# VigiSense

11

Generated by Doxygen 1.10.0

# Chapter 1

# Namespace Index

## 1.1 Namespace List

Here is a list of all namespaces with brief descriptions:

# Chapter 2

# Hierarchical Index

## 2.1 Class Hierarchy

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

# Chapter 3

# Class Index

## 3.1 Class List

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

# Chapter 4

# File Index

## 4.1 File List

Here is a list of all files with brief descriptions:

# Chapter 5

# Namespace Documentation

## 5.1 eprosima Namespace Reference

**Namespaces**

- namespace fastcdr

## 5.2 eprosima::fastcdr Namespace Reference

## 5.3 tps Namespace Reference

**Functions**

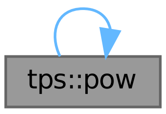
- template<typename T >
  constexpr T pow (T input, unsigned int power)

### 5.3.1 Function Documentation

#### 5.3.1.1 pow()

```
template<typename T >
constexpr T tps::pow (
             T input,
             unsigned int power )  [constexpr]
```

Here is the call graph for this function:



# Chapter 6

# Class Documentation

## 6.1 alert Struct Reference

This class represents the structure alert defined by the user in the IDL file.

```
#include <alert.h>
```

### Public Member Functions

- eProsima_user_DllExport alert ()

    *Default constructor.*
- eProsima_user_DllExport ∼alert ()

    *Default destructor.*
- eProsima_user_DllExport alert (const alert &x)

    *Copy constructor.*
- eProsima_user_DllExport alert (alert &&x) noexcept

    *Move constructor.*
- eProsima_user_DllExport alert & operator= (const alert &x)

    *Copy assignment.*
- eProsima_user_DllExport alert & operator= (alert &&x) noexcept

    *Move assignment.*
- eProsima_user_DllExport bool operator== (const alert &x) const

    *Comparison operator.*
- eProsima_user_DllExport bool operator!= (const alert &x) const

    *Comparison operator.*
- eProsima_user_DllExport void index (uint32_t _index)

    *This function sets a value in member index.*
- eProsima_user_DllExport uint32_t index () const

    *This function returns the value of member index.*
- eProsima_user_DllExport uint32_t & index ()

    *This function returns a reference to member index.*
- eProsima_user_DllExport void message (const std::string &_message)

    *This function copies the value in member message.*
- eProsima_user_DllExport void message (std::string &&_message)

    *This function moves the value in member message.*

- eProsima_user_DllExport const std::string & message () const

  *This function returns a constant reference to member message.*
- eProsima_user_DllExport std::string & message ()

  *This function returns a reference to member message.*
- eProsima_user_DllExport void serialize (eprosima::fastcdr::Cdr &cdr) const

  *This function serializes an object using CDR serialization.*
- eProsima_user_DllExport void deserialize (eprosima::fastcdr::Cdr &cdr)

  *This function deserializes an object using CDR serialization.*
- eProsima_user_DllExport void serializeKey (eprosima::fastcdr::Cdr &cdr) const

  *This function serializes the key members of an object using CDR serialization.*

**Static Public Member Functions**

- static eProsima_user_DllExport size_t getMaxCdrSerializedSize (size_t current_alignment=0)

  *This function returns the maximum serialized size of an object depending on the buffer alignment.*
- static eProsima_user_DllExport size_t getCdrSerializedSize (const alert &data, size_t current_alignment=0)

  *This function returns the serialized size of a data depending on the buffer alignment.*
- static eProsima_user_DllExport size_t getKeyMaxCdrSerializedSize (size_t current_alignment=0)

  *This function returns the maximum serialized size of the Key of an object depending on the buffer alignment.*
- static eProsima_user_DllExport bool isKeyDefined ()

  *This function tells you if the Key has been defined for this type.*

**Public Attributes**

- unsigned long index

  *This function sets a value in member index.*
- string message

  *This function copies the value in member message.*

**Private Attributes**

- uint32_t m_index
- std::string m_message

## 6.1.1 Detailed Description

This class represents the structure alert defined by the user in the IDL file.

## 6.1.2 Constructor & Destructor Documentation

### 6.1.2.1 alert() [1/3]

```
alert::alert ( )
```

Default constructor.

#### 6.1.2.2 ∼alert()

```
alert::∼alert ( )
```

Default destructor.

#### 6.1.2.3 alert() [2/3]

```
alert::alert (
            const alert & x )
```

Copy constructor.

**Parameters**

| | |
|---|---|
| *x* | Reference to the object alert that will be copied. |

#### 6.1.2.4 alert() [3/3]

```
alert::alert (
            alert && x )  [noexcept]
```

Move constructor.

**Parameters**

| | |
|---|---|
| *x* | Reference to the object alert that will be copied. |

### 6.1.3 Member Function Documentation

#### 6.1.3.1 deserialize()

```
void alert::deserialize (
            eprosima::fastcdr::Cdr & cdr )
```

This function deserializes an object using CDR serialization.

**Parameters**

| | |
|---|---|
| *cdr* | CDR serialization object. |

#### 6.1.3.2 getCdrSerializedSize()

```
size_t alert::getCdrSerializedSize (
            const alert & data,
            size_t current_alignment = 0 )  [static]
```

This function returns the serialized size of a data depending on the buffer alignment.

**Parameters**

| | |
|---|---|
| *data* | Data which is calculated its serialized size. |
| *current_alignment* | Buffer alignment. |

**Returns**

Serialized size.

### 6.1.3.3 getKeyMaxCdrSerializedSize()

```
size_t alert::getKeyMaxCdrSerializedSize (
              size_t current_alignment = 0 )  [static]
```

This function returns the maximum serialized size of the Key of an object depending on the buffer alignment.

**Parameters**

| | |
|---|---|
| *current_alignment* | Buffer alignment. |

**Returns**

Maximum serialized size.

### 6.1.3.4 getMaxCdrSerializedSize()

```
size_t alert::getMaxCdrSerializedSize (
              size_t current_alignment = 0 )  [static]
```

This function returns the maximum serialized size of an object depending on the buffer alignment.

**Parameters**

| | |
|---|---|
| *current_alignment* | Buffer alignment. |

**Returns**

Maximum serialized size.

### 6.1.3.5 index() [1/3]

```
eProsima_user_DllExport uint32_t & alert::index ( )
```

This function returns a reference to member index.

**Returns**

Reference to member index

### 6.1.3.6 index() [2/3]

```
eProsima_user_DllExport uint32_t alert::index ( ) const
```

This function returns the value of member index.

**Returns**

Value of member index

### 6.1.3.7 index() [3/3]

```
eProsima_user_DllExport void alert::index (
              uint32_t _index )
```

This function sets a value in member index.

**Parameters**

| | |
|---|---|
| *_index* | New value for member index |

### 6.1.3.8 isKeyDefined()

```
bool alert::isKeyDefined ( )  [static]
```

This function tells you if the Key has been defined for this type.

### 6.1.3.9 message() [1/4]

```
eProsima_user_DllExport std::string & alert::message ( )
```

This function returns a reference to member message.

**Returns**

Reference to member message

### 6.1.3.10 message() [2/4]

```
eProsima_user_DllExport const std::string & alert::message ( ) const
```

This function returns a constant reference to member message.

**Returns**

Constant reference to member message

### 6.1.3.11 message() [3/4]

```
eProsima_user_DllExport void alert::message (
            const std::string & _message )
```

This function copies the value in member message.

**Parameters**

| | |
|---|---|
| *_message* | New value to be copied in member message |

### 6.1.3.12 message() [4/4]

```
eProsima_user_DllExport void alert::message (
            std::string && _message )
```

This function moves the value in member message.

**Parameters**

| | |
|---|---|
| *_message* | New value to be moved in member message |

### 6.1.3.13 operator"!=()

```
bool alert::operator!= (
            const alert & x ) const
```

Comparison operator.

**Parameters**

| | |
|---|---|
| *x* | alert object to compare. |

### 6.1.3.14 operator=() [1/2]

```
alert & alert::operator= (
            alert && x )  [noexcept]
```

Move assignment.

**Parameters**

| | |
|---|---|
| *x* | Reference to the object alert that will be copied. |

#### 6.1.3.15 operator=() [2/2]

```
alert & alert::operator= (
            const alert & x )
```

Copy assignment.

**Parameters**

| | |
|---|---|
| *x* | Reference to the object alert that will be copied. |

#### 6.1.3.16 operator==()

```
bool alert::operator== (
            const alert & x ) const
```

Comparison operator.

**Parameters**

| | |
|---|---|
| *x* | alert object to compare. |

#### 6.1.3.17 serialize()

```
void alert::serialize (
            eprosima::fastcdr::Cdr & cdr ) const
```

This function serializes an object using CDR serialization.

**Parameters**

| | |
|---|---|
| *cdr* | CDR serialization object. |

#### 6.1.3.18 serializeKey()

```
void alert::serializeKey (
            eprosima::fastcdr::Cdr & cdr ) const
```

This function serializes the key members of an object using CDR serialization.

**Parameters**

| | |
|---|---|
| *cdr* | CDR serialization object. |

### 6.1.4 Member Data Documentation

#### 6.1.4.1 index

```
uint32_t & alert::index
```

This function sets a value in member index.

This function returns a reference to member index.

This function returns the value of member index.

**Parameters**

| | |
|---|---|
| *_index* | New value for member index |

**Returns**

Value of member index

Reference to member index

#### 6.1.4.2 m_index

```
uint32_t alert::m_index  [private]
```

#### 6.1.4.3 m_message

```
std::string alert::m_message  [private]
```

#### 6.1.4.4 message

```
std::string & alert::message
```

This function copies the value in member message.

This function returns a reference to member message.

This function returns a constant reference to member message.

This function moves the value in member message.

**Parameters**

| | |
|---|---|
| *_message* | New value to be copied in member message |
| *_message* | New value to be moved in member message |

**Returns**

Constant reference to member message

Reference to member message

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

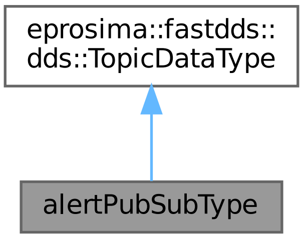
- /home/sitcomlab/Projects/VigiSense/src/alert.h
- /home/sitcomlab/Projects/VigiSense/src/alert.idl
- /home/sitcomlab/Projects/VigiSense/src/alert.cxx

## 6.2 alertPubSubType Class Reference

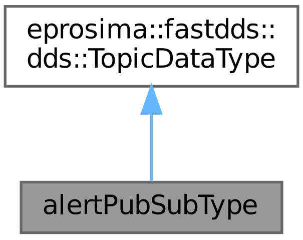
This class represents the TopicDataType of the type alert defined by the user in the IDL file.

```
#include <alertPubSubTypes.h>
```

Inheritance diagram for alertPubSubType:



Collaboration diagram for alertPubSubType:



**Public Types**

- typedef alert type

**Public Member Functions**

- eProsima_user_DllExport alertPubSubType ()
- virtual eProsima_user_DllExport ∼alertPubSubType () override
- virtual eProsima_user_DllExport bool serialize (void ∗data, eprosima::fastrtps::rtps::SerializedPayload_↩
t ∗payload) override
- virtual eProsima_user_DllExport bool deserialize (eprosima::fastrtps::rtps::SerializedPayload_t ∗payload, void ∗data) override
- virtual eProsima_user_DllExport std::function< uint32_t()> getSerializedSizeProvider (void ∗data) override
- virtual eProsima_user_DllExport bool getKey (void ∗data, eprosima::fastrtps::rtps::InstanceHandle_↩
t ∗ihandle, bool force_md5=false) override
- virtual eProsima_user_DllExport void ∗ createData () override
- virtual eProsima_user_DllExport void deleteData (void ∗data) override

**Public Attributes**

- MD5 m_md5
- unsigned char ∗ m_keyBuffer

## 6.2.1 Detailed Description

This class represents the TopicDataType of the type alert defined by the user in the IDL file.

## 6.2.2 Member Typedef Documentation

### 6.2.2.1 type

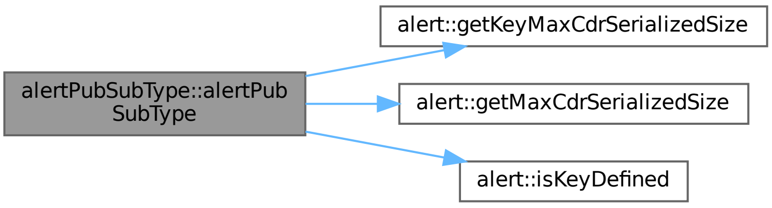
```
typedef alert alertPubSubType::type
```

## 6.2.3 Constructor & Destructor Documentation

### 6.2.3.1 alertPubSubType()

```
alertPubSubType::alertPubSubType ( )
```

Here is the call graph for this function:


alertPubSubType::alertPub SubType
alert::getKeyMaxCdrSerializedSize
alert::getMaxCdrSerializedSize
alert::isKeyDefined

#### 6.2.3.2 ∼alertPubSubType()

```
alertPubSubType::∼alertPubSubType ( )  [override], [virtual]
```

### 6.2.4 Member Function Documentation

#### 6.2.4.1 createData()

```
void * alertPubSubType::createData ( )  [override], [virtual]
```

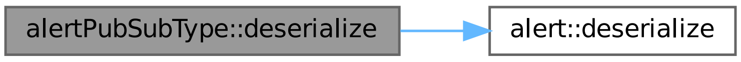
#### 6.2.4.2 deleteData()

```
void alertPubSubType::deleteData (
             void * data )  [override], [virtual]
```

#### 6.2.4.3 deserialize()

```
bool alertPubSubType::deserialize (
             eprosima::fastrtps::rtps::SerializedPayload_t * payload,
             void * data )  [override], [virtual]
```

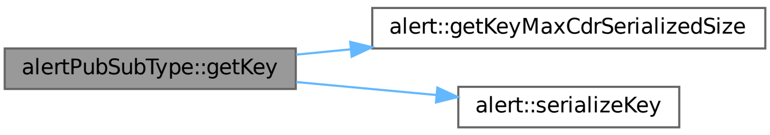
Here is the call graph for this function:



#### 6.2.4.4 getKey()

```
bool alertPubSubType::getKey (
             void * data,
             eprosima::fastrtps::rtps::InstanceHandle_t * ihandle,
             bool force_md5 = false )  [override], [virtual]
```

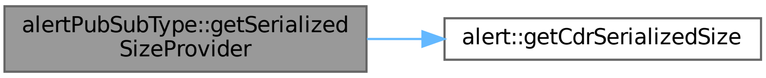
Here is the call graph for this function:



#### 6.2.4.5 getSerializedSizeProvider()

```
std::function< uint32_t()> alertPubSubType::getSerializedSizeProvider (
            void * data )  [override], [virtual]
```

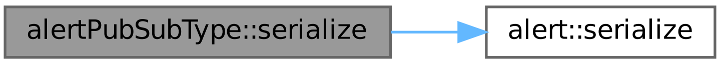
Here is the call graph for this function:


alertPubSubType::getSerialized SizeProvider
alert::getCdrSerializedSize

#### 6.2.4.6 serialize()

```
bool alertPubSubType::serialize (
            void * data,
            eprosima::fastrtps::rtps::SerializedPayload_t * payload )  [override], [virtual]
```

Here is the call graph for this function:



### 6.2.5 Member Data Documentation

#### 6.2.5.1 m_keyBuffer

```
unsigned char* alertPubSubType::m_keyBuffer
```

#### 6.2.5.2 m_md5

```
MD5 alertPubSubType::m_md5
```

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

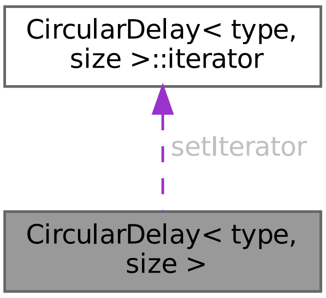
- /home/sitcomlab/Projects/VigiSense/src/alertPubSubTypes.h
- /home/sitcomlab/Projects/VigiSense/src/alertPubSubTypes.cxx

## 6.3 CircularDelay< type, size > Class Template Reference

A class that functions as a sample buffer.

```
#include <CircularDelay.hpp>
```

Collaboration diagram for CircularDelay< type, size >:


CircularDelay< type, size >::iterator
setIterator
CircularDelay< type, size >

### Classes

- class const_iterator
- class const_reverse_iterator
- class iterator
- class reverse_iterator

### Public Member Functions

- CircularDelay ()

    *Constructor that initializes that buffer and its set index.*
- type push (type input)

    *With this function you can insert a new sample into the buffer.*
- type get (size_t delay)

    *With this function you can retrieve a sample from the past.*
- iterator end ()
- iterator begin ()
- reverse_iterator rend ()
- reverse_iterator rbegin ()

### Private Attributes

- type data [size+1]
- iterator setIterator = iterator(data, data)

## 6.3.1 Detailed Description

**template<typename type, size_t size>**
**class CircularDelay< type, size >**

A class that functions as a sample buffer.

**Copyright**

GPL V3

Circular delay software library. Here data can be stored and retrieved is a LiFo manner. Copyright (C) 2018 Jimmy van den Berg

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see `http://www.gnu.org/licenses/`.

You can use this to insert samples and use the get function to get a sample from the past.

**Template Parameters**

| | |
|---|---|
| *type* | Type of sample that needs to be stored. |
| *size* | Size of how big the history buffer is. |

## 6.3.2 Constructor & Destructor Documentation

### 6.3.2.1 CircularDelay()

```
template<typename type , size_t size>
CircularDelay< type, size >::CircularDelay ( )
```

Constructor that initializes that buffer and its set index.

**Template Parameters**

| | |
|---|---|
| *type* | Type of sample that needs to be stored. |
| *size* | Size of how big the history buffer is. |

### 6.3.3 Member Function Documentation

#### 6.3.3.1 begin()

```
template<typename type , size_t size>
iterator CircularDelay< type, size >::begin ( )  [inline]
```

#### 6.3.3.2 end()

```
template<typename type , size_t size>
iterator CircularDelay< type, size >::end ( )  [inline]
```

#### 6.3.3.3 get()

```
template<typename type , size_t size>
type CircularDelay< type, size >::get (
             size_t delay )
```

With this function you can retrieve a sample from the past.

Maximum delay is the size of the CircularDelay - 1.

**Parameters**

| | |
|---|---|
| *delay* | How many samples you ago you want to get. |

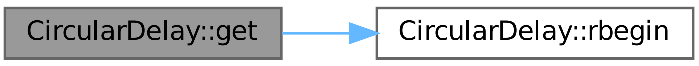
**Template Parameters**

| | |
|---|---|
| *type* | Type of sample that needs to be stored. |
| *size* | Size of how big the history buffer is. |

**Returns**

The sample of delay ago.

Here is the call graph for this function:



#### 6.3.3.4 push()

```
template<typename type , size_t size>
type CircularDelay< type, size >::push (
             type input )
```

With this function you can insert a new sample into the buffer.

**Parameters**

| | |
|---|---|
| *input* | Sample to push into. |

**Template Parameters**

| | |
|---|---|
| *type* | Type of sample that needs to be stored. |
| *size* | Size of how big the history buffer is. |

**Returns**

Value that has been pushed.

#### 6.3.3.5 rbegin()

```
template<typename type , size_t size>
reverse_iterator CircularDelay< type, size >::rbegin ( )  [inline]
```

#### 6.3.3.6 rend()

```
template<typename type , size_t size>
reverse_iterator CircularDelay< type, size >::rend ( )  [inline]
```

### 6.3.4 Member Data Documentation

#### 6.3.4.1 data

```
template<typename type , size_t size>
type CircularDelay< type, size >::data[size+1]  [private]
```

#### 6.3.4.2 setIterator

```
template<typename type , size_t size>
iterator CircularDelay< type, size >::setIterator = iterator(data, data)  [private]
```

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

- /home/sitcomlab/Projects/VigiSense/src/CircularDelay.hpp

## 6.4 CircularDelay< type, size >::const_iterator Class Reference

```
#include <CircularDelay.hpp>
```

### Public Types

- typedef const_iterator self_type
- typedef std::bidirectional_iterator_tag iterator_category
- typedef int difference_type

### Public Member Functions

- const_iterator (const CircularDelay< type, size >::const_iterator &it)
- self_type operator++ ()
- self_type operator++ (int)
- self_type operator-- ()
- self_type operator-- (int)
- const type & operator* ()
- const type * operator-> ()
- const type & operator[ ] (unsigned int index)
- bool operator== (const self_type &rhs)
- bool operator!= (const self_type &rhs)

### Private Member Functions

- const_iterator (type *data, type *ptr)

### Private Attributes

- type * data_ = nullptr
- type * ptr_ = nullptr

### Friends

- class CircularDelay

### 6.4.1 Member Typedef Documentation

#### 6.4.1.1 difference_type

```
template<typename type , size_t size>
typedef int CircularDelay< type, size >::const_iterator::difference_type
```

#### 6.4.1.2 iterator_category

```
template<typename type , size_t size>
typedef std::bidirectional_iterator_tag CircularDelay< type, size >::const_iterator::iterator←
_category
```

#### 6.4.1.3 self_type

```
template<typename type , size_t size>
typedef const_iterator CircularDelay< type, size >::const_iterator::self_type
```

### 6.4.2 Constructor & Destructor Documentation

#### 6.4.2.1 const_iterator() [1/2]

```
template<typename type , size_t size>
CircularDelay< type, size >::const_iterator::const_iterator (
             const CircularDelay< type, size >::const_iterator & it )  [inline]
```

#### 6.4.2.2 const_iterator() [2/2]

```
template<typename type , size_t size>
CircularDelay< type, size >::const_iterator::const_iterator (
             type * data,
             type * ptr )  [inline], [private]
```

### 6.4.3 Member Function Documentation

#### 6.4.3.1 operator"!=()

```
template<typename type , size_t size>
bool CircularDelay< type, size >::const_iterator::operator!= (
             const self_type & rhs )  [inline]
```

#### 6.4.3.2 operator*()

```
template<typename type , size_t size>
const type & CircularDelay< type, size >::const_iterator::operator* ( )  [inline]
```

#### 6.4.3.3 operator++() [1/2]

```
template<typename type , size_t size>
self_type CircularDelay< type, size >::const_iterator::operator++ ( )  [inline]
```

#### 6.4.3.4 operator++() [2/2]

```
template<typename type , size_t size>
self_type CircularDelay< type, size >::const_iterator::operator++ (
             int  )  [inline]
```

#### 6.4.3.5 operator--() [1/2]

```
template<typename type , size_t size>
self_type CircularDelay< type, size >::const_iterator::operator-- ( )  [inline]
```

#### 6.4.3.6 operator--() [2/2]

```
template<typename type , size_t size>
self_type CircularDelay< type, size >::const_iterator::operator-- (
             int  )  [inline]
```

#### 6.4.3.7 operator->()

```
template<typename type , size_t size>
const type * CircularDelay< type, size >::const_iterator::operator-> ( )  [inline]
```

#### 6.4.3.8 operator==()

```
template<typename type , size_t size>
bool CircularDelay< type, size >::const_iterator::operator== (
             const self_type & rhs )  [inline]
```

#### 6.4.3.9 operator[]()

```
template<typename type , size_t size>
const type & CircularDelay< type, size >::const_iterator::operator[] (
             unsigned int index )  [inline]
```

### 6.4.4 Friends And Related Symbol Documentation

#### 6.4.4.1 CircularDelay

```
template<typename type , size_t size>
friend class CircularDelay  [friend]
```

### 6.4.5 Member Data Documentation

#### 6.4.5.1 data_

```
template<typename type , size_t size>
type* CircularDelay< type, size >::const_iterator::data_ = nullptr  [private]
```

#### 6.4.5.2 ptr_

```
template<typename type , size_t size>
type* CircularDelay< type, size >::const_iterator::ptr_ = nullptr  [private]
```

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

- /home/sitcomlab/Projects/VigiSense/src/CircularDelay.hpp

## 6.5 CircularDelay< type, size >::const_reverse_iterator Class Reference

```
#include <CircularDelay.hpp>
```

### Public Types

- typedef const_reverse_iterator self_type
- typedef std::bidirectional_iterator_tag iterator_category
- typedef int difference_type

### Public Member Functions

- const_reverse_iterator (const CircularDelay< type, size >::const_reverse_iterator &it)
- self_type operator++ ()
- self_type operator++ (int)
- self_type operator-- ()
- self_type operator-- (int)
- const type & operator∗ ()
- const type ∗ operator-> ()
- const type & operator[ ] (int index)
- bool operator== (const self_type &rhs)
- bool operator!= (const self_type &rhs)

### Private Member Functions

- const_reverse_iterator (type ∗data, type ∗ptr)

### Private Attributes

- type ∗ data_ = nullptr
- type ∗ ptr_ = nullptr

### Friends

- class CircularDelay

### 6.5.1 Member Typedef Documentation

#### 6.5.1.1 difference_type

```
template<typename type , size_t size>
typedef int CircularDelay< type, size >::const_reverse_iterator::difference_type
```

#### 6.5.1.2 iterator_category

```
template<typename type , size_t size>
typedef std::bidirectional_iterator_tag CircularDelay< type, size >::const_reverse_iterator↩
::iterator_category
```

#### 6.5.1.3 self_type

```
template<typename type , size_t size>
typedef const_reverse_iterator CircularDelay< type, size >::const_reverse_iterator::self_type
```

### 6.5.2 Constructor & Destructor Documentation

#### 6.5.2.1 const_reverse_iterator() [1/2]

```
template<typename type , size_t size>
CircularDelay< type, size >::const_reverse_iterator::const_reverse_iterator (
             const CircularDelay< type, size >::const_reverse_iterator & it )  [inline]
```

#### 6.5.2.2 const_reverse_iterator() [2/2]

```
template<typename type , size_t size>
CircularDelay< type, size >::const_reverse_iterator::const_reverse_iterator (
             type * data,
             type * ptr )  [inline], [private]
```

### 6.5.3 Member Function Documentation

#### 6.5.3.1 operator"!=()

```
template<typename type , size_t size>
bool CircularDelay< type, size >::const_reverse_iterator::operator!= (
             const self_type & rhs )  [inline]
```

#### 6.5.3.2 operator*()

```
template<typename type , size_t size>
const type & CircularDelay< type, size >::const_reverse_iterator::operator* ( )  [inline]
```

#### 6.5.3.3 operator++() [1/2]

```
template<typename type , size_t size>
self_type CircularDelay< type, size >::const_reverse_iterator::operator++ ( )  [inline]
```

#### 6.5.3.4 operator++() [2/2]

```
template<typename type , size_t size>
self_type CircularDelay< type, size >::const_reverse_iterator::operator++ (
             int  )  [inline]
```

#### 6.5.3.5 operator--() [1/2]

```
template<typename type , size_t size>
self_type CircularDelay< type, size >::const_reverse_iterator::operator-- ( )  [inline]
```

#### 6.5.3.6 operator--() [2/2]

```
template<typename type , size_t size>
self_type CircularDelay< type, size >::const_reverse_iterator::operator-- (
             int  )  [inline]
```

#### 6.5.3.7 operator->()

```
template<typename type , size_t size>
const type * CircularDelay< type, size >::const_reverse_iterator::operator-> ( )  [inline]
```

#### 6.5.3.8 operator==()

```
template<typename type , size_t size>
bool CircularDelay< type, size >::const_reverse_iterator::operator== (
             const self_type & rhs )  [inline]
```

#### 6.5.3.9 operator[]()

```
template<typename type , size_t size>
const type & CircularDelay< type, size >::const_reverse_iterator::operator[] (
             int index )  [inline]
```

### 6.5.4 Friends And Related Symbol Documentation

#### 6.5.4.1 CircularDelay

```
template<typename type , size_t size>
friend class CircularDelay  [friend]
```

### 6.5.5 Member Data Documentation

#### 6.5.5.1 data_

```
template<typename type , size_t size>
type* CircularDelay< type, size >::const_reverse_iterator::data_ = nullptr  [private]
```

#### 6.5.5.2 ptr_

```
template<typename type , size_t size>
type* CircularDelay< type, size >::const_reverse_iterator::ptr_ = nullptr  [private]
```

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

- /home/sitcomlab/Projects/VigiSense/src/CircularDelay.hpp

## 6.6 contact Struct Reference

**Public Attributes**

- string name
- string email
- long long phoneNum

### 6.6.1 Member Data Documentation

#### 6.6.1.1 email

```
string contact::email
```

#### 6.6.1.2 name

```
string contact::name
```

#### 6.6.1.3 phoneNum

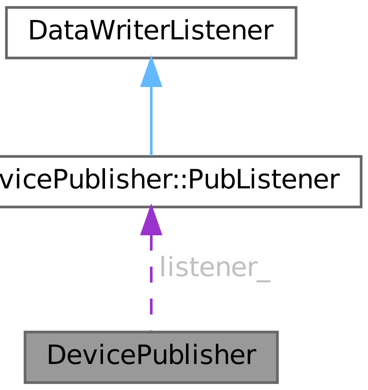
```
long long contact::phoneNum
```

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

- /home/sitcomlab/Projects/VigiSense/src/User.cpp

## 6.7 DevicePublisher Class Reference

Collaboration diagram for DevicePublisher:



#### Classes

- class PubListener

#### Public Member Functions

- DevicePublisher ()
- virtual ∼DevicePublisher ()
- bool init ()

  *Initialize the publisher.*
- bool publish (alert &hello)

  *Send a publication.*

#### Private Attributes

- DomainParticipant ∗ participant_ = nullptr
- Publisher ∗ publisher_ = nullptr
- Topic ∗ topic_ = nullptr
- DataWriter ∗ writer_ = nullptr
- TypeSupport type_
- DevicePublisher::PubListener listener_

### 6.7.1 Constructor & Destructor Documentation

#### 6.7.1.1 DevicePublisher()

```
DevicePublisher::DevicePublisher ( )  [inline]
```

#### 6.7.1.2 ∼DevicePublisher()

```
virtual DevicePublisher::∼DevicePublisher ( )  [inline], [virtual]
```

### 6.7.2 Member Function Documentation

#### 6.7.2.1 init()

```
bool DevicePublisher::init ( )  [inline]
```

Initialize the publisher.

#### 6.7.2.2 publish()

```
bool DevicePublisher::publish (
             alert & hello )  [inline]
```

Send a publication.

### 6.7.3 Member Data Documentation

#### 6.7.3.1 listener_

```
DevicePublisher::PubListener DevicePublisher::listener_  [private]
```

#### 6.7.3.2 participant_

```
DomainParticipant* DevicePublisher::participant_ = nullptr  [private]
```

#### 6.7.3.3 publisher_

```
Publisher* DevicePublisher::publisher_ = nullptr  [private]
```

#### 6.7.3.4 topic_

```
Topic* DevicePublisher::topic_ = nullptr  [private]
```

#### 6.7.3.5 type_

```
TypeSupport DevicePublisher::type_  [private]
```

#### 6.7.3.6 writer_

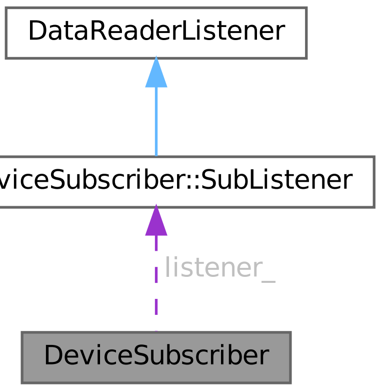
```
DataWriter* DevicePublisher::writer_ = nullptr  [private]
```

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

- /home/sitcomlab/Projects/VigiSense/src/DevicePublisher.cpp

## 6.8 DeviceSubscriber Class Reference

Collaboration diagram for DeviceSubscriber:



**Classes**

- class SubListener

**Public Member Functions**

- DeviceSubscriber ()
- virtual ∼DeviceSubscriber ()
- bool init ()

    *Initialize the subscriber.*

**Private Attributes**

- DomainParticipant ∗ participant_ = nullptr
- Subscriber ∗ subscriber_ = nullptr
- DataReader ∗ reader_ = nullptr
- Topic ∗ topic_ = nullptr
- TypeSupport type_
- DeviceSubscriber::SubListener listener_

### 6.8.1 Constructor & Destructor Documentation

#### 6.8.1.1 DeviceSubscriber()

```
DeviceSubscriber::DeviceSubscriber ( )  [inline]
```

#### 6.8.1.2 ∼DeviceSubscriber()

```
virtual DeviceSubscriber::∼DeviceSubscriber ( )  [inline], [virtual]
```

### 6.8.2 Member Function Documentation

#### 6.8.2.1 init()

```
bool DeviceSubscriber::init ( )  [inline]
```

Initialize the subscriber.

