Pub

Generated by Doxygen 1.8.17

Chapter 1

Class Index

1.1 Class List

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

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

Chapter 3

Class Documentation

3.1 rt_pub Class Reference

real-time publisher

```
#include <rt_pub.hpp>
```

Public Member Functions

- rt_pub (const rt_pub &)=delete

    copy constructor deleted
- rt_pub & operator= (const rt_pub &)=delete

    assignment operator deleted
- rt_pub (int const signal_type, int const var_id)

    constructor of real-time publisher
- ∼rt_pub ()

    destructor
- void add_subscriber (const pid_t pid)

    add subscriber to the subscriber list of the publisher
- void remove_subscriber (const pid_t pid)

    remove subscriber from subscriber list of the publisher
- void notify ()

    notify all subscriber

3.1.1 Detailed Description

real-time publisher

Author

karthik

Since

Mon Jan 04 2021

3.1.2 Constructor & Destructor Documentation

3.1.2.1 rt_pub() [1/2]

```
rt_pub::rt_pub (
             const rt_pub & )  [delete]
```

copy constructor deleted

3.1.2.2 rt_pub() [2/2]

```
rt_pub::rt_pub (
             int const signal_type,
             int const var_id )  [inline]
```

constructor of real-time publisher

Parameters

signal_type	Linux signal number
index	index variable to which publisher belongs

3.1.2.3 ∼rt_pub()

```
rt_pub::∼rt_pub ( )  [inline]
```

destructor

3.1.3 Member Function Documentation

3.1.3.1 add_subscriber()

```
void rt_pub::add_subscriber (
             const pid_t pid )  [inline]
```

add subscriber to the subscriber list of the publisher

Parameters

pid	process id of the subscriber

Returns

(void)

3.1.3.2 notify()

```
void rt_pub::notify ( )  [inline]
```

notify all subscriber

Returns

(void)

3.1.3.3 operator=()

```
rt_pub& rt_pub::operator= (
                const rt_pub & )  [delete]
```

assignment operator deleted

3.1.3.4 remove_subscriber()

```
void rt_pub::remove_subscriber (
                const pid_t pid )  [inline]
```

remove subscriber from subscriber list of the publisher

Parameters

pid	process id

Returns

(void)

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

- src/rt_pub.hpp

3.2 rt_sub Class Reference

real-time subscriber

`#include <rt_sub.hpp>`

Public Types

- using sig_handler = void(∗)(int signo, siginfo_t ∗info, void ∗extra)

  signal handler function type

Public Member Functions

- rt_sub (const rt_sub &)=delete

  copy constructor deleted
- rt_sub & operator= (const rt_sub &)=delete

  assignment operator deleted
- ∼rt_sub ()

  destrucor
- void block ()

  blocks the signals
- void unblock ()

  unblocks the signal
- void unblock_n_await ()

  unblocks and awaits for the signal
- void await ()

  awaits for the signal

Static Public Member Functions

- static rt_sub ∗ init (int const signal_type, const sig_handler signal_handler)

  initialization
- static rt_sub ∗ getInstance ()

  gets instance of the subscriber

3.2.1 Detailed Description

real-time subscriber

Author

Since

Mon Jan 04 2021

3.2.2 Member Typedef Documentation

3.2.2.1 sig_handler

```
using rt_sub::sig_handler = void (*)(int signo, siginfo_t *info, void *extra)
```

signal handler function type

Parameters

signo	signal number
info	signal information
extra	extra

Returns

void

3.2.3 Constructor & Destructor Documentation

3.2.3.1 rt_sub()

```
rt_sub::rt_sub (
             const rt_sub &  )  [delete]
```

copy constructor deleted

3.2.3.2 ∼rt_sub()

```
rt_sub::∼rt_sub ( )  [inline]
```

destrucor

3.2.4 Member Function Documentation

3.2.4.1 await()

```
void rt_sub::await ( )  [inline]
```

awaits for the signal

Returns

(void)

