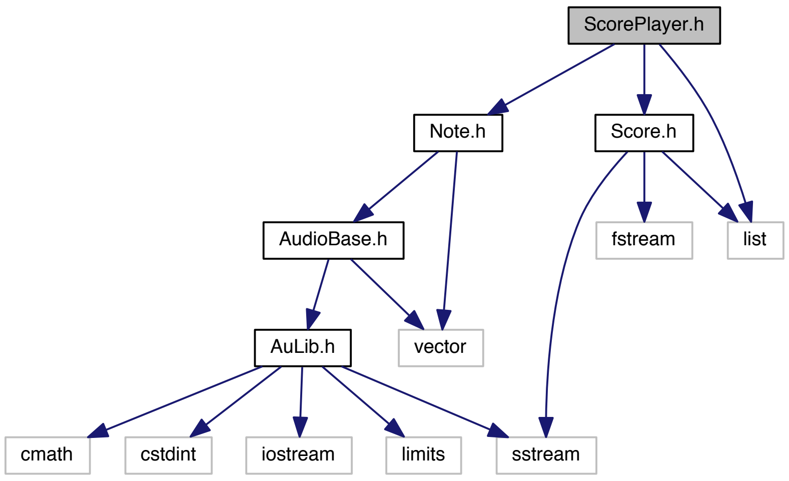
Namespaces

- AuLib

8.56 ScorePlayer.h File Reference

```
#include <Note.h>
#include <Score.h>
#include <list>
```

Include dependency graph for ScorePlayer.h:



Classes

- class AuLib::ScorePlayer

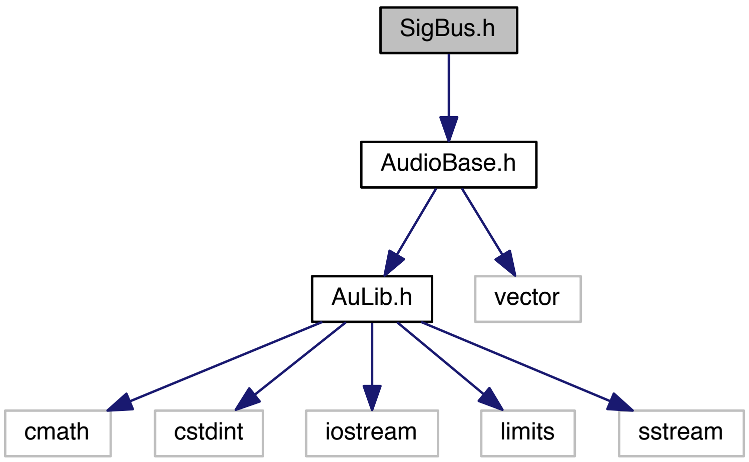
Namespaces

- AuLib

8.57 SigBus.h File Reference

```
#include "AudioBase.h"
```

Include dependency graph for SigBus.h:



Classes

- class AuLib::SigBus

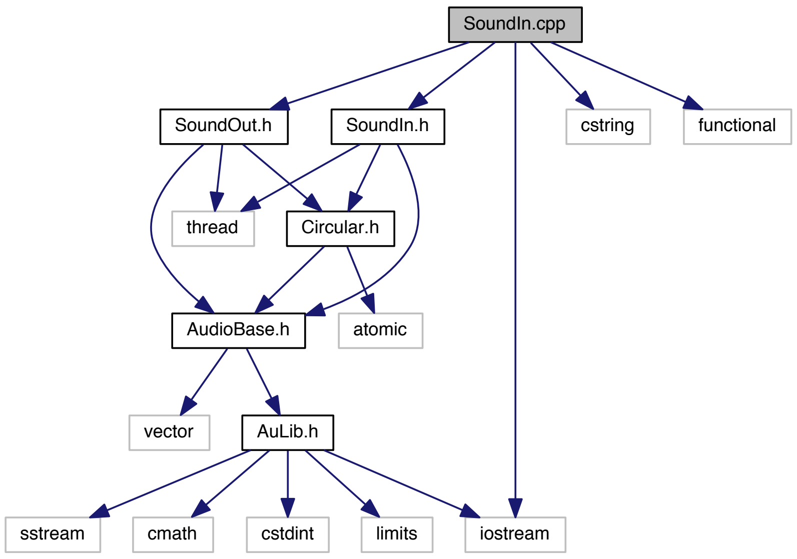
Namespaces

- AuLib

8.58 SoundIn.cpp File Reference

```
#include "SoundIn.h"
#include "SoundOut.h"
#include <cstring>
#include <functional>
#include <iostream>
```

Include dependency graph for SoundIn.cpp:



Namespaces

- AuLib

Typedefs

- typedef int(∗ AuLib::pa_callback_t) (const void ∗, void ∗, unsigned long, const PaStreamCallbackTimeInfo ∗, unsigned long, void ∗)

Functions

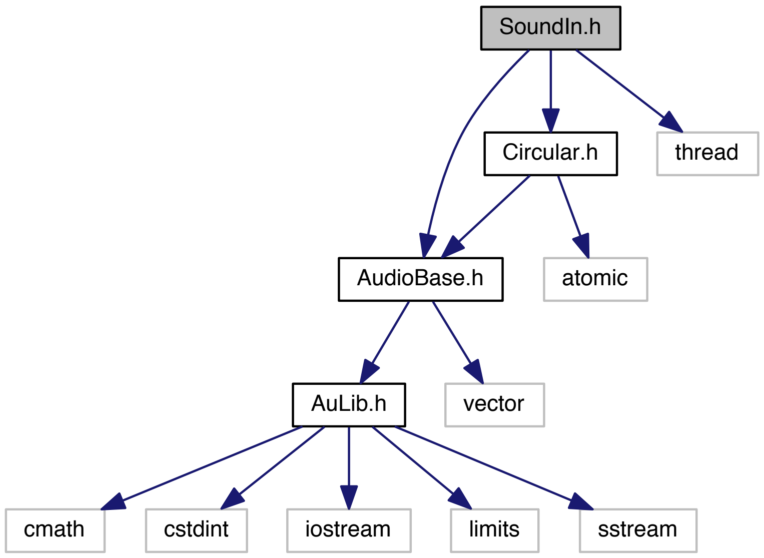
- int AuLib::rt_audio (const float ∗input, float ∗output, unsigned long frameCount, const void ∗timeInfo, unsigned long statusFlags, SoundIn ∗userData)
- void AuLib::audio (SoundIn &obj)