### 6.8.3 Member Data Documentation

#### 6.8.3.1 listener_

```
DeviceSubscriber::SubListener DeviceSubscriber::listener_  [private]
```

#### 6.8.3.2 participant_

```
DomainParticipant* DeviceSubscriber::participant_ = nullptr  [private]
```

#### 6.8.3.3 reader_

```
DataReader* DeviceSubscriber::reader_ = nullptr  [private]
```

#### 6.8.3.4 subscriber_

```
Subscriber* DeviceSubscriber::subscriber_ = nullptr  [private]
```

#### 6.8.3.5 topic_

```
Topic* DeviceSubscriber::topic_ = nullptr  [private]
```

#### 6.8.3.6 type_

```
TypeSupport DeviceSubscriber::type_  [private]
```

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

- /home/sitcomlab/Projects/VigiSense/src/DeviceSubscriber.cpp

## 6.9 diagnosis::DiagnosesTable Struct Reference

```
#include <Diagnosis.h>
```

**Public Attributes**

- std::vector< DiagnosisRange > Diagnoses

### 6.9.1 Member Data Documentation

#### 6.9.1.1 Diagnoses

```
std::vector<DiagnosisRange> diagnosis::DiagnosesTable::Diagnoses
```

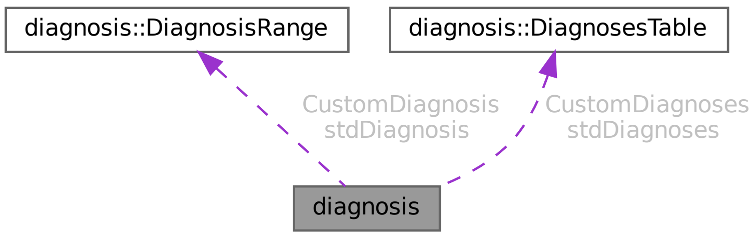
The documentation for this struct was generated from the following file:

- /home/sitcomlab/Projects/VigiSense/src/Diagnosis.h

## 6.10 diagnosis Class Reference

```
#include <Diagnosis.h>
```

Collaboration diagram for diagnosis:



**Classes**

- struct DiagnosesTable
- struct DiagnosisRange

**Public Types**

- typedef struct diagnosis::DiagnosisRange Diagnosis_Range
- typedef struct diagnosis::DiagnosesTable _Diagnoses

**Public Member Functions**

- void SetdiagnosisRanges (float minimum, float maximum, std::string diagnosis)
- std::string determineDiagnosis ()
- void critCheck ()
- void findMinMax ()
- virtual void displayDiagnosis ()
- virtual void critRangeAlert ()

**Public Attributes**

- Diagnosis_Range stdDiagnosis
- Diagnosis_Range CustomDiagnosis
- _Diagnoses stdDiagnoses
- _Diagnoses CustomDiagnoses
- float CriticalLow
- float CriticalHigh

## 6.10.1 Member Typedef Documentation

### 6.10.1.1 _Diagnoses

```
typedef struct diagnosis::DiagnosesTable diagnosis::_Diagnoses
```

### 6.10.1.2 Diagnosis_Range

```
typedef struct diagnosis::DiagnosisRange diagnosis::Diagnosis_Range
```

## 6.10.2 Member Function Documentation

### 6.10.2.1 critCheck()

```
void diagnosis::critCheck ( )  [inline]
```

### 6.10.2.2 critRangeAlert()

```
virtual void diagnosis::critRangeAlert ( )  [virtual]
```

#### 6.10.2.3 determineDiagnosis()

```
std::string diagnosis::determineDiagnosis ( )  [inline]
```

#### 6.10.2.4 displayDiagnosis()

```
virtual void diagnosis::displayDiagnosis ( )  [virtual]
```

#### 6.10.2.5 findMinMax()

```
void diagnosis::findMinMax ( )  [inline]
```

#### 6.10.2.6 SetdiagnosisRanges()

```
void diagnosis::SetdiagnosisRanges (
             float minimum,
             float maximum,
             std::string diagnosis )
```

### 6.10.3 Member Data Documentation

#### 6.10.3.1 CriticalHigh

```
float diagnosis::CriticalHigh
```

#### 6.10.3.2 CriticalLow

```
float diagnosis::CriticalLow
```

#### 6.10.3.3 CustomDiagnoses

```
_Diagnoses diagnosis::CustomDiagnoses
```

#### 6.10.3.4 CustomDiagnosis

```
Diagnosis_Range diagnosis::CustomDiagnosis
```

#### 6.10.3.5 stdDiagnoses

```
_Diagnoses diagnosis::stdDiagnoses
```

#### 6.10.3.6 stdDiagnosis

```
Diagnosis_Range diagnosis::stdDiagnosis
```

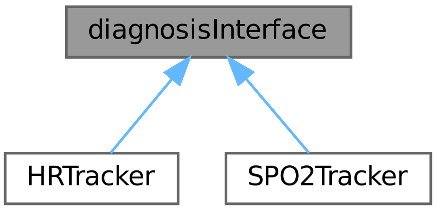
The documentation for this class was generated from the following files:

- /home/sitcomlab/Projects/VigiSense/src/Diagnosis.h
- /home/sitcomlab/Projects/VigiSense/src/Diagnosis.cpp

## 6.11 diagnosisInterface Class Reference

```
#include <DiagnosisInterface.h>
```

Inheritance diagram for diagnosisInterface:



#### Public Member Functions

- virtual void start ()=0
- virtual void stop ()=0
- virtual void ping ()=0
- virtual int getVal ()=0

#### Static Public Member Functions

- static std::string determineSymptom (std::vector< symptomRange > symptomRanges, int val)

#### Protected Attributes

- std::vector< symptomRange > symptomRanges

### 6.11.1 Member Function Documentation

#### 6.11.1.1 determineSymptom()

```
static std::string diagnosisInterface::determineSymptom (
             std::vector< symptomRange > symptomRanges,
             int val )  [inline], [static]
```

#### 6.11.1.2 getVal()

```
virtual int diagnosisInterface::getVal ( ) [pure virtual]
```

Implemented in HRTracker, and SPO2Tracker.

#### 6.11.1.3 ping()

```
virtual void diagnosisInterface::ping ( ) [pure virtual]
```

Implemented in HRTracker, and SPO2Tracker.

#### 6.11.1.4 start()

```
virtual void diagnosisInterface::start ( ) [pure virtual]
```

Implemented in HRTracker, and SPO2Tracker.

#### 6.11.1.5 stop()

```
virtual void diagnosisInterface::stop ( ) [pure virtual]
```

Implemented in HRTracker, and SPO2Tracker.

### 6.11.2 Member Data Documentation

#### 6.11.2.1 symptomRanges

```
std::vector<symptomRange> diagnosisInterface::symptomRanges [protected]
```

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

- /home/sitcomlab/Projects/VigiSense/src/DiagnosisInterface.h

## 6.12 diagnosis::DiagnosisRange Struct Reference

```
#include <Diagnosis.h>
```

**Public Attributes**

- float min
- float max
- std::string diagnosis

### 6.12.1 Member Data Documentation

#### 6.12.1.1 diagnosis

```
std::string diagnosis::DiagnosisRange::diagnosis
```

#### 6.12.1.2 max

```
float diagnosis::DiagnosisRange::max
```

#### 6.12.1.3 min

```
float diagnosis::DiagnosisRange::min
```

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

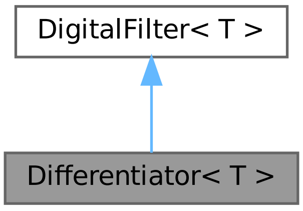
- /home/sitcomlab/Projects/VigiSense/src/Diagnosis.h

## 6.13 Differentiator< T > Class Template Reference

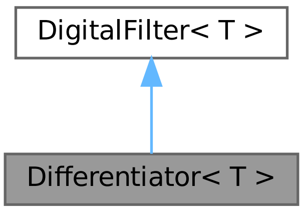
Class for differentiator.

`#include <DigitalFilters.h>`

Inheritance diagram for Differentiator< T >:



Collaboration diagram for Differentiator< T >:



**Public Member Functions**

- Differentiator (T sampleTime)
- T update (T input)
- T getOutput ()

**Private Attributes**

- const T sampleTime
- T y = 0
- T x1 = 0

## 6.13.1 Detailed Description

**template<typename T>**
**class Differentiator< T >**

Class for differentiator.

## 6.13.2 Constructor & Destructor Documentation

### 6.13.2.1 Differentiator()

```
template<typename T >
Differentiator< T >::Differentiator (
             T sampleTime )  [inline]
```

## 6.13.3 Member Function Documentation

### 6.13.3.1 getOutput()

```
template<typename T >
T Differentiator< T >::getOutput ( )  [inline], [virtual]
```

Implements DigitalFilter< T >.

### 6.13.3.2 update()

```
template<typename T >
T Differentiator< T >::update (
             T input )  [inline], [virtual]
```

Implements DigitalFilter< T >.

### 6.13.4 Member Data Documentation

#### 6.13.4.1 sampleTime

```
template<typename T >
const T Differentiator< T >::sampleTime  [private]
```

#### 6.13.4.2 x1

```
template<typename T >
T Differentiator< T >::x1 = 0  [private]
```

#### 6.13.4.3 y

```
template<typename T >
T Differentiator< T >::y = 0  [private]
```

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

- /home/sitcomlab/Projects/VigiSense/src/DigitalFilters.h

## 6.14 DigitalFilter< Type > Class Template Reference

Abstract base class for digital moving filters.

```
#include <DigitalFilters.h>
```

**Public Member Functions**

- virtual Type update (Type newValue)=0
- virtual Type getOutput ()=0

### 6.14.1 Detailed Description

**template<typename Type>**
**class DigitalFilter< Type >**

Abstract base class for digital moving filters.

```
        Moving filter are real time filter used for applications where
        continuous filtering is necessary as it can be part of an control
        system.
```

**Template Parameters**

| | |
|---|---|
| *Type* | Floating point type used. |

## 6.14.2 Member Function Documentation

### 6.14.2.1 getOutput()

```
template<typename Type >
virtual Type DigitalFilter< Type >::getOutput ( )  [pure virtual]
```

Implemented in Differentiator< T >, LowPassFilter, LowPassFilter2, HighPassFilter, HighPassFilter3, LowPassFilter3, LowPassFilter3MatchedZ, and LowPassFilter3DiffApprox.

### 6.14.2.2 update()

```
template<typename Type >
virtual Type DigitalFilter< Type >::update (
              Type newValue )  [pure virtual]
```

Implemented in LowPassFilter, LowPassFilter2, HighPassFilter, HighPassFilter3, LowPassFilter3, LowPassFilter3MatchedZ, LowPassFilter3DiffApprox, and Differentiator< T >.

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

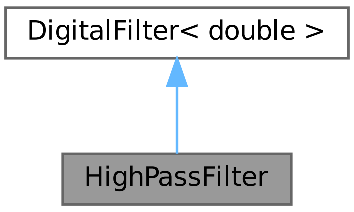
- /home/sitcomlab/Projects/VigiSense/src/DigitalFilters.h

## 6.15 HighPassFilter Class Reference

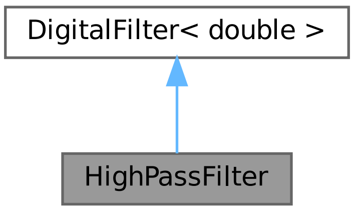
Class for high pass filter using bilinear transform.

```
#include <DigitalFilters.h>
```

Inheritance diagram for HighPassFilter:



Collaboration diagram for HighPassFilter:



#### Public Member Functions

- HighPassFilter (double idt, double omega_c)

  *Constructor to set sample time and the tau constant.*
- double update (double newValue) final

  *Update function to push new value into the low pass filter.*
- double getOutput () final

  *Gets the output.*
- void configOutput (double newOutput)

  *Force the output to a desired value.*
- const double ∗ outputPointer ()

#### Private Attributes

- const double amplFac
- const double y1c
- const double dt
- double x1 = 0
- double output = 0

## 6.15.1 Detailed Description

Class for high pass filter using bilinear transform.

## 6.15.2 Constructor & Destructor Documentation

### 6.15.2.1 HighPassFilter()

```
HighPassFilter::HighPassFilter (
             double idt,
             double omega_c )  [inline]
```

Constructor to set sample time and the tau constant.

**Parameters**

| | | |
|---|---|---|
| in | *idt* | Sample time for the low pass filter |
| in | *itua↩ _c* | Or $\tau_c$ The time constant for the filter. Note that $\tau_c = \frac{1}{2pif_c}$ where $f_c$ is the cutoff frequency |

## 6.15.3 Member Function Documentation

### 6.15.3.1 configOutput()

```
void HighPassFilter::configOutput (
             double newOutput )  [inline]
```

Force the output to a desired value.

X

```
        This can be useful when the output needs to be forced in case
        of extreme inputs or such
```

**Parameters**

| | | |
|---|---|---|
| in | *newOutput* | The new output |

### 6.15.3.2 getOutput()

```
double HighPassFilter::getOutput ( )  [inline], [final], [virtual]
```

Gets the output.

**Returns**

The output.

Implements DigitalFilter< double >.

### 6.15.3.3 outputPointer()

```
const double * HighPassFilter::outputPointer ( )  [inline]
```

### 6.15.3.4 update()

```
double HighPassFilter::update (
             double newValue )  [inline], [final], [virtual]
```

Update function to push new value into the low pass filter.

**Parameters**

|  |  |  |
|---|---|---|
| in | *newValue* | The new value after dt time |

**Returns**

The new output value

Implements DigitalFilter< double >.

### 6.15.4 Member Data Documentation

#### 6.15.4.1 amplFac

```
const double HighPassFilter::amplFac  [private]
```

#### 6.15.4.2 dt

```
const double HighPassFilter::dt  [private]
```

#### 6.15.4.3 output

```
double HighPassFilter::output = 0  [private]
```

#### 6.15.4.4 x1

```
double HighPassFilter::x1 = 0  [private]
```

#### 6.15.4.5 y1c

```
const double HighPassFilter::y1c  [private]
```

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

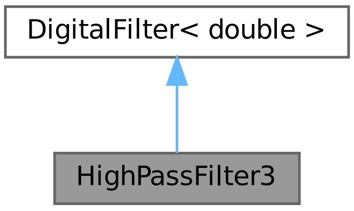
- /home/sitcomlab/Projects/VigiSense/src/DigitalFilters.h

## 6.16 HighPassFilter3 Class Reference

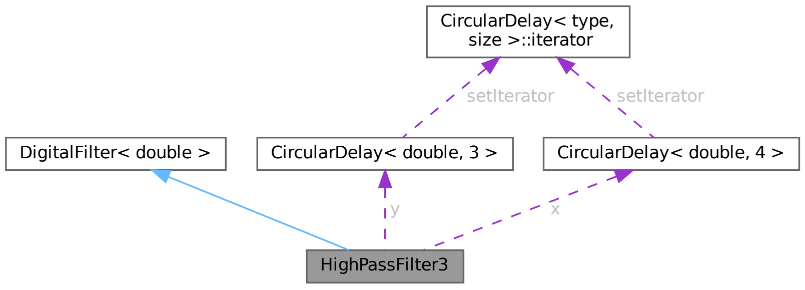
Class for third order high pass filter. This is designed using the bilinear transform.

```
#include <DigitalFilters.h>
```

Inheritance diagram for HighPassFilter3:



Collaboration diagram for HighPassFilter3:


CircularDelay< type, size >::iterator
setIterator
setIterator
DigitalFilter< double >
CircularDelay< double, 3 >
CircularDelay< double, 4 >
y
x
HighPassFilter3

### Public Member Functions

- HighPassFilter3 (double sampleTime, double omega_c, double ioutput=0)
- double update (double newValue) final
- double getOutput () final

### Private Attributes

- const double xc [4]
- const double yc [4]
- CircularDelay< double, 3 > y
- CircularDelay< double, 4 > x

### 6.16.1 Detailed Description

Class for third order high pass filter. This is designed using the bilinear transform.

### 6.16.2 Constructor & Destructor Documentation

#### 6.16.2.1 HighPassFilter3()

```
HighPassFilter3::HighPassFilter3 (
            double sampleTime,
            double omega_c,
            double ioutput = 0 )  [inline]
```

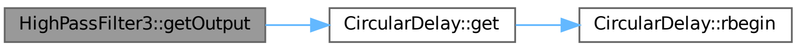
### 6.16.3 Member Function Documentation

#### 6.16.3.1 getOutput()

```
double HighPassFilter3::getOutput ( )  [inline], [final], [virtual]
```

Implements DigitalFilter< double >.

Here is the call graph for this function:

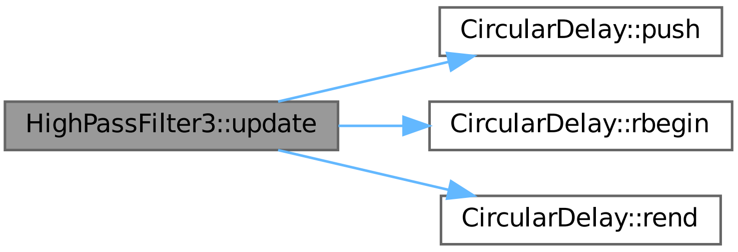


#### 6.16.3.2 update()

```
double HighPassFilter3::update (
            double newValue )  [inline], [final], [virtual]
```

Implements DigitalFilter< double >.

Here is the call graph for this function:



### 6.16.4 Member Data Documentation

#### 6.16.4.1 x

```
CircularDelay<double, 4> HighPassFilter3::x  [private]
```

#### 6.16.4.2 xc

```
const double HighPassFilter3::xc[4]  [private]
```

#### 6.16.4.3 y

```
CircularDelay<double, 3> HighPassFilter3::y  [private]
```

#### 6.16.4.4 yc

```
const double HighPassFilter3::yc[4]  [private]
```

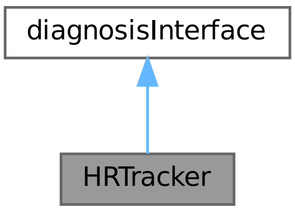
The documentation for this class was generated from the following file:

- /home/sitcomlab/Projects/VigiSense/src/DigitalFilters.h

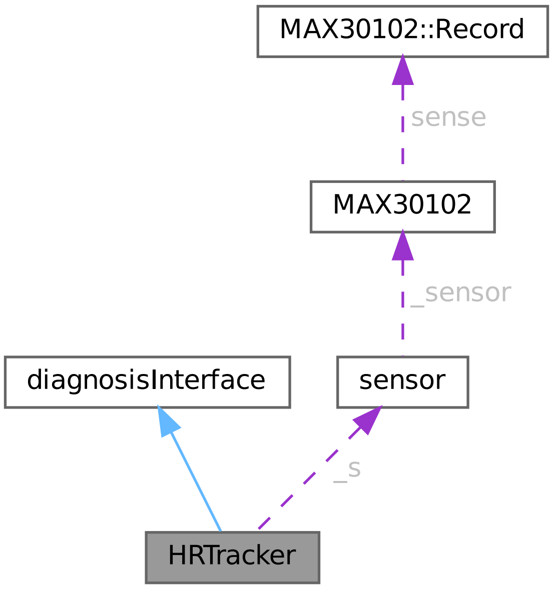
## 6.17 HRTracker Class Reference

```
#include <HRTracker.h>
```

Inheritance diagram for HRTracker:



Collaboration diagram for HRTracker:



**Public Member Functions**

- HRTracker (sensor ∗s)
- ∼HRTracker ()
- void start ()
- void stop ()
- void ping ()
- int getVal ()
- void tracker ()

**Protected Member Functions**

- void pingThread ()

**Protected Attributes**

- sensor ∗ _s
- bool threadRunning = false
- std::vector< symptomRange > symptomRanges

## Protected Attributes inherited from diagnosisInterface

- std::vector< symptomRange > symptomRanges

**Additional Inherited Members**

### Static Public Member Functions inherited from diagnosisInterface

- static std::string determineSymptom (std::vector< symptomRange > symptomRanges, int val)

## 6.17.1 Constructor & Destructor Documentation

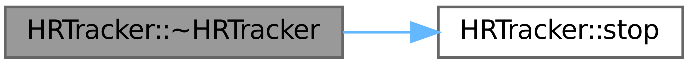
### 6.17.1.1 HRTracker()

```
HRTracker::HRTracker (
          sensor * s )
```

### 6.17.1.2 ∼HRTracker()

```
HRTracker::∼HRTracker ( )
```

Here is the call graph for this function:



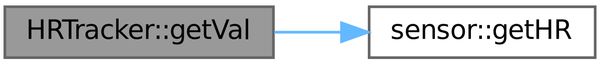
## 6.17.2 Member Function Documentation

### 6.17.2.1 getVal()

```
int HRTracker::getVal ( ) [virtual]
```

Implements diagnosisInterface.

Here is the call graph for this function:

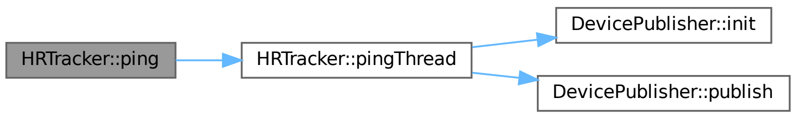


#### 6.17.2.2 ping()

```
void HRTracker::ping ( )  [virtual]
```

Implements diagnosisInterface.

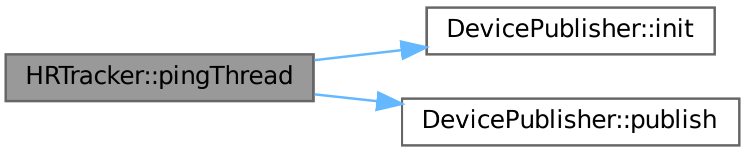
Here is the call graph for this function:



#### 6.17.2.3 pingThread()

```
void HRTracker::pingThread ( )  [protected]
```

Here is the call graph for this function:

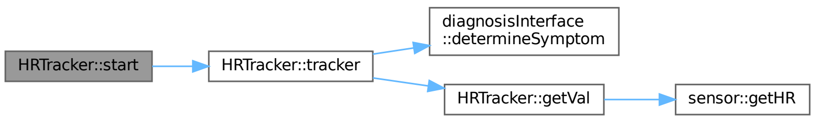


#### 6.17.2.4 start()

```
void HRTracker::start ( )  [virtual]
```

Implements diagnosisInterface.

Here is the call graph for this function:



#### 6.17.2.5 stop()

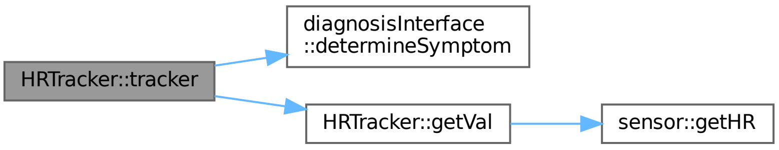
```
void HRTracker::stop ( )  [virtual]
```

Implements diagnosisInterface.

#### 6.17.2.6 tracker()

```
void HRTracker::tracker ( )
```

Here is the call graph for this function:



### 6.17.3 Member Data Documentation

#### 6.17.3.1 _s

```
sensor* HRTracker::_s  [protected]
```

#### 6.17.3.2 symptomRanges

```
std::vector<symptomRange> HRTracker::symptomRanges  [protected]
```

**Initial value:**

```
{
            {0,60,"Bradycardia"},
            {60,100,"Normal resting heart rate"},
            {100,200,"Tachyacardia"}}
```

#### 6.17.3.3 threadRunning

```
bool HRTracker::threadRunning = false  [protected]
```

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

- /home/sitcomlab/Projects/VigiSense/src/HRTracker.h
- /home/sitcomlab/Projects/VigiSense/src/HRTracker.cpp

## 6.18 i2c_msg Struct Reference

```
#include <i2c-dev.h>
```

### Public Attributes

- __u16 addr
- unsigned short flags
- short len
- char ∗ buf

### 6.18.1 Member Data Documentation

#### 6.18.1.1 addr

```
__u16 i2c_msg::addr
```

#### 6.18.1.2 buf

```
char* i2c_msg::buf
```

#### 6.18.1.3 flags

```
unsigned short i2c_msg::flags
```

#### 6.18.1.4 len

```
short i2c_msg::len
```

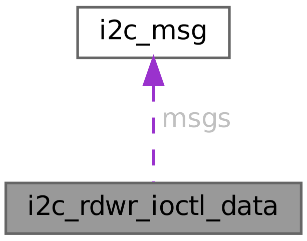
The documentation for this struct was generated from the following file:

- /home/sitcomlab/Projects/VigiSense/src/i2c-dev.h

## 6.19 i2c_rdwr_ioctl_data Struct Reference

```
#include <i2c-dev.h>
```

Collaboration diagram for i2c_rdwr_ioctl_data:



**Public Attributes**

- struct i2c_msg ∗ msgs
- __u32 nmsgs

### 6.19.1 Member Data Documentation

#### 6.19.1.1 msgs

```
struct i2c_msg* i2c_rdwr_ioctl_data::msgs
```

#### 6.19.1.2 nmsgs

```
__u32 i2c_rdwr_ioctl_data::nmsgs
```

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

- /home/sitcomlab/Projects/VigiSense/src/i2c-dev.h

## 6.20 i2c_smbus_data Union Reference

```
#include <i2c-dev.h>
```

**Public Attributes**

- __u8 byte
- __u16 word
- __u8 block [I2C_SMBUS_BLOCK_MAX+2]

### 6.20.1 Member Data Documentation

#### 6.20.1.1 block

```
__u8 i2c_smbus_data::block[I2C_SMBUS_BLOCK_MAX+2]
```

#### 6.20.1.2 byte

```
__u8 i2c_smbus_data::byte
```

#### 6.20.1.3 word

```
__u16 i2c_smbus_data::word
```

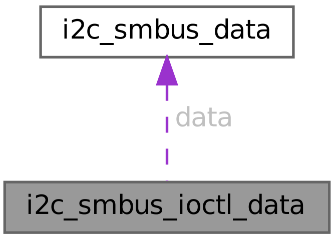
The documentation for this union was generated from the following file:

- /home/sitcomlab/Projects/VigiSense/src/i2c-dev.h

## 6.21 i2c_smbus_ioctl_data Struct Reference

```
#include <i2c-dev.h>
```

Collaboration diagram for i2c_smbus_ioctl_data:



**Public Attributes**

- __u8 read_write
- __u8 command
- __u32 size
- union i2c_smbus_data ∗ data

### 6.21.1 Member Data Documentation

#### 6.21.1.1 command

```
__u8 i2c_smbus_ioctl_data::command
```

#### 6.21.1.2 data

```
union i2c_smbus_data* i2c_smbus_ioctl_data::data
```

#### 6.21.1.3 read_write

```
__u8 i2c_smbus_ioctl_data::read_write
```

#### 6.21.1.4 size

```
__u32 i2c_smbus_ioctl_data::size
```

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

- /home/sitcomlab/Projects/VigiSense/src/i2c-dev.h

## 6.22 CircularDelay< type, size >::iterator Class Reference

```
#include <CircularDelay.hpp>
```

### Public Types

- typedef iterator self_type
- typedef std::bidirectional_iterator_tag iterator_category
- typedef int difference_type

### Public Member Functions

- iterator (const CircularDelay< type, size >::iterator &it)
- self_type operator++ ()
- self_type operator++ (int)
- self_type operator-- ()
- self_type operator-- (int)
- type & operator* ()
- type * operator-> ()
- type & operator[ ] (unsigned int index)
- bool operator== (const self_type &rhs)
- bool operator!= (const self_type &rhs)

### Private Member Functions

- iterator (type *data, type *ptr)

### Private Attributes

- type * data_ = nullptr
- type * ptr_ = nullptr

### Friends

- class CircularDelay

### 6.22.1 Member Typedef Documentation

#### 6.22.1.1 difference_type

```
template<typename type , size_t size>
typedef int CircularDelay< type, size >::iterator::difference_type
```

#### 6.22.1.2 iterator_category

```
template<typename type , size_t size>
typedef std::bidirectional_iterator_tag CircularDelay< type, size >::iterator::iterator_↩
category
```

### 6.22.1.3 self_type

```
template<typename type , size_t size>
typedef iterator CircularDelay< type, size >::iterator::self_type
```

## 6.22.2 Constructor & Destructor Documentation

### 6.22.2.1 iterator() [1/2]

```
template<typename type , size_t size>
CircularDelay< type, size >::iterator::iterator (
             const CircularDelay< type, size >::iterator & it )  [inline]
```

### 6.22.2.2 iterator() [2/2]

```
template<typename type , size_t size>
CircularDelay< type, size >::iterator::iterator (
             type * data,
             type * ptr )  [inline], [private]
```

## 6.22.3 Member Function Documentation

### 6.22.3.1 operator"!=()

```
template<typename type , size_t size>
bool CircularDelay< type, size >::iterator::operator!= (
             const self_type & rhs )  [inline]
```

### 6.22.3.2 operator*()

```
template<typename type , size_t size>
type & CircularDelay< type, size >::iterator::operator* ( )  [inline]
```

### 6.22.3.3 operator++() [1/2]

```
template<typename type , size_t size>
self_type CircularDelay< type, size >::iterator::operator++ ( )  [inline]
```

### 6.22.3.4 operator++() [2/2]

```
template<typename type , size_t size>
self_type CircularDelay< type, size >::iterator::operator++ (
             int  )  [inline]
```

#### 6.22.3.5 operator--() [1/2]

```
template<typename type , size_t size>
self_type CircularDelay< type, size >::iterator::operator-- ( )  [inline]
```

#### 6.22.3.6 operator--() [2/2]

```
template<typename type , size_t size>
self_type CircularDelay< type, size >::iterator::operator-- (
             int  )  [inline]
```

#### 6.22.3.7 operator->()

```
template<typename type , size_t size>
type * CircularDelay< type, size >::iterator::operator-> ( )  [inline]
```

#### 6.22.3.8 operator==()

```
template<typename type , size_t size>
bool CircularDelay< type, size >::iterator::operator== (
             const self_type & rhs )  [inline]
```

#### 6.22.3.9 operator[]()

```
template<typename type , size_t size>
type & CircularDelay< type, size >::iterator::operator[] (
             unsigned int index )  [inline]
```

### 6.22.4 Friends And Related Symbol Documentation

#### 6.22.4.1 CircularDelay

```
template<typename type , size_t size>
friend class CircularDelay  [friend]
```

### 6.22.5 Member Data Documentation

#### 6.22.5.1 data_

```
template<typename type , size_t size>
type* CircularDelay< type, size >::iterator::data_ = nullptr  [private]
```

#### 6.22.5.2 ptr_

```
template<typename type , size_t size>
type* CircularDelay< type, size >::iterator::ptr_ = nullptr  [private]
```

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

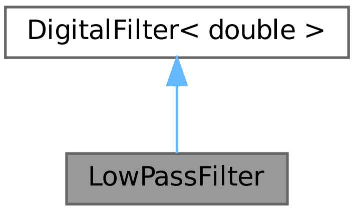
- /home/sitcomlab/Projects/VigiSense/src/CircularDelay.hpp

## 6.23 LowPassFilter Class Reference

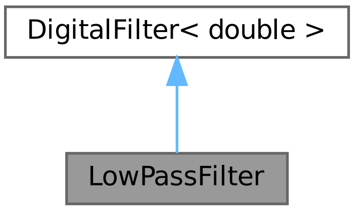
Class for a low pass filter.

```
#include <DigitalFilters.h>
```

Inheritance diagram for LowPassFilter:



Collaboration diagram for LowPassFilter:



**Public Member Functions**

- LowPassFilter (double idt, double omega_c, double ioutput=0)

    *Constructor to set sample time and the tau constant.*
- double update (double newValue) final

    *Update function to push new value into the low pass filter.*
- double getOutput () final

    *Gets the output.*
- void configOutput (double newOutput)

    *Force the output to a desired value.*
- const double $*$ outputPointer ()

**Private Attributes**

- const double epow
- const double dt

    *one time calculation constant*
- double output

### 6.23.1 Detailed Description

Class for a low pass filter.

```
        Design to be a first order Butterworth low pass filter.
        Transformation done using the matched-Z-transform method
```

### 6.23.2 Constructor & Destructor Documentation

#### 6.23.2.1 LowPassFilter()

```
LowPassFilter::LowPassFilter (
            double idt,
            double omega_c,
            double ioutput = 0 )  [inline]
```

Constructor to set sample time and the tau constant.

**Parameters**

| | | |
|---|---|---|
| in | *idt* | Sample time for the low pass filter |
| in | *itua↩_c* | Or $\tau_c$ The time constant for the filter. Note that $\tau_c = \frac{1}{2pif_c}$ where $f_c$ is the cutoff frequency |

### 6.23.3 Member Function Documentation

#### 6.23.3.1 configOutput()

```
void LowPassFilter::configOutput (
             double newOutput )  [inline]
```

Force the output to a desired value.

```
        This can be useful when the output needs to be forced in case
        of extreme inputs or such
```

**Parameters**

| | | |
|---|---|---|
| in | *newOutput* | The new output |

#### 6.23.3.2 getOutput()

```
double LowPassFilter::getOutput ( )  [inline], [final], [virtual]
```

Gets the output.

**Returns**

The output.

Implements DigitalFilter< double >.

#### 6.23.3.3 outputPointer()

```
const double * LowPassFilter::outputPointer ( )  [inline]
```

#### 6.23.3.4 update()

```
double LowPassFilter::update (
             double newValue )  [inline], [final], [virtual]
```

Update function to push new value into the low pass filter.

**Parameters**

| | | |
|---|---|---|
| in | *newValue* | The new value after dt time |

**Returns**

The new output value

Implements DigitalFilter< double >.

### 6.23.4 Member Data Documentation

#### 6.23.4.1 dt

```
const double LowPassFilter::dt  [private]
```

one time calculation constant

#### 6.23.4.2 epow

```
const double LowPassFilter::epow  [private]
```

#### 6.23.4.3 output

```
double LowPassFilter::output  [private]
```

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

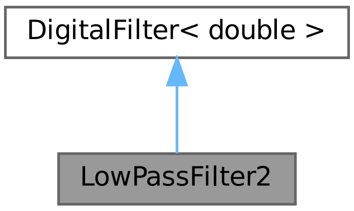
- /home/sitcomlab/Projects/VigiSense/src/DigitalFilters.h

## 6.24 LowPassFilter2 Class Reference

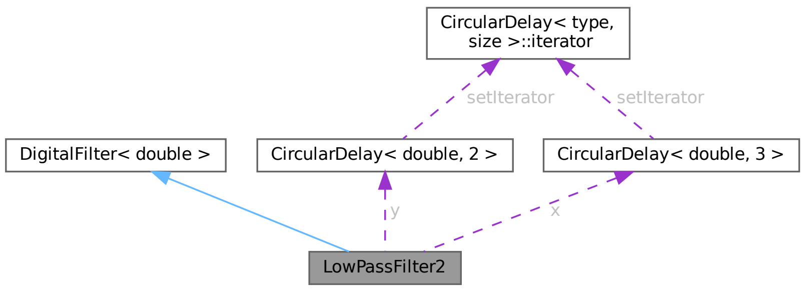
Class for a 2nd order low pass filter.

```
#include <DigitalFilters.h>
```

Inheritance diagram for LowPassFilter2:



Collaboration diagram for LowPassFilter2:


CircularDelay< type, size >::iterator
setIterator
setIterator
DigitalFilter< double >
CircularDelay< double, 2 >
CircularDelay< double, 3 >
y
x
LowPassFilter2

### Public Member Functions

- LowPassFilter2 (double dt, double tau_c, double ioutput=0)

  *Constructor to set sample time and the tau constant.*
- double update (double newValue) final

  *Update function to push new value into the low pass filter.*
- double getOutput () final

  *Gets the output.*
- void configOutput (double newOutput)

  *Force the output to a desired value.*

### Private Attributes

- const double yc [2]
- const double xc [3]
- CircularDelay< double, 2 > y
- CircularDelay< double, 3 > x

## 6.24.1 Detailed Description

Class for a 2nd order low pass filter.

```
        Design to be a 2nd order Butterworth low pass filter.
        Transformation done using the bilinear transform method
```

## 6.24.2 Constructor & Destructor Documentation

### 6.24.2.1 LowPassFilter2()

```
LowPassFilter2::LowPassFilter2 (
             double dt,
             double tau_c,
             double ioutput = 0 )  [inline]
```

Constructor to set sample time and the tau constant.

**Parameters**

| | | |
|---|---|---|
| in | *idt* | Sample time for the low pass filter |
| in | *itua←_c* | Or $\tau_c$ The time constant for the filter. Note that $\tau_c = \frac{1}{2pif_c}$ where $f_c$ is the cutoff frequency |

### 6.24.3 Member Function Documentation

#### 6.24.3.1 configOutput()

```
void LowPassFilter2::configOutput (
             double newOutput )  [inline]
```

Force the output to a desired value.

```
        This can be useful when the output needs to be forced in case
        of extreme inputs or such
```

**Parameters**

| | | |
|---|---|---|
| in | *newOutput* | The new output |

#### 6.24.3.2 getOutput()

```
double LowPassFilter2::getOutput ( )  [inline], [final], [virtual]
```

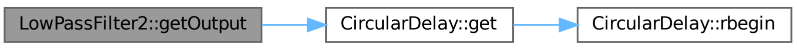
Gets the output.

**Returns**

The output.

Implements DigitalFilter< double >.

Here is the call graph for this function:



#### 6.24.3.3 update()

```
double LowPassFilter2::update (
             double newValue )  [inline], [final], [virtual]
```

Update function to push new value into the low pass filter.