3.2.4.2 block()

```
void rt_sub::block ( )  [inline]
```

blocks the signals

Returns

(void)

3.2.4.3 getInstance()

```
static rt_sub* rt_sub::getInstance ( )  [inline], [static]
```

gets instance of the subscriber

Returns

pointers to the subscriber instance

3.2.4.4 init()

```
static rt_sub* rt_sub::init (
             int const signal_type,
             const sig_handler signal_handler )  [inline], [static]
```

initialization

Parameters

signal_type	signal value to be registered
signal_handler	signal handler to be attached

Returns

3.2.4.5 operator=()

```
rt_sub& rt_sub::operator= (
             const rt_sub &  )  [delete]
```

assignment operator deleted

3.2.4.6 unblock()

```
void rt_sub::unblock ( ) [inline]
```

unblocks the signal

Returns

(void)

3.2.4.7 unblock_n_await()

```
void rt_sub::unblock_n_await ( ) [inline]
```

unblocks and awaits for the signal

Returns

(void)

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

- src/rt_sub.hpp

3.3 shm_block Class Reference

shared memory block class

```
#include <shm_block.hpp>
```

Public Member Functions

- shm_block (const shm_block &)=delete

  copy constructor is deleted
- shm_block & operator= (const shm_block &)=delete

  assignment operator is deleted
- shm_block (const char ∗const name, unsigned int const block_id, unsigned int const block_size)

  constructor
- ∼shm_block () noexcept

  destructor
- template<typename T >
  void read (const int &index, T &val) const

  reads value from shared memory location of specified index variable
- template<typename T >
  void read (const int &index, volatile std::atomic< T > &val) const

  reads value from shared memory location of specified index variable
- template<typename T >
  void write (const int index, const T &value) const

  write data to shared memory of variable with specified index
- template<typename T >
  void write_ (const int index, std::atomic< T > &value) const

  write data to shared memory of variable with specified index

3.3.1 Detailed Description

shared memory block class

Author

karthik

Since

Mon Jan 04 2021

3.3.2 Constructor & Destructor Documentation

3.3.2.1 shm_block() [1/2]

```
shm_block::shm_block (
              const shm_block &  )  [delete]
```

copy constructor is deleted

3.3.2.2 shm_block() [2/2]

```
shm_block::shm_block (
              const char *const name,
              unsigned int const block_id,
              unsigned int const block_size )  [inline], [explicit]
```

constructor

Parameters

name	user defined name to the shared memory
block_id	user defined ID to the shared memory
block_size	size of the shared memory

3.3.2.3 ∼shm_block()

```
shm_block::∼shm_block ( )  [inline], [noexcept]
```

destructor

3.3.3 Member Function Documentation

3.3.3.1 operator=()

```
shm_block& shm_block::operator= (
             const shm_block & )  [delete]
```

assignment operator is deleted

3.3.3.2 read() [1/2]

```
template<typename T >
void shm_block::read (
             const int & index,
             T & val ) const  [inline]
```

reads value from shared memory location of specified index variable

Parameters

index	index of the variable
val	variable in which read value is stored

Returns

void

3.3.3.3 read() [2/2]

```
template<typename T >
void shm_block::read (
             const int & index,
             volatile std::atomic< T > & val ) const  [inline]
```

reads value from shared memory location of specified index variable

Parameters

index	index of the variable
val	atomic variable in which read value is stored

Returns

3.3.3.4 write()

```
template<typename T >
void shm_block::write (
            const int index,
            const T & value ) const  [inline]
```

write data to shared memory of variable with specified index

Parameters

index	index of the variable
value	value to be written

Returns

void

3.3.3.5 write_()

```
template<typename T >
void shm_block::write_ (
            const int index,
            std::atomic< T > & value ) const  [inline]
```

write data to shared memory of variable with specified index

Parameters

index	index index of the variable
value	atomic variable containing value to be written