8.59 SoundIn.h File Reference

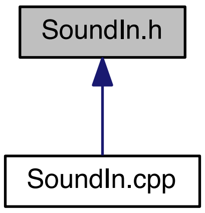
```
#include "AudioBase.h"
#include "Circular.h"
```

```
#include <thread>
```

Include dependency graph for SoundIn.h:



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



Classes

- class AuLib::SoundIn

Namespaces

- AuLib

Enumerations

- enum { AuLib::SOUNDIN_RT = 1, AuLib::SOUNDIN_STDIN, AuLib::SOUNDIN_SNDFILE }

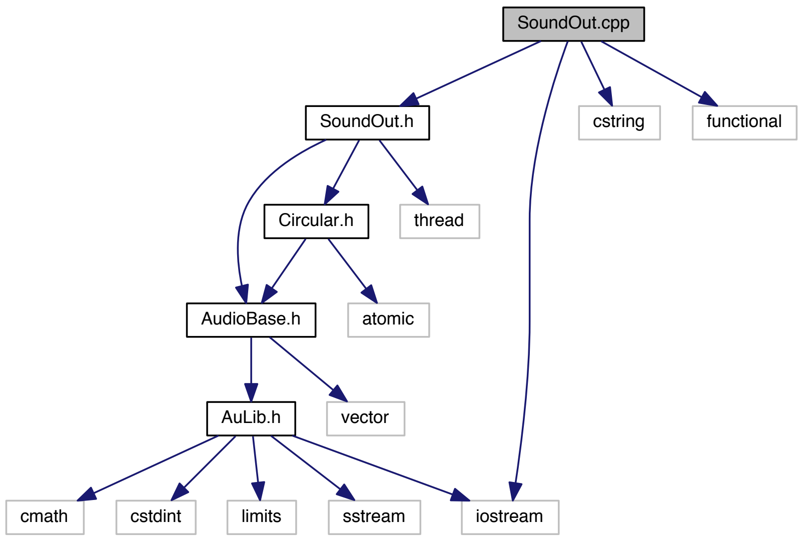
Variables

- const std::string AuLib::soundin_error [ ]

8.60 SoundOut.cpp File Reference

```
#include "SoundOut.h"
#include <cstring>
#include <functional>
#include <iostream>
```

Include dependency graph for SoundOut.cpp:



Namespaces

- AuLib

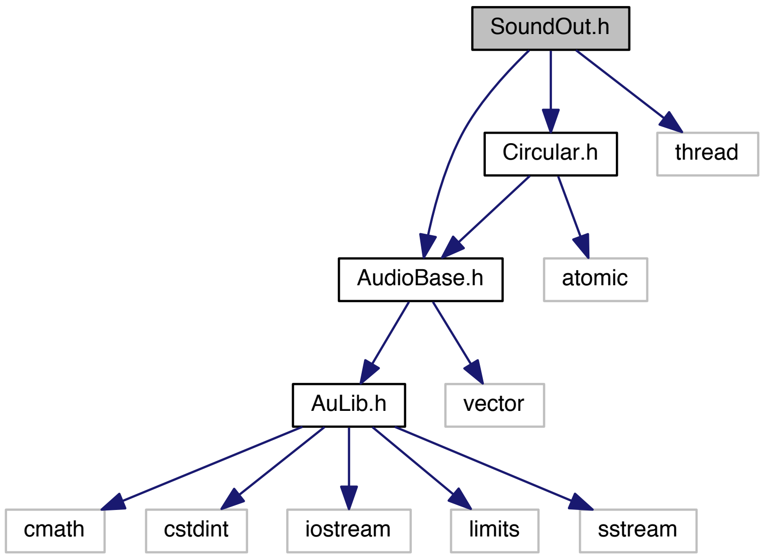
Functions

- int AuLib::rt_audio (const float ∗input, float ∗output, unsigned long frameCount, const void ∗timeInfo, unsigned long statusFlags, SoundOut ∗userData)
- void AuLib::audio (SoundOut &obj)

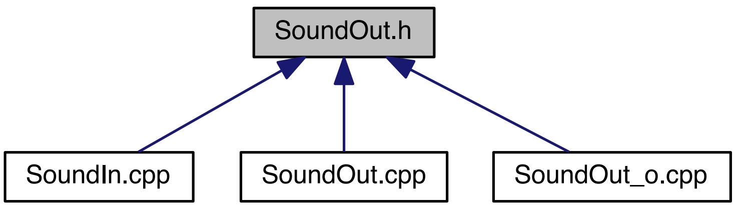
8.61 SoundOut.h File Reference

```
#include "AudioBase.h"
#include "Circular.h"
#include <thread>
```

Include dependency graph for SoundOut.h:



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



Classes

- class AuLib::SoundOut

Namespaces

- AuLib

Enumerations

- enum AuLib::dest_types {
  AuLib::SOUNDOUT_RT = 1, AuLib::SOUNDOUT_STDOUT, AuLib::SOUNDOUT_SNDFILE, AuLib::SOU↩NDOUT_RT = 1,
  AuLib::SOUNDOUT_STDOUT, AuLib::SOUNDOUT_SNDFILE }
- enum AuLib::soundout_error_codes {
  AuLib::AULIB_FOPEN_ERROR = AULIB_ERROR + 1, AuLib::AULIB_RTINIT_ERROR, AuLib::AULIB_R↩TOPEN_ERROR, AuLib::AULIB_RTSTREAM_ERROR,
  AuLib::AULIB_NOIO_ERROR, AuLib::AULIB_SOUNDIO_ERROR, AuLib::AULIB_FOPEN_ERROR = AU↩LIB_ERROR + 1, AuLib::AULIB_RTINIT_ERROR,
  AuLib::AULIB_RTOPEN_ERROR, AuLib::AULIB_RTSTREAM_ERROR, AuLib::AULIB_NOIO_ERROR,
  AuLib::AULIB_SOUNDIO_ERROR }

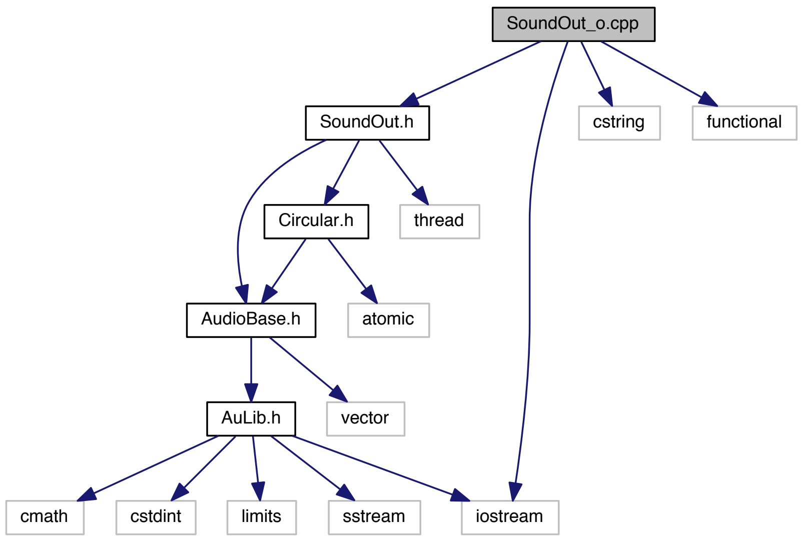
Variables

- const std::string AuLib::soundout_error [ ]

8.62 SoundOut_o.cpp File Reference

```
#include "SoundOut.h"
#include <cstring>
#include <functional>
#include <iostream>
```

Include dependency graph for SoundOut_o.cpp:



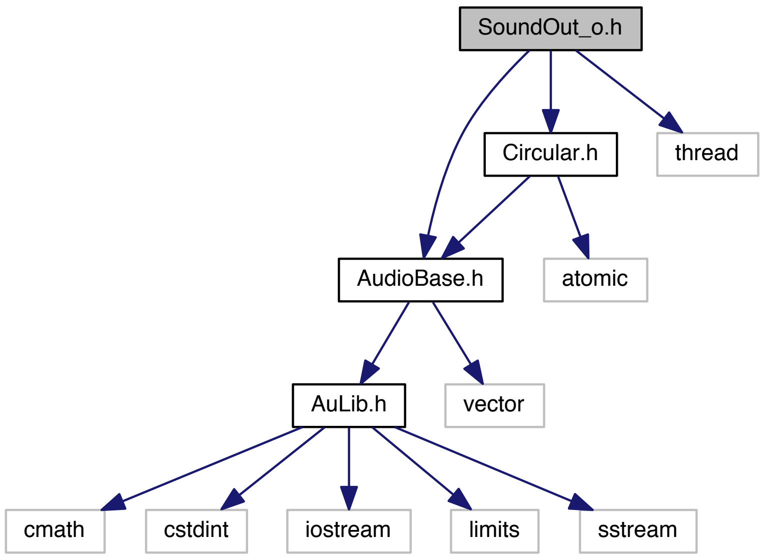
Namespaces

- AuLib

8.63 SoundOut_o.h File Reference

```
#include "AudioBase.h"
#include "Circular.h"
#include <thread>
```

Include dependency graph for SoundOut_o.h:



Classes

- class AuLib::SoundOut

Namespaces

- AuLib

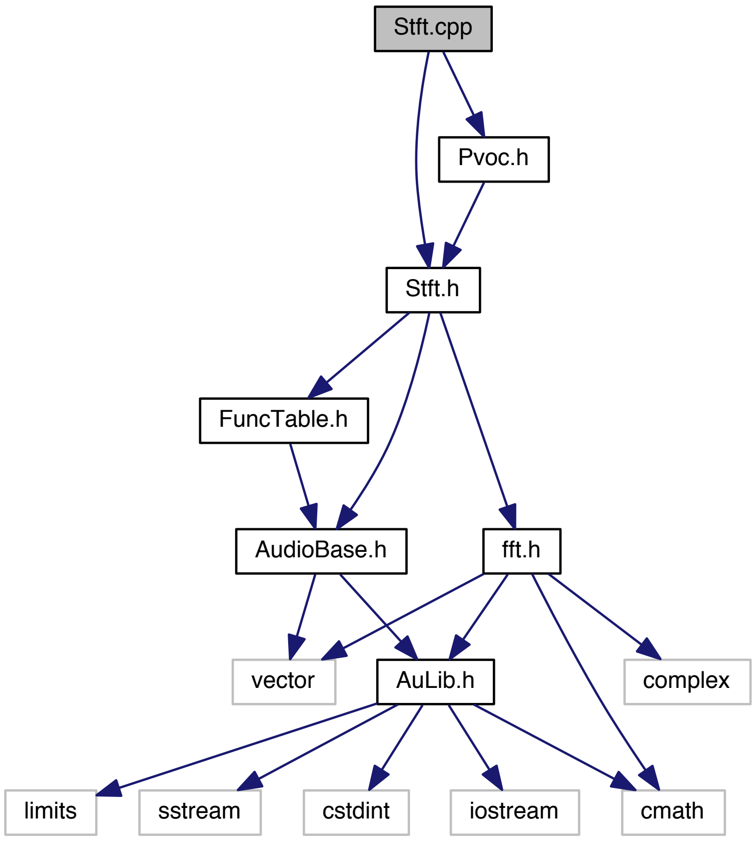
Enumerations

- enum AuLib::dest_types {
  AuLib::SOUNDOUT_RT = 1, AuLib::SOUNDOUT_STDOUT, AuLib::SOUNDOUT_SNDFILE, AuLib::SOU↩
  NDOUT_RT = 1,
  AuLib::SOUNDOUT_STDOUT, AuLib::SOUNDOUT_SNDFILE }
- enum AuLib::soundout_error_codes {
  AuLib::AULIB_FOPEN_ERROR = AULIB_ERROR + 1, AuLib::AULIB_RTINIT_ERROR, AuLib::AULIB_R↩
  TOPEN_ERROR, AuLib::AULIB_RTSTREAM_ERROR,
  AuLib::AULIB_NOIO_ERROR, AuLib::AULIB_SOUNDIO_ERROR, AuLib::AULIB_FOPEN_ERROR = AU↩
  LIB_ERROR + 1, AuLib::AULIB_RTINIT_ERROR,
  AuLib::AULIB_RTOPEN_ERROR, AuLib::AULIB_RTSTREAM_ERROR, AuLib::AULIB_NOIO_ERROR,
  AuLib::AULIB_SOUNDIO_ERROR }

8.64 Stft.cpp File Reference

```
#include "Stft.h"
#include "Pvoc.h"
```

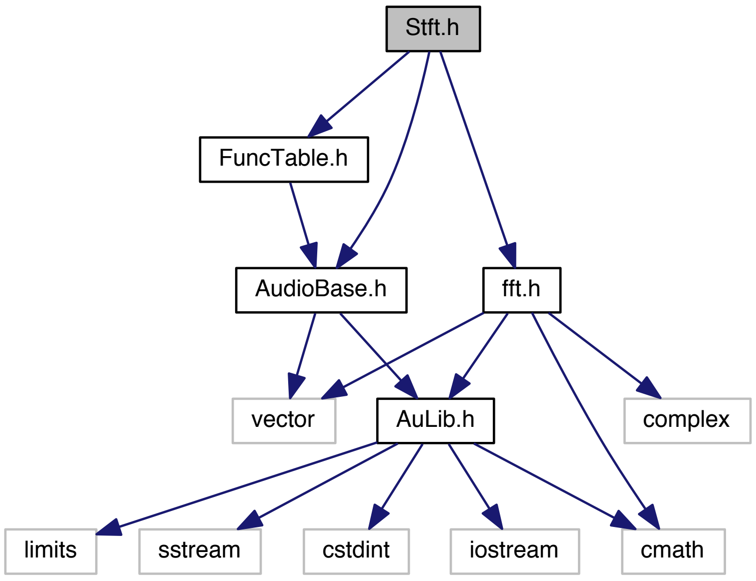
Include dependency graph for Stft.cpp:



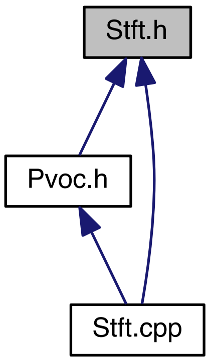
8.65 Stft.h File Reference

```
#include "AudioBase.h"
#include "FuncTable.h"
#include "fft.h"
```

Include dependency graph for Stft.h:



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



Classes

- class AuLib::Stft

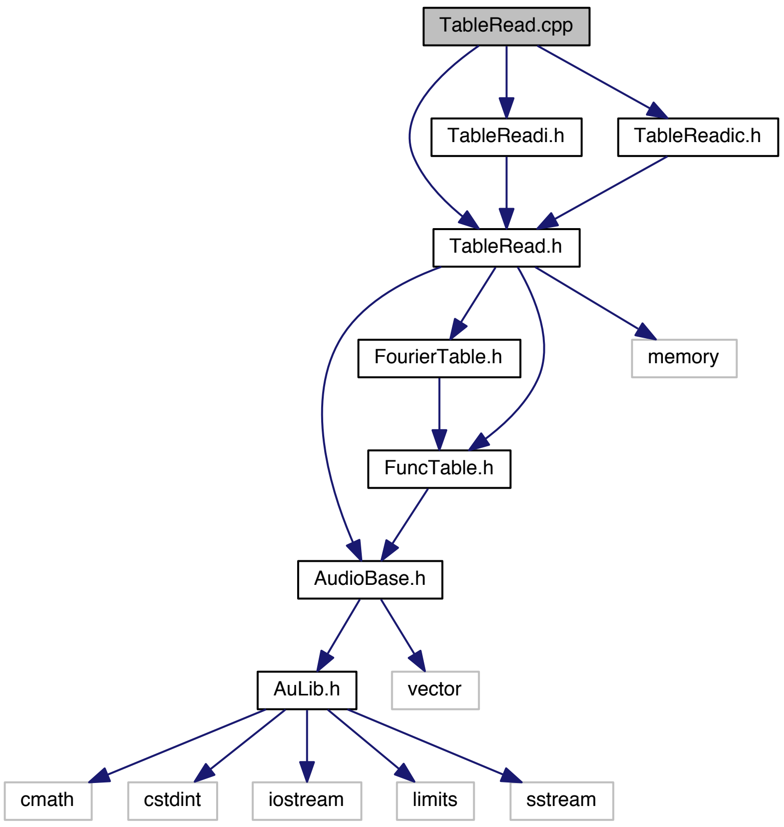
Namespaces

- AuLib

8.66 TableRead.cpp File Reference

```
#include "TableRead.h"
#include "TableReadi.h"
#include "TableReadic.h"
```

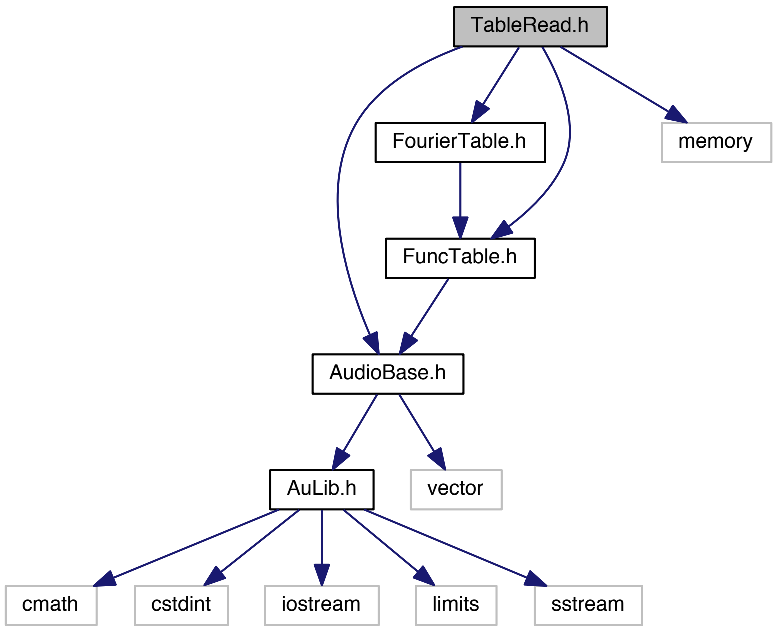
Include dependency graph for TableRead.cpp:



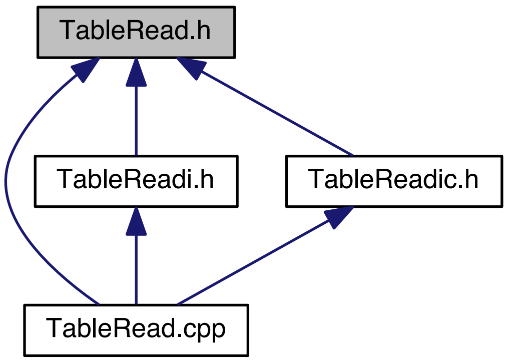
8.67 TableRead.h File Reference

```
#include "AudioBase.h"
#include "FourierTable.h"
#include "FuncTable.h"
#include <memory>
```

Include dependency graph for TableRead.h:



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



Classes

- class AuLib::TableRead

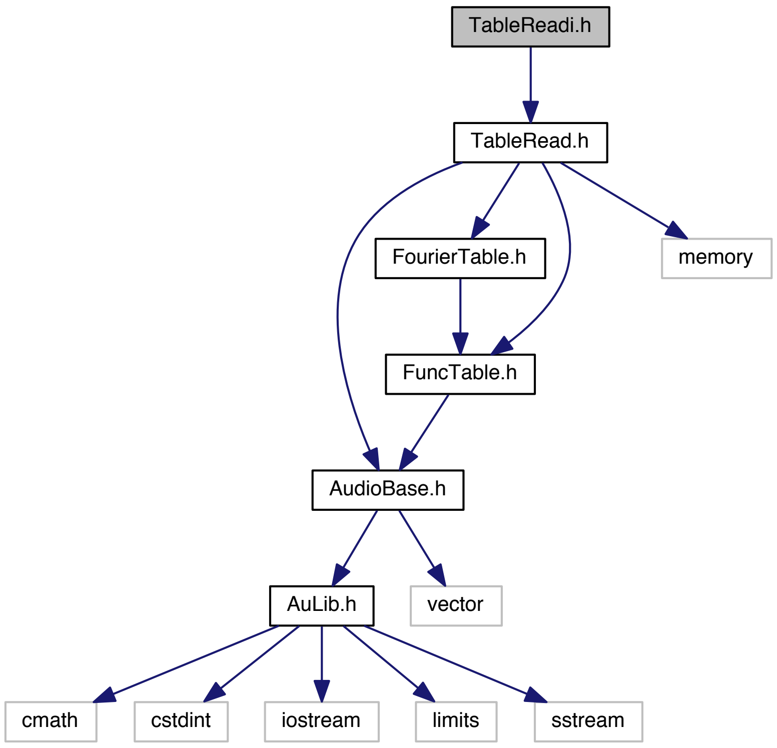
Namespaces

- AuLib

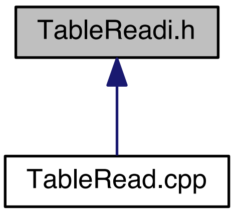
8.68 TableReadi.h File Reference

```
#include "TableRead.h"
```

Include dependency graph for TableReadi.h:



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



Classes

- class AuLib::TableReadi

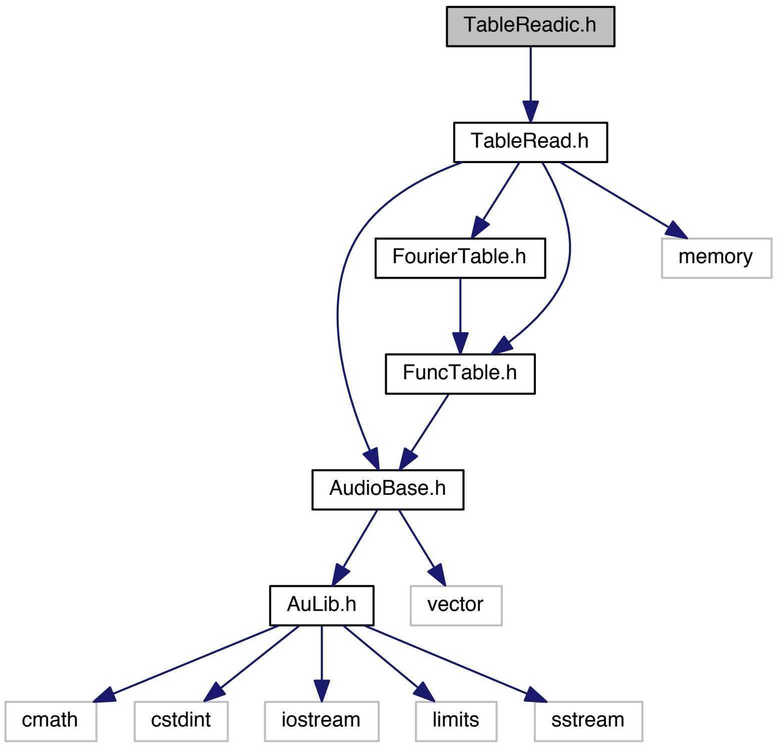
Namespaces

- AuLib

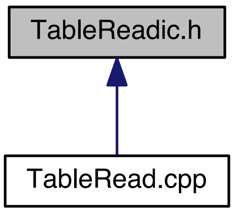
8.69 TableReadic.h File Reference

```
#include "TableRead.h"
```

Include dependency graph for TableReadic.h:



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



Classes

- class AuLib::TableReadic

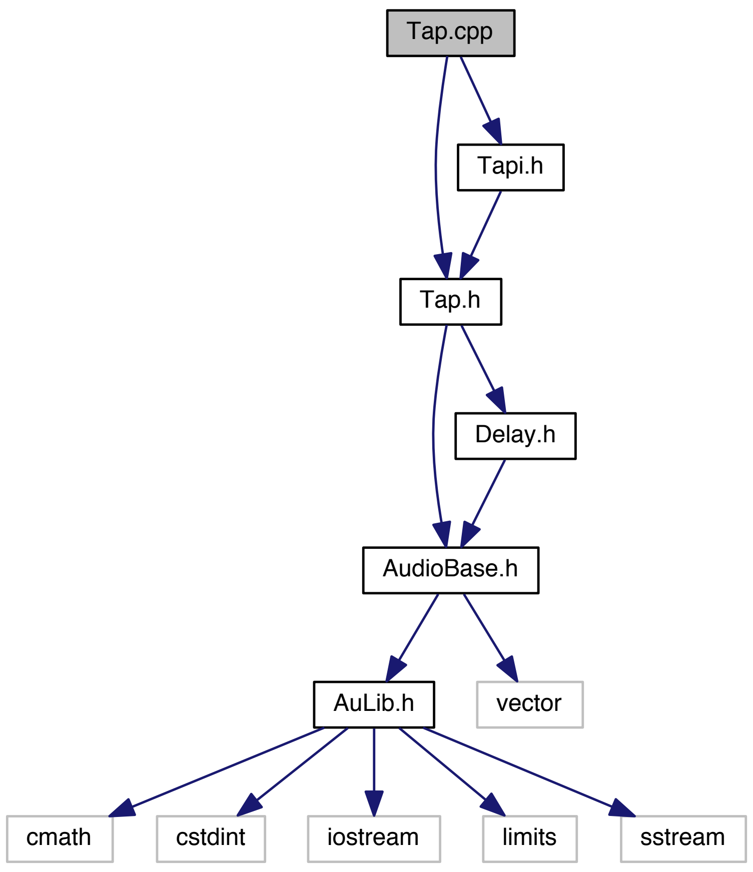
Namespaces

- AuLib

8.70 Tap.cpp File Reference

```
#include "Tap.h"
#include "Tapi.h"
```

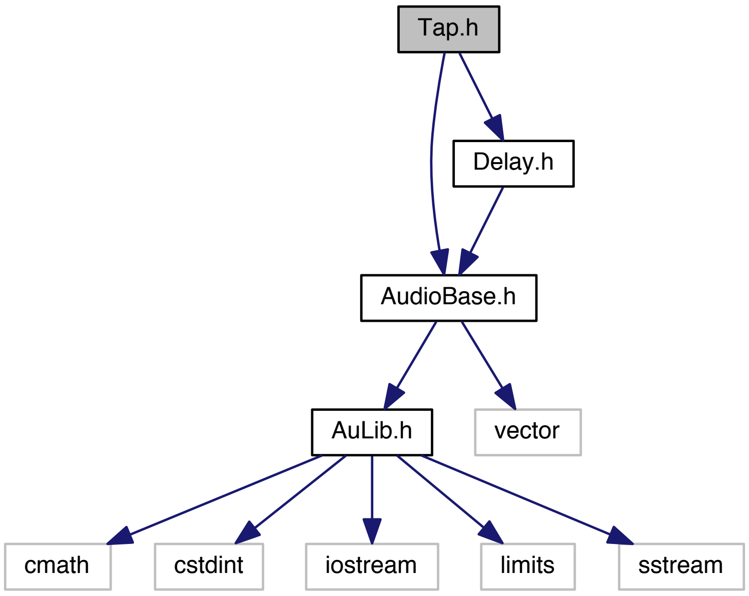
Include dependency graph for Tap.cpp:



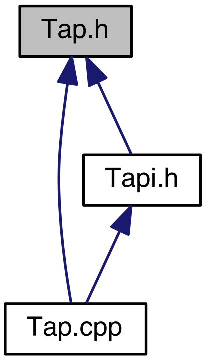
8.71 Tap.h File Reference

```
#include "AudioBase.h"
#include "Delay.h"
```

Include dependency graph for Tap.h:



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



Classes

- class AuLib::Tap

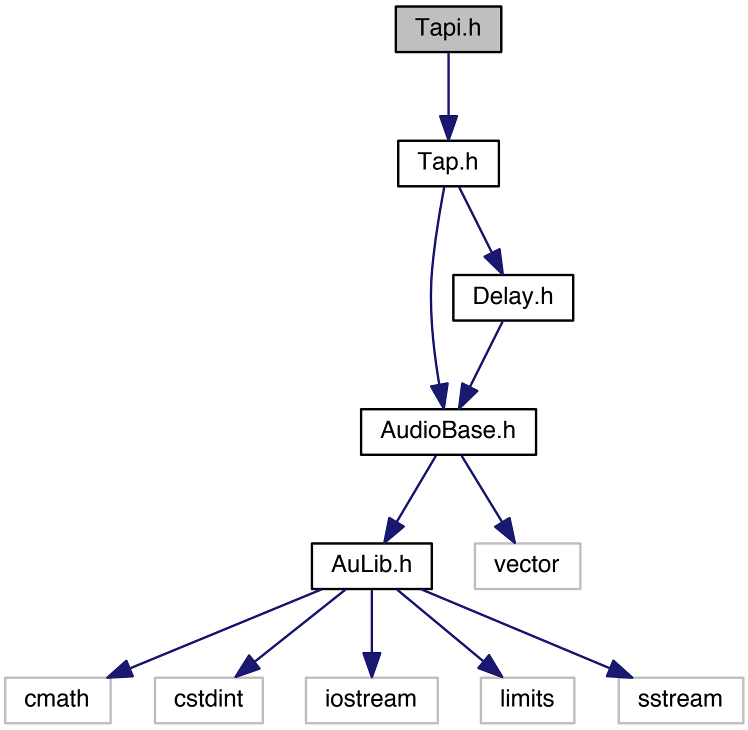
Namespaces

- AuLib

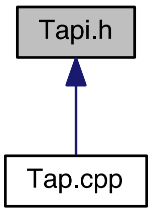
8.72 Tapi.h File Reference

```
#include "Tap.h"
```

Include dependency graph for Tapi.h:



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



Classes

- class AuLib::Tapi

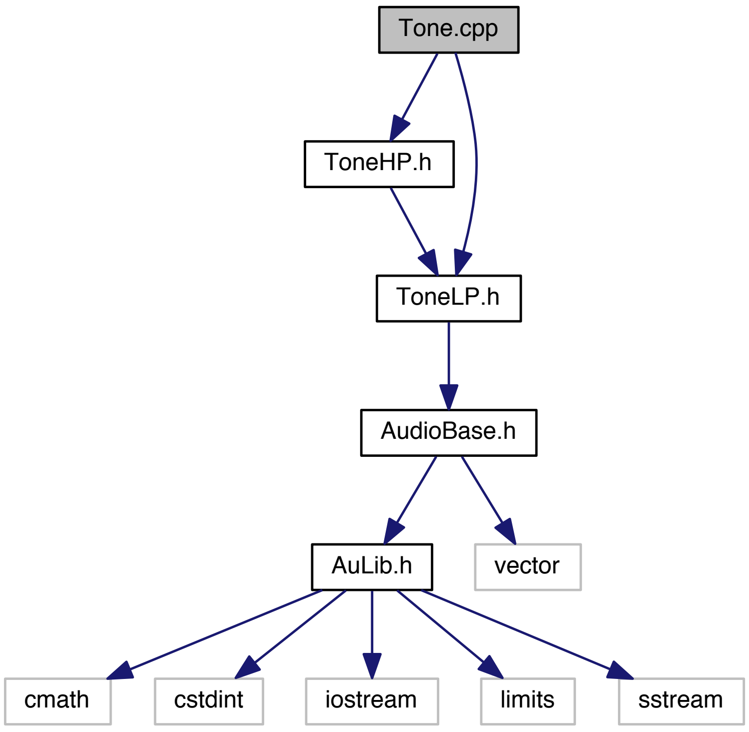
Namespaces

- AuLib

8.73 Tone.cpp File Reference

```
#include "ToneHP.h"
#include "ToneLP.h"
```

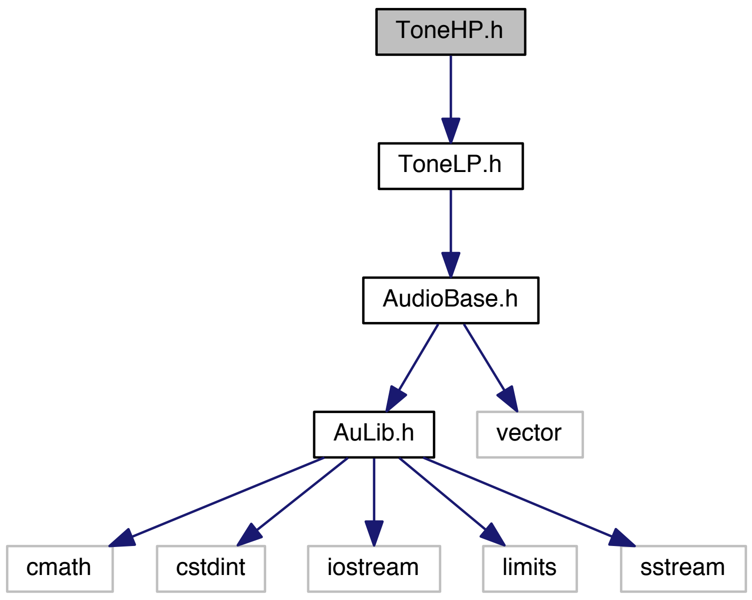
Include dependency graph for Tone.cpp:



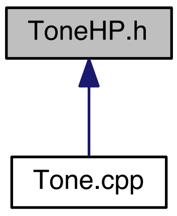
8.74 ToneHP.h File Reference

```
#include "ToneLP.h"
```

Include dependency graph for ToneHP.h:



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



Classes

- class AuLib::ToneHP

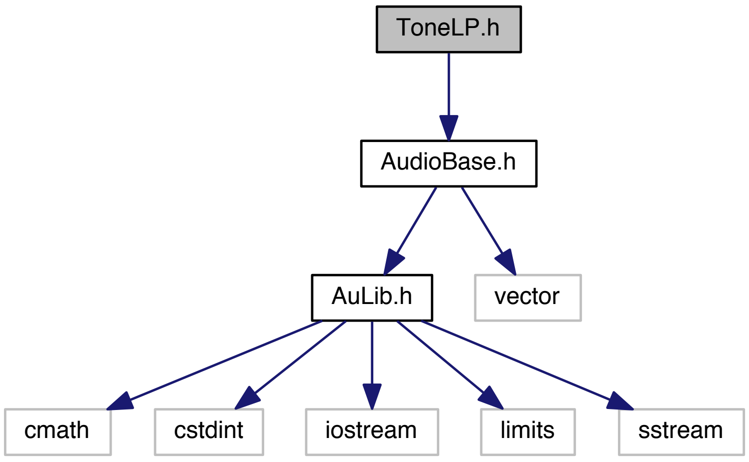
Namespaces

- AuLib

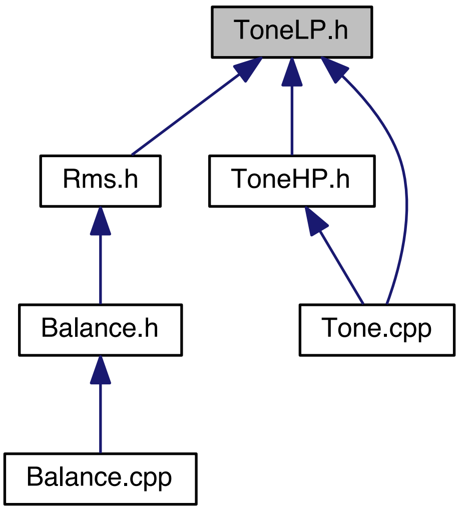
8.75 ToneLP.h File Reference

```
#include "AudioBase.h"
```

Include dependency graph for ToneLP.h:



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



Classes

- class AuLib::ToneLP

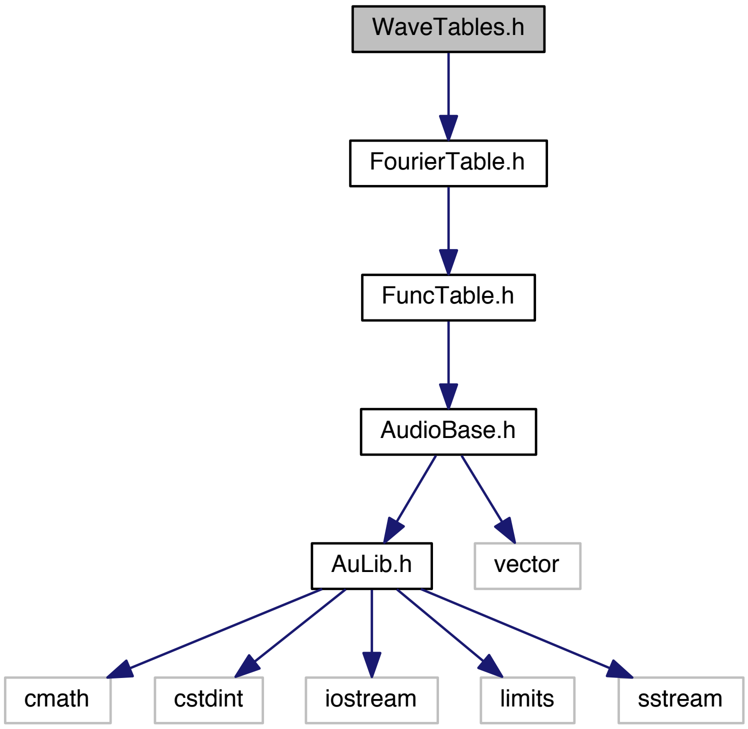
Namespaces

- AuLib

8.76 WaveTables.h File Reference

```
#include "FourierTable.h"
```

Include dependency graph for WaveTables.h:



Classes

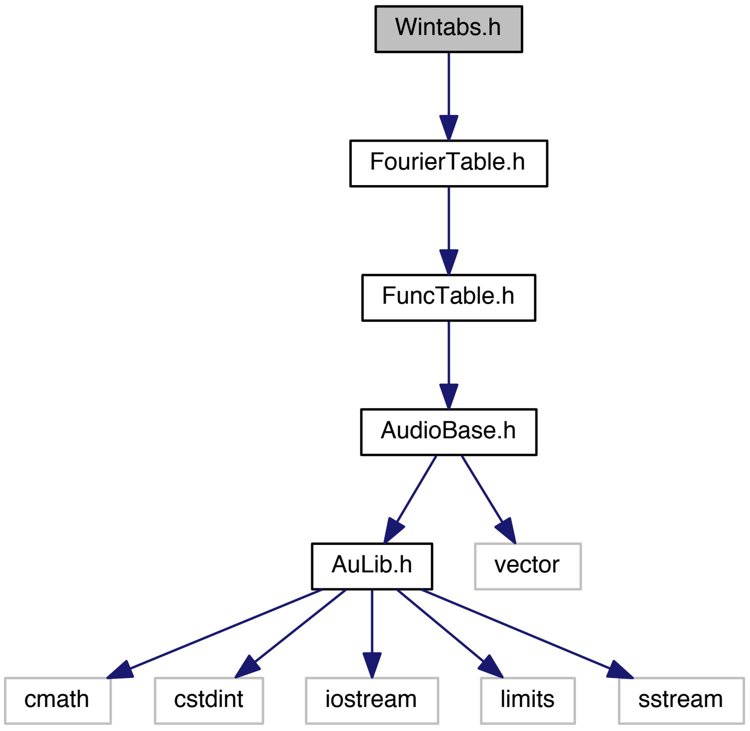
- class AuLib::SawTable
- class AuLib::SquareTable
- class AuLib::TriangleTable

Namespaces

- AuLib

8.77 Wintabs.h File Reference

```
#include "FourierTable.h"
```
Include dependency graph for Wintabs.h:



Classes

- struct AuLib::Hann
- struct AuLib::Hamming

Namespaces

- AuLib

Index