**Parameters**

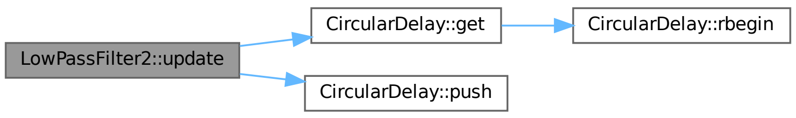
| | | |
|---|---|---|
| in | *newValue* | The new value after dt time |

**Returns**

The new output value

Implements DigitalFilter< double >.

Here is the call graph for this function:



## 6.24.4 Member Data Documentation

### 6.24.4.1 x

```
CircularDelay<double, 3> LowPassFilter2::x  [private]
```

### 6.24.4.2 xc

```
const double LowPassFilter2::xc[3]  [private]
```

### 6.24.4.3 y

```
CircularDelay<double, 2> LowPassFilter2::y  [private]
```

### 6.24.4.4 yc

```
const double LowPassFilter2::yc[2]  [private]
```

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

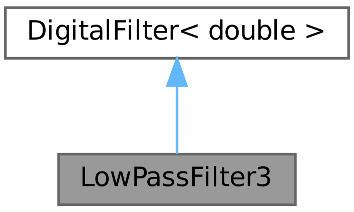
- /home/sitcomlab/Projects/VigiSense/src/DigitalFilters.h

## 6.25 LowPassFilter3 Class Reference

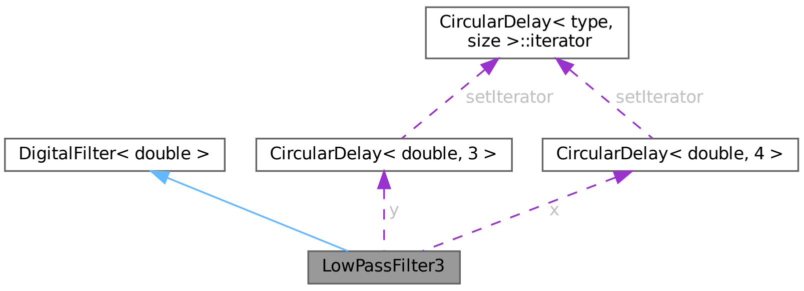
Class for third order high pass filter. This is designed using the bilinear transform.

```
#include <DigitalFilters.h>
```

Inheritance diagram for LowPassFilter3:



Collaboration diagram for LowPassFilter3:


CircularDelay< type, size >::iterator
setIterator
setIterator
DigitalFilter< double >
CircularDelay< double, 3 >
CircularDelay< double, 4 >
y
x
LowPassFilter3

### Public Member Functions

- LowPassFilter3 (long double sampleTime, long double omega_c, long double ioutput=0)
- double update (double newValue) final
- double getOutput () final

### Private Attributes

- const double yc [4]
- const double xc [4]
- CircularDelay< double, 3 > y
- CircularDelay< double, 4 > x

### 6.25.1 Detailed Description

Class for third order high pass filter. This is designed using the bilinear transform.

### 6.25.2 Constructor & Destructor Documentation

#### 6.25.2.1 LowPassFilter3()

```
LowPassFilter3::LowPassFilter3 (
             long double sampleTime,
             long double omega_c,
             long double ioutput = 0 )  [inline]
```

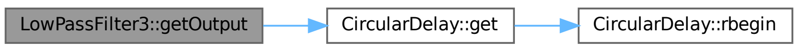
### 6.25.3 Member Function Documentation

#### 6.25.3.1 getOutput()

```
double LowPassFilter3::getOutput ( )  [inline], [final], [virtual]
```

Implements DigitalFilter< double >.

Here is the call graph for this function:

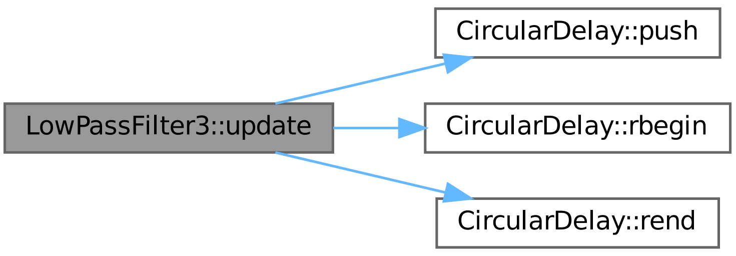


#### 6.25.3.2 update()

```
double LowPassFilter3::update (
             double newValue )  [inline], [final], [virtual]
```

Implements DigitalFilter< double >.

Here is the call graph for this function:



### 6.25.4 Member Data Documentation

#### 6.25.4.1 x

```
CircularDelay<double, 4> LowPassFilter3::x  [private]
```

#### 6.25.4.2 xc

```
const double LowPassFilter3::xc[4]  [private]
```

#### 6.25.4.3 y

```
CircularDelay<double, 3> LowPassFilter3::y  [private]
```

#### 6.25.4.4 yc

```
const double LowPassFilter3::yc[4]  [private]
```

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

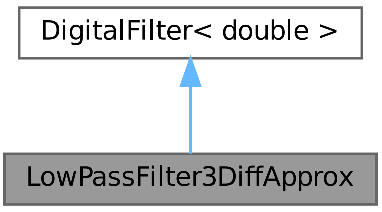
- /home/sitcomlab/Projects/VigiSense/src/DigitalFilters.h

## 6.26 LowPassFilter3DiffApprox Class Reference

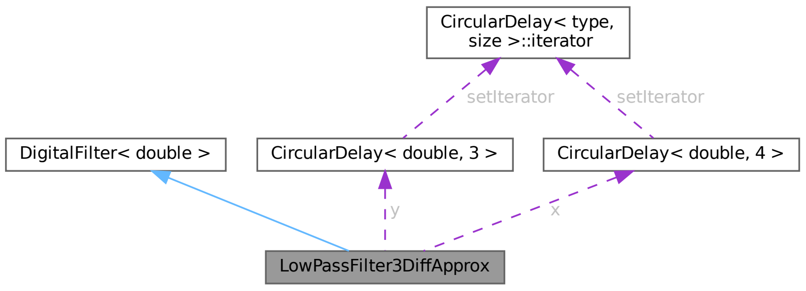
Class for third order high pass filter. This is designed using the approximated differtial approuch where s=(Z-1)/(Z∗T).

```
#include <DigitalFilters.h>
```

Inheritance diagram for LowPassFilter3DiffApprox:



Collaboration diagram for LowPassFilter3DiffApprox:


CircularDelay< type, size >::iterator
setIterator
setIterator
DigitalFilter< double >
CircularDelay< double, 3 >
CircularDelay< double, 4 >
y
x
LowPassFilter3DiffApprox

### Public Member Functions

- LowPassFilter3DiffApprox (double sampleTime, double omega_c, double ioutput=0)
- double update (double newValue) final
- double getOutput () final

### Private Attributes

- const double xc [4]
- const double yc [4]
- CircularDelay< double, 3 > y
- CircularDelay< double, 4 > x

## 6.26.1 Detailed Description

Class for third order high pass filter. This is designed using the approximated differtial approuch where s=(Z-1)/(Z∗T).

## 6.26.2 Constructor & Destructor Documentation

### 6.26.2.1 LowPassFilter3DiffApprox()

```
LowPassFilter3DiffApprox::LowPassFilter3DiffApprox (
            double sampleTime,
            double omega_c,
            double ioutput = 0 )  [inline]
```

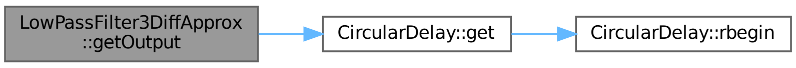
### 6.26.3 Member Function Documentation

#### 6.26.3.1 getOutput()

```
double LowPassFilter3DiffApprox::getOutput ( )  [inline], [final], [virtual]
```

Implements DigitalFilter< double >.

Here is the call graph for this function:

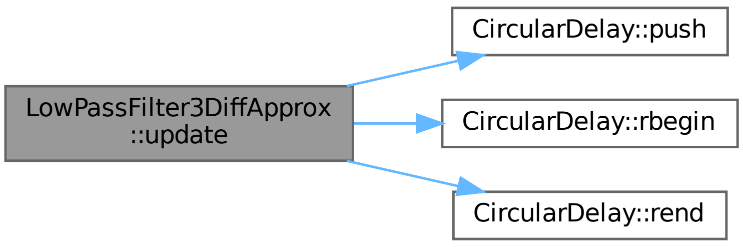


#### 6.26.3.2 update()

```
double LowPassFilter3DiffApprox::update (
             double newValue )  [inline], [final], [virtual]
```

Implements DigitalFilter< double >.

Here is the call graph for this function:



### 6.26.4 Member Data Documentation

#### 6.26.4.1 x

```
CircularDelay<double, 4> LowPassFilter3DiffApprox::x  [private]
```

#### 6.26.4.2 xc

```
const double LowPassFilter3DiffApprox::xc[4]  [private]
```

#### 6.26.4.3 y

```
CircularDelay<double, 3> LowPassFilter3DiffApprox::y  [private]
```

#### 6.26.4.4 yc

```
const double LowPassFilter3DiffApprox::yc[4]  [private]
```

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

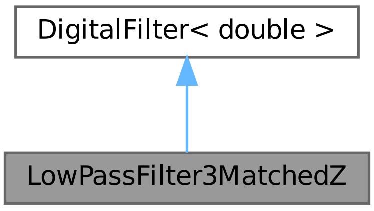
- /home/sitcomlab/Projects/VigiSense/src/DigitalFilters.h

## 6.27 LowPassFilter3MatchedZ Class Reference

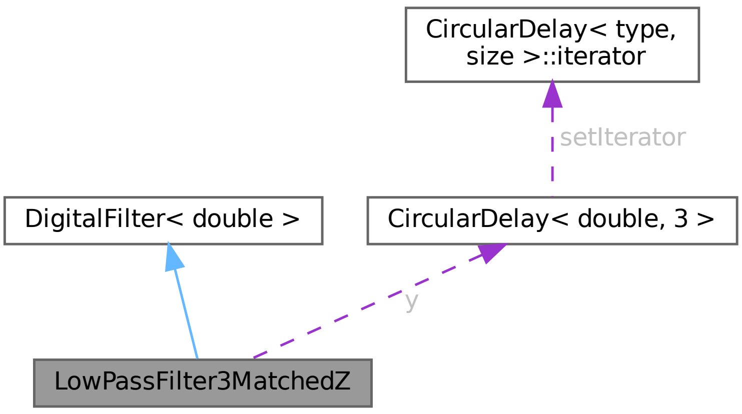
Class for third order high pass filter. This is designed using the matched Z transform.

```
#include <DigitalFilters.h>
```

Inheritance diagram for LowPassFilter3MatchedZ:



Collaboration diagram for LowPassFilter3MatchedZ:


CircularDelay< type, size >::iterator
setIterator
DigitalFilter< double >
CircularDelay< double, 3 >
y
LowPassFilter3MatchedZ

**Public Member Functions**

- LowPassFilter3MatchedZ (long double sampleTime, long double omega_c)
- double update (double newValue) final
- double getOutput () final

**Private Attributes**

- const double amplFac
- const double yc [3]
- CircularDelay< double, 3 > y

## 6.27.1 Detailed Description

Class for third order high pass filter. This is designed using the matched Z transform.

## 6.27.2 Constructor & Destructor Documentation

### 6.27.2.1 LowPassFilter3MatchedZ()

```
LowPassFilter3MatchedZ::LowPassFilter3MatchedZ (
             long double sampleTime,
             long double omega_c )  [inline]
```

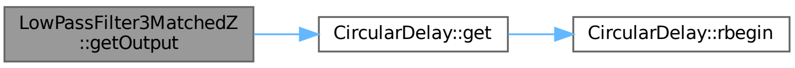
## 6.27.3 Member Function Documentation

### 6.27.3.1 getOutput()

```
double LowPassFilter3MatchedZ::getOutput ( )  [inline], [final], [virtual]
```

Implements DigitalFilter< double >.

Here is the call graph for this function:

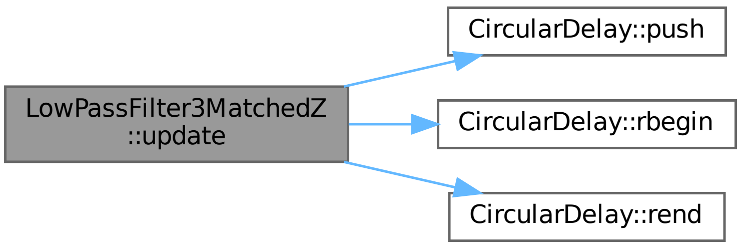


#### 6.27.3.2 update()

```
double LowPassFilter3MatchedZ::update (
             double newValue )  [inline], [final], [virtual]
```

Implements DigitalFilter< double >.

Here is the call graph for this function:



### 6.27.4 Member Data Documentation

#### 6.27.4.1 amplFac

```
const double LowPassFilter3MatchedZ::amplFac  [private]
```

#### 6.27.4.2 y

```
CircularDelay<double, 3> LowPassFilter3MatchedZ::y  [private]
```

#### 6.27.4.3 yc

```
const double LowPassFilter3MatchedZ::yc[3]  [private]
```

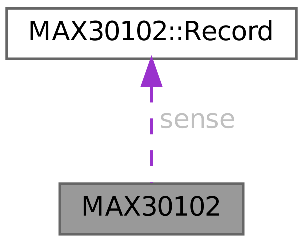
The documentation for this class was generated from the following file:

- /home/sitcomlab/Projects/VigiSense/src/DigitalFilters.h

## 6.28 MAX30102 Class Reference

```
#include <MAX30102.h>
```

Collaboration diagram for MAX30102:



### Classes

- struct Record

### Public Member Functions

- MAX30102 (void)
- int begin (uint32_t i2cSpeed=I2C_SPEED_STANDARD, uint8_t i2cAddr=MAX30102_ADDRESS)
- uint32_t getRed (void)
- uint32_t getIR (void)
- bool safeCheck (uint8_t maxTimeToCheck)
- void wakeUp ()
- void shutDown ()
- void softReset ()
- void setLEDMode (uint8_t mode)
- void setADCRange (uint8_t adcRange)
- void setSampleRate (uint8_t sampleRate)
- void setPulseWidth (uint8_t pulseWidth)
- void setPulseAmplitudeRed (uint8_t value)
- void setPulseAmplitudeIR (uint8_t value)
- void setPulseAmplitudeProximity (uint8_t value)
- void setProximityThreshold (uint8_t thresMSB)
- void enableSlot (uint8_t slotNumber, uint8_t device)
- void disableSlots (void)
- uint8_t getINT1 (void)
- uint8_t getINT2 (void)
- void enableAFULL (void)
- void disableAFULL (void)
- void enableDATARDY (void)
- void disableDATARDY (void)
- void enableALCOVF (void)
- void disableALCOVF (void)

- void enablePROXINT (void)
- void disablePROXINT (void)
- void enableDIETEMPRDY (void)
- void disableDIETEMPRDY (void)
- void setFIFOAverage (uint8_t samples)
- void enableFIFORollover ()
- void disableFIFORollover ()
- void setFIFOAlmostFull (uint8_t samples)
- uint16_t check (void)
- uint8_t available (void)
- void nextSample (void)
- uint32_t getFIFORed (void)
- uint32_t getFIFOIR (void)
- uint8_t getWritePointer (void)
- uint8_t getReadPointer (void)
- void clearFIFO (void)
- void setPROXINTTHRESH (uint8_t val)
- float readTemperature ()
- float readTemperatureF ()
- uint8_t getRevisionID ()
- uint8_t readPartID ()
- virtual void hasSample ()
- void setup (uint8_t powerLevel=0x1F, uint8_t sampleAverage=4, uint8_t ledMode=2, int sampleRate=400, int pulseWidth=411, int adcRange=4096)

### Private Types

- typedef struct MAX30102::Record sense_struct

### Private Member Functions

- void readRevisionID ()
- void bitMask (uint8_t reg, uint8_t mask, uint8_t thing)
- std::vector< uint8_t > readMany (uint8_t address, uint8_t length)
- void dataReady ()

### Static Private Member Functions

- static void gpioISR (int, int, uint32_t, void ∗userdata)

### Private Attributes

- int _i2c
- uint8_t _i2caddr
- uint8_t activeLEDs
- uint8_t revisionID
- sense_struct sense

## 6.28.1 Member Typedef Documentation

### 6.28.1.1 sense_struct

```
typedef struct MAX30102::Record MAX30102::sense_struct  [private]
```

## 6.28.2 Constructor & Destructor Documentation

### 6.28.2.1 MAX30102()

```
MAX30102::MAX30102 (
            void  )
```

## 6.28.3 Member Function Documentation

### 6.28.3.1 available()

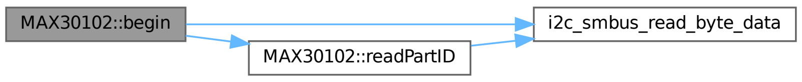
```
uint8_t MAX30102::available (
            void  )
```

Returns the number of samples available.

### 6.28.3.2 begin()

```
int MAX30102::begin (
            uint32_t i2cSpeed = I2C_SPEED_STANDARD,
            uint8_t i2cAddr = MAX30102_ADDRESS )
```

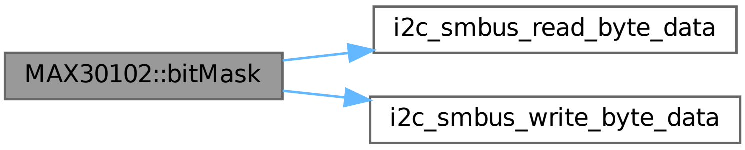
Initializes sensor. Returns negative number on failure. Returns sensor revision on success. Here is the call graph for this function:



### 6.28.3.3 bitMask()

```
void MAX30102::bitMask (
             uint8_t reg,
             uint8_t mask,
             uint8_t thing )  [private]
```

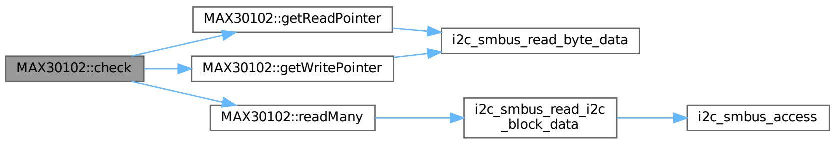
Set certain thing in register. Here is the call graph for this function:



### 6.28.3.4 check()

```
uint16_t MAX30102::check (
             void  )
```

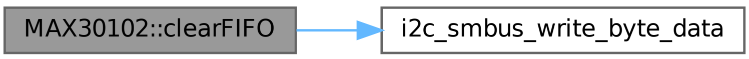
Here is the call graph for this function:



### 6.28.3.5 clearFIFO()

```
void MAX30102::clearFIFO (
             void  )
```

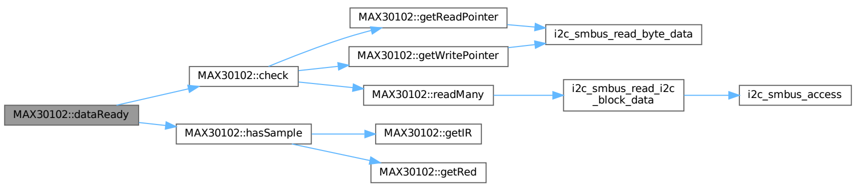
Resets all points to start in a known state. Recommended to clear FIFO before beginning a read. Here is the call graph for this function:



### 6.28.3.6 dataReady()

```
void MAX30102::dataReady ( )  [private]
```

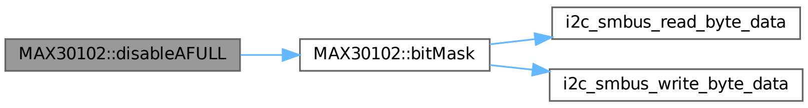
Here is the call graph for this function:



### 6.28.3.7 disableAFULL()

```
void MAX30102::disableAFULL (
             void   )
```

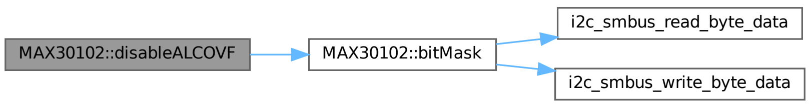
Here is the call graph for this function:



### 6.28.3.8 disableALCOVF()

```
void MAX30102::disableALCOVF (
             void   )
```

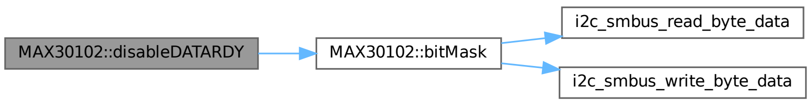
Here is the call graph for this function:



#### 6.28.3.9 disableDATARDY()

```
void MAX30102::disableDATARDY (
             void )
```

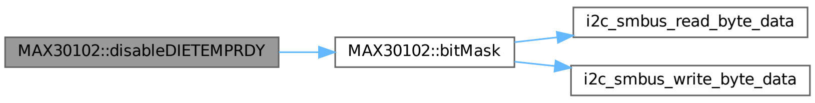
Here is the call graph for this function:



#### 6.28.3.10 disableDIETEMPRDY()

```
void MAX30102::disableDIETEMPRDY (
             void )
```

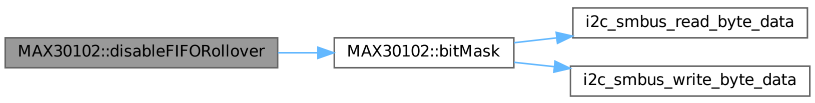
Here is the call graph for this function:



#### 6.28.3.11 disableFIFORollover()

```
void MAX30102::disableFIFORollover (
             void )
```

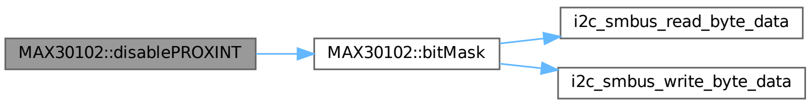
Disable roll over if FIFO over flows. Here is the call graph for this function:



#### 6.28.3.12 disablePROXINT()

```
void MAX30102::disablePROXINT (
            void  )
```

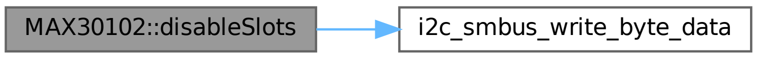
Here is the call graph for this function:



#### 6.28.3.13 disableSlots()

```
void MAX30102::disableSlots (
            void  )
```

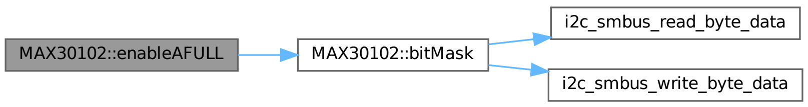
Clears all slot assignments. Here is the call graph for this function:



#### 6.28.3.14 enableAFULL()

```
void MAX30102::enableAFULL (
            void  )
```

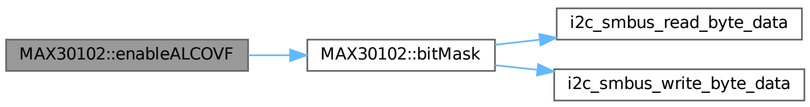
Here is the call graph for this function:



#### 6.28.3.15 enableALCOVF()

```
void MAX30102::enableALCOVF (
             void  )
```

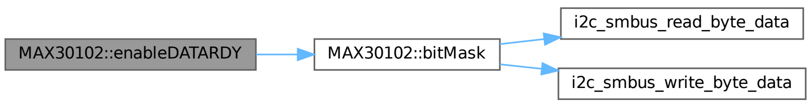
Here is the call graph for this function:



#### 6.28.3.16 enableDATARDY()

```
void MAX30102::enableDATARDY (
             void  )
```

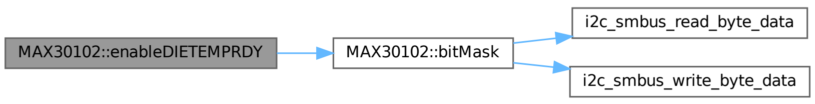
Here is the call graph for this function:



#### 6.28.3.17 enableDIETEMPRDY()

```
void MAX30102::enableDIETEMPRDY (
             void  )
```

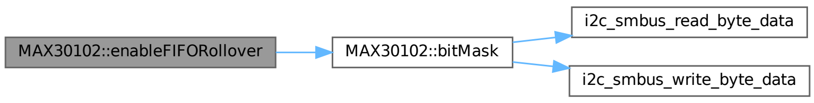
Here is the call graph for this function:



#### 6.28.3.18 enableFIFORollover()

```
void MAX30102::enableFIFORollover (
            void  )
```

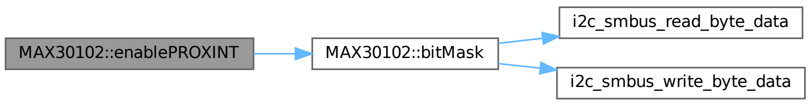
Enable roll over if FIFO over flows. Here is the call graph for this function:



#### 6.28.3.19 enablePROXINT()

```
void MAX30102::enablePROXINT (
            void  )
```

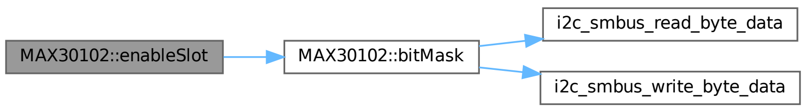
Here is the call graph for this function:



#### 6.28.3.20 enableSlot()

```
void MAX30102::enableSlot (
            uint8_t slotNumber,
            uint8_t device )
```

Given a slot number assign a thing to it. Devices are SLOT_RED_LED or SLOT_RED_PILOT (proximity) Assigning a SLOT_RED_LED will pulse LED Assigning a SLOT_RED_PILOT will ?? Here is the call graph for this function:



#### 6.28.3.21 getFIFOIR()

```
uint32_t MAX30102::getFIFOIR (
             void  )
```

Report the next IR value in FIFO.

#### 6.28.3.22 getFIFORed()

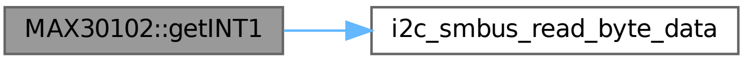
```
uint32_t MAX30102::getFIFORed (
             void  )
```

Report the next Red value in FIFO.

#### 6.28.3.23 getINT1()

```
uint8_t MAX30102::getINT1 (
             void  )
```

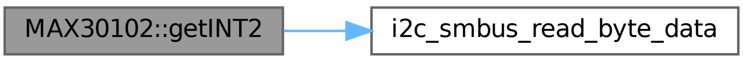
Here is the call graph for this function:



#### 6.28.3.24 getINT2()

```
uint8_t MAX30102::getINT2 (
             void  )
```

Here is the call graph for this function:



#### 6.28.3.25 getIR()

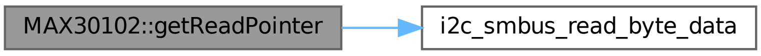
```
uint32_t MAX30102::getIR (
             void )
```

Report the most recent IR value. change to only return without calling checks

#### 6.28.3.26 getReadPointer()

```
uint8_t MAX30102::getReadPointer (
             void )
```

Read the FIFO Read Pointer. Here is the call graph for this function:



#### 6.28.3.27 getRed()

```
uint32_t MAX30102::getRed (
             void )
```

Report the most recent Red value. change to only return without calling checks

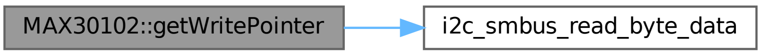
#### 6.28.3.28 getRevisionID()

```
uint8_t MAX30102::getRevisionID ( )
```

#### 6.28.3.29 getWritePointer()

```
uint8_t MAX30102::getWritePointer (
             void )
```

Read the FIFO Write Pointer. Here is the call graph for this function:



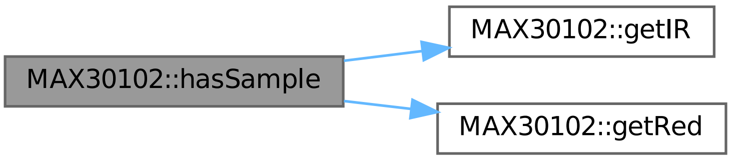
#### 6.28.3.30 gpioISR()

```
static void MAX30102::gpioISR (
             int ,
             int ,
             uint32_t ,
             void * userdata )  [inline], [static], [private]
```

#### 6.28.3.31 hasSample()

```
void MAX30102::hasSample ( )  [virtual]
```

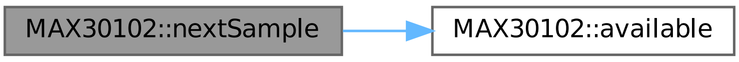
Here is the call graph for this function:



#### 6.28.3.32 nextSample()

```
void MAX30102::nextSample (
             void  )
```

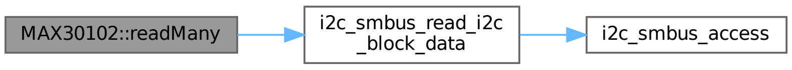
Advance the tail. Here is the call graph for this function:



### 6.28.3.33 readMany()

```
std::vector< uint8_t > MAX30102::readMany (
            uint8_t address,
            uint8_t length ) [private]
```

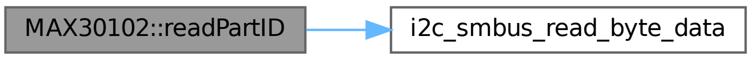
Read multiple bytes from register. Here is the call graph for this function:



### 6.28.3.34 readPartID()

```
uint8_t MAX30102::readPartID ( )
```

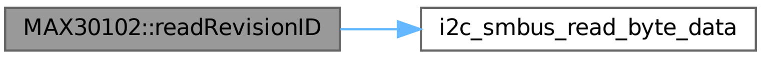
Here is the call graph for this function:



### 6.28.3.35 readRevisionID()

```
void MAX30102::readRevisionID ( ) [private]
```

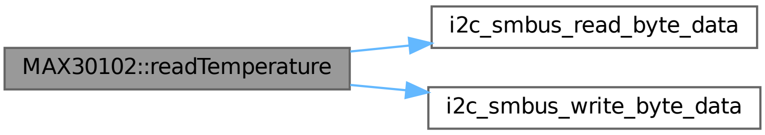
Here is the call graph for this function:



#### 6.28.3.36 readTemperature()

```
float MAX30102::readTemperature ( )
```

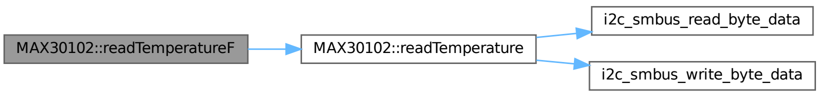
Die Temperature. Returns temperature in C. Here is the call graph for this function:



#### 6.28.3.37 readTemperatureF()

```
float MAX30102::readTemperatureF ( )
```

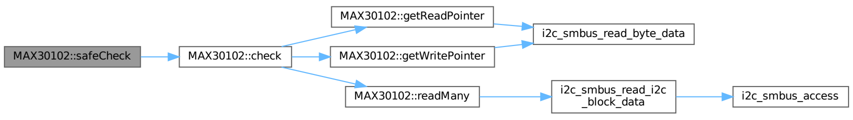
Returns die temperature in F. Here is the call graph for this function:


MAX30102::readTemperatureF
MAX30102::readTemperature
i2c_smbus_read_byte_data
i2c_smbus_write_byte_data

#### 6.28.3.38 safeCheck()

```
bool MAX30102::safeCheck (
             uint8_t maxTimeToCheck )
```

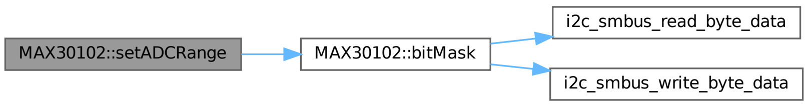
Check for new data but give up after a certain amount of time. Returns true if new data was found. Returns false if new data was not found. Here is the call graph for this function:



#### 6.28.3.39 setADCRange()

```
void MAX30102::setADCRange (
            uint8_t adcRange )
```

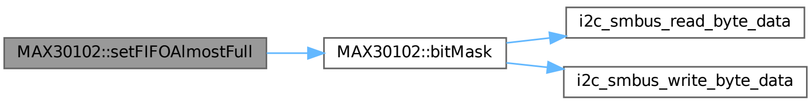
Sets ADC Range. Available ADC Range: 2048, 4096, 8192, 16384 Here is the call graph for this function:



#### 6.28.3.40 setFIFOAlmostFull()

```
void MAX30102::setFIFOAlmostFull (
            uint8_t numberOfSamples )
```

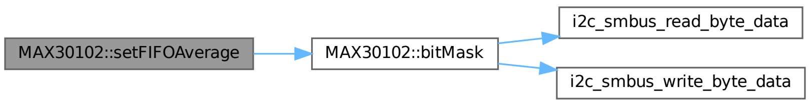
Sets number of samples to trigger the almost full interrupt. Power on deafult is 32 samples. Here is the call graph for this function:



#### 6.28.3.41 setFIFOAverage()

```
void MAX30102::setFIFOAverage (
            uint8_t numberOfSamples )
```

Sets sample average. Here is the call graph for this function:



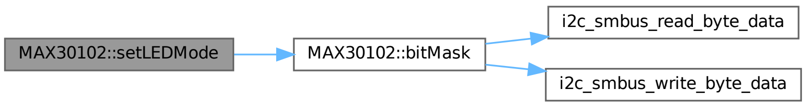
#### 6.28.3.42 setLEDMode()

```
void MAX30102::setLEDMode (
             uint8_t mode )
```

Sets which LEDs are used for sampling.