Returns

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

- src/shm_block.hpp

3.4 Timer Class Reference

```
#include <benchmark.hpp>
```

Public Member Functions

- Timer ()
- ~Timer ()
- void start ()
- void stop ()

3.4.1 Constructor & Destructor Documentation

3.4.1.1 Timer()

```
Timer::Timer ( )  [inline]
```

3.4.1.2 ~Timer()

```
Timer::~Timer ( )  [inline]
```

3.4.2 Member Function Documentation

3.4.2.1 start()

```
void Timer::start ( )  [inline]
```

3.4.2.2 stop()

```
void Timer::stop ( )  [inline]
```

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

- src/benchmark.hpp

3.5 variable_info_t Class Reference

variable information type

```
#include <shm_variable_info_t.hpp>
```

Public Member Functions

- variable_info_t (const variable_info_t &)=delete

  copy constructor is deleted
- variable_info_t & operator= (const variable_info_t &)=delete

  asignment operator is deleted
- variable_info_t (const int index, const uint32_t mem_offset, const dtype_t type_info)

  constructor
- const int getindex () const

  gets index value of the variable
- const uint32_t getoffset () const

  gets offset value of memory location from shared memory's base address
- const uint8_t getTypeInfo ()

  gets data type information of the variable

3.5.1 Detailed Description

variable information type

Author

karthik

Since

Fri Jan 08 2021

3.5.2 Constructor & Destructor Documentation

3.5.2.1 variable_info_t() [1/2]

```
variable_info_t::variable_info_t (
             const variable_info_t &  )  [delete]
```

copy constructor is deleted

3.5.2.2 variable_info_t() [2/2]

```
variable_info_t::variable_info_t (
             const int index,
             const uint32_t mem_offset,
             const dtype_t type_info )  [inline], [explicit]
```

constructor

Parameters

index	index value of the variable
mem_offset	memory offset from base address of the shared memory
type_info	data type information (enumerated to dtype_t)

See also

dtype_t

3.5.3 Member Function Documentation

3.5.3.1 getindex()

```
const int variable_info_t::getindex ( ) const  [inline]
```

gets index value of the variable

Returns

index value in int

3.5.3.2 getoffset()

```
const uint32_t variable_info_t::getoffset ( ) const  [inline]
```

gets offset value of memory location from shared memory's base address

Returns

mem_offset value in uint32_t

3.5.3.3 getTypeInfo()

```
const uint8_t variable_info_t::getTypeInfo ( )  [inline]
```

gets data type information of the variable

Returns

uint8_t type value which enumerated by dtype_t

3.5.3.4 operator=()

```
variable_info_t& variable_info_t::operator= (
             const variable_info_t &  )  [delete]
```

asignment operator is deleted

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

- src/shm_variable_info_t.hpp

3.6 variable_t< T > Class Template Reference

Data dictionary variable type.

```
#include <shm_variables.hpp>
```

Public Member Functions

- variable_t (int const &index, int const signal_type)

  constructor
- void write (const T &value)

  Writes data into shared memory and notifies all subscriber.
- void read (T &value)

  reads data from shared memory
- void update ()

  updates value store of the variable from shared memory (called inside signal handler only)
- void add_subscriber (const pid_t pid)

  adds subscriber to the variable subscriber list
- void remove_subscriber (const pid_t pid)

  removes subscriber from the subscriber list of the variable

3.6.1 Detailed Description

template<class T>
class variable_t< T >

Data dictionary variable type.