- Red only
- Red+IR only
- Custom

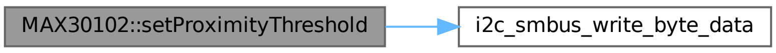
Here is the call graph for this function:



#### 6.28.3.43 setProximityThreshold()

```
void MAX30102::setProximityThreshold (
             uint8_t threshMSB )
```

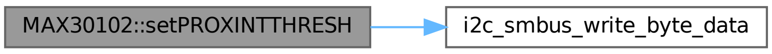
Set the IR ADC count that will trigger the beginning of particle-sensing mode. The threshMSB signifies only the 8 most significant-bits of the ADC count. Here is the call graph for this function:



#### 6.28.3.44 setPROXINTTHRESH()

```
void MAX30102::setPROXINTTHRESH (
             uint8_t val )
```

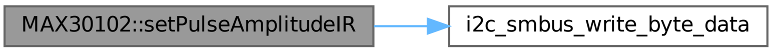
Sets the PROX_INT_THRESHold. Here is the call graph for this function:



#### 6.28.3.45 setPulseAmplitudeIR()

```
void MAX30102::setPulseAmplitudeIR (
            uint8_t amplitude )
```

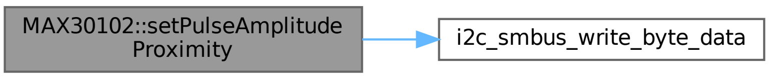
Sets IR LED Pulse Amplitude. Here is the call graph for this function:



#### 6.28.3.46 setPulseAmplitudeProximity()

```
void MAX30102::setPulseAmplitudeProximity (
            uint8_t value )
```

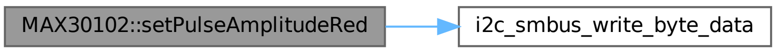
Here is the call graph for this function:


MAX30102::setPulseAmplitude Proximity
i2c_smbus_write_byte_data

#### 6.28.3.47 setPulseAmplitudeRed()

```
void MAX30102::setPulseAmplitudeRed (
            uint8_t amplitude )
```

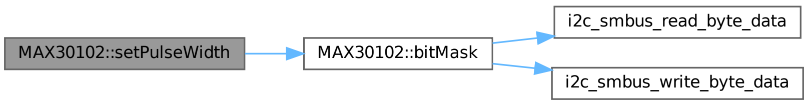
Sets Red LED Pulse Amplitude. Here is the call graph for this function:



#### 6.28.3.48 setPulseWidth()

```
void MAX30102::setPulseWidth (
             uint8_t pulseWidth )
```

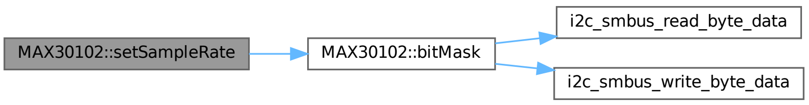
Sets Pulse Width. Available Pulse Width: 69, 188, 215, 411 Here is the call graph for this function:



#### 6.28.3.49 setSampleRate()

```
void MAX30102::setSampleRate (
             uint8_t sampleRate )
```

Sets Sample Rate. Available Sample Rates: 50, 100, 200, 400, 800, 1000, 1600, 3200 Here is the call graph for this function:

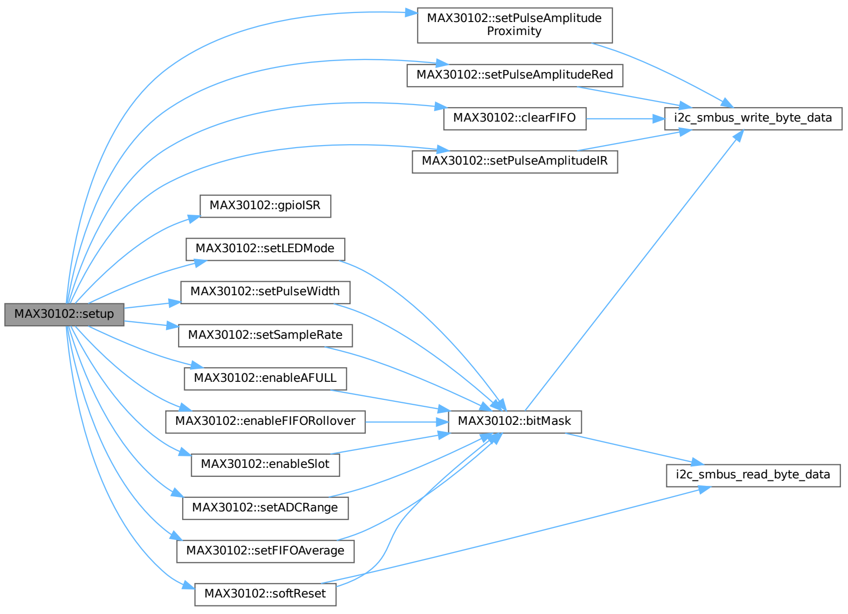


#### 6.28.3.50 setup()

```
void MAX30102::setup (
             uint8_t powerLevel = 0x1F,
             uint8_t sampleAverage = 4,
             uint8_t ledMode = 2,
             int sampleRate = 400,
             int pulseWidth = 411,
             int adcRange = 4096 )
```

Setup the Sensor use led mode 2 where ir and red is used. Default averages 4 samples, with samepleRate of 400, resulting in effective sampleRate of 100, giving a period of 10ms between each averaged sample. Meaning that the 32 sample FIFO will fill up in approx 320ms

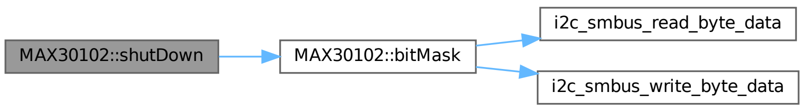
Defaults: powerLevel = 0x1F (6.2mA) sampleAverage = 4 ledMode = 2 sampleRate = 400 pulseWidth = 411 adc↩
Range = 4096 Here is the call graph for this function:



### 6.28.3.51 shutDown()

```
void MAX30102::shutDown (
            void )
```

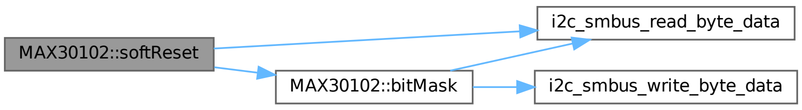
Put sensor into low power mode. During this mode the sensor will continue to respond to I2C commands but will not update or take new readings, such as temperature. Here is the call graph for this function:



### 6.28.3.52 softReset()

```
void MAX30102::softReset (
            void )
```

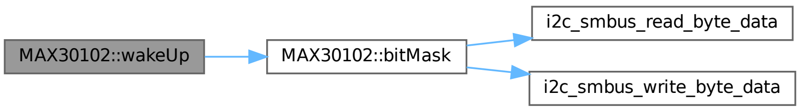
All configuration, threshold, and data registers are reset to their power-on state through a power-on reset. The reset bit is cleared back to zero after reset finishes. Here is the call graph for this function:



#### 6.28.3.53 wakeUp()

```
void MAX30102::wakeUp (
             void  )
```

Pull sensor out of low power mode. Here is the call graph for this function:



### 6.28.4 Member Data Documentation

#### 6.28.4.1 _i2c

```
int MAX30102::_i2c  [private]
```

#### 6.28.4.2 _i2caddr

```
uint8_t MAX30102::_i2caddr  [private]
```

#### 6.28.4.3 activeLEDs

```
uint8_t MAX30102::activeLEDs  [private]
```

#### 6.28.4.4 revisionID

```
uint8_t MAX30102::revisionID  [private]
```

#### 6.28.4.5 sense

```
sense_struct MAX30102::sense  [private]
```

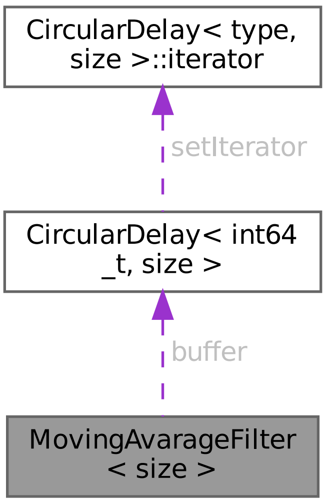
The documentation for this class was generated from the following files:

- /home/sitcomlab/Projects/VigiSense/src/MAX30102.h
- /home/sitcomlab/Projects/VigiSense/src/MAX30102.cpp

## 6.29 MovingAvarageFilter< size > Class Template Reference

```
#include <DigitalFilters.h>
```

Collaboration diagram for MovingAvarageFilter< size >:


CircularDelay< type, size >::iterator
setIterator
CircularDelay< int64_t, size >
buffer
MovingAvarageFilter< size >

### Public Member Functions

- double update (double input)

### Private Attributes

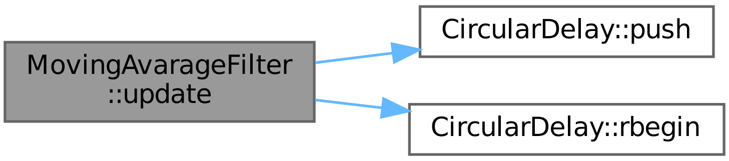
- int64_t output = 0
- CircularDelay< int64_t, size > buffer

### 6.29.1 Member Function Documentation

#### 6.29.1.1 update()

```
template<size_t size>
double MovingAvarageFilter< size >::update (
             double input )  [inline]
```

Here is the call graph for this function:



### 6.29.2 Member Data Documentation

#### 6.29.2.1 buffer

```
template<size_t size>
CircularDelay<int64_t, size> MovingAvarageFilter< size >::buffer  [private]
```

#### 6.29.2.2 output

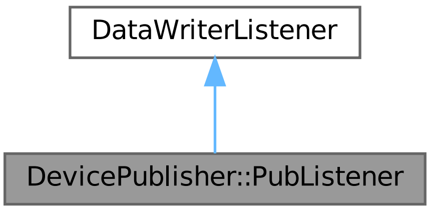
```
template<size_t size>
int64_t MovingAvarageFilter< size >::output = 0  [private]
```

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

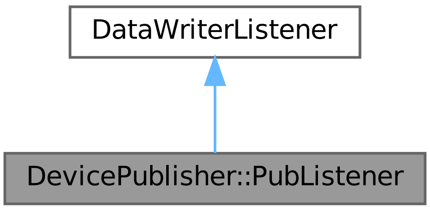
- /home/sitcomlab/Projects/VigiSense/src/DigitalFilters.h

## 6.30 DevicePublisher::PubListener Class Reference

Inheritance diagram for DevicePublisher::PubListener:



Collaboration diagram for DevicePublisher::PubListener:



**Public Member Functions**

- PubListener ()
- ∼PubListener () override
- void on_publication_matched (DataWriter ∗, const PublicationMatchedStatus &info) override

**Public Attributes**

- std::atomic_int matched_

### 6.30.1 Constructor & Destructor Documentation

#### 6.30.1.1 PubListener()

```
DevicePublisher::PubListener::PubListener ( )  [inline]
```

#### 6.30.1.2 ∼PubListener()

```
DevicePublisher::PubListener::∼PubListener ( )  [inline], [override]
```

### 6.30.2 Member Function Documentation

#### 6.30.2.1 on_publication_matched()

```
void DevicePublisher::PubListener::on_publication_matched (
             DataWriter * ,
             const PublicationMatchedStatus & info )  [inline], [override]
```

### 6.30.3 Member Data Documentation

#### 6.30.3.1 matched_

```
std::atomic_int DevicePublisher::PubListener::matched_
```

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

- /home/sitcomlab/Projects/VigiSense/src/DevicePublisher.cpp

## 6.31 MAX30102::Record Struct Reference

**Public Attributes**

- uint32_t red [STORAGE_SIZE]
- uint32_t IR [STORAGE_SIZE]
- uint8_t head
- uint8_t tail

### 6.31.1 Member Data Documentation

#### 6.31.1.1 head

```
uint8_t MAX30102::Record::head
```

#### 6.31.1.2 IR

```
uint32_t MAX30102::Record::IR[STORAGE_SIZE]
```

#### 6.31.1.3 red

```
uint32_t MAX30102::Record::red[STORAGE_SIZE]
```

#### 6.31.1.4 tail

```
uint8_t MAX30102::Record::tail
```

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

- /home/sitcomlab/Projects/VigiSense/src/MAX30102.h

## 6.32 CircularDelay< type, size >::reverse_iterator Class Reference

```
#include <CircularDelay.hpp>
```

### Public Types

- typedef reverse_iterator self_type
- typedef std::bidirectional_iterator_tag iterator_category
- typedef int difference_type

### Public Member Functions

- reverse_iterator (const CircularDelay< type, size >::reverse_iterator &it)
- self_type operator++ ()
- self_type operator++ (int)
- self_type operator-- ()
- self_type operator-- (int)
- type & operator* ()
- type * operator-> ()
- type & operator[ ] (int index)
- bool operator== (const self_type &rhs)
- bool operator!= (const self_type &rhs)

### Private Member Functions

- reverse_iterator (type *data, type *ptr)

### Private Attributes

- type * data_ = nullptr
- type * ptr_ = nullptr

### Friends

- class CircularDelay

## 6.32.1 Member Typedef Documentation

### 6.32.1.1 difference_type

```
template<typename type , size_t size>
typedef int CircularDelay< type, size >::reverse_iterator::difference_type
```

### 6.32.1.2 iterator_category

```
template<typename type , size_t size>
typedef std::bidirectional_iterator_tag CircularDelay< type, size >::reverse_iterator::iterator↩
_category
```

### 6.32.1.3 self_type

```
template<typename type , size_t size>
typedef reverse_iterator CircularDelay< type, size >::reverse_iterator::self_type
```

## 6.32.2 Constructor & Destructor Documentation

### 6.32.2.1 reverse_iterator() [1/2]

```
template<typename type , size_t size>
CircularDelay< type, size >::reverse_iterator::reverse_iterator (
             const CircularDelay< type, size >::reverse_iterator & it )  [inline]
```

### 6.32.2.2 reverse_iterator() [2/2]

```
template<typename type , size_t size>
CircularDelay< type, size >::reverse_iterator::reverse_iterator (
             type * data,
             type * ptr )  [inline], [private]
```

## 6.32.3 Member Function Documentation

### 6.32.3.1 operator"!=()

```
template<typename type , size_t size>
bool CircularDelay< type, size >::reverse_iterator::operator!= (
             const self_type & rhs )  [inline]
```

### 6.32.3.2 operator*()

```
template<typename type , size_t size>
type & CircularDelay< type, size >::reverse_iterator::operator* ( )  [inline]
```

#### 6.32.3.3 operator++() [1/2]

```
template<typename type , size_t size>
self_type CircularDelay< type, size >::reverse_iterator::operator++ ( )  [inline]
```

#### 6.32.3.4 operator++() [2/2]

```
template<typename type , size_t size>
self_type CircularDelay< type, size >::reverse_iterator::operator++ (
             int  )  [inline]
```

#### 6.32.3.5 operator--() [1/2]

```
template<typename type , size_t size>
self_type CircularDelay< type, size >::reverse_iterator::operator-- ( )  [inline]
```

#### 6.32.3.6 operator--() [2/2]

```
template<typename type , size_t size>
self_type CircularDelay< type, size >::reverse_iterator::operator-- (
             int  )  [inline]
```

#### 6.32.3.7 operator->()

```
template<typename type , size_t size>
type * CircularDelay< type, size >::reverse_iterator::operator-> ( )  [inline]
```

#### 6.32.3.8 operator==()

```
template<typename type , size_t size>
bool CircularDelay< type, size >::reverse_iterator::operator== (
             const self_type & rhs )  [inline]
```

#### 6.32.3.9 operator[]()

```
template<typename type , size_t size>
type & CircularDelay< type, size >::reverse_iterator::operator[] (
             int index )  [inline]
```

### 6.32.4 Friends And Related Symbol Documentation

#### 6.32.4.1 CircularDelay

```
template<typename type , size_t size>
friend class CircularDelay  [friend]
```

### 6.32.5 Member Data Documentation

#### 6.32.5.1 data_

```
template<typename type , size_t size>
type* CircularDelay< type, size >::reverse_iterator::data_ = nullptr  [private]
```

#### 6.32.5.2 ptr_

```
template<typename type , size_t size>
type* CircularDelay< type, size >::reverse_iterator::ptr_ = nullptr  [private]
```

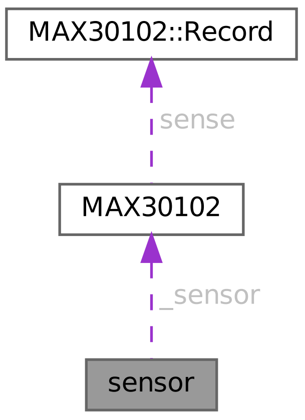
The documentation for this class was generated from the following file:

- /home/sitcomlab/Projects/VigiSense/src/CircularDelay.hpp

## 6.33 sensor Class Reference

```
#include <Sensor.h>
```

Collaboration diagram for sensor:



**Public Member Functions**

- sensor (MAX30102 ∗sensor)
- ∼sensor ()
- void begin ()
- void stop ()
- float getLatestTemperatureF ()
- void HRcalc ()
- void stopHRcalc ()
- int getSpO2 ()
- int getHR ()

### Protected Member Functions

- void loopThread ()
- void runHRCalculationLoop ()
- void updateTemperature ()
- void resetCalculations ()
- int32_t Derivative (int32_t data)
- int32_t getPeakThreshold ()
- bool peakDetect (int32_t data)

### Protected Attributes

- MAX30102 ∗ _sensor
- bool running = false
- bool runningHR = false
- int32_t bpmBuffer [BPM_BUFFER_SIZE]
- int nextBPMBufferIndex = 0
- int32_t spo2Buffer [SPO2_BUFFER_SIZE]
- int nextSPO2BufferIndex = 0
- std::chrono::time_point< std::chrono::system_clock > timeLastLoopRan
- std::chrono::time_point< std::chrono::system_clock > timeLastIRHeartBeat
- int32_t irLastValue
- int latestIRBPM
- int averageIRBPM
- std::chrono::time_point< std::chrono::system_clock > timeLastRedHeartBeat
- uint64_t redLastValue
- int latestRedBPM
- float latestTemperature = -999
- int32_t localMaximaIR
- int32_t localMinimaIR
- int32_t localMaximaRed
- int32_t localMinimaRed
- int32_t pastMaximasIR [PAST_PEAKS_SIZE]
- int32_t pastMinimasIR [PAST_PEAKS_SIZE]
- int32_t pastMaximasRed [PAST_PEAKS_SIZE]
- int32_t pastMinimasRed [PAST_PEAKS_SIZE]
- int R
- int latestSpO2

### Static Protected Attributes

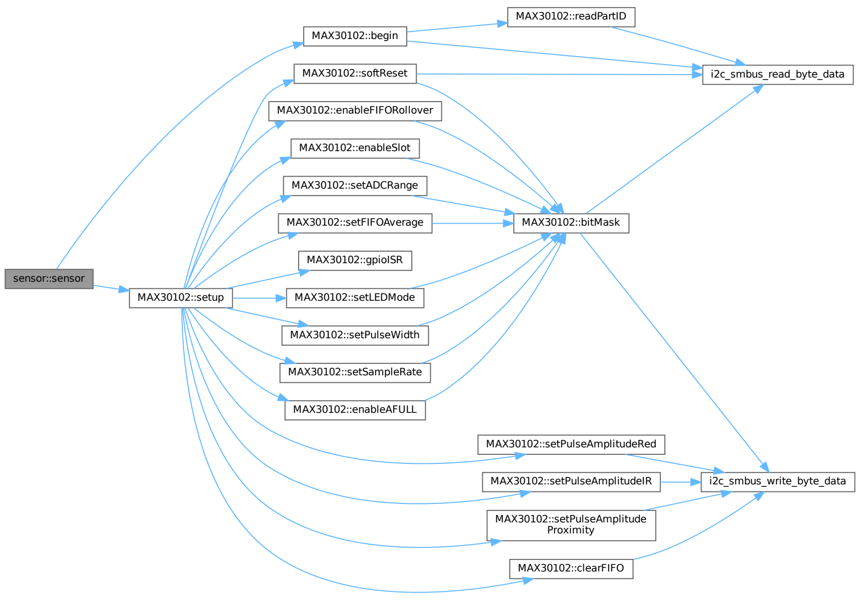
- const static int BPM_BUFFER_SIZE = 4
- const static int SPO2_BUFFER_SIZE = 4
- static const int8_t PAST_PEAKS_SIZE = 2

### 6.33.1 Constructor & Destructor Documentation

#### 6.33.1.1 sensor()

```
sensor::sensor (
            MAX30102 * s )
```

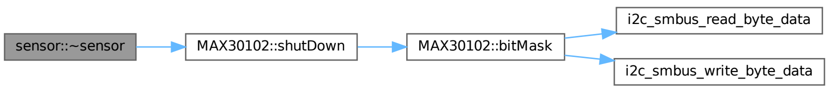
Constructor to initialize the MAX30102 sensor with the default I2C address and start communication Could also change the class name to "MAX30102Sensor" OR have "MAX30102Sensor" inherit from "sensor".
Here is the call graph for this function:



#### 6.33.1.2 ∼sensor()

```
sensor::∼sensor ( )
```

Here is the call graph for this function:



### 6.33.2 Member Function Documentation

#### 6.33.2.1 begin()

```
void sensor::begin ( )
```

#### 6.33.2.2 Derivative()

```
int32_t sensor::Derivative (
            int32_t data )  [protected]
```

#### 6.33.2.3 getHR()

```
int sensor::getHR ( )
```

Returns the latest calculated IR heart rate. (unchecked!) Returns the average measured heart rate. This method ignores heart rate values greater than 150 or lower than 45.

#### 6.33.2.4 getLatestTemperatureF()

```
float sensor::getLatestTemperatureF ( )
```

Returns the latest calculated Red heart rate. (unchecked!)

#### 6.33.2.5 getPeakThreshold()

```
int32_t sensor::getPeakThreshold ( )  [protected]
```

Detects peaks in heart data. Returns true when input data is a peak.

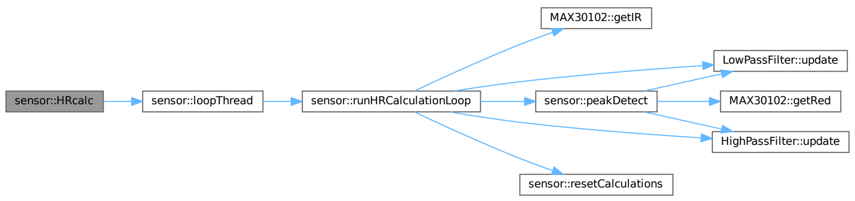
#### 6.33.2.6 getSpO2()

```
int sensor::getSpO2 ( )
```

#### 6.33.2.7 HRcalc()

```
void sensor::HRcalc ( )
```

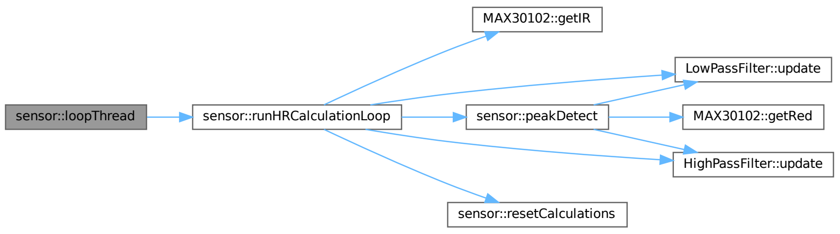
Here is the call graph for this function:



#### 6.33.2.8 loopThread()

```
void sensor::loopThread ( ) [protected]
```

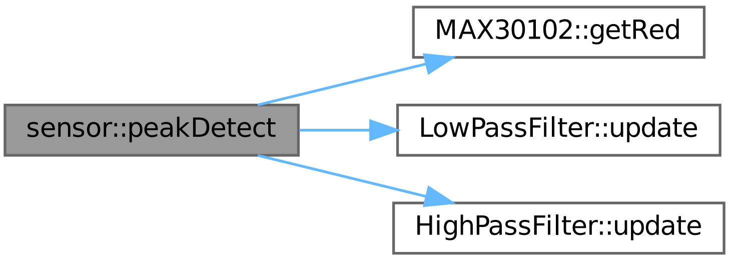
Here is the call graph for this function:



#### 6.33.2.9 peakDetect()

```
bool sensor::peakDetect (
              int32_t data ) [protected]
```

Here is the call graph for this function:



#### 6.33.2.10 resetCalculations()

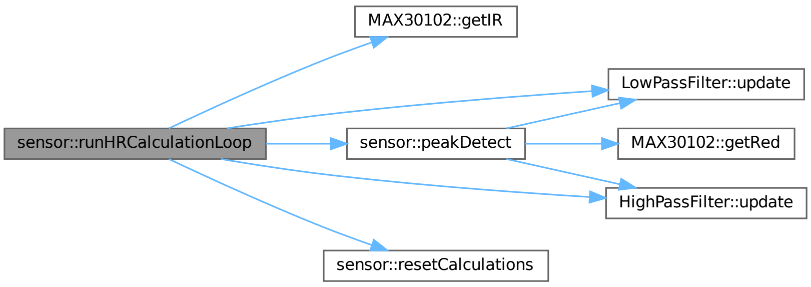
```
void sensor::resetCalculations ( ) [protected]
```

Clears all calculations.

#### 6.33.2.11 runHRCalculationLoop()

```
void sensor::runHRCalculationLoop ( )  [protected]
```

Here is the call graph for this function:



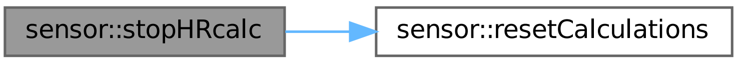
#### 6.33.2.12 stop()

```
void sensor::stop ( )
```

#### 6.33.2.13 stopHRcalc()

```
void sensor::stopHRcalc ( )
```

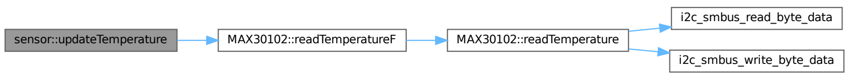
Stops the calculation loop. You may no longer get heart rate data after calling this function. Here is the call graph for this function:



#### 6.33.2.14 updateTemperature()

```
void sensor::updateTemperature ( )  [protected]
```

Updates the temperature variable. Here is the call graph for this function:


sensor::updateTemperature
MAX30102::readTemperatureF
MAX30102::readTemperature
i2c_smbus_read_byte_data
i2c_smbus_write_byte_data

### 6.33.3 Member Data Documentation

#### 6.33.3.1 _sensor

```
MAX30102* sensor::_sensor  [protected]
```

#### 6.33.3.2 averageIRBPM

```
int sensor::averageIRBPM  [protected]
```

#### 6.33.3.3 BPM_BUFFER_SIZE

```
const static int sensor::BPM_BUFFER_SIZE = 4  [static], [protected]
```

#### 6.33.3.4 bpmBuffer

```
int32_t sensor::bpmBuffer[BPM_BUFFER_SIZE]  [protected]
```

#### 6.33.3.5 irLastValue

```
int32_t sensor::irLastValue  [protected]
```

#### 6.33.3.6 latestIRBPM

```
int sensor::latestIRBPM  [protected]
```

#### 6.33.3.7 latestRedBPM

```
int sensor::latestRedBPM  [protected]
```

#### 6.33.3.8 latestSpO2

```
int sensor::latestSpO2  [protected]
```

#### 6.33.3.9 latestTemperature

```
float sensor::latestTemperature = -999  [protected]
```

#### 6.33.3.10 localMaximaIR

```
int32_t sensor::localMaximaIR  [protected]
```

#### 6.33.3.11 localMaximaRed

```
int32_t sensor::localMaximaRed [protected]
```

#### 6.33.3.12 localMinimaIR

```
int32_t sensor::localMinimaIR [protected]
```

#### 6.33.3.13 localMinimaRed

```
int32_t sensor::localMinimaRed [protected]
```

#### 6.33.3.14 nextBPMBufferIndex

```
int sensor::nextBPMBufferIndex = 0 [protected]
```

#### 6.33.3.15 nextSPO2BufferIndex

```
int sensor::nextSPO2BufferIndex = 0 [protected]
```

#### 6.33.3.16 PAST_PEAKS_SIZE

```
const int8_t sensor::PAST_PEAKS_SIZE = 2 [static], [protected]
```

#### 6.33.3.17 pastMaximasIR

```
int32_t sensor::pastMaximasIR[PAST_PEAKS_SIZE] [protected]
```

#### 6.33.3.18 pastMaximasRed

```
int32_t sensor::pastMaximasRed[PAST_PEAKS_SIZE] [protected]
```

#### 6.33.3.19 pastMinimasIR

```
int32_t sensor::pastMinimasIR[PAST_PEAKS_SIZE] [protected]
```

#### 6.33.3.20 pastMinimasRed

```
int32_t sensor::pastMinimasRed[PAST_PEAKS_SIZE] [protected]
```

#### 6.33.3.21 R

```
int sensor::R  [protected]
```

#### 6.33.3.22 redLastValue

```
uint64_t sensor::redLastValue  [protected]
```

#### 6.33.3.23 running

```
bool sensor::running = false  [protected]
```

#### 6.33.3.24 runningHR

```
bool sensor::runningHR = false  [protected]
```

#### 6.33.3.25 SPO2_BUFFER_SIZE

```
const static int sensor::SPO2_BUFFER_SIZE = 4  [static], [protected]
```

#### 6.33.3.26 spo2Buffer

```
int32_t sensor::spo2Buffer[SPO2_BUFFER_SIZE]  [protected]
```

#### 6.33.3.27 timeLastIRHeartBeat

```
std::chrono::time_point<std::chrono::system_clock> sensor::timeLastIRHeartBeat  [protected]
```

#### 6.33.3.28 timeLastLoopRan

```
std::chrono::time_point<std::chrono::system_clock> sensor::timeLastLoopRan  [protected]
```

#### 6.33.3.29 timeLastRedHeartBeat

```
std::chrono::time_point<std::chrono::system_clock> sensor::timeLastRedHeartBeat  [protected]
```

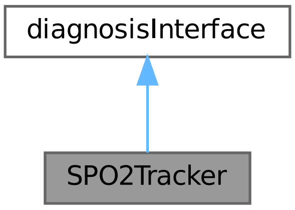
The documentation for this class was generated from the following files:

- /home/sitcomlab/Projects/VigiSense/src/Sensor.h
- /home/sitcomlab/Projects/VigiSense/src/Sensor.cpp

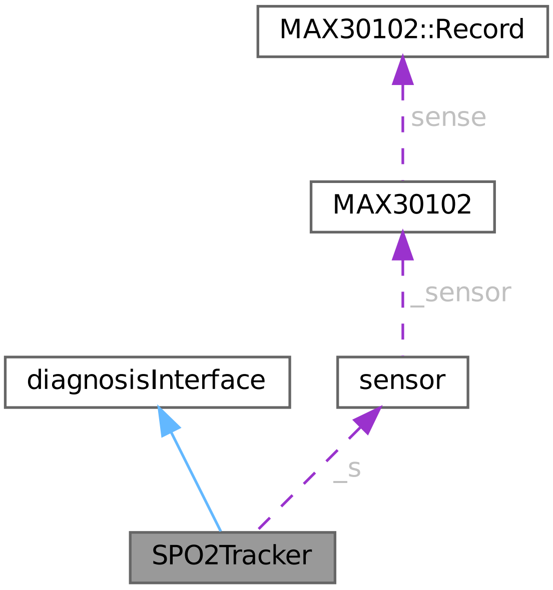
## 6.34 SPO2Tracker Class Reference

```
#include <SPO2Tracker.h>
```

Inheritance diagram for SPO2Tracker:



Collaboration diagram for SPO2Tracker:



### Public Member Functions

- SPO2Tracker (sensor ∗s)
- ∼SPO2Tracker ()
- void start ()
- void stop ()
- void ping ()
- int getVal ()
- void tracker ()

**Protected Member Functions**

- void pingThread ()

**Protected Attributes**

- sensor ∗ _s
- bool threadRunning = false
- std::vector< symptomRange > symptomRanges

## Protected Attributes inherited from diagnosisInterface

- std::vector< symptomRange > symptomRanges

**Additional Inherited Members**

## Static Public Member Functions inherited from diagnosisInterface

- static std::string determineSymptom (std::vector< symptomRange > symptomRanges, int val)

## 6.34.1 Constructor & Destructor Documentation

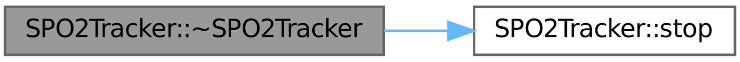
### 6.34.1.1 SPO2Tracker()

```
SPO2Tracker::SPO2Tracker (
            sensor * s )
```

### 6.34.1.2 ∼SPO2Tracker()

```
SPO2Tracker::∼SPO2Tracker ( )
```

Here is the call graph for this function:



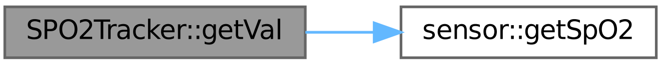
### 6.34.2 Member Function Documentation

#### 6.34.2.1 getVal()

```
int SPO2Tracker::getVal ( ) [virtual]
```

Implements diagnosisInterface.

Here is the call graph for this function:

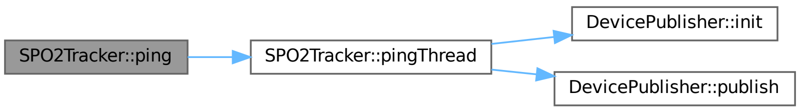


#### 6.34.2.2 ping()

```
void SPO2Tracker::ping ( ) [virtual]
```

Implements diagnosisInterface.

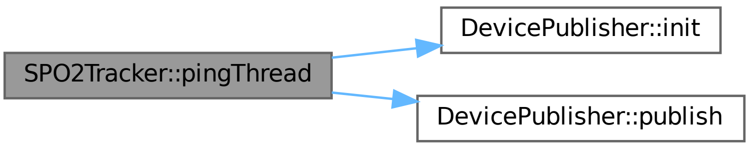
Here is the call graph for this function:



#### 6.34.2.3 pingThread()

```
void SPO2Tracker::pingThread ( ) [protected]
```

Here is the call graph for this function:

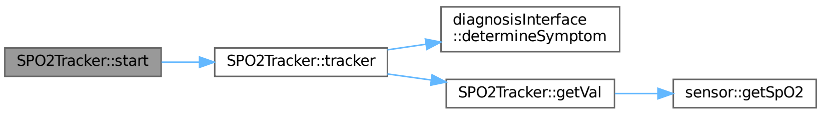


#### 6.34.2.4 start()

```
void SPO2Tracker::start ( )  [virtual]
```

Implements diagnosisInterface.

Here is the call graph for this function:



#### 6.34.2.5 stop()

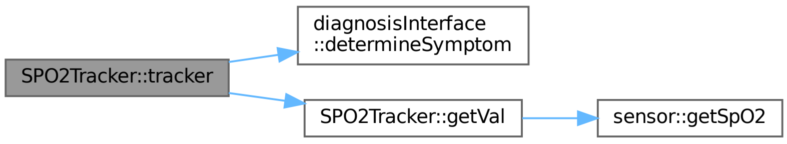
```
void SPO2Tracker::stop ( )  [virtual]
```

Implements diagnosisInterface.

#### 6.34.2.6 tracker()

```
void SPO2Tracker::tracker ( )
```

Here is the call graph for this function:



### 6.34.3 Member Data Documentation

#### 6.34.3.1 _s

```
sensor* SPO2Tracker::_s  [protected]
```

#### 6.34.3.2 symptomRanges

```
std::vector<symptomRange> SPO2Tracker::symptomRanges  [protected]
```

**Initial value:**

```
{
            {0,88,"Critically Low Oxygen concentration"},
            {88,92,"Concerningly Low Oxygen Concentration"},
            {92,100,"Healthy Oxygen Concentration"}}
```

#### 6.34.3.3 threadRunning

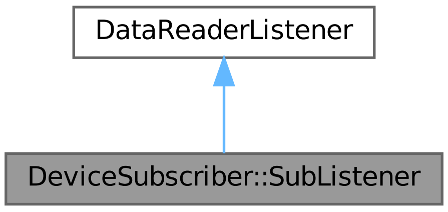
```
bool SPO2Tracker::threadRunning = false  [protected]
```

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

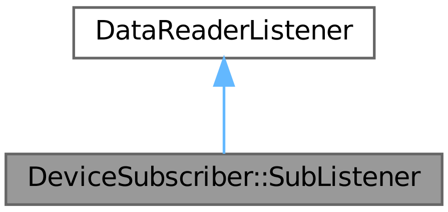
- /home/sitcomlab/Projects/VigiSense/src/SPO2Tracker.h
- /home/sitcomlab/Projects/VigiSense/src/SPO2Tracker.cpp

## 6.35 DeviceSubscriber::SubListener Class Reference

Inheritance diagram for DeviceSubscriber::SubListener:



Collaboration diagram for DeviceSubscriber::SubListener:



**Public Member Functions**

- SubListener ()
- ∼SubListener () override
- void on_subscription_matched (DataReader ∗, const SubscriptionMatchedStatus &info) override
- void on_data_available (DataReader ∗reader) override

## 6.35.1 Constructor & Destructor Documentation

### 6.35.1.1 SubListener()

```
DeviceSubscriber::SubListener::SubListener ( )  [inline]
```

### 6.35.1.2 ∼SubListener()

```
DeviceSubscriber::SubListener::∼SubListener ( )  [inline], [override]
```

## 6.35.2 Member Function Documentation

### 6.35.2.1 on_data_available()

```
void DeviceSubscriber::SubListener::on_data_available (
             DataReader ∗ reader )  [inline], [override]
```

### 6.35.2.2 on_subscription_matched()

```
void DeviceSubscriber::SubListener::on_subscription_matched (
             DataReader ∗ ,
             const SubscriptionMatchedStatus & info )  [inline], [override]
```

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

- /home/sitcomlab/Projects/VigiSense/src/DeviceSubscriber.cpp

## 6.36 symptomRange Struct Reference

```
#include <DiagnosisInterface.h>
```

**Public Attributes**

- float min
- float max
- std::string symptom

### 6.36.1 Member Data Documentation

#### 6.36.1.1 max

```
float symptomRange::max
```

#### 6.36.1.2 min

```
float symptomRange::min
```

#### 6.36.1.3 symptom

```
std::string symptomRange::symptom
```

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

- /home/sitcomlab/Projects/VigiSense/src/DiagnosisInterface.h

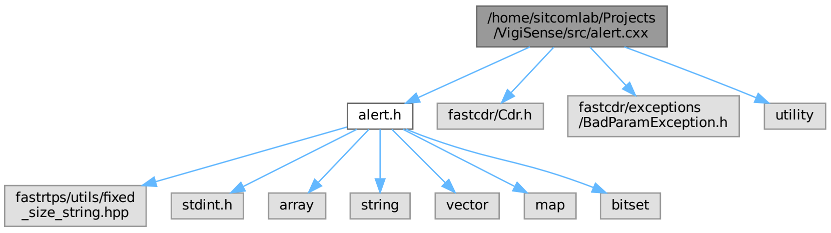
# Chapter 7

# File Documentation

## 7.1 /home/sitcomlab/Projects/VigiSense/src/alert.cxx File Reference

```
#include "alert.h"
#include <fastcdr/Cdr.h>
#include <fastcdr/exceptions/BadParamException.h>
#include <utility>
```

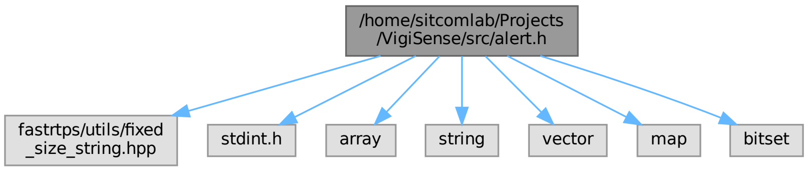
Include dependency graph for alert.cxx:



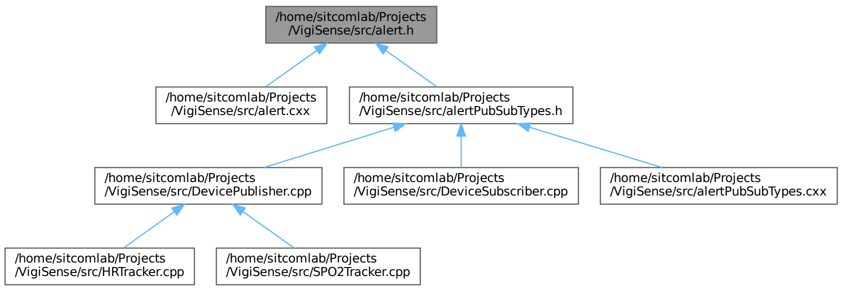
## 7.2 /home/sitcomlab/Projects/VigiSense/src/alert.h File Reference

```
#include <fastrtps/utils/fixed_size_string.hpp>
#include <stdint.h>
#include <array>
#include <string>
#include <vector>
#include <map>
```

```
#include <bitset>
```
Include dependency graph for alert.h:


/home/sitcomlab/Projects /VigiSense/src/alert.h
fastrtps/utils/fixed _size_string.hpp
stdint.h
array
string
vector
map
bitset

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


/home/sitcomlab/Projects /VigiSense/src/alert.h
/home/sitcomlab/Projects /VigiSense/src/alert.cxx
/home/sitcomlab/Projects /VigiSense/src/alertPubSubTypes.h
/home/sitcomlab/Projects /VigiSense/src/DevicePublisher.cpp
/home/sitcomlab/Projects /VigiSense/src/DeviceSubscriber.cpp
/home/sitcomlab/Projects /VigiSense/src/alertPubSubTypes.cxx
/home/sitcomlab/Projects /VigiSense/src/HRTracker.cpp
/home/sitcomlab/Projects /VigiSense/src/SPO2Tracker.cpp

### Classes

- struct alert

  *This class represents the structure alert defined by the user in the IDL file.*

### Namespaces

- namespace eprosima
- namespace eprosima::fastcdr

### Macros

- #define eProsima_user_DllExport
- #define alert_DllAPI

## 7.2.1 Detailed Description

This header file contains the declaration of the described types in the IDL file.

This file was generated by the tool gen.

### 7.2.2 Macro Definition Documentation

#### 7.2.2.1 alert_DllAPI

```
#define alert_DllAPI
```

#### 7.2.2.2 eProsima_user_DllExport

```
#define eProsima_user_DllExport
```

## 7.3 alert.h

Go to the documentation of this file.

```
// Copyright 2016 Proyectos y Sistemas de Mantenimiento SL (eProsima).
//
// Licensed under the Apache License, Version 2.0 (the "License");
// you may not use this file except in compliance with the License.
// You may obtain a copy of the License at
//
//     http://www.apache.org/licenses/LICENSE-2.0
//
// Unless required by applicable law or agreed to in writing, software
// distributed under the License is distributed on an "AS IS" BASIS,
// WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
// See the License for the specific language governing permissions and
// limitations under the License.

#ifndef _FAST_DDS_GENERATED_ALERT_H_
#define _FAST_DDS_GENERATED_ALERT_H_


#include <fastrtps/utils/fixed_size_string.hpp>

#include <stdint.h>
#include <array>
#include <string>
#include <vector>
#include <map>
#include <bitset>

#if defined(_WIN32)
#if defined(EPROSIMA_USER_DLL_EXPORT)
#define eProsima_user_DllExport __declspec( dllexport )
#else
#define eProsima_user_DllExport
#endif  // EPROSIMA_USER_DLL_EXPORT
#else
#define eProsima_user_DllExport
#endif  // _WIN32

#if defined(_WIN32)
#if defined(EPROSIMA_USER_DLL_EXPORT)
#if defined(alert_SOURCE)
#define alert_DllAPI __declspec( dllexport )
#else
#define alert_DllAPI __declspec( dllimport )
#endif // alert_SOURCE
#else
#define alert_DllAPI
#endif  // EPROSIMA_USER_DLL_EXPORT
#else
#define alert_DllAPI
#endif // _WIN32

namespace eprosima {
namespace fastcdr {
class Cdr;
} // namespace fastcdr
} // namespace eprosima


class alert
{
```

```
public:

    eProsima_user_DllExport alert();

    eProsima_user_DllExport ~alert();

    eProsima_user_DllExport alert(
            const alert& x);

    eProsima_user_DllExport alert(
            alert&& x) noexcept;

    eProsima_user_DllExport alert& operator =(
            const alert& x);

    eProsima_user_DllExport alert& operator =(
            alert&& x) noexcept;

    eProsima_user_DllExport bool operator ==(
            const alert& x) const;

    eProsima_user_DllExport bool operator !=(
            const alert& x) const;

    eProsima_user_DllExport void index(
            uint32_t _index);

    eProsima_user_DllExport uint32_t index() const;

    eProsima_user_DllExport uint32_t& index();

    eProsima_user_DllExport void message(
            const std::string& _message);

    eProsima_user_DllExport void message(
            std::string&& _message);

    eProsima_user_DllExport const std::string& message() const;

    eProsima_user_DllExport std::string& message();

    eProsima_user_DllExport static size_t getMaxCdrSerializedSize(
            size_t current_alignment = 0);

    eProsima_user_DllExport static size_t getCdrSerializedSize(
            const alert& data,
            size_t current_alignment = 0);


    eProsima_user_DllExport void serialize(
            eprosima::fastcdr::Cdr& cdr) const;

    eProsima_user_DllExport void deserialize(
            eprosima::fastcdr::Cdr& cdr);



    eProsima_user_DllExport static size_t getKeyMaxCdrSerializedSize(
            size_t current_alignment = 0);

    eProsima_user_DllExport static bool isKeyDefined();

    eProsima_user_DllExport void serializeKey(
            eprosima::fastcdr::Cdr& cdr) const;

private:

    uint32_t m_index;
    std::string m_message;
};

#endif // _FAST_DDS_GENERATED_ALERT_H_
```

## 7.4 /home/sitcomlab/Projects/VigiSense/src/alert.idl File Reference

### Classes

- struct alert

    *This class represents the structure alert defined by the user in the IDL file.*

## 7.5 alert.idl

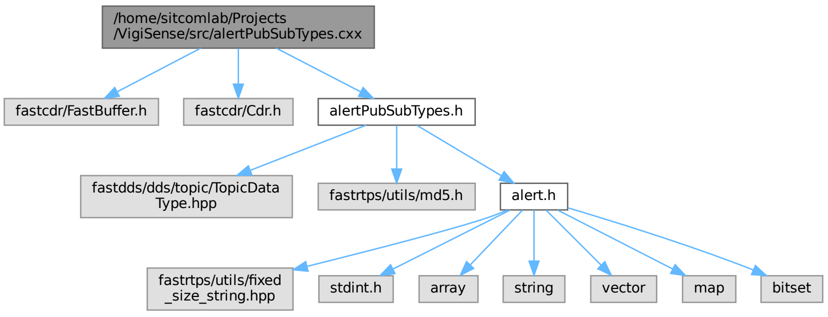
Go to the documentation of this file.

```
struct alert
{
    unsigned long index;
    string message;
};
```

## 7.6 /home/sitcomlab/Projects/VigiSense/src/alertPubSubTypes.cxx File Reference

```
#include <fastcdr/FastBuffer.h>
#include <fastcdr/Cdr.h>
#include "alertPubSubTypes.h"
```

Include dependency graph for alertPubSubTypes.cxx:



### Typedefs

- using SerializedPayload_t = eprosima::fastrtps::rtps::SerializedPayload_t
- using InstanceHandle_t = eprosima::fastrtps::rtps::InstanceHandle_t

### 7.6.1 Typedef Documentation

#### 7.6.1.1 InstanceHandle_t

```
using InstanceHandle_t = eprosima::fastrtps::rtps::InstanceHandle_t
```

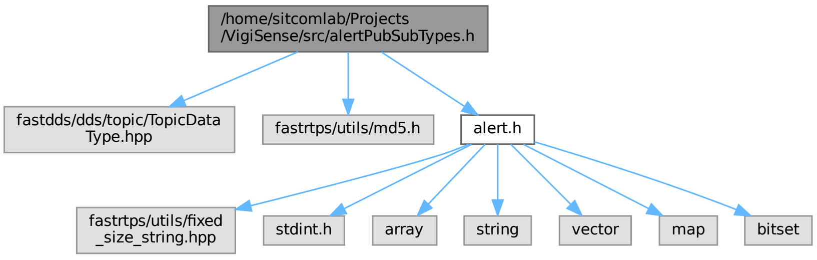
#### 7.6.1.2 SerializedPayload_t

```
using SerializedPayload_t = eprosima::fastrtps::rtps::SerializedPayload_t
```

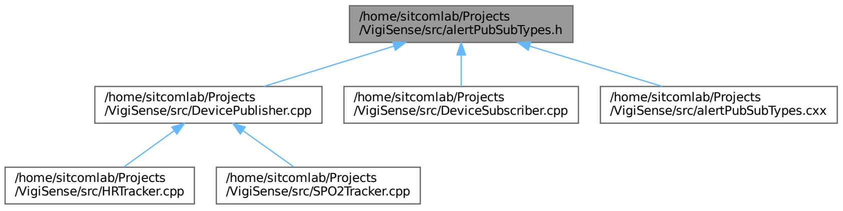
## 7.7 /home/sitcomlab/Projects/VigiSense/src/alertPubSubTypes.h File Reference

```
#include <fastdds/dds/topic/TopicDataType.hpp>
#include <fastrtps/utils/md5.h>
#include "alert.h"
```

Include dependency graph for alertPubSubTypes.h:


/home/sitcomlab/Projects /VigiSense/src/alertPubSubTypes.h
fastdds/dds/topic/TopicDataType.hpp
fastrtps/utils/md5.h
alert.h
fastrtps/utils/fixed_size_string.hpp
stdint.h
array
string
vector
map
bitset

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


/home/sitcomlab/Projects /VigiSense/src/alertPubSubTypes.h
/home/sitcomlab/Projects /VigiSense/src/DevicePublisher.cpp
/home/sitcomlab/Projects /VigiSense/src/DeviceSubscriber.cpp
/home/sitcomlab/Projects /VigiSense/src/alertPubSubTypes.cxx
/home/sitcomlab/Projects /VigiSense/src/HRTracker.cpp
/home/sitcomlab/Projects /VigiSense/src/SPO2Tracker.cpp

### Classes

- class alertPubSubType

  *This class represents the TopicDataType of the type alert defined by the user in the IDL file.*

### 7.7.1 Detailed Description

This header file contains the declaration of the serialization functions.

This file was generated by the tool fastcdrgen.

## 7.8 alertPubSubTypes.h

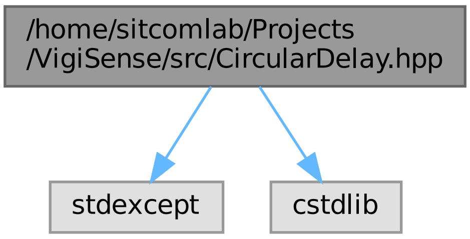
Go to the documentation of this file.

```
// Copyright 2016 Proyectos y Sistemas de Mantenimiento SL (eProsima).
//
// Licensed under the Apache License, Version 2.0 (the "License");
// you may not use this file except in compliance with the License.
// You may obtain a copy of the License at
//
//     http://www.apache.org/licenses/LICENSE-2.0
//
// Unless required by applicable law or agreed to in writing, software
// distributed under the License is distributed on an "AS IS" BASIS,
// WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
// See the License for the specific language governing permissions and
// limitations under the License.

#ifndef _FAST_DDS_GENERATED_ALERT_PUBSUBTYPES_H_
#define _FAST_DDS_GENERATED_ALERT_PUBSUBTYPES_H_

#include <fastdds/dds/topic/TopicDataType.hpp>
#include <fastrtps/utils/md5.h>

#include "alert.h"

#if !defined(GEN_API_VER) || (GEN_API_VER != 1)
#error \
    Generated alert is not compatible with current installed Fast DDS. Please, regenerate it with
fastddsgen.
#endif  // GEN_API_VER

class alertPubSubType : public eprosima::fastdds::dds::TopicDataType
{
public:

    typedef alert type;

    eProsima_user_DllExport alertPubSubType();

    eProsima_user_DllExport virtual ~alertPubSubType() override;

    eProsima_user_DllExport virtual bool serialize(
            void* data,
            eprosima::fastrtps::rtps::SerializedPayload_t* payload) override;

    eProsima_user_DllExport virtual bool deserialize(
            eprosima::fastrtps::rtps::SerializedPayload_t* payload,
            void* data) override;

    eProsima_user_DllExport virtual std::function<uint32_t()> getSerializedSizeProvider(
            void* data) override;

    eProsima_user_DllExport virtual bool getKey(
            void* data,
            eprosima::fastrtps::rtps::InstanceHandle_t* ihandle,
            bool force_md5 = false) override;

    eProsima_user_DllExport virtual void* createData() override;

    eProsima_user_DllExport virtual void deleteData(
            void* data) override;

#ifdef TOPIC_DATA_TYPE_API_HAS_IS_BOUNDED
    eProsima_user_DllExport inline bool is_bounded() const override
    {
        return false;
    }

#endif  // TOPIC_DATA_TYPE_API_HAS_IS_BOUNDED

#ifdef TOPIC_DATA_TYPE_API_HAS_IS_PLAIN
    eProsima_user_DllExport inline bool is_plain() const override
    {
        return false;
    }

#endif  // TOPIC_DATA_TYPE_API_HAS_IS_PLAIN

#ifdef TOPIC_DATA_TYPE_API_HAS_CONSTRUCT_SAMPLE
    eProsima_user_DllExport inline bool construct_sample(
            void* memory) const override
    {
        (void)memory;
        return false;
    }
#endif  // TOPIC_DATA_TYPE_API_HAS_CONSTRUCT_SAMPLE

    MD5 m_md5;
    unsigned char* m_keyBuffer;
};

#endif // _FAST_DDS_GENERATED_ALERT_PUBSUBTYPES_H_
```

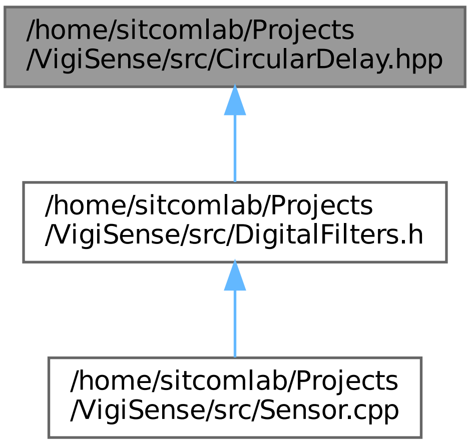
## 7.9 /home/sitcomlab/Projects/VigiSense/src/CircularDelay.hpp File Reference

```
#include <stdexcept>
#include <cstdlib>
```

Include dependency graph for CircularDelay.hpp:



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


/home/sitcomlab/Projects
/VigiSense/src/CircularDelay.hpp
/home/sitcomlab/Projects
/VigiSense/src/DigitalFilters.h
/home/sitcomlab/Projects
/VigiSense/src/Sensor.cpp

## Classes

- class CircularDelay< type, size >

  *A class that functions as a sample buffer.*
- class CircularDelay< type, size >::iterator
- class CircularDelay< type, size >::const_iterator
- class CircularDelay< type, size >::reverse_iterator
- class CircularDelay< type, size >::const_reverse_iterator

## 7.10 CircularDelay.hpp

Go to the documentation of this file.

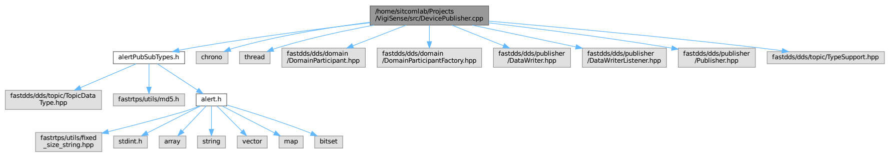
```

#include <stdexcept>
#include <cstdlib>
template<typename type, size_t size>
class CircularDelay{
public:
    CircularDelay();
    class iterator{
    public:
        typedef iterator self_type;
        typedef std::bidirectional_iterator_tag iterator_category;
        typedef int difference_type;
        iterator(const CircularDelay<type, size>::iterator& it):data_(it.data_), ptr_(it.ptr_){}
        self_type operator++() {
            if(++ptr_ == data_ + size + 1){
                ptr_ = data_;
            }
            return *this;
        }
        self_type operator++(int) {self_type ret = *this; ++*this; return ret;}
        self_type operator--() {
            if(ptr_ == data_){
                ptr_ = data_ + size;
            }
            else{
                --ptr_;
            }
            return *this;
        }
        self_type operator--(int) {self_type ret = *this; --*this; return ret;}
        type& operator*() { return *ptr_; }
        type* operator->() { return ptr_; }
        type& operator[](unsigned int index){
            // Note that ptr_ - data_ is converting ptr_ to index value;
            index += ptr_ - data_;
            if(index >= size + 1){
                index -= size + 1;
            }
            return data_[index];
        }
        bool operator==(const self_type& rhs) { return ptr_ == rhs.ptr_; }
        bool operator!=(const self_type& rhs) { return ptr_ != rhs.ptr_; }
    private:
        iterator(type* data, type* ptr) : data_(data), ptr_(ptr) { }
        friend class CircularDelay;
        type* data_ = nullptr;
        type* ptr_ = nullptr;
    };
    class const_iterator{
    public:
        typedef const_iterator self_type;
        typedef std::bidirectional_iterator_tag iterator_category;
        typedef int difference_type;
        const_iterator(const CircularDelay<type, size>::const_iterator& it):data_(it.data_),
ptr_(it.ptr_){}
        self_type operator++() {
            if(++ptr_ == data_ + size + 1){
                ptr_ = data_;
            }
            return *this;
        }
        self_type operator++(int) {self_type ret = *this; ++*this; return ret;}
        self_type operator--() {
            if(ptr_ == data_){
                ptr_ = data_ + size;
            }
            else{
                --ptr_;
            }
            return *this;
        }
        self_type operator--(int) {self_type ret = *this; --*this; return ret;}
        const type& operator*() { return *ptr_; }
        const type* operator->() { return ptr_; }
        const type& operator[](unsigned int index){
            // Note that ptr_ - data_ is converting ptr_ to index value;
            index += ptr_ - data_;
            if(index >= size + 1){
                index -= size + 1;
            }
            return data_[index];
        }
        bool operator==(const self_type& rhs) { return ptr_ == rhs.ptr_; }
        bool operator!=(const self_type& rhs) { return ptr_ != rhs.ptr_; }
    private:
        const_iterator(type* data, type* ptr) : data_(data), ptr_(ptr) { }
        friend class CircularDelay;
        type* data_ = nullptr;
        type* ptr_ = nullptr;
    };
    class reverse_iterator{
    public:
        typedef reverse_iterator self_type;
        typedef std::bidirectional_iterator_tag iterator_category;
        typedef int difference_type;
        reverse_iterator(const CircularDelay<type, size>::reverse_iterator& it):data_(it.data_),
ptr_(it.ptr_){}
        self_type operator++() {
            if(ptr_ == data_){
                ptr_ = data_ + size;
            }
            else{
                --ptr_;
            }
            return *this;
        }
        self_type operator++(int) {self_type ret = *this; ++*this; return ret;}
        self_type operator--() {
            if(++ptr_ == data_ + size + 1){
                ptr_ = data_;
            }
            return *this;
        }
        self_type operator--(int) {self_type ret = *this; --*this; return ret;}
        type& operator*() { return *ptr_; }
        type* operator->() { return ptr_; }
        type& operator[](int index){
            // Convert ptr_ to index value;
            index = ptr_ - data_ - index;
            if(index < 0){
                index += size + 1;
            }
            return data_[index];
        }
        bool operator==(const self_type& rhs) { return ptr_ == rhs.ptr_; }
        bool operator!=(const self_type& rhs) { return ptr_ != rhs.ptr_; }
    private:
        reverse_iterator(type* data, type* ptr) : data_(data), ptr_(ptr) { }
        friend class CircularDelay;
        type* data_ = nullptr;
        type* ptr_ = nullptr;
    };
    class const_reverse_iterator{
    public:
        typedef const_reverse_iterator self_type;
        typedef std::bidirectional_iterator_tag iterator_category;
        typedef int difference_type;
        const_reverse_iterator(const CircularDelay<type, size>::const_reverse_iterator&
it):data_(it.data_), ptr_(it.ptr_){}
        self_type operator++() {
            if(ptr_ == data_){
                ptr_ = data_ + size;
            }
            else{
                --ptr_;
            }
            return *this;
        }
        self_type operator++(int) {self_type ret = *this; ++*this; return ret;}
        self_type operator--() {
            if(++ptr_ == data_ + size + 1){
                ptr_ = data_;
            }
            return *this;
        }
        self_type operator--(int) {self_type ret = *this; --*this; return ret;}
        const type& operator*() { return *ptr_; }
        const type* operator->() { return ptr_; }
        const type& operator[](int index){
            // Convert ptr_ to index value;
            index = ptr_ - data_ - index;
            if(index < 0){
                index += size + 1;
            }
            return data_[index];
        }
        bool operator==(const self_type& rhs) { return ptr_ == rhs.ptr_; }
        bool operator!=(const self_type& rhs) { return ptr_ != rhs.ptr_; }
    private:
        const_reverse_iterator(type* data, type* ptr) : data_(data), ptr_(ptr) { }
        friend class CircularDelay;
        type* data_ = nullptr;
        type* ptr_ = nullptr;
    };
    type push(type input);
    type get(size_t delay);
    iterator end(){return setIterator;}
    iterator begin(){
        iterator it(setIterator);
        ++it;
        return it;
    }
    reverse_iterator rend(){return reverse_iterator(data, setIterator.ptr_);}
    reverse_iterator rbegin(){
        reverse_iterator it(data, setIterator.ptr_);
        ++it;
        return it;
    }
private:
    type data[size + 1];
    iterator setIterator = iterator(data, data);
};

template<typename type, size_t size>
CircularDelay<type, size>::CircularDelay(){
    for (size_t i = 0; i < size + 1; ++i){
        data[i] = 0;
    }
}

template<typename type, size_t size>
type CircularDelay<type, size>::push(type input){
    *setIterator = input;
    setIterator++;
    return input;
}

template<typename type, size_t size>
type CircularDelay<type, size>::get(size_t delay){
    if(delay >= size + 1)
        throw(std::domain_error("Tried to get a value that is longer ago than the size of a
CircularDelay."));
    reverse_iterator itRBegin(rbegin());
    return itRBegin[delay];
}
```

## 7.11 /home/sitcomlab/Projects/VigiSense/src/DevicePublisher.cpp File Reference

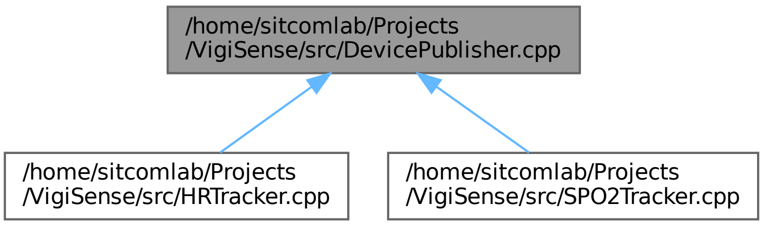
```
#include "alertPubSubTypes.h"
#include <chrono>
#include <thread>
#include <fastdds/dds/domain/DomainParticipant.hpp>
#include <fastdds/dds/domain/DomainParticipantFactory.hpp>
#include <fastdds/dds/publisher/DataWriter.hpp>
#include <fastdds/dds/publisher/DataWriterListener.hpp>
#include <fastdds/dds/publisher/Publisher.hpp>
```

```
#include <fastdds/dds/topic/TypeSupport.hpp>
```

Include dependency graph for DevicePublisher.cpp:



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


/home/sitcomlab/Projects
/VigiSense/src/DevicePublisher.cpp
/home/sitcomlab/Projects
/VigiSense/src/HRTracker.cpp
/home/sitcomlab/Projects
/VigiSense/src/SPO2Tracker.cpp

### Classes

- class DevicePublisher
- class DevicePublisher::PubListener

## 7.12 DevicePublisher.cpp

Go to the documentation of this file.

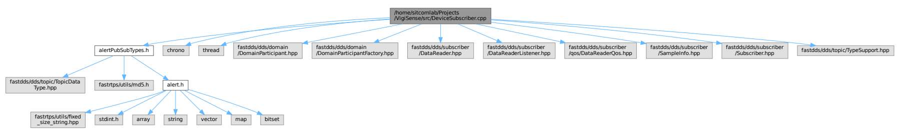
```
// Copyright 2016 Proyectos y Sistemas de Mantenimiento SL (eProsima).
//
// Licensed under the Apache License, Version 2.0 (the "License");
// you may not use this file except in compliance with the License.
// You may obtain a copy of the License at
//
//     http://www.apache.org/licenses/LICENSE-2.0
//
// Unless required by applicable law or agreed to in writing, software
// distributed under the License is distributed on an "AS IS" BASIS,
// WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
// See the License for the specific language governing permissions and
// limitations under the License.

#include "alertPubSubTypes.h"

#include <chrono>
#include <thread>

#include <fastdds/dds/domain/DomainParticipant.hpp>
#include <fastdds/dds/domain/DomainParticipantFactory.hpp>
#include <fastdds/dds/publisher/DataWriter.hpp>
#include <fastdds/dds/publisher/DataWriterListener.hpp>
#include <fastdds/dds/publisher/Publisher.hpp>
#include <fastdds/dds/topic/TypeSupport.hpp>

using namespace eprosima::fastdds::dds;
```

```

class DevicePublisher
{
private:

    DomainParticipant* participant_ = nullptr;

    Publisher* publisher_ = nullptr;

    Topic* topic_ = nullptr;

    DataWriter* writer_ = nullptr;

    TypeSupport type_;

    class PubListener : public DataWriterListener
    {
    public:

        PubListener()
            : matched_(0)
        {
        }

        ~PubListener() override
        {
        }

        void on_publication_matched(
                DataWriter*,
                const PublicationMatchedStatus& info) override
        {
            if (info.current_count_change == 1)
            {
                matched_ = info.total_count;
                std::cout « "Publisher matched." « std::endl;
            }
            else if (info.current_count_change == -1)
            {
                matched_ = info.total_count;
                std::cout « "Publisher unmatched." « std::endl;
            }
            else
            {
                std::cout « info.current_count_change
                        « " is not a valid value for PublicationMatchedStatus current count change." «
std::endl;
            }
        }

        std::atomic_int matched_;

    } listener_;

public:

    DevicePublisher() : type_(new alertPubSubType()) {}

    virtual ~DevicePublisher()
    {
        if (writer_ != nullptr)
        {
            publisher_->delete_datawriter(writer_);
        }
        if (publisher_ != nullptr)
        {
            participant_->delete_publisher(publisher_);
        }
        if (topic_ != nullptr)
        {
            participant_->delete_topic(topic_);
        }
        DomainParticipantFactory::get_instance()->delete_participant(participant_);
    }

    bool init()
    {
        DomainParticipantQos participantQos;
        participantQos.name("Participant_publisher");
        participant_ = DomainParticipantFactory::get_instance()->create_participant(0,
participantQos);

        if (participant_ == nullptr)
        {
            return false;
        }

        // Register the Type
        type_.register_type(participant_);

        // Create the publications Topic
    // !! Important that this matches with the name of message defined in HelloWorldMsg.idl !!
        topic_ = participant_->create_topic("HelloWorldTopic", "alert", TOPIC_QOS_DEFAULT);

        if (topic_ == nullptr)
        {
            return false;
        }

        // Create the Publisher
        publisher_ = participant_->create_publisher(PUBLISHER_QOS_DEFAULT, nullptr);

        if (publisher_ == nullptr)
        {
            return false;
        }

        // Create the DataWriter
        writer_ = publisher_->create_datawriter(topic_, DATAWRITER_QOS_DEFAULT, &listener_);

        if (writer_ == nullptr)
        {
            return false;
        }
        return true;
    }

    bool publish(alert& hello)
    {
        if (listener_.matched_ > 0)
        {
            writer_->write(&hello);
            return true;
        }
        return false;
    }

};
```

## 7.13 /home/sitcomlab/Projects/VigiSense/src/DeviceSubscriber.cpp File Reference

```
#include "alertPubSubTypes.h"
#include <chrono>
#include <thread>
#include <fastdds/dds/domain/DomainParticipant.hpp>
#include <fastdds/dds/domain/DomainParticipantFactory.hpp>
#include <fastdds/dds/subscriber/DataReader.hpp>
#include <fastdds/dds/subscriber/DataReaderListener.hpp>
#include <fastdds/dds/subscriber/qos/DataReaderQos.hpp>
#include <fastdds/dds/subscriber/SampleInfo.hpp>
#include <fastdds/dds/subscriber/Subscriber.hpp>
#include <fastdds/dds/topic/TypeSupport.hpp>
```

Include dependency graph for DeviceSubscriber.cpp:


/home/sitcomlab/Projects /VigiSense/src/DeviceSubscriber.cpp
alertPubSubTypes.h
chrono
thread
fastdds/dds/domain /DomainParticipant.hpp
fastdds/dds/domain /DomainParticipantFactory.hpp
fastdds/dds/subscriber /DataReader.hpp
fastdds/dds/subscriber /DataReaderListener.hpp
fastdds/dds/subscriber /qos/DataReaderQos.hpp
fastdds/dds/subscriber /SampleInfo.hpp
fastdds/dds/subscriber /Subscriber.hpp
fastdds/dds/topic/TypeSupport.hpp
fastdds/dds/topic/TopicData Type.hpp
fastrtps/utils/md5.h
alert.h
fastrtps/utils/fixed _size_string.hpp
stdint.h
array
string
vector
map
bitset

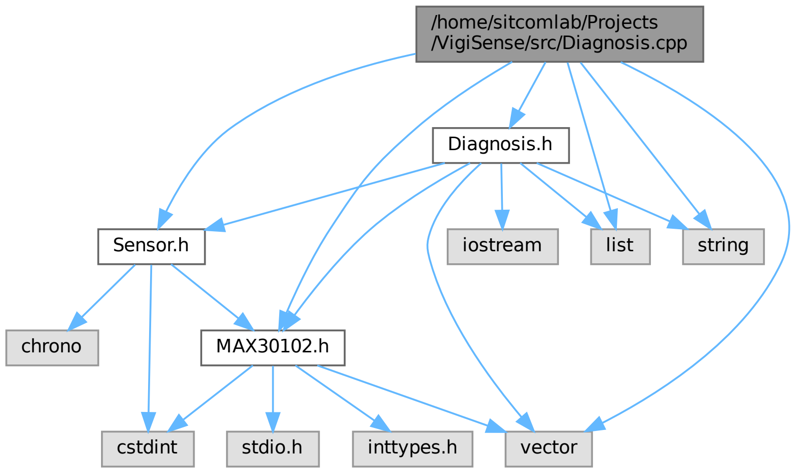
### Classes

- class DeviceSubscriber
- class DeviceSubscriber::SubListener

## 7.14 /home/sitcomlab/Projects/VigiSense/src/Diagnosis.cpp File Reference

```
#include <vector>
#include <list>
#include <string>
#include "MAX30102.h"
#include "Sensor.h"
#include "Diagnosis.h"
```

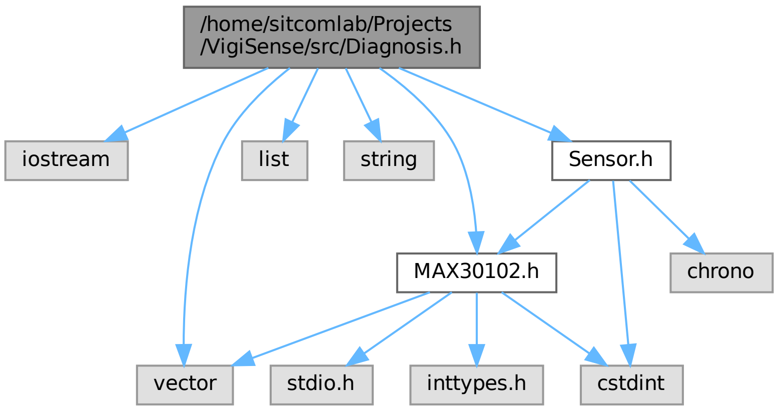
Include dependency graph for Diagnosis.cpp:



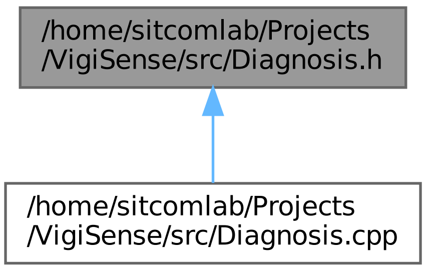
## 7.15 /home/sitcomlab/Projects/VigiSense/src/Diagnosis.h File Reference

```
#include <iostream>
#include <vector>
#include <list>
#include <string>
#include "MAX30102.h"
#include "Sensor.h"
```

Include dependency graph for Diagnosis.h:



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



### Classes

- class diagnosis
- struct diagnosis::DiagnosisRange
- struct diagnosis::DiagnosesTable

## 7.16 Diagnosis.h

Go to the documentation of this file.