Author

Karthik

Since

Fri Jan 08 2021

3.6.2 Constructor & Destructor Documentation

3.6.2.1 variable_t()

```
template<class T >
variable_t< T >::variable_t (
             int const & index,
             int const signal_type )  [inline], [explicit]
```

constructor

Parameters

index	index value of the variable
signal_type	Linux signal value

3.6.3 Member Function Documentation

3.6.3.1 add_subscriber()

```
template<class T >
void variable_t< T >::add_subscriber (
             const pid_t pid )  [inline]
```

adds subscriber to the variable subscriber list

Parameters

pid	process ID of the subscriber

Returns

(void)

3.6.3.2 read()

```
template<class T >
void variable_t< T >::read (
             T & value )  [inline]
```

reads data from shared memory

Parameters

value	value read from value store (not from shared memory)

Returns

(void)

3.6.3.3 remove_subscriber()

```
template<class T >
void variable_t< T >::remove_subscriber (
             const pid_t pid )  [inline]
```

removes subscriber from the subscriber list of the variable

Parameters

pid	process ID of the subscriber

Returns

(void)

3.6.3.4 update()

```
template<class T >
void variable_t< T >::update ( )  [inline]
```

updates value store of the variable from shared memory (called inside signal handler only)

Returns

(void)

3.6.3.5 write()

```
template<class T >
void variable_t< T >::write (
             const T & value )  [inline]
```

Writes data into shared memory and notifies all subscriber.

Parameters

value	value to be written

Returns

(void)

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

- src/shm_variables.hpp

Chapter 4

File Documentation

4.1 src/benchmark.hpp File Reference

```
#include <chrono>
#include <iostream>
```

Include dependency graph for benchmark.hpp:

4.2 src/data_dict.cpp File Reference

```
#include "benchmark.hpp"
#include "data_dict.hpp"
#include "shm_block.hpp"
#include <sys/types.h>
#include <sys/wait.h>
#include <utility>
```

Include dependency graph for data_dict.cpp:

Functions

- void rt_sub_handler (int signo, siginfo_t ∗info, void ∗extra)

Variables

- shm_block var_space ("/dev/shm/var", 65, 1024)
- shm_block proc_space ("/dev/shm/proc", 64, 128)
- variable_t< float > var_index_0 (0, SIGRTMIN+1)
- variable_info_t info_index_0 (0, 0, dtype_t::FLOAT32)
- variable_t< int8_t > var_index_1 (1, SIGRTMIN+1)
- variable_info_t info_index_1 (1, 4, dtype_t::SIGNED8)
- variable_t< int8_t > var_index_2 (2, SIGRTMIN+1)
- variable_info_t info_index_2 (2, 5, dtype_t::SIGNED8)
- variable_info_t ∗ indices [3] = {&info_index_0, &info_index_1, &info_index_2}

4.2.1 Function Documentation

4.2.1.1 rt_sub_handler()

```
void rt_sub_handler (
             int signo,
             siginfo_t * info,
             void * extra )
```

4.2.2 Variable Documentation

4.2.2.1 indices

```
variable_info_t* indices[3] = {&info_index_0, &info_index_1, &info_index_2}
```

4.2.2.2 info_index_0

```
variable_info_t info_index_0(0, 0, dtype_t::FLOAT32)
```

4.2.2.3 info_index_1

```
variable_info_t info_index_1(1, 4, dtype_t::SIGNED8)
```

4.2.2.4 info_index_2

```
variable_info_t info_index_2(2, 5, dtype_t::SIGNED8)
```

4.2.2.5 proc_space

```
shm_block proc_space("/dev/shm/proc", 64, 128)
```

variable space in shared memory

4.2.2.6 var_index_0

```
variable_t<float> var_index_0(0, SIGRTMIN+1)
```

4.2.2.7 var_index_1

```
variable_t<int8_t> var_index_1(1, SIGRTMIN+1)
```

4.2.2.8 var_index_2

```
variable_t<int8_t> var_index_2(2, SIGRTMIN+1)
```

4.2.2.9 var_space

```
shm_block var_space("/dev/shm/var", 65, 1024)
```

4.3 src/data_dict.hpp File Reference

```
#include "shm_variables.hpp"
```
Include dependency graph for data_dict.hpp: This graph shows which files directly or indirectly include this file:

Functions

- void rt_sub_handler (int signo, siginfo_t ∗info, void ∗extra)

Variables

- variable_t< float > var_index_0
- variable_t< int8_t > var_index_1
- variable_t< int8_t > var_index_2

4.3.1 Function Documentation

4.3.1.1 rt_sub_handler()

```
void rt_sub_handler (
             int signo,
             siginfo_t * info,
             void * extra )
```

4.3.2 Variable Documentation

4.3.2.1 var_index_0

```
variable_t<float> var_index_0
```

4.3.2.2 var_index_1

```
variable_t<int8_t> var_index_1
```

4.3.2.3 var_index_2

```
variable_t<int8_t> var_index_2
```

4.4 src/main.cpp File Reference

```
#include "data_dict.hpp"
#include "rt_sub.hpp"
#include <iostream>
#include <ostream>
#include <sys/types.h>
#include <unistd.h>
#include <sys/wait.h>
#include <utility>
```

Include dependency graph for main.cpp:

Functions

- int main (int argc, char **argv)

  Example application.

4.4.1 Function Documentation

4.4.1.1 main()

```
int main (
            int argc,
            char ** argv )
```

Example application.

Parameters

argc	
argv	

Returns

4.5 src/rt_pub.hpp File Reference

```
#include <stddef.h>
#include <stdint.h>
#include <stdio.h>
#include <signal.h>
#include <list>
#include <stdexcept>
#include <iostream>
```

Include dependency graph for rt_pub.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class rt_pub

  real-time publisher

4.6 src/rt_sub.hpp File Reference

```
#include <stddef.h>
#include <stdint.h>
#include <stdio.h>
#include <signal.h>
#include <list>
#include <stdexcept>
#include <iostream>
#include <exception>
```

Include dependency graph for rt_sub.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class rt_sub

  real-time subscriber

4.7 src/shm_block.hpp File Reference

```
#include <iostream>
#include <atomic>
#include <stddef.h>
#include <stdint.h>
#include <stdio.h>
#include <sys/ipc.h>
#include <sys/shm.h>
#include <cstdint>
#include <cstring>
#include <exception>
#include <shared_mutex>
#include <stdexcept>
#include "shm_variable_info_t.hpp"
```

Include dependency graph for shm_block.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class shm_block

  shared memory block class

Variables

- variable_info_t ∗ indices [ ]
- shm_block var_space
- shm_block proc_space

4.7.1 Variable Documentation

4.7.1.1 indices

```
variable_info_t* indices[]
```

4.7.1.2 proc_space

```
shm_block proc_space
```

variable space in shared memory

4.7.1.3 var_space

```
shm_block var_space
```

4.8 src/shm_variable_info_t.hpp File Reference

```
#include <stddef.h>
#include <stdint.h>
#include <stdio.h>
#include <array>
#include <atomic>
```

Include dependency graph for shm_variable_info_t.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class variable_info_t

  variable information type

Typedefs

- using dtype_t = enum { UNSIGNED8=0, UNSIGNED16, UNSIGNED32, UNSIGNED64, SIGNED8, SIGNE↩ D16, SIGNED32, SIGNED64, FLOAT32, FLOAT64 }

  enumeration of data type

4.8.1 Typedef Documentation

4.8.1.1 dtype_t

```
using dtype_t = enum { UNSIGNED8 = 0, UNSIGNED16, UNSIGNED32, UNSIGNED64, SIGNED8, SIGNED16,
SIGNED32, SIGNED64, FLOAT32, FLOAT64 }
```

enumeration of data type

4.9 src/shm_variables.hpp File Reference

```
#include "shm_block.hpp"
#include "rt_pub.hpp"
#include <stddef.h>
#include <stdint.h>
#include <stdio.h>
#include <atomic>
```

Include dependency graph for shm_variables.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class variable_t< T >

  Data dictionary variable type.