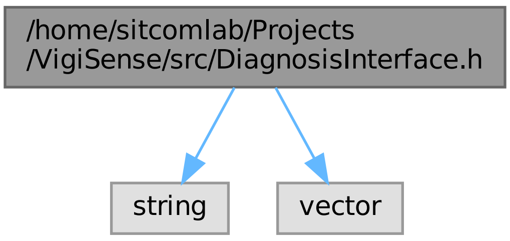
```
#include <iostream>
#include <vector>
#include <list>
#include <string>
#include "MAX30102.h"
#include "Sensor.h"
```

```

//This is going to be an abstract class that will be inherited by the diagnosis classes for each
biosignal

class diagnosis {
public:
    // Public member functions and variables

    //Struct for the individual diagnosis range
    typedef struct DiagnosisRange{
    float min;
    float max;
    std::string diagnosis;
    } Diagnosis_Range;

    Diagnosis_Range stdDiagnosis; // Creates a standard symptom range for the diagnosis
    Diagnosis_Range CustomDiagnosis; //Creates a custom symptom range for the diagnosis

    //Struct for the diagnoses table
    typedef struct DiagnosesTable {
        std::vector<DiagnosisRange> Diagnoses; //Contains all the possible diagnoses for the biosignal
    } _Diagnoses;

    _Diagnoses stdDiagnoses; //Creates a standard diagnoses table for the biosignal
    _Diagnoses CustomDiagnoses; //Creates a custom diagnoses table for the biosignal

    //Function to set customized symptom ranges for the biosignal
    void SetdiagnosisRanges(float minimum, float maximum, std::string diagnosis);//Sets the symptom
ranges for the biosignal

    //Function to determine the symptom based on the biosignal value
    std::string determineDiagnosis() { return ""; }

    void critCheck() {} //Checks if the biosignal value crosses the critical value and initiates an
immediate alert

    void findMinMax() {} //Finds the minimum and maximum values in the diagnoses table

    float CriticalLow; //Variable to store the critical low value
    float CriticalHigh; //Variable to store the critical high value

    //function to display the diagnosis
    virtual void displayDiagnosis(); //Pure virtual function to display the diagnosis

    //function for critical value crossing
    virtual void critRangeAlert(); //Pure virtual function for critical value crossing alert
private:
    // Private member functions and variables

};
```

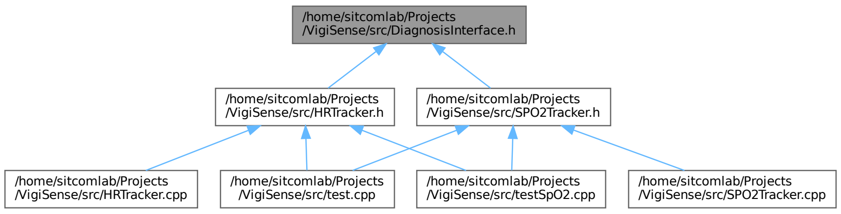
## 7.17 /home/sitcomlab/Projects/VigiSense/src/DiagnosisInterface.h File Reference

```
#include <string>
#include <vector>
```

Include dependency graph for DiagnosisInterface.h:



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


/home/sitcomlab/Projects /VigiSense/src/DiagnosisInterface.h
/home/sitcomlab/Projects /VigiSense/src/HRTracker.h
/home/sitcomlab/Projects /VigiSense/src/SPO2Tracker.h
/home/sitcomlab/Projects /VigiSense/src/HRTracker.cpp
/home/sitcomlab/Projects /VigiSense/src/test.cpp
/home/sitcomlab/Projects /VigiSense/src/testSpO2.cpp
/home/sitcomlab/Projects /VigiSense/src/SPO2Tracker.cpp

### Classes

- struct symptomRange
- class diagnosisInterface

## 7.18 DiagnosisInterface.h

Go to the documentation of this file.

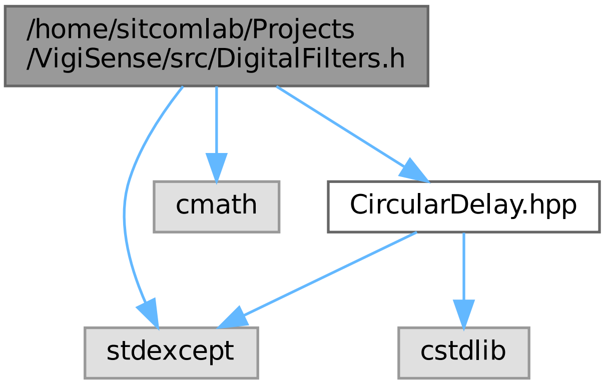
```
#pragma once
#include <string>
#include <vector>

struct symptomRange{
    float min;
    float max;
    std::string symptom;
};

class diagnosisInterface {
    public:
        virtual void start() = 0;
        virtual void stop() = 0;
        virtual void ping() = 0;
        virtual int getVal() = 0;

        static std::string determineSymptom(std::vector<symptomRange> symptomRanges, int val){
            for (int i = 0; i < symptomRanges.size(); ++i){
                if (val>symptomRanges[i].min && val<symptomRanges[i].max){
                    return symptomRanges[i].symptom;
                }
            }
            if (val < symptomRanges[0].min) {
                return "critLow";
            } else if (val > symptomRanges[symptomRanges.size() -1].max)
            {
                return "critHigh";
            } else {
                return "Out of range";
            }
        };


    protected:
        std::vector<symptomRange> symptomRanges;
};
```

## 7.19 /home/sitcomlab/Projects/VigiSense/src/DigitalFilters.h File Reference

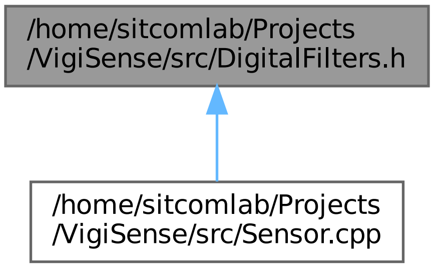
```
#include <stdexcept>
#include <cmath>
```

```
#include "CircularDelay.hpp"
```

Include dependency graph for DigitalFilters.h:



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



### Classes

- class DigitalFilter< Type >

  *Abstract base class for digital moving filters.*
- class Differentiator< T >

  *Class for differentiator.*
- class LowPassFilter

  *Class for a low pass filter.*
- class LowPassFilter2

  *Class for a 2nd order low pass filter.*
- class HighPassFilter

  *Class for high pass filter using bilinear transform.*
- class HighPassFilter3

*Class for third order high pass filter. This is designed using the bilinear transform.*
- class LowPassFilter3

  *Class for third order high pass filter. This is designed using the bilinear transform.*
- class LowPassFilter3MatchedZ

  *Class for third order high pass filter. This is designed using the matched Z transform.*
- class LowPassFilter3DiffApprox

  *Class for third order high pass filter. This is designed using the approximated differtial approuch where s=(Z-1)/(Z∗T).*
- class MovingAvarageFilter< size >

### Namespaces

- namespace tps

### Functions

- template<typename T >
  constexpr T squareOf (T input)
- template<typename T >
  constexpr T tps::pow (T input, unsigned int power)
- template<typename T >
  constexpr T calcC_Cr (T k, T m)

## 7.19.1 Function Documentation

### 7.19.1.1 calcC_Cr()

```
template<typename T >
constexpr T calcC_Cr (
             T k,
             T m )  [constexpr]
```

### 7.19.1.2 squareOf()

```
template<typename T >
constexpr T squareOf (
             T input )  [constexpr]
```

## 7.20 DigitalFilters.h

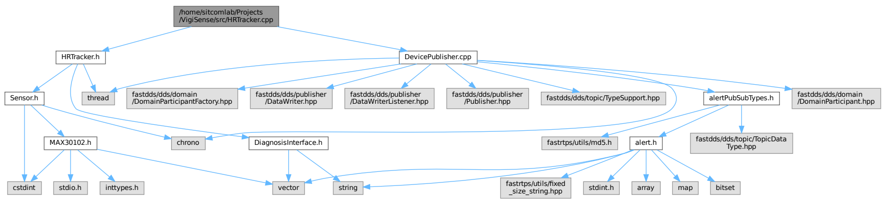
Go to the documentation of this file.

```
#ifndef _DIGITAL_FILTERS_H_
#define _DIGITAL_FILTERS_H_

#include <stdexcept>
#include <cmath>
#include "CircularDelay.hpp"

template<typename T>
constexpr T squareOf(T input){return input * input;}

namespace tps{
    template<typename T>
    constexpr T pow(T input, unsigned int power){return (power == 0 ? 1 : input * (power <= 1 ? 1 : tps::pow(input, power-1)));}
}

template<typename T>
constexpr T calcC_Cr(T k, T m){return 2 * sqrt(k * m);}

template<typename Type>
class DigitalFilter{
public:
    virtual Type update(Type newValue) = 0;
    virtual Type getOutput() = 0;
};

template<typename T>
class Differentiator : public DigitalFilter<T> {
public:
    Differentiator(T sampleTime):
        sampleTime(sampleTime)
    {}
    T update(T input){
        y = (input - x1) / sampleTime;
        x1 = input;
        return y;
    }
    T getOutput(){return y;}
private:
    const T sampleTime;
    T y = 0;
    T x1 = 0;
};

class LowPassFilter : public DigitalFilter<double> {
public:
    LowPassFilter(double idt, double omega_c, double ioutput = 0):
        epow(exp(-idt * omega_c)),
        dt(idt),
        output(ioutput){
            if(omega_c < idt){
                throw std::domain_error("LowPassFilter constructor error: tua_c is smaller than the sample time dt.");
            }
        }
    double update(double newValue) final{return output = (output-newValue) * epow + newValue;}
    double getOutput() final{return output;}
    void configOutput(double newOutput){output = newOutput;}
    const double* outputPointer(){return &output;}
private:
    const double epow;
    const double dt;
    double output;
};

class LowPassFilter2 : public DigitalFilter<double> {
public:
    LowPassFilter2(double dt, double tau_c, double ioutput = 0):
        yc{
            -2 * (pow(dt, 2) - 4 * pow(tau_c, 2)) / (pow(dt, 2) + 2 * sqrt(2) * tau_c * dt + 4 * pow(tau_c, 2)),
            (-pow(dt, 2) + 2 * sqrt(2) * tau_c * dt - 4 * pow(tau_c, 2)) / (pow(dt, 2) + 2 * sqrt(2) * tau_c * dt + 4 * pow(tau_c, 2))
        },
        xc{
            pow(dt, 2) / (pow(dt, 2) + 2 * sqrt(2) * tau_c * dt + 4 * pow(tau_c, 2)),
            2 * pow(dt, 2) / (pow(dt, 2) + 2 * sqrt(2) * tau_c * dt + 4 * pow(tau_c, 2)),
            pow(dt, 2) / (pow(dt, 2) + 2 * sqrt(2) * tau_c * dt + 4 * pow(tau_c, 2))
        }
        {
            if(tau_c < M_PI * dt){
                throw std::domain_error("LowPassFilter constructor error: tua_c is smaller than the sample time dt.");
            }
        }
    double update(double newValue) final{
        x.push(newValue);
        double output = 0;
        for (int i = 0; i < 2; ++i)
            output += y.get(i) * yc[i];
        for (int i = 0; i < 3; ++i)
            output += x.get(i) * xc[i];
        return y.push(output);
    }
    double getOutput() final{return y.get(0);}
    void configOutput(double newOutput){
        for(auto& it : x){
            it = newOutput;
        }
        for(auto& it : y){
            it = newOutput;
        }
    }
private:
    const double yc[2];
    const double xc[3];
    CircularDelay<double, 2> y;
    CircularDelay<double, 3> x;
};

class HighPassFilter : public DigitalFilter<double> {
public:
    HighPassFilter(double idt, double omega_c):
        amplFac(1/((idt * omega_c / 2) + 1)),
        y1c((idt * omega_c / 2) - 1),
        dt(idt){
            if(omega_c < idt){
                throw std::domain_error("LowPassFilter constructor error: tua_c is smaller than the
sample time dt.");
            }
        }
    double update(double newValue) final{
        // Note that output before assignment equals y1 being y[n-1]
        output = amplFac * (newValue - x1 - output * y1c);
        x1 = newValue;
        return output;
    }
    double getOutput() final{return output;}
    void configOutput(double newOutput){output = newOutput;}
    const double* outputPointer(){return &output;}
private:
    const double amplFac; // one time calculation constant
    const double y1c; // one time calculation constant
    const double dt;
    double x1 = 0;
    double output = 0;
};

class HighPassFilter3 : public DigitalFilter<double> {
public:
    HighPassFilter3(double sampleTime, double omega_c, double ioutput = 0):
        xc{
            8
            ,
            -24
            ,
            24
            ,
            -8
        },
        yc{
            1 / (1 * tps::pow(sampleTime * omega_c, 3) + 4 * tps::pow(sampleTime * omega_c, 2) + 8 *
sampleTime * omega_c + 8),
                3 * tps::pow(sampleTime * omega_c, 3) + 4 * tps::pow(sampleTime * omega_c, 2) - 8 *
sampleTime * omega_c - 24,
                3 * tps::pow(sampleTime * omega_c, 3) - 4 * tps::pow(sampleTime * omega_c, 2) - 8 *
sampleTime * omega_c + 24,
                1 * tps::pow(sampleTime * omega_c, 3) - 4 * tps::pow(sampleTime * omega_c, 2) + 8 *
sampleTime * omega_c - 8
        }
        {
            if(omega_c < sampleTime){
                throw std::domain_error("LowPassFilter constructor error: tua_c is smaller than the
sample time dt.");
            }
        }
    double update(double newValue) final{
        x.push(newValue);
        double y0 = 0;
        const double* doubleP = xc;
        for (auto it = x.rbegin(); it != x.rend(); it++)
        {
            y0 += *it * *doubleP++;
        }
        doubleP = yc + 1;
        for (auto it = y.rbegin(); it != y.rend(); it++)
        {
            y0 -= *it * *doubleP++;
        }
        return y.push(yc[0] * y0);
        // return y.push(yc[0] * (
        //      x.get(0) * xc[0] + x.get(1) * xc[1] + x.get(2) * xc[2] + x.get(3) * xc[3] -
        //      y.get(0) * yc[1] - y.get(1) * yc[2] - y.get(2) * yc[3]
        //  )
        // );
    }
    double getOutput() final{return y.get(0);}
private:
    const double xc[4];
    const double yc[4];
    CircularDelay<double, 3> y;
    CircularDelay<double, 4> x;
};

class LowPassFilter3 : public DigitalFilter<double> {
public:
    LowPassFilter3(long double sampleTime, long double omega_c, long double ioutput = 0):
        yc{
            1
            ,
            (double)((3 * tps::pow(sampleTime * omega_c, 3) + 4 * tps::pow(sampleTime * omega_c, 2) -
8 * sampleTime * omega_c - 24)
            /
            (1 * tps::pow(sampleTime * omega_c, 3) + 4 * tps::pow(sampleTime * omega_c, 2) + 8 *
sampleTime * omega_c + 8))
            ,
            (double)((3 * tps::pow(sampleTime * omega_c, 3) - 4 * tps::pow(sampleTime * omega_c, 2) -
8 * sampleTime * omega_c + 24)
            /
            (1 * tps::pow(sampleTime * omega_c, 3) + 4 * tps::pow(sampleTime * omega_c, 2) + 8 *
sampleTime * omega_c + 8))
            ,
            (double)((1 * tps::pow(sampleTime * omega_c, 3) - 4 * tps::pow(sampleTime * omega_c, 2) +
8 * sampleTime * omega_c - 8)
            /
            (1 * tps::pow(sampleTime * omega_c, 3) + 4 * tps::pow(sampleTime * omega_c, 2) + 8 *
sampleTime * omega_c + 8))
        },
        xc{
            (double)(1 * tps::pow(sampleTime * omega_c, 3)
            /
            (1 * tps::pow(sampleTime * omega_c, 3) + 4 * tps::pow(sampleTime * omega_c, 2) + 8 *
sampleTime * omega_c + 8))
            ,
            (double)(3 * tps::pow(sampleTime * omega_c, 3)
            /
            (1 * tps::pow(sampleTime * omega_c, 3) + 4 * tps::pow(sampleTime * omega_c, 2) + 8 *
sampleTime * omega_c + 8))
            ,
            (double)(3 * tps::pow(sampleTime * omega_c, 3)
            /
            (1 * tps::pow(sampleTime * omega_c, 3) + 4 * tps::pow(sampleTime * omega_c, 2) + 8 *
sampleTime * omega_c + 8))
            ,
            (double)(1 * tps::pow(sampleTime * omega_c, 3)
            /
            (1 * tps::pow(sampleTime * omega_c, 3) + 4 * tps::pow(sampleTime * omega_c, 2) + 8 *
sampleTime * omega_c + 8))
        }
        {
            if(omega_c < sampleTime){
                throw std::domain_error("LowPassFilter constructor error: tua_c is smaller than the
sample time dt.");
            }
        }
    double update(double newValue) final{
        x.push(newValue);
        double y0 = 0;
        const double* doubleP = xc;
        for (auto it = x.rbegin(); it != x.rend(); it++)
        {
            y0 += *it * *doubleP++;
        }
        doubleP = yc + 1;
        for (auto it = y.rbegin(); it != y.rend(); it++)
        {
            y0 -= *it * *doubleP++;
        }
        return y.push(yc[0] * y0);
    }
    double getOutput() final{return y.get(0);}
private:
    const double yc[4];
    const double xc[4];
    CircularDelay<double, 3> y;
    CircularDelay<double, 4> x;
};

class LowPassFilter3MatchedZ : public DigitalFilter<double> {
public:
    LowPassFilter3MatchedZ(long double sampleTime, long double omega_c):
        amplFac(-(2*(expl(3 * omega_c * sampleTime) - expl(2 * omega_c * sampleTime))*cosl(sqrtl(3) * omega_c * sampleTime / 2) - expl(7 * omega_c * sampleTime / 2) + expl(3 * omega_c * sampleTime / 2))*expl(-7 * omega_c * sampleTime / 2)),
        yc{
            (double)(-(2 * cosl(sqrtl(3) * omega_c * sampleTime / 2) * expl(omega_c * sampleTime * 5 / 2) + expl(2 * omega_c * sampleTime)) * expl(-3 * omega_c * sampleTime))
            ,
            (double)((2 * cosl(sqrtl(3) * omega_c * sampleTime / 2) * expl(omega_c * sampleTime * 3 / 2) + expl(2 * omega_c * sampleTime)) * expl(-3 * omega_c * sampleTime))
            ,
            (double)(-expl(-2 * omega_c * sampleTime))
        }
        {
            if(omega_c / (2 * M_PI) < sampleTime){
                throw std::domain_error("LowPassFilter3MatchedZ constructor error: tua_c is smaller than the sample time dt.");
            }
        }
    double update(double newValue) final{
        double y0 = newValue * amplFac;
        const double* doubleP = yc;
        for (auto it = y.rbegin(); it != y.rend(); it++)
        {
            y0 -= *it * *doubleP++;
        }
        return y.push(y0);
    }
    double getOutput() final{return y.get(0);}
private:
    const double amplFac;
    const double yc[3];
    CircularDelay<double, 3> y;
};

class LowPassFilter3DiffApprox : public DigitalFilter<double> {
public:
    LowPassFilter3DiffApprox(double sampleTime, double omega_c, double ioutput = 0):
        xc{
            1 * tps::pow(sampleTime * omega_c, 3)
            ,
            0
            ,
            0
            ,
            0
        },
        yc{
            1 / (1 * tps::pow(sampleTime * omega_c, 3) + 2 * tps::pow(sampleTime * omega_c, 2) + 2 * sampleTime * omega_c + 1),
                0 * tps::pow(sampleTime * omega_c, 3) - 2 * tps::pow(sampleTime * omega_c, 2) - 4 * sampleTime * omega_c - 3,
                0 * tps::pow(sampleTime * omega_c, 3) + 0 * tps::pow(sampleTime * omega_c, 2) + 2 * sampleTime * omega_c + 3,
                0 * tps::pow(sampleTime * omega_c, 3) - 0 * tps::pow(sampleTime * omega_c, 2) + 0 * sampleTime * omega_c - 1
        }
        {
            if(omega_c < sampleTime){
                throw std::domain_error("LowPassFilter constructor error: tua_c is smaller than the sample time dt.");
            }
        }
    double update(double newValue) final{
        x.push(newValue);
        double y0 = 0;
        const double* doubleP = xc;
        for (auto it = x.rbegin(); it != x.rend(); it++)
        {
            y0 += *it * *doubleP++;
        }
        doubleP = yc + 1;
        for (auto it = y.rbegin(); it != y.rend(); it++)
        {
            y0 -= *it * *doubleP++;
        }
        return y.push(yc[0] * y0);
    }
    double getOutput() final{return y.get(0);}
private:
    const double xc[4];
    const double yc[4];
    CircularDelay<double, 3> y;
    CircularDelay<double, 4> x;
};

template<size_t size>
class MovingAvarageFilter{
public:
    double update(double input){
        input *= 1000;
        output += int64_t(input) - *buffer.rbegin();
        buffer.push(input);
        return double(output) / (1000);
    }
private:
    int64_t output = 0;
    CircularDelay<int64_t, size> buffer;
};

#endif
```

## 7.21 /home/sitcomlab/Projects/VigiSense/src/HRTracker.cpp File Reference

```
#include "HRTracker.h"
#include "DevicePublisher.cpp"
```

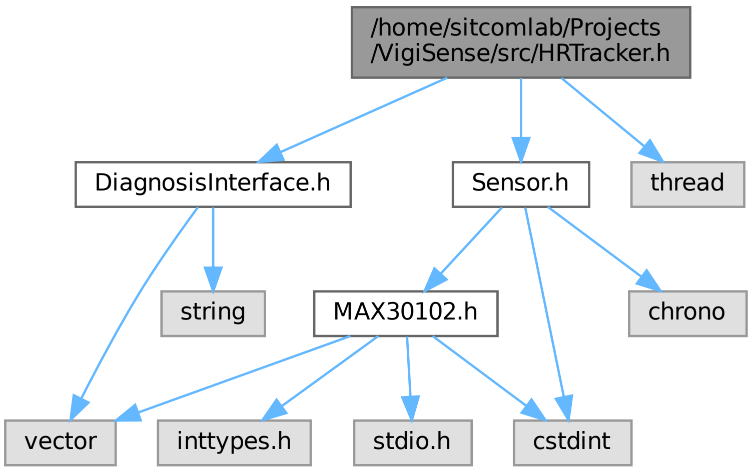
Include dependency graph for HRTracker.cpp:


/home/sitcomlab/Projects /VigiSense/src/HRTracker.cpp
HRTracker.h
DevicePublisher.cpp
Sensor.h
thread
fastdds/dds/domain /DomainParticipantFactory.hpp
fastdds/dds/publisher /DataWriter.hpp
fastdds/dds/publisher /DataWriterListener.hpp
fastdds/dds/publisher /Publisher.hpp
fastdds/dds/topic/TypeSupport.hpp
alertPubSubTypes.h
fastdds/dds/domain /DomainParticipant.hpp
MAX30102.h
chrono
DiagnosisInterface.h
fastrtps/utils/md5.h
alert.h
fastdds/dds/topic/TopicData Type.hpp
cstdint
stdio.h
inttypes.h
vector
string
fastrtps/utils/fixed _size_string.hpp
stdint.h
array
map
bitset

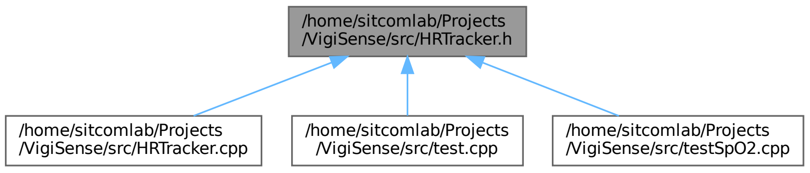
## 7.22 /home/sitcomlab/Projects/VigiSense/src/HRTracker.h File Reference

```
#include "DiagnosisInterface.h"
#include "Sensor.h"
#include <thread>
```

Include dependency graph for HRTracker.h:



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


/home/sitcomlab/Projects /VigiSense/src/HRTracker.h
/home/sitcomlab/Projects /VigiSense/src/HRTracker.cpp
/home/sitcomlab/Projects /VigiSense/src/test.cpp
/home/sitcomlab/Projects /VigiSense/src/testSpO2.cpp

### Classes

- class HRTracker

## 7.23 HRTracker.h

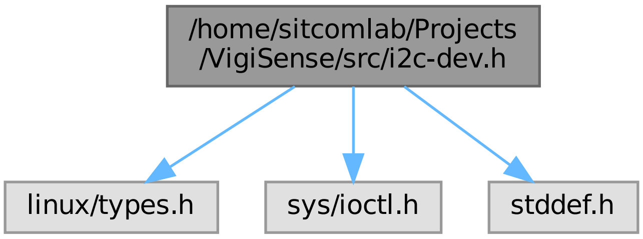
Go to the documentation of this file.

```
#pragma once
#include "DiagnosisInterface.h"
#include "Sensor.h"
#include <thread>

class HRTracker:public diagnosisInterface {
    public:
        HRTracker(sensor *s);
        ~HRTracker();
        void start();
        void stop();
        void ping();
        int getVal();
        void tracker();
    protected:
        sensor* _s;
        bool threadRunning = false;
        void pingThread();
        // define symptom table here
        std::vector<symptomRange> symptomRanges {
            {0,60,"Bradycardia"},
            {60,100,"Normal resting heart rate"},
            {100,200,"Tachyacardia"}};

};
```

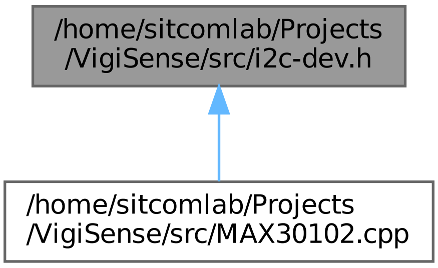
## 7.24 /home/sitcomlab/Projects/VigiSense/src/i2c-dev.h File Reference

```
#include <linux/types.h>
#include <sys/ioctl.h>
#include <stddef.h>
```

Include dependency graph for i2c-dev.h:



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



### Classes

- struct i2c_msg
- union i2c_smbus_data
- struct i2c_smbus_ioctl_data
- struct i2c_rdwr_ioctl_data

## Macros

- #define I2C_M_TEN 0x10 /∗ we have a ten bit chip address ∗/
- #define I2C_M_RD 0x01
- #define I2C_M_NOSTART 0x4000
- #define I2C_M_REV_DIR_ADDR 0x2000
- #define I2C_M_IGNORE_NAK 0x1000
- #define I2C_M_NO_RD_ACK 0x0800
- #define I2C_FUNC_I2C 0x00000001
- #define I2C_FUNC_10BIT_ADDR 0x00000002
- #define I2C_FUNC_PROTOCOL_MANGLING 0x00000004 /∗ I2C_M_{REV_DIR_ADDR,NOSTART,..} ∗/
- #define I2C_FUNC_SMBUS_PEC 0x00000008
- #define I2C_FUNC_SMBUS_BLOCK_PROC_CALL 0x00008000 /∗ SMBus 2.0 ∗/
- #define I2C_FUNC_SMBUS_QUICK 0x00010000
- #define I2C_FUNC_SMBUS_READ_BYTE 0x00020000
- #define I2C_FUNC_SMBUS_WRITE_BYTE 0x00040000
- #define I2C_FUNC_SMBUS_READ_BYTE_DATA 0x00080000
- #define I2C_FUNC_SMBUS_WRITE_BYTE_DATA 0x00100000
- #define I2C_FUNC_SMBUS_READ_WORD_DATA 0x00200000
- #define I2C_FUNC_SMBUS_WRITE_WORD_DATA 0x00400000
- #define I2C_FUNC_SMBUS_PROC_CALL 0x00800000
- #define I2C_FUNC_SMBUS_READ_BLOCK_DATA 0x01000000
- #define I2C_FUNC_SMBUS_WRITE_BLOCK_DATA 0x02000000
- #define I2C_FUNC_SMBUS_READ_I2C_BLOCK 0x04000000 /∗ I2C-like block xfer ∗/
- #define I2C_FUNC_SMBUS_WRITE_I2C_BLOCK 0x08000000 /∗ w/ 1-byte reg. addr. ∗/
- #define I2C_FUNC_SMBUS_BYTE
- #define I2C_FUNC_SMBUS_BYTE_DATA
- #define I2C_FUNC_SMBUS_WORD_DATA
- #define I2C_FUNC_SMBUS_BLOCK_DATA
- #define I2C_FUNC_SMBUS_I2C_BLOCK
- #define I2C_FUNC_SMBUS_HWPEC_CALC I2C_FUNC_SMBUS_PEC
- #define I2C_SMBUS_BLOCK_MAX 32 /∗ As specified in SMBus standard ∗/
- #define I2C_SMBUS_I2C_BLOCK_MAX 32 /∗ Not specified but we use same structure ∗/
- #define I2C_SMBUS_READ 1
- #define I2C_SMBUS_WRITE 0
- #define I2C_SMBUS_QUICK 0
- #define I2C_SMBUS_BYTE 1
- #define I2C_SMBUS_BYTE_DATA 2
- #define I2C_SMBUS_WORD_DATA 3
- #define I2C_SMBUS_PROC_CALL 4
- #define I2C_SMBUS_BLOCK_DATA 5
- #define I2C_SMBUS_I2C_BLOCK_BROKEN 6
- #define I2C_SMBUS_BLOCK_PROC_CALL 7 /∗ SMBus 2.0 ∗/
- #define I2C_SMBUS_I2C_BLOCK_DATA 8
- #define I2C_RETRIES
- #define I2C_TIMEOUT 0x0702 /∗ set timeout in units of 10 ms ∗/
- #define I2C_SLAVE 0x0703 /∗ Use this slave address ∗/
- #define I2C_SLAVE_FORCE
- #define I2C_TENBIT 0x0704 /∗ 0 for 7 bit addrs, != 0 for 10 bit ∗/
- #define I2C_FUNCS 0x0705 /∗ Get the adapter functionality mask ∗/
- #define I2C_RDWR 0x0707 /∗ Combined R/W transfer (one STOP only) ∗/
- #define I2C_PEC 0x0708 /∗ != 0 to use PEC with SMBus ∗/
- #define I2C_SMBUS 0x0720 /∗ SMBus transfer ∗/
- #define I2C_RDRW_IOCTL_MAX_MSGS 42

**Functions**

- static __s32 i2c_smbus_access (int file, char read_write, __u8 command, int size, union i2c_smbus_data ∗data)
- static __s32 i2c_smbus_write_quick (int file, __u8 value)
- static __s32 i2c_smbus_read_byte (int file)
- static __s32 i2c_smbus_write_byte (int file, __u8 value)
- static __s32 i2c_smbus_read_byte_data (int file, __u8 command)
- static __s32 i2c_smbus_write_byte_data (int file, __u8 command, __u8 value)
- static __s32 i2c_smbus_read_word_data (int file, __u8 command)
- static __s32 i2c_smbus_write_word_data (int file, __u8 command, __u16 value)
- static __s32 i2c_smbus_process_call (int file, __u8 command, __u16 value)
- static __s32 i2c_smbus_read_block_data (int file, __u8 command, __u8 ∗values)
- static __s32 i2c_smbus_write_block_data (int file, __u8 command, __u8 length, const __u8 ∗values)
- static __s32 i2c_smbus_read_i2c_block_data (int file, __u8 command, __u8 length, __u8 ∗values)
- static __s32 i2c_smbus_write_i2c_block_data (int file, __u8 command, __u8 length, const __u8 ∗values)
- static __s32 i2c_smbus_block_process_call (int file, __u8 command, __u8 length, __u8 ∗values)

## 7.24.1 Macro Definition Documentation

### 7.24.1.1 I2C_FUNC_10BIT_ADDR

```
#define I2C_FUNC_10BIT_ADDR 0x00000002
```

### 7.24.1.2 I2C_FUNC_I2C

```
#define I2C_FUNC_I2C 0x00000001
```

### 7.24.1.3 I2C_FUNC_PROTOCOL_MANGLING

```
#define I2C_FUNC_PROTOCOL_MANGLING 0x00000004 /* I2C_M_{REV_DIR_ADDR,NOSTART,..} */
```

### 7.24.1.4 I2C_FUNC_SMBUS_BLOCK_DATA

```
#define I2C_FUNC_SMBUS_BLOCK_DATA
```

**Value:**

```
(I2C_FUNC_SMBUS_READ_BLOCK_DATA | \
I2C_FUNC_SMBUS_WRITE_BLOCK_DATA)
```

### 7.24.1.5 I2C_FUNC_SMBUS_BLOCK_PROC_CALL

```
#define I2C_FUNC_SMBUS_BLOCK_PROC_CALL 0x00008000 /* SMBus 2.0 */
```

#### 7.24.1.6 I2C_FUNC_SMBUS_BYTE

```
#define I2C_FUNC_SMBUS_BYTE
```

**Value:**

```
(I2C_FUNC_SMBUS_READ_BYTE | \
I2C_FUNC_SMBUS_WRITE_BYTE)
```

#### 7.24.1.7 I2C_FUNC_SMBUS_BYTE_DATA

```
#define I2C_FUNC_SMBUS_BYTE_DATA
```

**Value:**

```
(I2C_FUNC_SMBUS_READ_BYTE_DATA | \
I2C_FUNC_SMBUS_WRITE_BYTE_DATA)
```

#### 7.24.1.8 I2C_FUNC_SMBUS_HWPEC_CALC

```
#define I2C_FUNC_SMBUS_HWPEC_CALC I2C_FUNC_SMBUS_PEC
```

#### 7.24.1.9 I2C_FUNC_SMBUS_I2C_BLOCK

```
#define I2C_FUNC_SMBUS_I2C_BLOCK
```

**Value:**

```
(I2C_FUNC_SMBUS_READ_I2C_BLOCK | \
I2C_FUNC_SMBUS_WRITE_I2C_BLOCK)
```

#### 7.24.1.10 I2C_FUNC_SMBUS_PEC

```
#define I2C_FUNC_SMBUS_PEC 0x00000008
```

#### 7.24.1.11 I2C_FUNC_SMBUS_PROC_CALL

```
#define I2C_FUNC_SMBUS_PROC_CALL 0x00800000
```

#### 7.24.1.12 I2C_FUNC_SMBUS_QUICK

```
#define I2C_FUNC_SMBUS_QUICK 0x00010000
```

#### 7.24.1.13 I2C_FUNC_SMBUS_READ_BLOCK_DATA

```
#define I2C_FUNC_SMBUS_READ_BLOCK_DATA 0x01000000
```

#### 7.24.1.14 I2C_FUNC_SMBUS_READ_BYTE

```
#define I2C_FUNC_SMBUS_READ_BYTE 0x00020000
```

#### 7.24.1.15 I2C_FUNC_SMBUS_READ_BYTE_DATA

```
#define I2C_FUNC_SMBUS_READ_BYTE_DATA 0x00080000
```

#### 7.24.1.16 I2C_FUNC_SMBUS_READ_I2C_BLOCK

```
#define I2C_FUNC_SMBUS_READ_I2C_BLOCK 0x04000000 /* I2C-like block xfer */
```

#### 7.24.1.17 I2C_FUNC_SMBUS_READ_WORD_DATA

```
#define I2C_FUNC_SMBUS_READ_WORD_DATA 0x00200000
```

#### 7.24.1.18 I2C_FUNC_SMBUS_WORD_DATA

```
#define I2C_FUNC_SMBUS_WORD_DATA
```

**Value:**

```
(I2C_FUNC_SMBUS_READ_WORD_DATA | \
I2C_FUNC_SMBUS_WRITE_WORD_DATA)
```

#### 7.24.1.19 I2C_FUNC_SMBUS_WRITE_BLOCK_DATA

```
#define I2C_FUNC_SMBUS_WRITE_BLOCK_DATA 0x02000000
```

#### 7.24.1.20 I2C_FUNC_SMBUS_WRITE_BYTE

```
#define I2C_FUNC_SMBUS_WRITE_BYTE 0x00040000
```

#### 7.24.1.21 I2C_FUNC_SMBUS_WRITE_BYTE_DATA

```
#define I2C_FUNC_SMBUS_WRITE_BYTE_DATA 0x00100000
```

#### 7.24.1.22 I2C_FUNC_SMBUS_WRITE_I2C_BLOCK

```
#define I2C_FUNC_SMBUS_WRITE_I2C_BLOCK 0x08000000 /* w/ 1-byte reg.  addr.  */
```

#### 7.24.1.23 I2C_FUNC_SMBUS_WRITE_WORD_DATA

```
#define I2C_FUNC_SMBUS_WRITE_WORD_DATA 0x00400000
```

#### 7.24.1.24 I2C_FUNCS

```
#define I2C_FUNCS 0x0705 /* Get the adapter functionality mask */
```

#### 7.24.1.25 I2C_M_IGNORE_NAK

```
#define I2C_M_IGNORE_NAK 0x1000
```

#### 7.24.1.26 I2C_M_NO_RD_ACK

```
#define I2C_M_NO_RD_ACK 0x0800
```

#### 7.24.1.27 I2C_M_NOSTART

```
#define I2C_M_NOSTART 0x4000
```

#### 7.24.1.28 I2C_M_RD

```
#define I2C_M_RD 0x01
```

#### 7.24.1.29 I2C_M_REV_DIR_ADDR

```
#define I2C_M_REV_DIR_ADDR 0x2000
```

#### 7.24.1.30 I2C_M_TEN

```
#define I2C_M_TEN 0x10 /* we have a ten bit chip address */
```

#### 7.24.1.31 I2C_PEC

```
#define I2C_PEC 0x0708 /* != 0 to use PEC with SMBus */
```

#### 7.24.1.32 I2C_RDWR_IOCTL_MAX_MSGS

```
#define I2C_RDWR_IOCTL_MAX_MSGS 42
```

#### 7.24.1.33 I2C_RDWR

```
#define I2C_RDWR 0x0707 /* Combined R/W transfer (one STOP only) */
```

#### 7.24.1.34 I2C_RETRIES

```
#define I2C_RETRIES
```

**Value:**

```
0x0701  /* number of times a device address should
be polled when not acknowledging */
```

#### 7.24.1.35 I2C_SLAVE

```
#define I2C_SLAVE 0x0703 /* Use this slave address */
```

#### 7.24.1.36 I2C_SLAVE_FORCE

```
#define I2C_SLAVE_FORCE
```

**Value:**

```
0x0706  /* Use this slave address, even if it
is already in use by a driver! */
```

#### 7.24.1.37 I2C_SMBUS

```
#define I2C_SMBUS 0x0720 /* SMBus transfer */
```

#### 7.24.1.38 I2C_SMBUS_BLOCK_DATA

```
#define I2C_SMBUS_BLOCK_DATA 5
```

#### 7.24.1.39 I2C_SMBUS_BLOCK_MAX

```
#define I2C_SMBUS_BLOCK_MAX 32 /* As specified in SMBus standard */
```

#### 7.24.1.40 I2C_SMBUS_BLOCK_PROC_CALL

```
#define I2C_SMBUS_BLOCK_PROC_CALL 7 /* SMBus 2.0 */
```

#### 7.24.1.41 I2C_SMBUS_BYTE

```
#define I2C_SMBUS_BYTE 1
```

#### 7.24.1.42 I2C_SMBUS_BYTE_DATA

```
#define I2C_SMBUS_BYTE_DATA 2
```

#### 7.24.1.43 I2C_SMBUS_I2C_BLOCK_BROKEN

```
#define I2C_SMBUS_I2C_BLOCK_BROKEN 6
```

#### 7.24.1.44 I2C_SMBUS_I2C_BLOCK_DATA

```
#define I2C_SMBUS_I2C_BLOCK_DATA 8
```

#### 7.24.1.45 I2C_SMBUS_I2C_BLOCK_MAX

```
#define I2C_SMBUS_I2C_BLOCK_MAX 32 /* Not specified but we use same structure */
```

#### 7.24.1.46 I2C_SMBUS_PROC_CALL

```
#define I2C_SMBUS_PROC_CALL 4
```

#### 7.24.1.47 I2C_SMBUS_QUICK

```
#define I2C_SMBUS_QUICK 0
```

#### 7.24.1.48 I2C_SMBUS_READ

```
#define I2C_SMBUS_READ 1
```

#### 7.24.1.49 I2C_SMBUS_WORD_DATA

```
#define I2C_SMBUS_WORD_DATA 3
```

#### 7.24.1.50 I2C_SMBUS_WRITE

```
#define I2C_SMBUS_WRITE 0
```

#### 7.24.1.51 I2C_TENBIT

```
#define I2C_TENBIT 0x0704 /* 0 for 7 bit addrs, != 0 for 10 bit */
```

#### 7.24.1.52 I2C_TIMEOUT

```
#define I2C_TIMEOUT 0x0702 /* set timeout in units of 10 ms */
```

## 7.24.2 Function Documentation

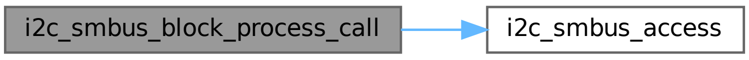
### 7.24.2.1 i2c_smbus_access()

```
static __s32 i2c_smbus_access (
             int file,
             char read_write,
             __u8 command,
             int size,
             union i2c_smbus_data * data ) [inline], [static]
```

### 7.24.2.2 i2c_smbus_block_process_call()

```
static __s32 i2c_smbus_block_process_call (
             int file,
             __u8 command,
             __u8 length,
             __u8 * values ) [inline], [static]
```

Here is the call graph for this function:



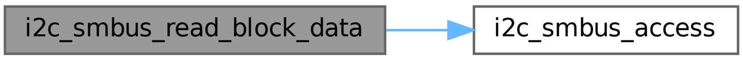
### 7.24.2.3 i2c_smbus_process_call()

```
static __s32 i2c_smbus_process_call (
             int file,
             __u8 command,
             __u16 value ) [inline], [static]
```

### 7.24.2.4 i2c_smbus_read_block_data()

```
static __s32 i2c_smbus_read_block_data (
             int file,
             __u8 command,
             __u8 * values ) [inline], [static]
```

Here is the call graph for this function:



#### 7.24.2.5 i2c_smbus_read_byte()

```
static __s32 i2c_smbus_read_byte (
             int file )  [inline], [static]
```

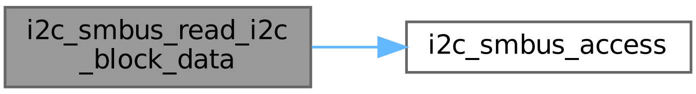
#### 7.24.2.6 i2c_smbus_read_byte_data()

```
static __s32 i2c_smbus_read_byte_data (
             int file,
             __u8 command )  [inline], [static]
```

#### 7.24.2.7 i2c_smbus_read_i2c_block_data()

```
static __s32 i2c_smbus_read_i2c_block_data (
             int file,
             __u8 command,
             __u8 length,
             __u8 * values )  [inline], [static]
```

Here is the call graph for this function:



#### 7.24.2.8 i2c_smbus_read_word_data()

```
static __s32 i2c_smbus_read_word_data (
             int file,
             __u8 command )  [inline], [static]
```

#### 7.24.2.9 i2c_smbus_write_block_data()

```
static __s32 i2c_smbus_write_block_data (
             int file,
             __u8 command,
             __u8 length,
             const __u8 * values )  [inline], [static]
```

#### 7.24.2.10 i2c_smbus_write_byte()

```
static __s32 i2c_smbus_write_byte (
             int file,
             __u8 value )  [inline], [static]
```

#### 7.24.2.11 i2c_smbus_write_byte_data()

```
static __s32 i2c_smbus_write_byte_data (
             int file,
             __u8 command,
             __u8 value )  [inline], [static]
```

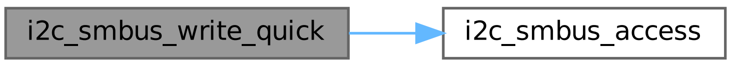
#### 7.24.2.12 i2c_smbus_write_i2c_block_data()

```
static __s32 i2c_smbus_write_i2c_block_data (
             int file,
             __u8 command,
             __u8 length,
             const __u8 * values )  [inline], [static]
```

#### 7.24.2.13 i2c_smbus_write_quick()

```
static __s32 i2c_smbus_write_quick (
             int file,
             __u8 value )  [inline], [static]
```

Here is the call graph for this function:



#### 7.24.2.14 i2c_smbus_write_word_data()

```
static __s32 i2c_smbus_write_word_data (
             int file,
             __u8 command,
             __u16 value )  [inline], [static]
```

## 7.25 i2c-dev.h

Go to the documentation of this file.

```
/*
    i2c-dev.h - i2c-bus driver, char device interface

    Copyright (C) 1995-97 Simon G. Vogl
    Copyright (C) 1998-99 Frodo Looijaard <frodol@dds.nl>

    This program is free software; you can redistribute it and/or modify
    it under the terms of the GNU General Public License as published by
    the Free Software Foundation; either version 2 of the License, or
    (at your option) any later version.

    This program is distributed in the hope that it will be useful,
    but WITHOUT ANY WARRANTY; without even the implied warranty of
    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
    GNU General Public License for more details.

    You should have received a copy of the GNU General Public License
    along with this program; if not, write to the Free Software
    Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston,
    MA 02110-1301 USA.
*/

#ifndef _LINUX_I2C_DEV_H
#define _LINUX_I2C_DEV_H

#include <linux/types.h>
#include <sys/ioctl.h>
#include <stddef.h>


/* -- i2c.h -- */


/*
 * I2C Message - used for pure i2c transaction, also from /dev interface
 */
struct i2c_msg {
        __u16 addr;     /* slave address                        */
        unsigned short flags;
#define I2C_M_TEN       0x10    /* we have a ten bit chip address       */
#define I2C_M_RD        0x01
#define I2C_M_NOSTART   0x4000
#define I2C_M_REV_DIR_ADDR      0x2000
#define I2C_M_IGNORE_NAK        0x1000
#define I2C_M_NO_RD_ACK         0x0800
        short len;              /* msg length                           */
        char *buf;              /* pointer to msg data                  */
};

/* To determine what functionality is present */

#define I2C_FUNC_I2C                    0x00000001
#define I2C_FUNC_10BIT_ADDR             0x00000002
#define I2C_FUNC_PROTOCOL_MANGLING      0x00000004 /* I2C_M_{REV_DIR_ADDR,NOSTART,..} */
#define I2C_FUNC_SMBUS_PEC              0x00000008
#define I2C_FUNC_SMBUS_BLOCK_PROC_CALL  0x00008000 /* SMBus 2.0 */
#define I2C_FUNC_SMBUS_QUICK            0x00010000
#define I2C_FUNC_SMBUS_READ_BYTE        0x00020000
#define I2C_FUNC_SMBUS_WRITE_BYTE       0x00040000
#define I2C_FUNC_SMBUS_READ_BYTE_DATA   0x00080000
#define I2C_FUNC_SMBUS_WRITE_BYTE_DATA  0x00100000
#define I2C_FUNC_SMBUS_READ_WORD_DATA   0x00200000
#define I2C_FUNC_SMBUS_WRITE_WORD_DATA  0x00400000
#define I2C_FUNC_SMBUS_PROC_CALL        0x00800000
#define I2C_FUNC_SMBUS_READ_BLOCK_DATA  0x01000000
#define I2C_FUNC_SMBUS_WRITE_BLOCK_DATA 0x02000000
#define I2C_FUNC_SMBUS_READ_I2C_BLOCK   0x04000000 /* I2C-like block xfer  */
#define I2C_FUNC_SMBUS_WRITE_I2C_BLOCK  0x08000000 /* w/ 1-byte reg. addr. */

#define I2C_FUNC_SMBUS_BYTE (I2C_FUNC_SMBUS_READ_BYTE | \
                             I2C_FUNC_SMBUS_WRITE_BYTE)
#define I2C_FUNC_SMBUS_BYTE_DATA (I2C_FUNC_SMBUS_READ_BYTE_DATA | \
                                  I2C_FUNC_SMBUS_WRITE_BYTE_DATA)
#define I2C_FUNC_SMBUS_WORD_DATA (I2C_FUNC_SMBUS_READ_WORD_DATA | \
                                  I2C_FUNC_SMBUS_WRITE_WORD_DATA)
#define I2C_FUNC_SMBUS_BLOCK_DATA (I2C_FUNC_SMBUS_READ_BLOCK_DATA | \
                                   I2C_FUNC_SMBUS_WRITE_BLOCK_DATA)
#define I2C_FUNC_SMBUS_I2C_BLOCK (I2C_FUNC_SMBUS_READ_I2C_BLOCK | \
                                  I2C_FUNC_SMBUS_WRITE_I2C_BLOCK)

/* Old name, for compatibility */
#define I2C_FUNC_SMBUS_HWPEC_CALC       I2C_FUNC_SMBUS_PEC
```

```c

/*
 * Data for SMBus Messages
 */
#define I2C_SMBUS_BLOCK_MAX     32      /* As specified in SMBus standard */
#define I2C_SMBUS_I2C_BLOCK_MAX 32      /* Not specified but we use same structure */
union i2c_smbus_data {
        __u8 byte;
        __u16 word;
        __u8 block[I2C_SMBUS_BLOCK_MAX + 2]; /* block[0] is used for length */
                                                    /* and one more for PEC */
};

/* smbus_access read or write markers */
#define I2C_SMBUS_READ  1
#define I2C_SMBUS_WRITE 0

/* SMBus transaction types (size parameter in the above functions)
   Note: these no longer correspond to the (arbitrary) PIIX4 internal codes! */
#define I2C_SMBUS_QUICK             0
#define I2C_SMBUS_BYTE              1
#define I2C_SMBUS_BYTE_DATA         2
#define I2C_SMBUS_WORD_DATA         3
#define I2C_SMBUS_PROC_CALL         4
#define I2C_SMBUS_BLOCK_DATA        5
#define I2C_SMBUS_I2C_BLOCK_BROKEN  6
#define I2C_SMBUS_BLOCK_PROC_CALL   7           /* SMBus 2.0 */
#define I2C_SMBUS_I2C_BLOCK_DATA    8


/* /dev/i2c-X ioctl commands.  The ioctl's parameter is always an
 * unsigned long, except for:
 *      - I2C_FUNCS, takes pointer to an unsigned long
 *      - I2C_RDWR, takes pointer to struct i2c_rdwr_ioctl_data
 *      - I2C_SMBUS, takes pointer to struct i2c_smbus_ioctl_data
 */
#define I2C_RETRIES     0x0701  /* number of times a device address should
                                   be polled when not acknowledging */
#define I2C_TIMEOUT     0x0702  /* set timeout in units of 10 ms */

/* NOTE: Slave address is 7 or 10 bits, but 10-bit addresses
 * are NOT supported! (due to code brokenness)
 */
#define I2C_SLAVE       0x0703  /* Use this slave address */
#define I2C_SLAVE_FORCE 0x0706  /* Use this slave address, even if it
                                   is already in use by a driver! */
#define I2C_TENBIT      0x0704  /* 0 for 7 bit addrs, != 0 for 10 bit */

#define I2C_FUNCS       0x0705  /* Get the adapter functionality mask */

#define I2C_RDWR        0x0707  /* Combined R/W transfer (one STOP only) */

#define I2C_PEC         0x0708  /* != 0 to use PEC with SMBus */
#define I2C_SMBUS       0x0720  /* SMBus transfer */


/* This is the structure as used in the I2C_SMBUS ioctl call */
struct i2c_smbus_ioctl_data {
        __u8 read_write;
        __u8 command;
        __u32 size;
        union i2c_smbus_data *data;
};

/* This is the structure as used in the I2C_RDWR ioctl call */
struct i2c_rdwr_ioctl_data {
        struct i2c_msg *msgs;   /* pointers to i2c_msgs */
        __u32 nmsgs;                    /* number of i2c_msgs */
};

#define  I2C_RDWR_IOCTL_MAX_MSGS        42


static inline __s32 i2c_smbus_access(int file, char read_write, __u8 command,
                                     int size, union i2c_smbus_data *data)
{
        struct i2c_smbus_ioctl_data args;

        args.read_write = read_write;
        args.command = command;
        args.size = size;
        args.data = data;
        return ioctl(file,I2C_SMBUS,&args);
}


static inline __s32 i2c_smbus_write_quick(int file, __u8 value)
```

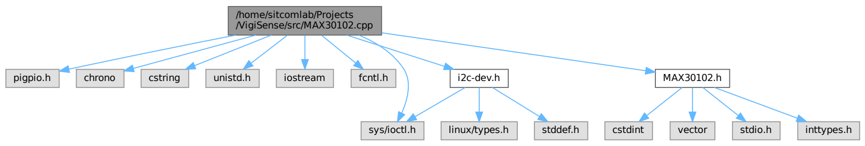
```
{
        return i2c_smbus_access(file,value,0,I2C_SMBUS_QUICK,NULL);
}

static inline __s32 i2c_smbus_read_byte(int file)
{
        union i2c_smbus_data data;
        if (i2c_smbus_access(file,I2C_SMBUS_READ,0,I2C_SMBUS_BYTE,&data))
                return -1;
        else
                return 0x0FF & data.byte;
}

static inline __s32 i2c_smbus_write_byte(int file, __u8 value)
{
        return i2c_smbus_access(file,I2C_SMBUS_WRITE,value,
                                I2C_SMBUS_BYTE,NULL);
}

static inline __s32 i2c_smbus_read_byte_data(int file, __u8 command)
{
        union i2c_smbus_data data;
        if (i2c_smbus_access(file,I2C_SMBUS_READ,command,
                             I2C_SMBUS_BYTE_DATA,&data))
                return -1;
        else
                return 0x0FF & data.byte;
}

static inline __s32 i2c_smbus_write_byte_data(int file, __u8 command,
                                              __u8 value)
{
        union i2c_smbus_data data;
        data.byte = value;
        return i2c_smbus_access(file,I2C_SMBUS_WRITE,command,
                                I2C_SMBUS_BYTE_DATA, &data);
}

static inline __s32 i2c_smbus_read_word_data(int file, __u8 command)
{
        union i2c_smbus_data data;
        if (i2c_smbus_access(file,I2C_SMBUS_READ,command,
                             I2C_SMBUS_WORD_DATA,&data))
                return -1;
        else
                return 0x0FFFF & data.word;
}

static inline __s32 i2c_smbus_write_word_data(int file, __u8 command,
                                              __u16 value)
{
        union i2c_smbus_data data;
        data.word = value;
        return i2c_smbus_access(file,I2C_SMBUS_WRITE,command,
                                I2C_SMBUS_WORD_DATA, &data);
}

static inline __s32 i2c_smbus_process_call(int file, __u8 command, __u16 value)
{
        union i2c_smbus_data data;
        data.word = value;
        if (i2c_smbus_access(file,I2C_SMBUS_WRITE,command,
                             I2C_SMBUS_PROC_CALL,&data))
                return -1;
        else
                return 0x0FFFF & data.word;
}


/* Returns the number of read bytes */
static inline __s32 i2c_smbus_read_block_data(int file, __u8 command,
                                              __u8 *values)
{
        union i2c_smbus_data data;
        int i;
        if (i2c_smbus_access(file,I2C_SMBUS_READ,command,
                             I2C_SMBUS_BLOCK_DATA,&data))
                return -1;
        else {
                for (i = 1; i <= data.block[0]; i++)
                        values[i-1] = data.block[i];
                return data.block[0];
        }
}

static inline __s32 i2c_smbus_write_block_data(int file, __u8 command,
                                               __u8 length, const __u8 *values)
```

```
{
        union i2c_smbus_data data;
        int i;
        if (length > 32)
                length = 32;
        for (i = 1; i <= length; i++)
                data.block[i] = values[i-1];
        data.block[0] = length;
        return i2c_smbus_access(file,I2C_SMBUS_WRITE,command,
                                I2C_SMBUS_BLOCK_DATA, &data);
}

/* Returns the number of read bytes */
/* Until kernel 2.6.22, the length is hardcoded to 32 bytes. If you
   ask for less than 32 bytes, your code will only work with kernels
   2.6.23 and later. */
static inline __s32 i2c_smbus_read_i2c_block_data(int file, __u8 command,
                                                  __u8 length, __u8 *values)
{
        union i2c_smbus_data data;
        int i;

        if (length > 32)
                length = 32;
        data.block[0] = length;
        if (i2c_smbus_access(file,I2C_SMBUS_READ,command,
                             length == 32 ? I2C_SMBUS_I2C_BLOCK_BROKEN :
                              I2C_SMBUS_I2C_BLOCK_DATA,&data))
                return -1;
        else {
                for (i = 1; i <= data.block[0]; i++)
                        values[i-1] = data.block[i];
                return data.block[0];
        }
}

static inline __s32 i2c_smbus_write_i2c_block_data(int file, __u8 command,
                                                   __u8 length,
                                                   const __u8 *values)
{
        union i2c_smbus_data data;
        int i;
        if (length > 32)
                length = 32;
        for (i = 1; i <= length; i++)
                data.block[i] = values[i-1];
        data.block[0] = length;
        return i2c_smbus_access(file,I2C_SMBUS_WRITE,command,
                                I2C_SMBUS_I2C_BLOCK_BROKEN, &data);
}

/* Returns the number of read bytes */
static inline __s32 i2c_smbus_block_process_call(int file, __u8 command,
                                                 __u8 length, __u8 *values)
{
        union i2c_smbus_data data;
        int i;
        if (length > 32)
                length = 32;
        for (i = 1; i <= length; i++)
                data.block[i] = values[i-1];
        data.block[0] = length;
        if (i2c_smbus_access(file,I2C_SMBUS_WRITE,command,
                             I2C_SMBUS_BLOCK_PROC_CALL,&data))
                return -1;
        else {
                for (i = 1; i <= data.block[0]; i++)
                        values[i-1] = data.block[i];
                return data.block[0];
        }
}


#endif /* _LINUX_I2C_DEV_H */
```

## 7.26 /home/sitcomlab/Projects/VigiSense/src/MAX30102.cpp File Reference

```
#include <pigpio.h>
#include <chrono>
```

```
#include <cstring>
#include <unistd.h>
#include <iostream>
#include <fcntl.h>
#include <sys/ioctl.h>
#include "i2c-dev.h"
#include "MAX30102.h"
```
Include dependency graph for MAX30102.cpp:



## Variables

- static const uint8_t REG_INTSTAT1 = 0x00
- static const uint8_t REG_INTSTAT2 = 0x01
- static const uint8_t REG_INTENABLE1 = 0x02
- static const uint8_t REG_INTENABLE2 = 0x03
- static const uint8_t REG_FIFOWRITEPTR = 0x04
- static const uint8_t REG_FIFOOVERFLOW = 0x05
- static const uint8_t REG_FIFOREADPTR = 0x06
- static const uint8_t REG_FIFODATA = 0x07
- static const uint8_t REG_FIFOCONFIG = 0x08
- static const uint8_t REG_MODECONFIG = 0x09
- static const uint8_t REG_PARTICLECONFIG = 0x0A
- static const uint8_t REG_LED1_PULSEAMP = 0x0C
- static const uint8_t REG_LED2_PULSEAMP = 0x0D
- static const uint8_t REG_LED_PROX_AMP = 0x10
- static const uint8_t REG_MULTILEDCONFIG1 = 0x11
- static const uint8_t REG_MULTILEDCONFIG2 = 0x12
- static const uint8_t REG_DIETEMPINT = 0x1F
- static const uint8_t REG_DIETEMPFRAC = 0x20
- static const uint8_t REG_DIETEMPCONFIG = 0x21
- static const uint8_t REG_PROXINTTHRESH = 0x30
- static const uint8_t REG_REVISIONID = 0xFE
- static const uint8_t REG_PARTID = 0xFF
- static const uint8_t MAX30102_EXPECTEDPARTID = 0x15
- static const uint8_t MASK_INT_A_FULL = (uint8_t)∼0b10000000
- static const uint8_t INT_A_FULL_ENABLE = 0x80
- static const uint8_t INT_A_FULL_DISABLE = 0x00
- static const uint8_t MASK_INT_DATA_RDY = (uint8_t)∼0b01000000
- static const uint8_t INT_DATA_RDY_ENABLE = 0x40
- static const uint8_t INT_DATA_RDY_DISABLE = 0x00
- static const uint8_t MASK_INT_ALC_OVF = (uint8_t)∼0b00100000
- static const uint8_t INT_ALC_OVF_ENABLE = 0x20
- static const uint8_t INT_ALC_OVF_DISABLE = 0x00
- static const uint8_t MASK_INT_PROX_INT = (uint8_t)∼0b00010000
- static const uint8_t INT_PROX_INT_ENABLE = 0x10
- static const uint8_t INT_PROX_INT_DISABLE = 0x00

- static const uint8_t MASK_INT_DIE_TEMP_RDY = (uint8_t)∼0b00000010
- static const uint8_t INT_DIE_TEMP_RDY_ENABLE = 0x02
- static const uint8_t INT_DIE_TEMP_RDY_DISABLE = 0x00
- static const uint8_t MASK_SAMPLEAVG = (uint8_t)∼0b11100000
- static const uint8_t SAMPLEAVG_1 = 0x00
- static const uint8_t SAMPLEAVG_2 = 0x20
- static const uint8_t SAMPLEAVG_4 = 0x40
- static const uint8_t SAMPLEAVG_8 = 0x60
- static const uint8_t SAMPLEAVG_16 = 0x80
- static const uint8_t SAMPLEAVG_32 = 0xA0
- static const uint8_t MASK_ROLLOVER = 0xEF
- static const uint8_t ROLLOVER_ENABLE = 0x10
- static const uint8_t ROLLOVER_DISABLE = 0x00
- static const uint8_t MASK_A_FULL = 0xF0
- static const uint8_t MASK_SHUTDOWN = 0x7f
- static const uint8_t SHUTDOWN = 0x80
- static const uint8_t WAKEUP = 0x00
- static const uint8_t MASK_RESET = 0xBF
- static const uint8_t RESET = 0X40
- static const uint8_t MASK_LEDMODE = 0xF8

  *IR led mode.*
- static const uint8_t LEDMODE_REDONLY = 0x02
- static const uint8_t LEDMODE_REDIRONLY = 0x03
- static const uint8_t LEDMODE_MULTILED = 0x07
- static const uint8_t MASK_ADCRANGE = 0x9F
- static const uint8_t ADCRANGE_2048 = 0x00
- static const uint8_t ADCRANGE_4096 = 0x20
- static const uint8_t ADCRANGE_8192 = 0x40
- static const uint8_t ADCRANGE_16384 = 0x60
- static const uint8_t MASK_SAMPLERATE = 0xE3
- static const uint8_t SAMPLERATE_50 = 0x00
- static const uint8_t SAMPLERATE_100 = 0x04
- static const uint8_t SAMPLERATE_200 = 0x08
- static const uint8_t SAMPLERATE_400 = 0x0C
- static const uint8_t SAMPLERATE_800 = 0x10
- static const uint8_t SAMPLERATE_1000 = 0x14
- static const uint8_t SAMPLERATE_1600 = 0x18
- static const uint8_t SAMPLERATE_3200 = 0x1C
- static const uint8_t MASK_PULSEWIDTH = 0xFC
- static const uint8_t PULSEWIDTH_69 = 0x00
- static const uint8_t PULSEWIDTH_118 = 0x01
- static const uint8_t PULSEWIDTH_215 = 0x02
- static const uint8_t PULSEWIDTH_411 = 0x03
- static const uint8_t MASK_SLOT1 = 0xF8
- static const uint8_t MASK_SLOT2 = 0x8F
- static const uint8_t MASK_SLOT3 = 0xF8
- static const uint8_t MASK_SLOT4 = 0x8F
- static const uint8_t SLOT_NONE = 0x00
- static const uint8_t SLOT_RED_LED = 0x01
- static const uint8_t SLOT_IR_LED = 0x02
- static const uint8_t SLOT_NONE_PILOT = 0x04
- static const uint8_t SLOT_RED_PILOT = 0x05
- static const uint8_t SLOT_IR_PILOT = 0x06

### 7.26.1 Variable Documentation

#### 7.26.1.1 ADCRANGE_16384

```
const uint8_t ADCRANGE_16384 = 0x60  [static]
```

#### 7.26.1.2 ADCRANGE_2048

```
const uint8_t ADCRANGE_2048 = 0x00  [static]
```

#### 7.26.1.3 ADCRANGE_4096

```
const uint8_t ADCRANGE_4096 = 0x20  [static]
```

#### 7.26.1.4 ADCRANGE_8192

```
const uint8_t ADCRANGE_8192 = 0x40  [static]
```

#### 7.26.1.5 INT_A_FULL_DISABLE

```
const uint8_t INT_A_FULL_DISABLE = 0x00  [static]
```

#### 7.26.1.6 INT_A_FULL_ENABLE

```
const uint8_t INT_A_FULL_ENABLE = 0x80  [static]
```

#### 7.26.1.7 INT_ALC_OVF_DISABLE

```
const uint8_t INT_ALC_OVF_DISABLE = 0x00  [static]
```

#### 7.26.1.8 INT_ALC_OVF_ENABLE

```
const uint8_t INT_ALC_OVF_ENABLE = 0x20  [static]
```

#### 7.26.1.9 INT_DATA_RDY_DISABLE

```
const uint8_t INT_DATA_RDY_DISABLE = 0x00  [static]
```

#### 7.26.1.10 INT_DATA_RDY_ENABLE

```
const uint8_t INT_DATA_RDY_ENABLE = 0x40  [static]
```

#### 7.26.1.11 INT_DIE_TEMP_RDY_DISABLE

```
const uint8_t INT_DIE_TEMP_RDY_DISABLE = 0x00  [static]
```

#### 7.26.1.12 INT_DIE_TEMP_RDY_ENABLE

```
const uint8_t INT_DIE_TEMP_RDY_ENABLE = 0x02  [static]
```

#### 7.26.1.13 INT_PROX_INT_DISABLE

```
const uint8_t INT_PROX_INT_DISABLE = 0x00  [static]
```

#### 7.26.1.14 INT_PROX_INT_ENABLE

```
const uint8_t INT_PROX_INT_ENABLE = 0x10  [static]
```

#### 7.26.1.15 LEDMODE_MULTILED

```
const uint8_t LEDMODE_MULTILED = 0x07  [static]
```

#### 7.26.1.16 LEDMODE_REDIRONLY

```
const uint8_t LEDMODE_REDIRONLY = 0x03  [static]
```

#### 7.26.1.17 LEDMODE_REDONLY

```
const uint8_t LEDMODE_REDONLY = 0x02  [static]
```

#### 7.26.1.18 MASK_A_FULL

```
const uint8_t MASK_A_FULL = 0xF0  [static]
```

#### 7.26.1.19 MASK_ADCRANGE

```
const uint8_t MASK_ADCRANGE = 0x9F  [static]
```

#### 7.26.1.20 MASK_INT_A_FULL

```
const uint8_t MASK_INT_A_FULL = (uint8_t)~0b10000000  [static]
```

#### 7.26.1.21 MASK_INT_ALC_OVF

```
const uint8_t MASK_INT_ALC_OVF = (uint8_t)∼0b00100000  [static]
```

#### 7.26.1.22 MASK_INT_DATA_RDY

```
const uint8_t MASK_INT_DATA_RDY = (uint8_t)∼0b01000000  [static]
```

#### 7.26.1.23 MASK_INT_DIE_TEMP_RDY

```
const uint8_t MASK_INT_DIE_TEMP_RDY = (uint8_t)∼0b00000010  [static]
```

#### 7.26.1.24 MASK_INT_PROX_INT

```
const uint8_t MASK_INT_PROX_INT = (uint8_t)∼0b00010000  [static]
```

#### 7.26.1.25 MASK_LEDMODE

```
const uint8_t MASK_LEDMODE = 0xF8  [static]
```

IR led mode.

#### 7.26.1.26 MASK_PULSEWIDTH

```
const uint8_t MASK_PULSEWIDTH = 0xFC  [static]
```

#### 7.26.1.27 MASK_RESET

```
const uint8_t MASK_RESET = 0xBF  [static]
```

#### 7.26.1.28 MASK_ROLLOVER

```
const uint8_t MASK_ROLLOVER = 0xEF  [static]
```

#### 7.26.1.29 MASK_SAMPLEAVG

```
const uint8_t MASK_SAMPLEAVG = (uint8_t)∼0b11100000  [static]
```

#### 7.26.1.30 MASK_SAMPLERATE

```
const uint8_t MASK_SAMPLERATE = 0xE3  [static]
```

#### 7.26.1.31 MASK_SHUTDOWN

```
const uint8_t MASK_SHUTDOWN = 0x7f  [static]
```

#### 7.26.1.32 MASK_SLOT1

```
const uint8_t MASK_SLOT1 = 0xF8  [static]
```

#### 7.26.1.33 MASK_SLOT2

```
const uint8_t MASK_SLOT2 = 0x8F  [static]
```

#### 7.26.1.34 MASK_SLOT3

```
const uint8_t MASK_SLOT3 = 0xF8  [static]
```

#### 7.26.1.35 MASK_SLOT4

```
const uint8_t MASK_SLOT4 = 0x8F  [static]
```

#### 7.26.1.36 MAX30102_EXPECTEDPARTID

```
const uint8_t MAX30102_EXPECTEDPARTID = 0x15  [static]
```

#### 7.26.1.37 PULSEWIDTH_118

```
const uint8_t PULSEWIDTH_118 = 0x01  [static]
```

#### 7.26.1.38 PULSEWIDTH_215

```
const uint8_t PULSEWIDTH_215 = 0x02  [static]
```

#### 7.26.1.39 PULSEWIDTH_411

```
const uint8_t PULSEWIDTH_411 = 0x03  [static]
```

#### 7.26.1.40 PULSEWIDTH_69

```
const uint8_t PULSEWIDTH_69 = 0x00  [static]
```

#### 7.26.1.41 REG_DIETEMPCONFIG

```
const uint8_t REG_DIETEMPCONFIG = 0x21  [static]
```

#### 7.26.1.42 REG_DIETEMPFRAC

```
const uint8_t REG_DIETEMPFRAC = 0x20  [static]
```

#### 7.26.1.43 REG_DIETEMPINT

```
const uint8_t REG_DIETEMPINT = 0x1F  [static]
```

#### 7.26.1.44 REG_FIFOCONFIG

```
const uint8_t REG_FIFOCONFIG = 0x08  [static]
```

#### 7.26.1.45 REG_FIFODATA

```
const uint8_t REG_FIFODATA = 0x07  [static]
```

#### 7.26.1.46 REG_FIFOOVERFLOW

```
const uint8_t REG_FIFOOVERFLOW = 0x05  [static]
```

#### 7.26.1.47 REG_FIFOREADPTR

```
const uint8_t REG_FIFOREADPTR = 0x06  [static]
```

#### 7.26.1.48 REG_FIFOWRITEPTR

```
const uint8_t REG_FIFOWRITEPTR = 0x04  [static]
```

#### 7.26.1.49 REG_INTENABLE1

```
const uint8_t REG_INTENABLE1 = 0x02  [static]
```

#### 7.26.1.50 REG_INTENABLE2

```
const uint8_t REG_INTENABLE2 = 0x03  [static]
```

#### 7.26.1.51 REG_INTSTAT1

```
const uint8_t REG_INTSTAT1 = 0x00  [static]
```

#### 7.26.1.52 REG_INTSTAT2

```
const uint8_t REG_INTSTAT2 = 0x01  [static]
```

#### 7.26.1.53 REG_LED1_PULSEAMP

```
const uint8_t REG_LED1_PULSEAMP = 0x0C  [static]
```

#### 7.26.1.54 REG_LED2_PULSEAMP

```
const uint8_t REG_LED2_PULSEAMP = 0x0D  [static]
```

#### 7.26.1.55 REG_LED_PROX_AMP

```
const uint8_t REG_LED_PROX_AMP = 0x10  [static]
```

#### 7.26.1.56 REG_MODECONFIG

```
const uint8_t REG_MODECONFIG = 0x09  [static]
```

#### 7.26.1.57 REG_MULTILEDCONFIG1

```
const uint8_t REG_MULTILEDCONFIG1 = 0x11  [static]
```

#### 7.26.1.58 REG_MULTILEDCONFIG2

```
const uint8_t REG_MULTILEDCONFIG2 = 0x12  [static]
```

#### 7.26.1.59 REG_PARTICLECONFIG

```
const uint8_t REG_PARTICLECONFIG = 0x0A  [static]
```

#### 7.26.1.60 REG_PARTID

```
const uint8_t REG_PARTID = 0xFF  [static]
```

#### 7.26.1.61 REG_PROXINTTHRESH

```
const uint8_t REG_PROXINTTHRESH = 0x30  [static]
```

#### 7.26.1.62 REG_REVISIONID

```
const uint8_t REG_REVISIONID = 0xFE  [static]
```

#### 7.26.1.63 RESET

```
const uint8_t RESET = 0X40  [static]
```

#### 7.26.1.64 ROLLOVER_DISABLE

```
const uint8_t ROLLOVER_DISABLE = 0x00  [static]
```

#### 7.26.1.65 ROLLOVER_ENABLE

```
const uint8_t ROLLOVER_ENABLE = 0x10  [static]
```

#### 7.26.1.66 SAMPLEAVG_1

```
const uint8_t SAMPLEAVG_1 = 0x00  [static]
```

#### 7.26.1.67 SAMPLEAVG_16

```
const uint8_t SAMPLEAVG_16 = 0x80  [static]
```

#### 7.26.1.68 SAMPLEAVG_2

```
const uint8_t SAMPLEAVG_2 = 0x20  [static]
```

#### 7.26.1.69 SAMPLEAVG_32

```
const uint8_t SAMPLEAVG_32 = 0xA0  [static]
```

#### 7.26.1.70 SAMPLEAVG_4

```
const uint8_t SAMPLEAVG_4 = 0x40  [static]
```

#### 7.26.1.71 SAMPLEAVG_8

```
const uint8_t SAMPLEAVG_8 = 0x60  [static]
```

#### 7.26.1.72 SAMPLERATE_100

```
const uint8_t SAMPLERATE_100 = 0x04  [static]
```

#### 7.26.1.73 SAMPLERATE_1000

```
const uint8_t SAMPLERATE_1000 = 0x14  [static]
```

#### 7.26.1.74 SAMPLERATE_1600

```
const uint8_t SAMPLERATE_1600 = 0x18  [static]
```

#### 7.26.1.75 SAMPLERATE_200

```
const uint8_t SAMPLERATE_200 = 0x08  [static]
```

#### 7.26.1.76 SAMPLERATE_3200

```
const uint8_t SAMPLERATE_3200 = 0x1C  [static]
```

#### 7.26.1.77 SAMPLERATE_400

```
const uint8_t SAMPLERATE_400 = 0x0C  [static]
```

#### 7.26.1.78 SAMPLERATE_50

```
const uint8_t SAMPLERATE_50 = 0x00  [static]
```

#### 7.26.1.79 SAMPLERATE_800

```
const uint8_t SAMPLERATE_800 = 0x10  [static]
```

#### 7.26.1.80 SHUTDOWN

```
const uint8_t SHUTDOWN = 0x80  [static]
```

#### 7.26.1.81 SLOT_IR_LED

```
const uint8_t SLOT_IR_LED = 0x02  [static]
```

#### 7.26.1.82 SLOT_IR_PILOT

```
const uint8_t SLOT_IR_PILOT = 0x06  [static]
```

#### 7.26.1.83 SLOT_NONE

```
const uint8_t SLOT_NONE = 0x00  [static]
```

#### 7.26.1.84 SLOT_NONE_PILOT

```
const uint8_t SLOT_NONE_PILOT = 0x04  [static]
```

#### 7.26.1.85 SLOT_RED_LED

```
const uint8_t SLOT_RED_LED = 0x01  [static]
```

#### 7.26.1.86 SLOT_RED_PILOT

```
const uint8_t SLOT_RED_PILOT = 0x05  [static]
```

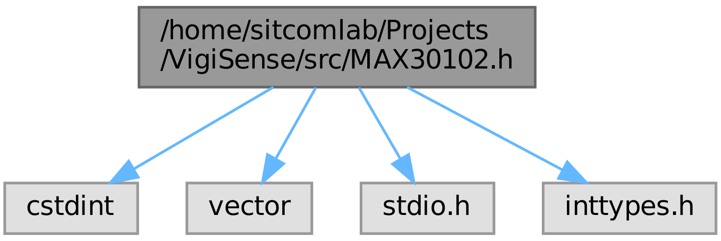
#### 7.26.1.87 WAKEUP

```
const uint8_t WAKEUP = 0x00  [static]
```

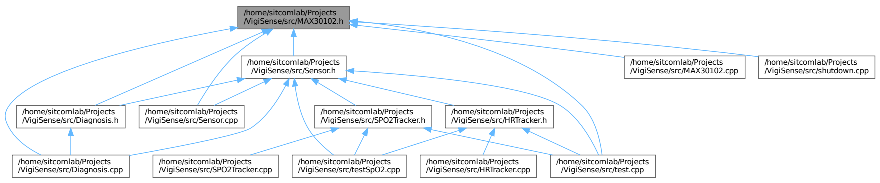
## 7.27 /home/sitcomlab/Projects/VigiSense/src/MAX30102.h File Reference

```
#include <cstdint>
#include <vector>
#include <stdio.h>
#include <inttypes.h>
```

Include dependency graph for MAX30102.h:



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


/home/sitcomlab/Projects /VigiSense/src/MAX30102.h
/home/sitcomlab/Projects /VigiSense/src/Sensor.h
/home/sitcomlab/Projects /VigiSense/src/MAX30102.cpp
/home/sitcomlab/Projects /VigiSense/src/shutdown.cpp
/home/sitcomlab/Projects /VigiSense/src/Diagnosis.h
/home/sitcomlab/Projects /VigiSense/src/Sensor.cpp
/home/sitcomlab/Projects /VigiSense/src/SPO2Tracker.h
/home/sitcomlab/Projects /VigiSense/src/HRTracker.h
/home/sitcomlab/Projects /VigiSense/src/Diagnosis.cpp
/home/sitcomlab/Projects /VigiSense/src/SPO2Tracker.cpp
/home/sitcomlab/Projects /VigiSense/src/testSpO2.cpp
/home/sitcomlab/Projects /VigiSense/src/HRTracker.cpp
/home/sitcomlab/Projects /VigiSense/src/test.cpp

### Classes

- class MAX30102
- struct MAX30102::Record

### Macros

- #define MAX30102_ADDRESS 0x57
- #define I2C_SPEED_STANDARD 100000
- #define I2C_SPEED_FAST 400000
- #define I2C_BUFFER_LENGTH 32
- #define DEFAULT_INT_GPIO 0
- #define STORAGE_SIZE 4

## 7.27.1 Macro Definition Documentation

### 7.27.1.1 DEFAULT_INT_GPIO

```
#define DEFAULT_INT_GPIO 0
```

### 7.27.1.2 I2C_BUFFER_LENGTH

```
#define I2C_BUFFER_LENGTH 32
```

### 7.27.1.3 I2C_SPEED_FAST

```
#define I2C_SPEED_FAST 400000
```

### 7.27.1.4 I2C_SPEED_STANDARD

```
#define I2C_SPEED_STANDARD 100000
```

### 7.27.1.5 MAX30102_ADDRESS

```
#define MAX30102_ADDRESS 0x57
```

#### 7.27.1.6 STORAGE_SIZE

```
#define STORAGE_SIZE 4
```

## 7.28 MAX30102.h

Go to the documentation of this file.

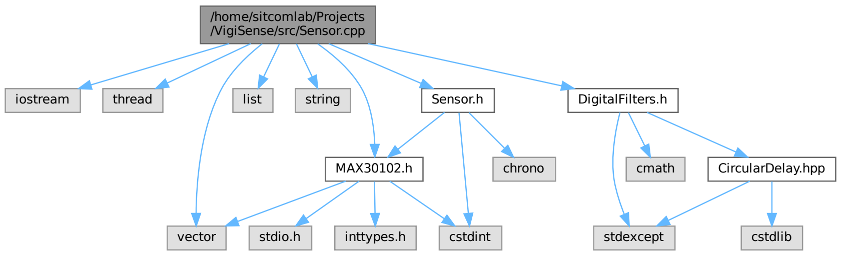
```
/*****************************************************
  This is a library written for the Maxim MAX30102 Optical Smoke Detector
  It should also work with the MAX30102. However, the MAX30102 does not have a Green LED.

  These sensors use I2C to communicate, as well as a single (optional)
  interrupt line that is not currently supported in this driver.

  Written by Garettluu (https://github.com/garrettluu)
  BSD license, all text above must be included in any redistribution.
 *****************************************************/

/*
Edited by Yu Kit Foo, to include data extraction using Interrupts
Functions changed: changed DEFAULT_INT_GPIO, added hasSample(), dataReady(), gpioISR()
*/
#pragma once
#include <cstdint>
#include <vector>
#include <stdio.h>
#include <inttypes.h>

#define MAX30102_ADDRESS    0x57

#define I2C_SPEED_STANDARD  100000
#define I2C_SPEED_FAST      400000

#define I2C_BUFFER_LENGTH   32

// define the GPIO used for the sensor here
#define DEFAULT_INT_GPIO    0

class MAX30102 {
    public:
        MAX30102(void);
        int begin(uint32_t i2cSpeed = I2C_SPEED_STANDARD, uint8_t i2cAddr = MAX30102_ADDRESS);

        uint32_t getRed(void); // Returns immediate red value
        uint32_t getIR(void); // Returns immediate IR value
        bool safeCheck(uint8_t maxTimeToCheck); // Given a max amount of time, checks for new data.

        // Configuration
        void wakeUp();
        void shutDown();
        void softReset();

        void setLEDMode(uint8_t mode);

        void setADCRange(uint8_t adcRange);
        void setSampleRate(uint8_t sampleRate);
        void setPulseWidth(uint8_t pulseWidth);

        void setPulseAmplitudeRed(uint8_t value);
        void setPulseAmplitudeIR(uint8_t value);
        void setPulseAmplitudeProximity(uint8_t value);

        void setProximityThreshold(uint8_t thresMSB);


        // Multi-LED configuration mode
        void enableSlot(uint8_t slotNumber, uint8_t device);
        void disableSlots(void);

        // Data Collection

        // Interrupts
        uint8_t getINT1(void);
        uint8_t getINT2(void);
        void enableAFULL(void);
        void disableAFULL(void);
        void enableDATARDY(void);
        void disableDATARDY(void);
        void enableALCOVF(void);
        void disableALCOVF(void);
        void enablePROXINT(void);
        void disablePROXINT(void);
        void enableDIETEMPRDY(void);
        void disableDIETEMPRDY(void);

        // FIFO Configurations
        void setFIFOAverage(uint8_t samples);
        void enableFIFORollover();
        void disableFIFORollover();
        void setFIFOAlmostFull(uint8_t samples);

        // FIFO Reading
        uint16_t check(void);
        uint8_t available(void);
        void nextSample(void);
        uint32_t getFIFORed(void);
        uint32_t getFIFOIR(void);

        uint8_t getWritePointer(void);
        uint8_t getReadPointer(void);
        void clearFIFO(void);

        // Proximity Mode Interrupt Threshold
        void setPROXINTTHRESH(uint8_t val);

        // Die Temperature
        float readTemperature();
        float readTemperatureF();

        // Detecting ID/Revision
        uint8_t getRevisionID();
        uint8_t readPartID();


        virtual void hasSample();

        // Setup the sensor with user selectable settings
        void setup(uint8_t powerLevel = 0x1F, uint8_t sampleAverage = 4, uint8_t ledMode = 2, int sampleRate = 400, int pulseWidth = 411, int adcRange = 4096);
    private:
        int _i2c;
        uint8_t _i2caddr;

        uint8_t activeLEDs;

        uint8_t revisionID;

        void readRevisionID();

        void bitMask(uint8_t reg, uint8_t mask, uint8_t thing);

        std::vector<uint8_t> readMany(uint8_t address, uint8_t length);

        #define STORAGE_SIZE 4
        typedef struct Record {
            uint32_t red[STORAGE_SIZE];
            uint32_t IR[STORAGE_SIZE];
            uint8_t head;
            uint8_t tail;
        } sense_struct;
        sense_struct sense;

        void dataReady();

        static void gpioISR(int, int, uint32_t, void* userdata) {
            ((MAX30102*)userdata)->dataReady();
        }
};
```

## 7.29 /home/sitcomlab/Projects/VigiSense/src/Sensor.cpp File Reference

```
#include <iostream>
#include <thread>
#include <vector>
#include <list>
#include <string>
#include "MAX30102.h"
```

```
#include "Sensor.h"
#include "DigitalFilters.h"
```

Include dependency graph for Sensor.cpp:



### Variables

- LowPassFilter lpf (0.08, M_PI)
- HighPassFilter hpf (0.08, M_PI)
- float R
- float SpO2
- bool crest = false
- bool trough = false
- uint8_t dataBeenIncreasing = 0
- uint8_t nextPastPeaksIndex = 0

## 7.29.1 Variable Documentation

### 7.29.1.1 crest

```
bool crest = false
```

### 7.29.1.2 dataBeenIncreasing

```
uint8_t dataBeenIncreasing = 0
```

### 7.29.1.3 hpf

```
HighPassFilter hpf(0.08, M_PI) (
             0.  08,
             M_PI  )
```

### 7.29.1.4 lpf

```
LowPassFilter lpf(0.08, M_PI) (
             0.  08,
             M_PI  )
```

#### 7.29.1.5 nextPastPeaksIndex

```
uint8_t nextPastPeaksIndex = 0
```

#### 7.29.1.6 R

```
float R
```

#### 7.29.1.7 SpO2

```
float SpO2
```

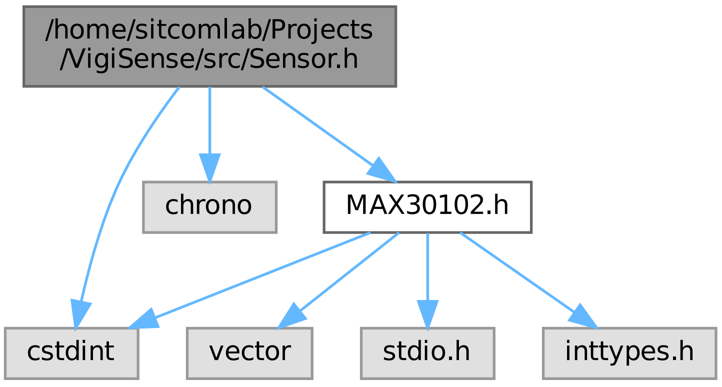
#### 7.29.1.8 trough

```
bool trough = false
```

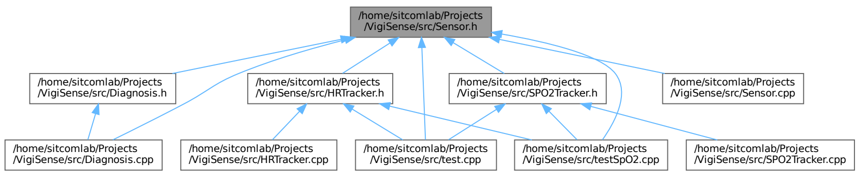
## 7.30 /home/sitcomlab/Projects/VigiSense/src/Sensor.h File Reference

```
#include <cstdint>
#include <chrono>
#include "MAX30102.h"
```

Include dependency graph for Sensor.h:



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



### Classes

- class sensor

## 7.31 Sensor.h

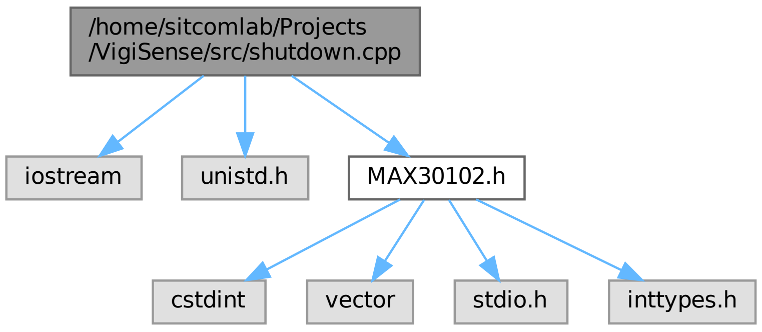
Go to the documentation of this file.

```
#pragma once
#include <cstdint>
#include <chrono>
#include "MAX30102.h"

// struct symptomRange{
//     float min;
//     float max;
//     std::string symptom;
// };

//Code refactored from HeartRate.h

class sensor{
    public:

        sensor(MAX30102 *sensor);
        ~sensor();
        //Check which functions are fine as is and which need editing/removing
        void begin();
        void stop();

        float getLatestTemperatureF();
        void HRcalc();
        void stopHRcalc();

        // getter for SPO2 and HR
        int getSpO2();
        int getHR();

    protected:
        MAX30102* _sensor;
        bool running = false;
        bool runningHR = false;

        //Check which functions are fine as is and which need editing/removing

        const int static BPM_BUFFER_SIZE = 4; // Change based on how many samples you want to average
        int32_t bpmBuffer[BPM_BUFFER_SIZE];
        int nextBPMBufferIndex = 0;

        const int static SPO2_BUFFER_SIZE = 4; // Change based on how many samples you want to average
        int32_t spo2Buffer[SPO2_BUFFER_SIZE];
        int nextSPO2BufferIndex = 0;

        std::chrono::time_point<std::chrono::system_clock> timeLastLoopRan;
        // IR Data
        std::chrono::time_point<std::chrono::system_clock> timeLastIRHeartBeat;
        int32_t irLastValue;
        int latestIRBPM;
        int averageIRBPM;
        // Red Data
        std::chrono::time_point<std::chrono::system_clock> timeLastRedHeartBeat;
        uint64_t redLastValue;
        int latestRedBPM;
        // Temperature Data
        float latestTemperature = -999;

        // For Peak Detection
        int32_t localMaximaIR;
        int32_t localMinimaIR;
        int32_t localMaximaRed;
        int32_t localMinimaRed;
        const static int8_t PAST_PEAKS_SIZE = 2;
        int32_t pastMaximasIR[PAST_PEAKS_SIZE];
        int32_t pastMinimasIR[PAST_PEAKS_SIZE];
        int32_t pastMaximasRed[PAST_PEAKS_SIZE];
        int32_t pastMinimasRed[PAST_PEAKS_SIZE];
        // SpO2 data
        int R;
        int latestSpO2;


        void loopThread();
        void runHRCalculationLoop();
        void updateTemperature();
        void resetCalculations();
        int32_t Derivative(int32_t data);
        int32_t getPeakThreshold();
        bool peakDetect(int32_t data);
};

// class spO2Measure : public sensor{
//  public:
//      std::vector<symptomRange> symptomRanges {
//          {0,88,"Critically Low Oxygen concentration"},
//          {88,92,"Concerningly Low Oxygen Concentration"},
//          {92,100,"Healthy Oxygen Concentration"}};
//      float critLow = 88;
//      int getSpO2();
//      std::string determineSymptom(float baseline);
// };

// class heartRateMeasure : public sensor{
//  public:
//      std::vector<symptomRange> symptomRanges {
//          {0,60,"Bradycardia"},
//          {60,100,"Normal resting heart rate"},
//          {100,200,"Tachyacardia"}};
//      float critHigh = 120;
//      float critLow = 40;
//      int getSafeIRHeartRate();
//      int getLatestIRHeartRate();
//      int getLatestRedHeartRate();
//      std::string determineSymptom(float baseline);
// };
```

## 7.32 /home/sitcomlab/Projects/VigiSense/src/shutdown.cpp File Reference

```
#include <iostream>
#include <unistd.h>
#include "MAX30102.h"
```

Include dependency graph for shutdown.cpp:



### Functions

- int main (void)

**Variables**

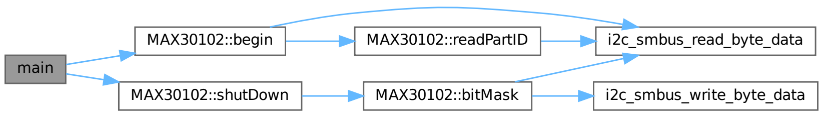
- MAX30102 heartSensor

### 7.32.1 Function Documentation

#### 7.32.1.1 main()

```
int main (
            void )
```

Here is the call graph for this function:



### 7.32.2 Variable Documentation

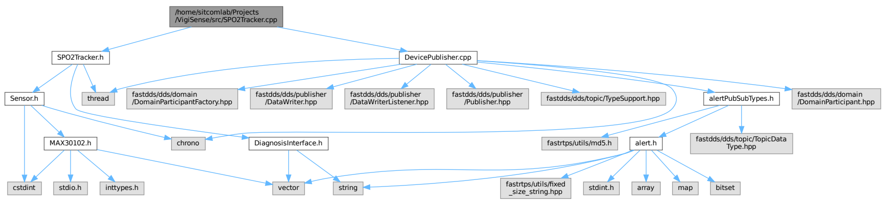
#### 7.32.2.1 heartSensor

```
MAX30102 heartSensor
```

## 7.33 /home/sitcomlab/Projects/VigiSense/src/SPO2Tracker.cpp File Reference

```
#include "SPO2Tracker.h"
#include "DevicePublisher.cpp"
```

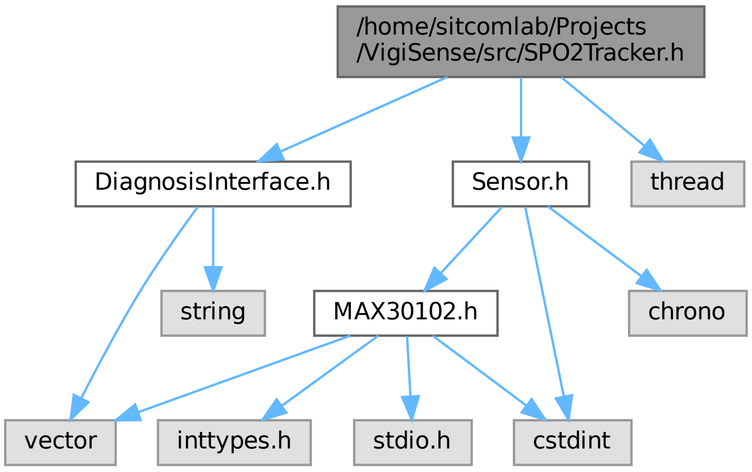
Include dependency graph for SPO2Tracker.cpp:


/home/sitcomlab/Projects /VigiSense/src/SPO2Tracker.cpp
SPO2Tracker.h
DevicePublisher.cpp
Sensor.h
thread
fastdds/dds/domain /DomainParticipantFactory.hpp
fastdds/dds/publisher /DataWriter.hpp
fastdds/dds/publisher /DataWriterListener.hpp
fastdds/dds/publisher /Publisher.hpp
fastdds/dds/topic/TypeSupport.hpp
alertPubSubTypes.h
fastdds/dds/domain /DomainParticipant.hpp
MAX30102.h
chrono
DiagnosisInterface.h
fastrtps/utils/md5.h
alert.h
fastdds/dds/topic/TopicData Type.hpp
cstdint
stdio.h
inttypes.h
vector
string
fastrtps/utils/fixed _size_string.hpp
stdint.h
array
map
bitset

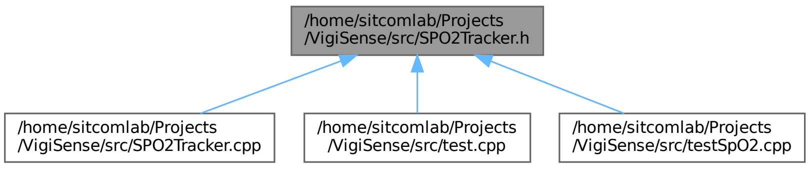
## 7.34 /home/sitcomlab/Projects/VigiSense/src/SPO2Tracker.h File Reference

```
#include "DiagnosisInterface.h"
#include "Sensor.h"
#include <thread>
```

Include dependency graph for SPO2Tracker.h:



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



### Classes

- class SPO2Tracker

## 7.35 SPO2Tracker.h

Go to the documentation of this file.

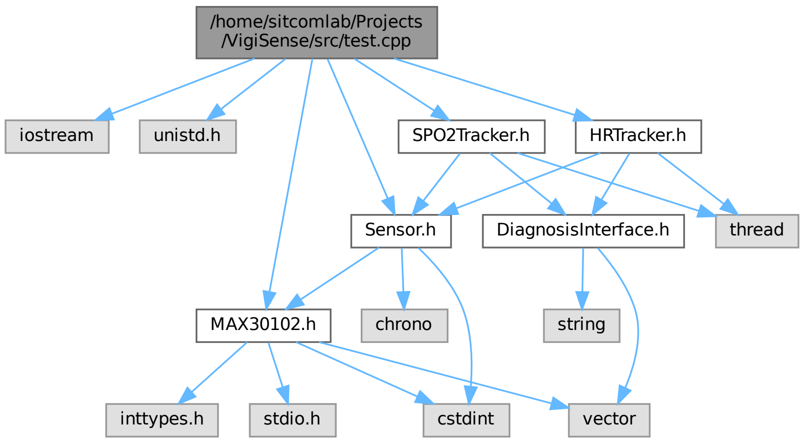
```
00001 #pragma once
00002 #include "DiagnosisInterface.h"
00003 #include "Sensor.h"
```

```
#include <thread>

class SPO2Tracker:public diagnosisInterface {
    public:
        SPO2Tracker(sensor *s);
        ~SPO2Tracker();
        void start();
        void stop();
        void ping();
        int getVal();
        void tracker();
    protected:
        sensor* _s;
        bool threadRunning = false;
        void pingThread();
        // define symptom table here
        std::vector<symptomRange> symptomRanges {
            {0,88,"Critically Low Oxygen concentration"},
            {88,92,"Concerningly Low Oxygen Concentration"},
            {92,100,"Healthy Oxygen Concentration"}};

};
```

## 7.36 /home/sitcomlab/Projects/VigiSense/src/test.cpp File Reference

```
#include <iostream>
#include <unistd.h>
#include "MAX30102.h"
#include "Sensor.h"
#include "SPO2Tracker.h"
#include "HRTracker.h"
```

Include dependency graph for test.cpp:



### Functions

- int main (void)

### Variables

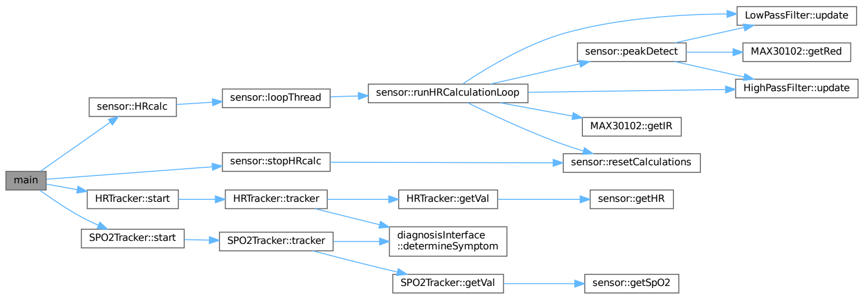
- MAX30102 heartSensor

### 7.36.1 Function Documentation

#### 7.36.1.1 main()

```
int main (
            void )
```

Here is the call graph for this function:



### 7.36.2 Variable Documentation

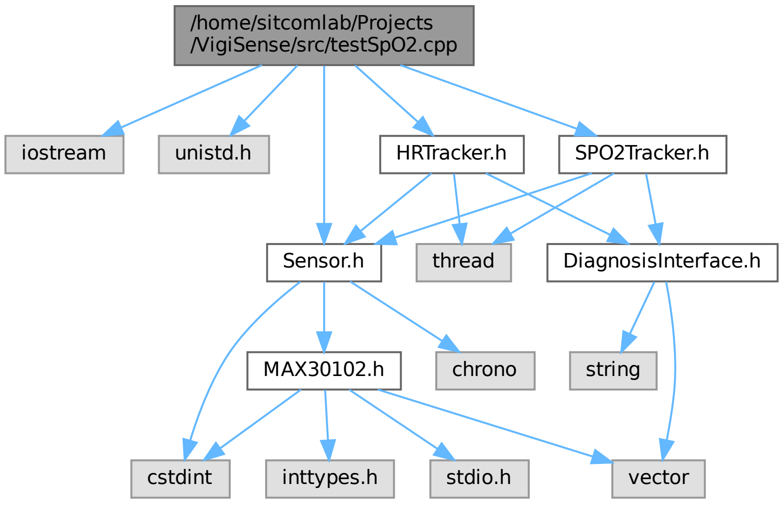
#### 7.36.2.1 heartSensor

```
MAX30102 heartSensor
```

## 7.37 /home/sitcomlab/Projects/VigiSense/src/testSpO2.cpp File Reference

```
#include <iostream>
#include <unistd.h>
#include "Sensor.h"
#include "SPO2Tracker.h"
#include "HRTracker.h"
```

Include dependency graph for testSpO2.cpp:



## Functions

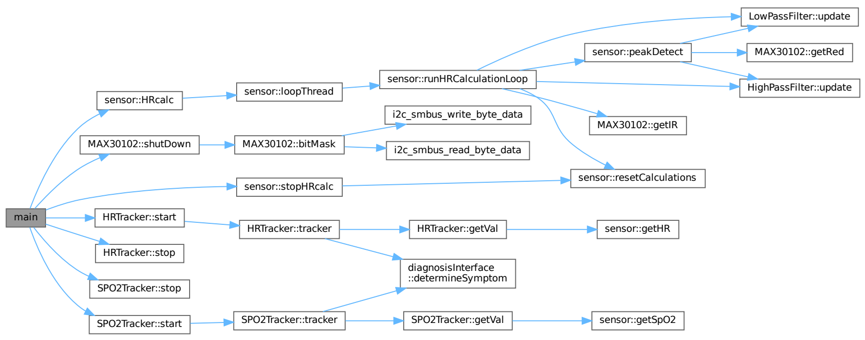
- int main ()

## 7.37.1 Function Documentation

### 7.37.1.1 main()

```
int main (
            void  )
```

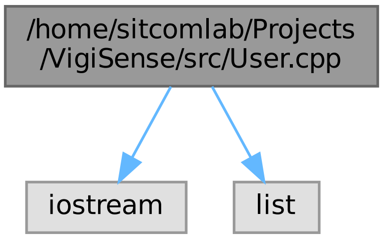
Here is the call graph for this function:



## 7.38 /home/sitcomlab/Projects/VigiSense/src/User.cpp File Reference

```
#include <iostream>
#include <list>
```

Include dependency graph for User.cpp:



#### Classes

- struct contact

#### Functions

- User (string n)
- addContact (string n, string e, long long p)

#### Variables

- struct contact name
- list< contact > contacts

### 7.38.1 Function Documentation

#### 7.38.1.1 addContact()

```
addContact (
            string n,
            string e,
            long long p )
```

#### 7.38.1.2 User()

```
User (
            string n )
```

### 7.38.2 Variable Documentation

#### 7.38.2.1 contacts

```
list<contact> contacts
```

#### 7.38.2.2 name

```
struct contact name
```

# Index